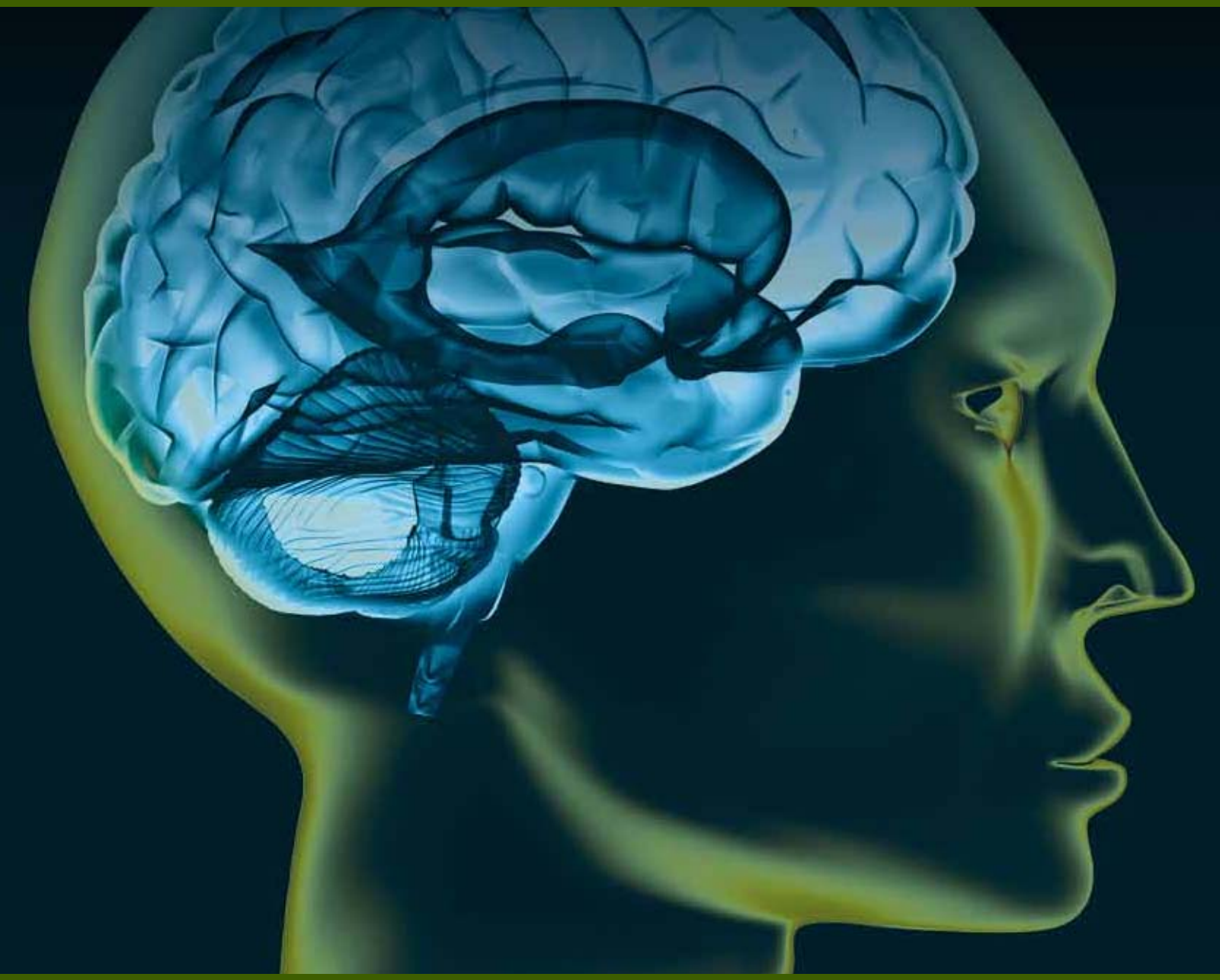
Computational Intelligence & Neuroscience

Academic Software Applications for Electromagnetic Brain Mapping Using MEG and EEG

Guest Editors: Sylvain Baillet, Karl Friston, and Robert Oostenveld



Academic Software Applications for Electromagnetic Brain Mapping Using MEG and EEG

Computational Intelligence and Neuroscience

Academic Software Applications for Electromagnetic Brain Mapping Using MEG and EEG

Guest Editors: Sylvain Baillet, Karl Friston, and Robert Oostenveld

Editorial Board

Contents

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 972050, 4 pages
doi:10.1155/2011/972050

Editorial

Academic Software Applications for Electromagnetic Brain Mapping Using MEG and EEG

Sylvain Baillet,[1] Karl Friston,[2] and Robert Oostenveld[3]

[1] *Departments of Neurology and Biophysics, Medical College of Wisconsin, Milwaukee, WI 53226, USA*
[2] *The Wellcome Trust Centre for Neuroimaging, UCL Institute of Neurology, Queen Square, London WC1N 3BG, UK*
[3] *Donders Institute for Brain, Cognition and Behaviour, Centre for Cognitive Neuroimaging, Radboud University Nijmegen, 6500 HB Nijmegen, The Netherlands*

Correspondence should be addressed to Sylvain Baillet, sbaillet@mcw.edu

Received 5 May 2011; Accepted 5 May 2011

The Hard Work and the Software

In the last decade, the imaging neuroscience community has become very amenable to data sharing and the exchange of people and ideas across imaging modalities. The development of academic software has accompanied this openness and has served to disseminate new and rapidly evolving methodologies and to facilitate the reproducibility of data analyses. The academic recognition of this effort is not straightforward. This special issue features 20 software contributions, all freely available to the research community. It represents a vast swath of MEG and EEG applications and research topics: from real-time data analysis for brain-machine interface applications to all-in-one applications for imaging, modeling and visualization. We hope this special issue offers a technical and citable reference to users of these software packages and will encourage more scientists to share and disseminate their ideas in this pragmatic fashion. Software development in academia has often been equated with prototyping *ad hoc* computing solutions to specific empirical problems. Until recently, in neuroscience, as in other highly specialized scientific fields, computer science skills have been regarded as insufficient to support the distribution of software packages beyond the institutions that developed them. Research groups often considered in-house software literally, as pieces of intellectual property that should not be shared with a community of potential competitors.

The explosion of neuroimaging advances in the 1990s triggered a shift in this perspective. Imaging neuroscience is essentially multidisciplinary: neuroscientists design paradigms, collect and then analyze data; physicists develop and implement new instrumentation, while mathematicians and theorists develop bespoke statistical analyses and the models needed to improve our understanding of how data are generated. In this context, software has become a vector for sharing, cross-fertilizing, disseminatiing and replicating scientific results. In essence, this new regard for software has paralleled the interest in open-source developments (Linux, Python libraries, etc.), public licensing (e.g., GNU General Public License, BSD) in computer science, and the emerging model of open-access peer-reviewed scientific publishing. It also has benefited enormously from rapid and accessible prototyping environments such as Matlab, which have enabled many neuroscientists to write robust and sophisticated software. We note that although these environments may not be free, their distributors usually grant privileged pricing options to academia. We are also aware that entirely free alternatives exist and are developing constantly (e.g., Octave, SciPy.org, and other Python initiatives).

Commercial software packages are readily available for some imaging modalities and could represent the best choice (or the only alternative) with respect to support, documentation, quality assurance, performance, and most importantly, regulatory and clinical compliance. However, in a field where methodological developments are evolving constantly to address new imaging challenges and techniques, the flexibility and reactivity of academic software ensure that it remains at the forefront of advances in brain imaging.

Nevertheless, software development in academia comes with a price. It takes a considerable amount of human

Table 1: Alphabetic list of the software contributions featured in this special issue. Three further prominent software packages (eConnectome, MNE and sLORETA) have been added, for reference and convenience of comparison. The main features of all software packages are itemized in terms of time-series analysis (from basic preprocessing of raw data to dynamic systems metrics), time-frequency analysis and decomposition, MEG/EEG forward modeling and/or source modeling (dipole fitting, beamforming/spatial filters, and source imaging), functional or effective connectivity measures and modeling, statistical analysis and inference, graphical user interface versus scripting libraries, interoperability with other software packages/the possibility for users to develop plugins, the computing environment and the operating system(s) (OS) required, and definitive feature(s) that characterizes each software package. Asterisks are for packages that are not featured as full-length articles in this special issue. Green cells indicate which attributes apply to each package.

Featured software packages	Time-series analysis	Time-frequency analysis	Forward modeling	Source modeling	Functional or e ective connectivity	Statistical analysis	Graphical user interface	Interoperability (other software packages/user plugins)	Environment/OS	Unique feature(s)
All-in-one packages										
BrainStorm http://neuroimage.usc.edu/brainstorm								User plugins	Matlab/multiple	Emphasis on user interaction through GUI or scripts
CARTOOL http://brainmapping.unige.ch/cartool									Windows	Quantitative assessment of EEG topographies; microstate segmentation
eConnectome (*) http://econnectome.umn.edu									Matlab/multiple	Focus on connectivity analysis and unique visualization tools
EEGLAB, SIFT, NFT, BCILAB & ERICA http://sccn.ucsd.edu/wiki/EEGLAB								EEGLAB, FieldTrip; user plugins	Matlab/multiple	Large user community; diverse toolkit: from data preprocessing to real-time analyses
ELAN http://elan.lyon.inserm.fr								Matlab, SPM, FieldTrip, NUTMEG, and so forth	Compiled C/multiple (virtual machine)	Wide range of preprocessing and signal analysis tools for electrophysiological data
EMEGS http://www.emegs.org/								SPM8; User plugins	Matlab/multiple	GUI-based data preprocessing
FieldTrip http://fieldtrip.fcdonders.nl/								BESA, EEGLAB, LORETA, SPM8, and so forth; userplugins	Matlab/multiple	Community-based development; extensive documentation; real-time features
MNE (*) http://www.nmr.mgh.harvard.edu/martinos/userInfo/data/sofMNE.php								Matlab	C/C++ (Linux/MacOS)	High-quality 3D graphics; computationally e cient
NUTMEG http://nutmeg.berkeley.edu								FieldTrip, SPM, OpenMEEG, ELAN, and so forth,	Matlab/multiple; Python	Multiple advanced source imaging techniques
sLORETA (*) http://www.uzh.ch/keyinst/loreta.htm									Windows	Large user community
SPM8 http://www.fil.ion.ucl.ac.uk/spm/								FieldTrip, BioSig; user plugins	Matlab/multiple	SPM tools and community; dynamic causal modeling

Table 1: Continued.

Featured software packages	Time-series analysis	Time-frequency analysis	Forward modeling	Source modeling	Functional or e ective connectivity	Statistical analysis	Graphical user interface	Interoperability (other software packages/user plugins)	Environment/OS	Unique feature(s)
Descriptive and inference statistics										
EEGIFT http://icatb.sourceforge.net/gift/eegift_startup.php	■						■	EEGLAB	Matlab/multiple	Group-level temporal ICA
LIMO EEG https://gforge.dcn.ed.ac.uk/gf/project/limo_eeg/	■					■	■	EEGLAB	Matlab/multiple	Simple hierarchical linear modeling of evoked responses
Ragu http://www.thomaskoenig.ch/Ragu_pkg.exe						■	■		Matlab/multiple	Multichannel randomization statistics
Brain-machine interface/Real-time data processing										
BioSig http://biosig.sourceforge.net/	■					■		FieldTrip, SPM, EEGLAB	Matlab, Octave, C/C++/multiple	Set of generic libraries for the analysis of biosignals; real-time features
Craniux http://hrnel.pitt.edu/Software.html	■						■		LabVIEW	Out-of-the-box modular brain-machine interface software framework
rtMEG http://fieldtrip.fcdonders.nl/development/realtime/neuromag	■							FieldTrip, BrainStorm	Matlab/multiple	Real-time signal processing and source localization
Special interest										
BrainNetVis http://www.ics.forth.gr/bmi/tools.html	■				■		■		Java/multiple	Network modeling and visualization
FASST http://www.montefiore.ulg.ac.be/~phillips/FASST.html	■						■	SPM8, EEGLAN, FSL	Matlab/Windows	Sleep scoring EEG-MRI artifact correction
HADES http://hades.dima.unige.it/		■		■			■	MNE	Matlab/multiple	Highly automated dipole fitting
OpenMEEG http://openmeeg.gforge.inria.fr			■					SPM, FieldTrip, BrainStorm	Multiple	High-quality realistic EEG and MEG head modeling
PyEEG http://pyeeg.sourceforge.net/	■								Python, SciPy/multiple	Fractal dimensions; entropy and complexity measures
TopoToolbox https://files.nyu.edu/xt235/public/	■						■	EEGLAB	Matlab/multiple	Using sensor topography to calculate psychologically meaningful measures

resources and skills, which are sometimes not fully appreciated by scientific peers and funding agencies. Although the vast majority of investigators have come to realize that good software underwrites the quality of scientific contributions and the profile of institutions disseminating software, the long-term investment of scientists who develop software is difficult to quantify and appreciate.

We have assembled this special issue to promote these efforts and to offer a bibliographic landmark to their authors. We have restricted the scope of this issue to software for electroencephalography (EEG) and magnetoencephalography (MEG) data analysis, for which a great variety of techniques have been developed. They embrace a scope that is representative of other neuroimaging modalities: from time series analysis to source modeling and image reconstruction; from mapping and localization to the modeling and quantification of functional and effective connectivity; from real-time data analyses to sophisticated group-level statistical inference.

We are grateful to the authors of the 20 manuscripts presented herewith (Table 1) and to our anonymous reviewers. We have organized the contents of this issue according to the nature of the software featured: all-in-one packages, tools for descriptive and inference statistics, tools for brain-machine interface applications and real-time data processing, and special-interest tools that usually feature a unique application. We believe these contributions offer a balanced reflection of the open issues and scientific challenges in MEG/EEG research. They also reveal the multiple faces of what academic software has come to offer: from specific toolkits that embody the methodological developments of a particular research group, to integrated software applications embracing multiple features, sophisticated user interfaces and libraries. It is worth mentioning that some of these tools have been developed and supported for over a decade. The requirements for acceptance to this special issue were, beyond technical validity, that the featured software is available for download and is free of charge and that user documentation is readily available.

We truly hope this special issue celebrates the communal efforts in our field and becomes the academic citation reference for the developers of the featured software packages and their users.

Sylvain Baillet
Karl Friston
Robert Oostenveld

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 879716, 13 pages
doi:10.1155/2011/879716

Research Article

Brainstorm: A User-Friendly Application for MEG/EEG Analysis

François Tadel,[1] Sylvain Baillet,[2] John C. Mosher,[3] Dimitrios Pantazis,[4] and Richard M. Leahy[1]

[1] *Signal & Image Processing Institute, University of Southern California, Los Angeles, CA 90089, USA*
[2] *MEG Program, Departments of Neurology & Biophysics, Froedtert & Medical College of Wisconsin, Milwaukee, WI 53226, USA*
[3] *Epilepsy Center, Cleveland Clinic Neurological Institute, Cleveland, OH 44195, USA*
[4] *MEG Lab, McGovern Institute for Brain Research, Massachusetts Institute of Technology, Cambridge, MA 02139, USA*

Correspondence should be addressed to François Tadel, tadel@usc.edu

Received 4 October 2010; Accepted 28 January 2011

Academic Editor: Robert Oostenveld

Brainstorm is a collaborative open-source application dedicated to magnetoencephalography (MEG) and electroencephalography (EEG) data visualization and processing, with an emphasis on cortical source estimation techniques and their integration with anatomical magnetic resonance imaging (MRI) data. The primary objective of the software is to connect MEG/EEG neuroscience investigators with both the best-established and cutting-edge methods through a simple and intuitive graphical user interface (GUI).

1. Introduction

Although MEG and EEG instrumentation is becoming more common in neuroscience research centers and hospitals, research software availability and standardization remain limited compared to the other functional brain imaging modalities. MEG/EEG source imaging poses a series of specific technical challenges that have, until recently, impeded academic software developments and their acceptance by users (e.g., the multidimensional nature of the data, the multitude of approaches to modeling head tissues and geometry, and the ambiguity of source modeling). Ideally, MEG/EEG imaging is multimodal: MEG and EEG recordings need to be registered to a source space that may be obtained from structural MRI data, which adds to the complexity of the analysis. Further, there is no widely accepted standard MEG/EEG data format, which has limited the distribution and sharing of data and created a major technical hurdle to academic software developers.

MEG/EEG data analysis and source imaging feature a multitude of possible approaches, which draw on a wide range of signal processing techniques. Forward head modeling for example, which maps elemental neuronal current sources to scalp potentials and external magnetic fields, is dependent on the shape and conductivity of head tissues and can be performed using a number of methods, ranging from simple spherical head models [1] to overlapping spheres [2] and boundary or finite element methods [3]. Inverse source modeling, which resolves the cortical sources that gave rise to MEG/EEG recordings, has been approached through a multitude of methods, ranging from dipole fitting [4] to distributed source imaging using Bayesian inference [5–7]. This diversity of models and methods reflects the ill-posed nature of electrophysiological imaging which requires restrictive models or regularization procedures to ensure a stable inverse solution.

The user's needs for analysis and visualization of MEG and EEG data vary greatly depending on their application. In a clinical environment, raw recordings are often used to identify and characterize abnormal brain activity, such as seizure events in epileptic patients [8]. Alternatively, ordering data into trials and averaging of an evoked response [9] remains the typical approach to revealing event-related cortical activity. Time-frequency decompositions [10] provide insight into induced responses and extend the analysis of MEG/EEG time series at the sensor and source levels to the spatial, temporal, and spectral dimensions. Many of these techniques give rise to computational and storage related

challenges. More recently, an increasing number of methods have been proposed to address the detection of functional and effective connectivity among brain regions: coherence [11], phase locking value [12], Granger causality [13, 14] and its multivariate extensions [15], and canonical correlation [16] among others. Finally, the low spatial resolution and nonisotropic covariance structure of measurements requires adequate approaches to their statistical analysis [17].

Despite such daunting diversity and complexity in user needs and methodological approaches, an integrated software solution would be beneficial to the imaging community and provide progressive automation, standardization and reproducibility of some of the most common analysis pathways. The Brainstorm project was initiated more than 10 years ago in collaboration between the University of Southern California in Los Angeles, the Salpêtrière Hospital in Paris, and the Los Alamos National Laboratory in New Mexico. The project has been supported by the National Institutes of Health (NIH) in the USA and the Centre National de la Recherche Scientifique (CNRS) in France. Its objective is to make a broad range of electromagnetic source imaging and visualization techniques accessible to nontechnical users, with an emphasis on the interaction of users with their data at multiple stages of the analysis. The first version of the software was released in 2000, [18] and a full graphic user interface (GUI) was added to Brainstorm 2 in 2004 [19]. As the number of users grew, the interface was completely redesigned and improved, as described in this paper. In response to the high demand from users, many other tools were integrated in Brainstorm to cover the whole processing and visualization pipeline of MEG/EEG recordings, from the importing of data files, from a large selection of formats, to the statistical analysis of source imaging maps. Brainstorm 3 was made available for download in June 2009 and was featured at the 15th Human Brain Mapping Conference in San Francisco. The software is now being improved and updated on a regular basis. There have been about 950 new registered users since June 2009, for a total of 4,000 since the beginning of the project.

Brainstorm is free and open source. Some recent publications using Brainstorm as a main analysis software tool are listed in [20–26]. This paper describes the Brainstorm project and the main features of the software, its connection to other projects, and some future developments that are planned for the next two years. This paper describes the software only; methodological background material is not presented here but can be found in multiple review articles and books, for example, [1, 27, 28].

2. Software Overview

Brainstorm is open-source software written almost entirely in Matlab scripts and distributed under the terms of the General Public License (GPL). Its interface is written in Java/Swing embedded in Matlab scripts, using Matlab's ability to work as a Java console. The use of Matlab and Java make Brainstorm a fully portable, cross-platform application.

The advantage of scripting languages in a research environment is the simplicity to maintain, modify, exchange, and reuse functions and libraries. Although Python might be a better choice for a new project because of its non-commercial open source license, Brainstorm was built from a vast amount of pre-existing lines of Matlab code as its methodological foundations for data analysis. The Matlab development environment is also a high-performance prototyping tool. One important feature for users who do not own a Matlab license is that a stand-alone version of Brainstorm, generated with the Matlab Compiler, is also available for download for the Windows and Linux operating systems.

All software functions are accessible through the GUI, without any direct interaction with the Matlab environment; hence, Brainstorm can be used without Matlab or programming experience. For more advanced users, it is also possible to run all processes and displays from Matlab scripts, and all data structures manipulated by Brainstorm can be easily accessed from the Matlab command window.

The source code is accessible for developers on an SVN server, and all related Brainstorm files are compressed daily into a zip file that is publicly available from the website, to facilitate download and updates for the end user. Brainstorm also features an automatic update system that checks at each startup if the software should be updated and whether downloading a new version is necessary.

User documentation is mainly organized in detailed online tutorials illustrated with numerous screen captures that guide the user step by step through all software features. The entire website is based on a MoinMoin wiki system [29]; hence, the community of users is able to edit the online documentation. Users can report bugs or ask questions through a VBulletin forum [30], also accessible from the main website.

3. Integrated Interface

Brainstorm is driven by its interface: it is not a library of functions on top of which a GUI has been added to simplify access but rather a generic environment structured around one unique interface in which specific functions were implemented (Figure 1). From the user perspective, its organization is contextual rather than linear: the multiple features from the software are not listed in long menus; they are accessible only when needed and are typically suggested within contextual popup menus or specific interface windows. This structure provides faster and easier access to requested functions.

Data files are saved in the Matlab.mat format and are organized in a structured database with three levels of classification: protocols, subjects, and experimental conditions. User data is always directly accessible from the database explorer, regardless of the actual file organization on the hard drive. This ensures immediate access to all protocol information and allows simultaneous display and comparison of recordings or sources from multiple runs, conditions, or subjects.

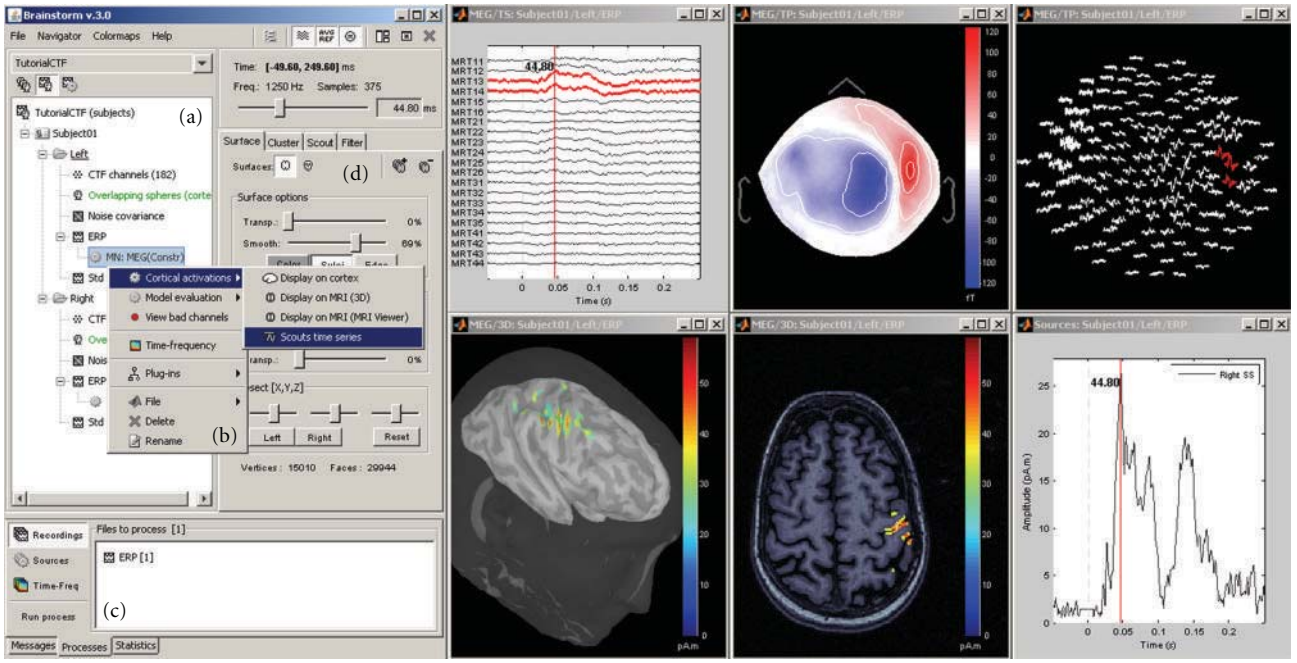


FIGURE 1: General overview of the Brainstorm interface. Considerable effort was made to make the design intuitive and easy to use. The interface includes: (a) a file database that provides direct access to all data (recordings, surfaces, etc.), (b) contextual menus that are available throughout the interface with a right-button click, (c) a batch tool that launches processes (filtering, averaging, statistical tests, etc.) for all files that were drag-and-dropped from the database; (right) multiple displays of information from the database, organized as individual figures and automatically positioned on the screen, and (d) properties of the currently active display.

4. Supported File Formats

Brainstorm requires three categories of inputs to proceed to MEG/EEG source analysis: the anatomy of the subject, the MEG/EEG recordings, and the 3D locations of the sensors. The anatomy input is usually a T1-weighted MRI of the full head, plus at least two tessellated surfaces representing the cerebral cortex and scalp. Supported MRI formats include Analyze, NIfTI, CTF, Neuromag, BrainVISA, and MGH. Brainstorm does not extract cortical and head surfaces from the MRI, but imports surfaces from external programs. Three popular and freely available surface formats are supported: BrainSuite [31], BrainVISA [32], and FreeSurfer [33].

The native file formats from three main MEG manufacturers are supported: Elekta-Neuromag, CTF, and BTi/4D-Neuroimaging. The generic file format developed at La Salpêtrière Hospital in Paris (LENA) is also supported. Supported EEG formats include: Neuroscan (cnt, eeg, avg), EGI (raw), BrainVision BrainAmp, EEGLab, and Cartool. Users can also import their data using generic ASCII text files.

Sensor locations are always included in MEG files; however, this is not the case for the majority of EEG file formats. Electrode locations need to be imported separately. Supported electrode definition files include: BESA, Polhemus Isotrak, Curry, EETrak, EGI, EMSE, Neuroscan, EEGLab, Cartool, and generic ASCII text files.

Other formats not yet supported by Brainstorm will be available shortly. Our strategy will merge Brainstorm's functions for the input and output from and to external file formats with the *fileio* module from the FieldTrip toolbox [34]. This independent library, also written in Matlab code, contains routines to read and write most of the file formats used in the MEG/EEG community and is already supported by the developers of multiple open-source software packages (EEGLab, SPM, and FieldTrip).

5. Data Preprocessing

Brainstorm features an extensive preprocessing pipeline for MEG/EEG data: visual or automatic detection of bad trials and bad channels, event marking and definition, baseline correction, frequency filtering, data resampling, averaging, and the estimation of noise statistics. Other preprocessing operations can be performed easily with other programs (EEGLab [35], FieldTrip, or MNE [36]) and results then imported into Brainstorm as described above.

Expanding preprocessing operations with the most popular techniques for noise reduction and automatic artifact detection is one of our priorities for the next few years of development.

6. Visualization of Sensor Data

Brainstorm provides a rich interface for displaying and interacting with MEG/EEG recordings (Figure 2) including various displays of time series (a)–(c), topographical mapping on 2D or 3D surfaces (d)-(e), generation of animations and series of snapshots of identical viewpoints at sequential

time points (f), the selection of channels and time segments, and the manipulation of clusters of sensors.

These visualization tools can be used either on segments of recordings that are fully copied into the Brainstorm database and saved in the Matlab.mat file format, or on typically larger, ongoing recordings, directly read from the original files and which remain stored in native file formats. The interface for reviewing raw recordings (Figure 3) also features event marking in a fast and intuitive way, and the simultaneous display of the corresponding source model (see below).

7. Visualization of Anatomical Surfaces and Volumes from MRI

Analysis can be performed on the individual subject anatomy (this requires the importation of the MRI and surfaces as described above) or using the Brainstorm's default anatomy (included in Brainstorm's distribution), which is derived from the MNI/Colin27 brain [37]. A number of options for surface visualization are available, including transparency, smoothing, and downsampling of the tessellated surface. Figure 4 shows some of the possible options to visualize MRI volumes and surfaces.

8. Registration of MEG/EEG with MRI

Analysis in Brainstorm involves integration of data from multiple sources: MEG and/or EEG recordings, structural MRI scans, and cortical and scalp surface tessellations. Their geometrical registration in the same coordinate system is essential to the accuracy of source imaging. Brainstorm aligns all data in a subject coordinate system (SCS), whose definition is based on 3 fiducial markers: the nasion, left preauricular, and right preauricular points: more details regarding the definition of the SCS are available at Brainstorm's website.

MRI-Surfaces. Aligning the MRI data volume with the surface tessellations of the head tissues is straightforward and automatic as both usually originate from the same volume of data. Nevertheless, Brainstorm features several options to manually align the surface tessellations with the MRI and to perform quality control of this critical step including definition of the reference points on the scalp surface (Figure 5(a)) and visual verification of the proper alignment of one of the surfaces in the 3D MRI (Figures 5(b), 5(c)).

Registration of MRI with MEG/EEG. The fiducial reference points need to be first defined in the MRI volume (see above and Figure 4) and are then pair matched with the coordinates of the same reference points as measured in the coordinate system of the MEG/EEG during acquisition. Alignment based on three points only is relatively inaccurate and can be advantageously complemented by an automatic refinement procedure when the locations of additional scalp points were acquired during the MEG/EEG session, using a 3D digitizer device. Brainstorm lets the user run this additional alignment, which is based on an iterated closest point algorithm, automatically.

It is common in EEG to run a study without collecting individual anatomical data (MRI volume data or individual electrode positions). Brainstorm has a tool that lets users define and edit the locations of the EEG electrodes at the surface of the individual or generic head (Figure 6). This tool can be used to manually adjust one of the standard EEG montages available in the software, including those already defined for the MNI/Colin27 template anatomy.

Volume and Surface Warping of the Template Anatomy. When the individual MRI data is not available for a subject, the MNI/Colin27 template can be warped to fit a set of head points digitized from the individual anatomy of the subject. This creates an approximation of the individual anatomy based on scalp morphology, as illustrated in Figure 7. Technical details are provided in [38]. This is particularly useful for EEG studies where MRI scans were not acquired and the locations of scalp points are available.

9. Forward Modeling

Forward modeling refers to the correspondence between neural currents and MEG/EEG sensor measurements. This step depends on the shape and conductivity of the head and can be computed using a number of methods, ranging from simple spherical head models [1] to overlapping spheres [2] and boundary or finite element methods [39].

Over the past ten years, multiple approaches to forward modeling have been prototyped, implemented, and tested in Brainstorm. The ones featured in the software today offer the best compromise between robustness (adaptability to any specific case) and accuracy (precision of the results). Other techniques will be added in the future. Current models include the single sphere and overlapping spheres methods for MEG [2] and Berg's three-layer sphere model for EEG [40]. For the single sphere methods, an interactive interface helps the user refine—after automatic estimation—the parameters of the sphere(s) that best fits the subject's head (Figure 8).

EEG is more sensitive to approximations in the geometry of the head as a volume conductor so that boundary element methods (BEMs) may improve model accuracy. A BEM approach for both MEG and EEG will soon be added to Brainstorm through a contribution from the OpenMEEG project [41], developed by the French National Institute for Research in Computer Science and Control (INRIA).

10. Inverse Modeling

Inverse modeling resolves the cortical sources that gave rise to a specific set of MEG or EEG recordings. In Brainstorm, the main method to estimate source activities is adapted from the depth-weighted minimum L2 norm estimator of cortical current density [42], which can subsequently be normalized using either the statistics of noise (dSPM [43]) or the data covariance (sLORETA [44]), as estimated

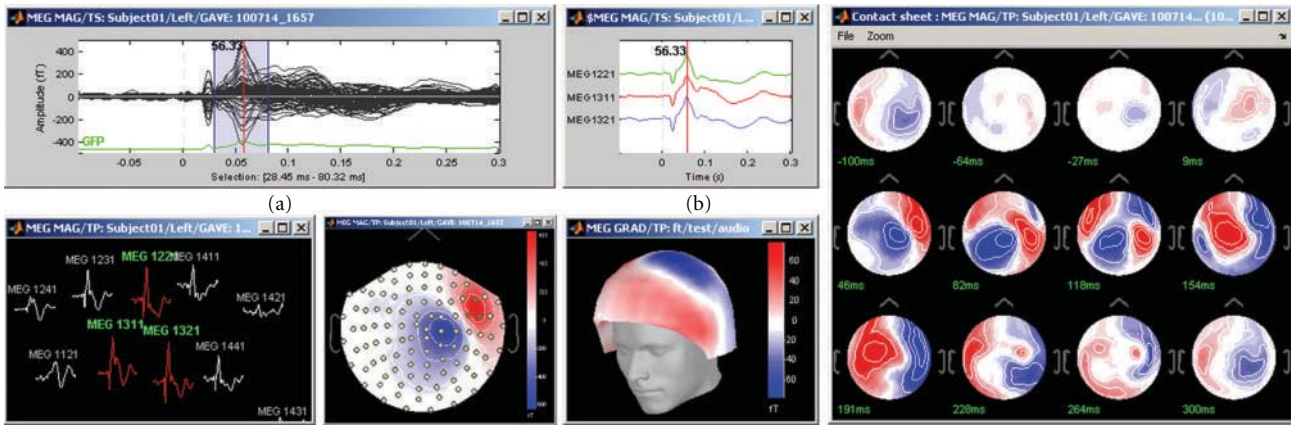


Figure 2: Brainstorm features multiple solutions for the visualization of MEG/EEG recordings.

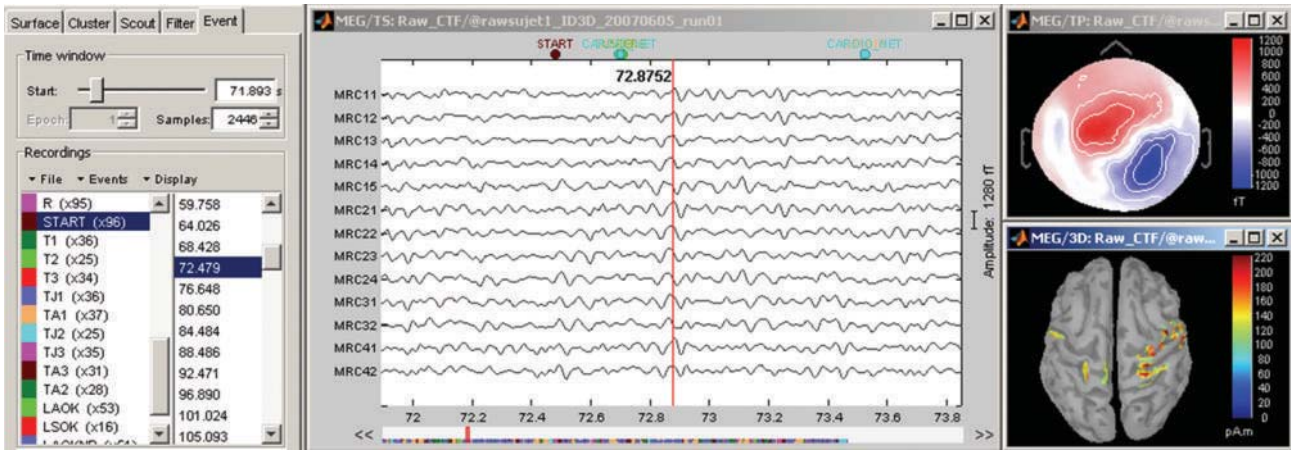


Figure 3: Interface for reviewing raw recordings and marking events.

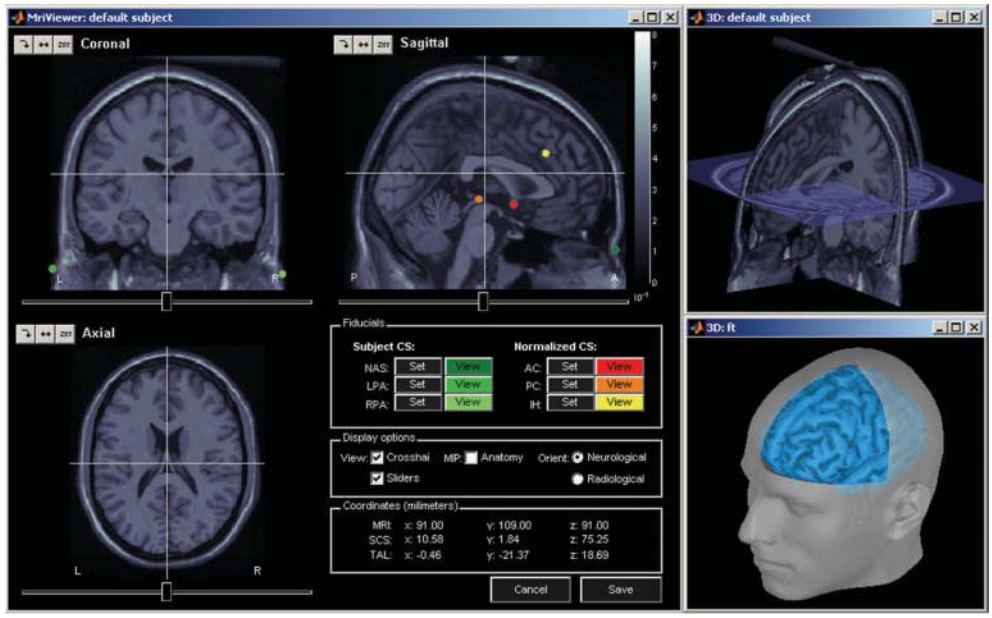


Figure 4: MRI and surface visualization.

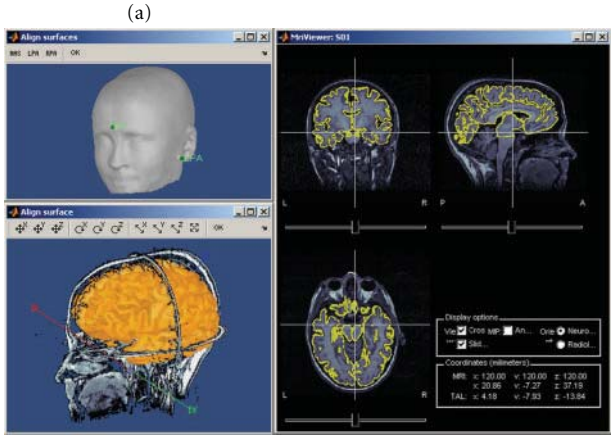


Figure 5: Registration of MRI data volumes with corresponding surface meshes.

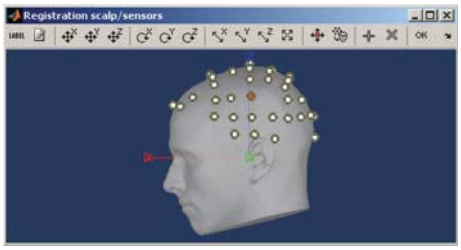


Figure 6: Brainstorm tool for editing of EEG electrode montages.

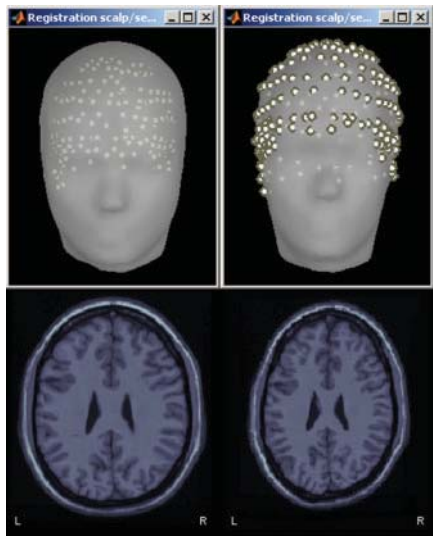


Figure 7: Warping of the MRI volume and corresponding tissue surface envelopes of the Colin27 template brain to fit a set a digitized head points (white dots in upper right corner): initial Colin27 anatomy (left) and warped to the scalp control points of another subject (right). Note how surfaces and MRI volumes are adjusted to the individual data.

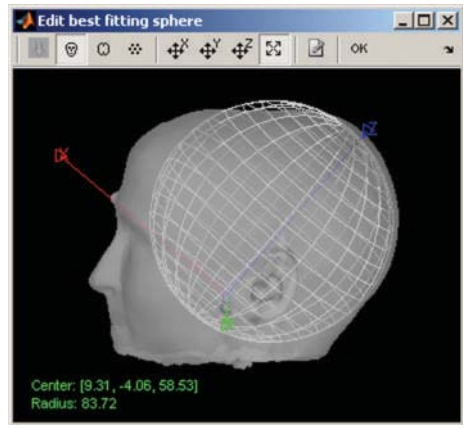


Figure 8: Interactive selection of the best-fitting sphere model parameter for MEG and EEG forward modeling.

from experimental recordings. For consistency and in an effort to promote standardization, the implementation of these estimators is similar to the ones available in the MNE software [36]. Two additional inverse models are available in Brainstorm: a linearly-constrained minimum variance (LCMV) beamformer [45] and the MUSIC signal classification technique [4, 46]. We also plan to add least squares multiple dipole fitting [4] to Brainstorm in the near future.

The region of support for these inverse methods can be either the entire head volume or restricted to the cortical surface, with or without constraints on source orientations. In the latter case, elementary dipole sources are distributed over the nodes of the surface mesh of the cortical surface. The orientation of the elementary dipoles can be left either unconstrained or constrained normally to the cortical surface. In all cases, the recommended number of dipoles to use for source estimation is about 15,000 (decimation of the original surface meshes can be performed within Brainstorm).

Brainstorm can manage the various types of sensors (EEG, MEG gradiometers, and MEG magnetometers) that may be available within a given dataset. When multiple sensor types are processed together in a joint source model, the empirical noise covariance matrix is used to estimate the weight of each individual sensor in the global reconstruction. The noise covariance statistics are typically obtained from an empty-room recording, which captures the typical instrumental and environmental fluctuations.

11. Source Visualization and Analysis

Brainstorm provides a large set of tools to display, visualize, and explore the spatio-temporal features of the estimated source maps (Figure 9), both on the cortical surface (a) and in the full head volume (b). The sources estimated on the cortical surface can be reprojected and displayed in the original volume of the MRI data (c) and on another mesh of the cortex at a higher or lower resolution. Reconstructed

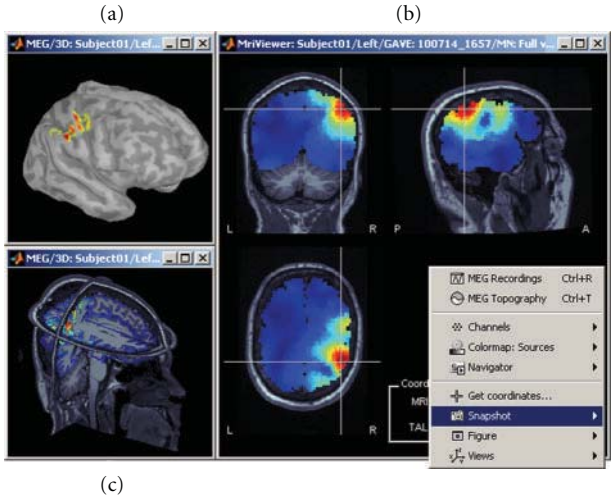


Figure 9: A variety of options for the visualization of estimated sources. (a) 3D rendering of the cortical surface, with control of surface smoothing; (c) 3D orthogonal planes of the MRI volumes; (b) conventional orthogonal views of the MRI volume with overlay of the MEG/EEG source density.

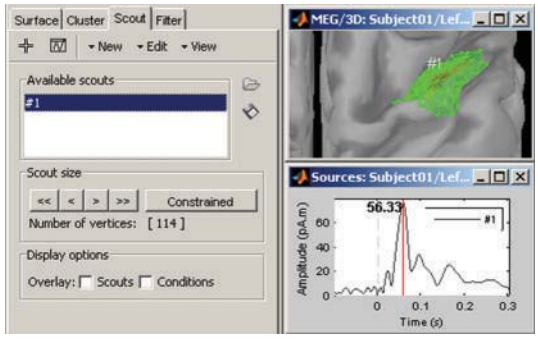


Figure 10: Selection of cortical regions of interest in Brainstorm and extraction of a representative time course of the elementary sources within.

current values can be smoothed in space or in time before performing group analysis.

A dedicated interface lets the user define and analyze the time courses of specific regions of interest, named *scouts* in Brainstorm (Figure 10). Brainstorm distribution includes two predefined segmentations of the default anatomy (MNI Colin27 [37]) into regions of interest, based on the anatomical atlases of Tzourio-Mazoyer et al. [47].

The rich contextual popup menus available in all visualization windows suggest predefined selections of views for creating a large variety of plots. The resulting views can be saved as images, movies, or contact sheets (Figure 9). Note that it is also possible to import dipoles estimated with the FDA-approved software Xfit from Elekta-Neuromag (Figure 11).

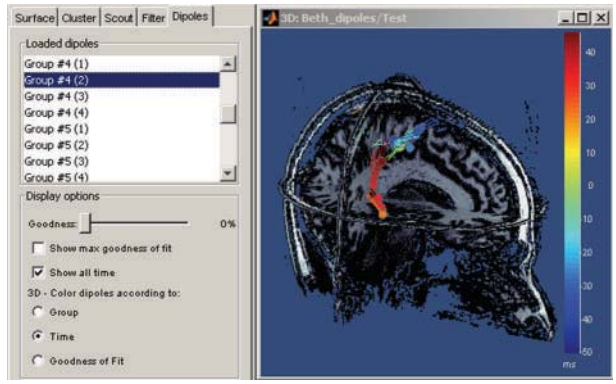


Figure 11: Temporal evolution of elementary dipole sources estimated with the external Xfit software. Data from a right-temporal epileptic spike. This component was implemented in collaboration with Elizabeth Bock, MEG Program, Medical College of Wisconsin.

12. Time-Frequency Analysis of Sensor and Source Signals

Brainstorm features a dedicated user interface for performing the time-frequency decomposition of MEG/EEG sensor and source time series using Morlet wavelets [10]. The shape—scaled versions of complex-valued sinusoids weighted by a Gaussian kernel—of the Morlet wavelets can efficiently capture bursts of oscillatory brain activity. For this reason, they are one of the most popular tools for time-frequency decompositions of electrophysiological data [26, 48]. The temporal and spectral resolution of the decomposition can be adjusted by the user, depending on the experiment and the specific requirements of the data analysis to be performed.

Time-frequency decompositions tend to increase the volume of data dramatically, as it is decomposed in the space, time, and frequency dimensions. Brainstorm has been efficiently designed to either store the transformed data or compute it on the fly.

Data can be analyzed as instantaneous measurements, or grouped into temporal and spectral bands of interest such as alpha (8–12 Hz) [26, 49], theta (5–7 Hz) [50–53], and so forth. Even though this reduces the resolution of the decomposition, it may benefit the analysis in multiple ways: reduced data storage requirements, improved signal-to-noise ratio, and a better control over the issue of multiple comparisons by reducing the number of concurrent hypothesis being tested.

Figure 12 illustrates some of the displays available to explore time-frequency decompositions: time-frequency maps of the times series from one sensor (a)-(b), one source (c) and one or more scouts (d), time courses of the power of the sensors for one frequency band (e), 2D/3D mappings (f), and cortical maps (g)-(h) of the power for one time and one frequency band.

13. Graphical Batching Interface

The main window includes a graphical batching interface (Figure 13) that directly benefits from the database

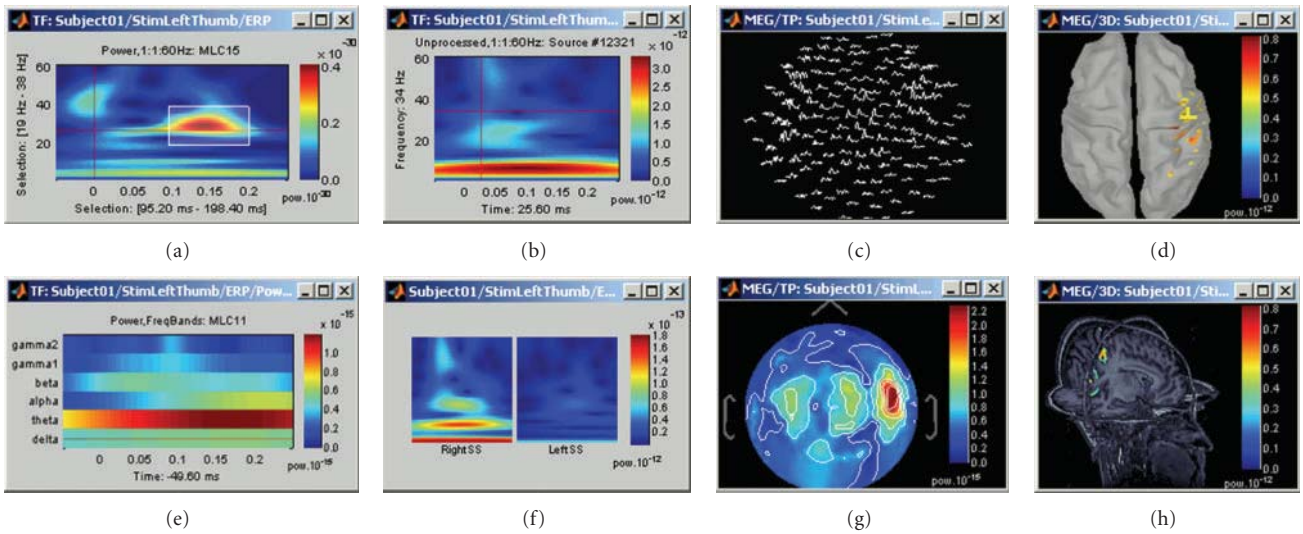


Figure 12: A variety of display options to visualize time-frequency decompositions using Brainstorm (see text for details).

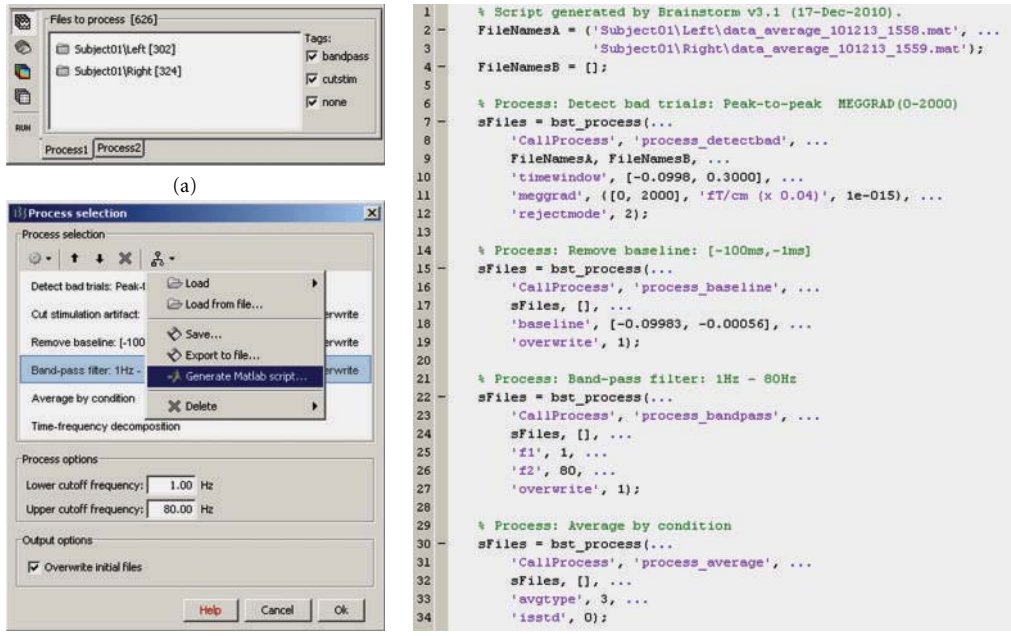


Figure 13: Graphical interface of the batching tool. (a) selection of the input files by drag-and-drop. (b) creation of an analysis pipeline. (c) example of Matlab script generated automatically.

display: files are organized as a tree of subjects and conditions, and simple drag-and-drop operations readily select files for subsequent batch processing. Most of the Brainstorm features are available through this interface, including preprocessing of the recordings, averaging, estimation of the sources, time-frequency decompositions, and computing statistics. A full analysis pipeline can be created in a few minutes, saved in the user's preferences and reloaded in one click, executed directly or exported as a Matlab script.

The available processes are organized in a plug-in structure. Any Matlab script that is added to the plug-in folder and has the right format will be automatically detected and made available in the GUI. This mechanism makes the contribution from other developers to Brainstorm very easy.

14. High-Level Scripting

For advanced users and visualization purposes, Brainstorm can be used as a high-level scripting environment. All Brainstorm operations have been designed to interact with the graphical interface and the database; therefore, they have very simple inputs: mouse clicks and keyboard presses. As

```
% ===== COMPUTE SOURCES =====
% Start brainstorm in script mode (main window is hidden)
brainstorm nogui
% Get from the database: condition "Right" for subject #1
[sStudy, iStudy] = bst_get('StudyWithCondition', 'Subject01/Right');
% Minimum norm source estimation for all the recordings in the condition
srcFiles = script_minnorm(iStudy, [], 'MN: kernel', 'MEG');

% ===== VISUALIZATION =====
% View sources on the cortex surface (first file in the folder)
hFig = script_view_sources(srcFiles(1), 'cortex');
% Set current time to 24ms
panel_time('SetCurrentTime', 0.024);
% Set data threshold to 35% of the maximal value
panel_surface('SetDataThreshold', hFig, 1, .35);
% Rotate view to visualize the cortex from the left
figure_3d('SetStandardView', hFig, 'left');
% Close all the figures and clear the allocated memory
bst_datasets('UnloadAll', 'Forced');
```

Figure 14: Example of Brainstorm script.

a result, the interface can be manipulated through Matlab scripts, and each mouse click can be translated into a line of script. Similar to working through the graphical interface, all contextual information is gathered from the interface and the database, so that most of the functions may be called with a limited number of parameters, and, for example, there is no need to keep track of file names. As a result, scripting with Brainstorm is intuitive and easy to use. Figure 14 shows an example of a Matlab script using Brainstorm.

15. Solutions for Performing Group Analyses with MEG/EEG Data and Source Models

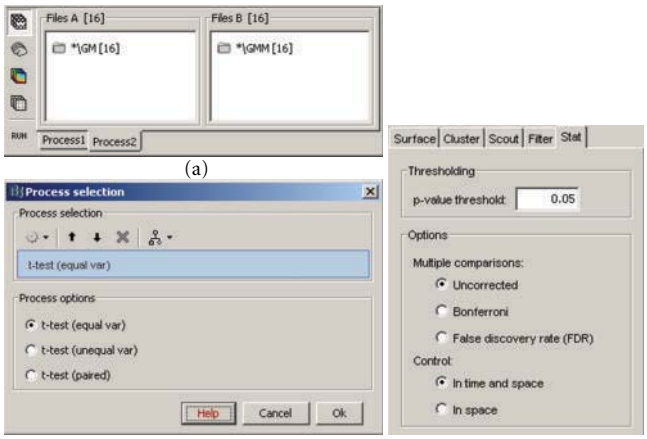
Brainstorm's "*Process2*" tab allows the comparison of two data samples. This corresponds to a single factor 2-level analysis and supported tests include simple difference, paired/unpaired Student t-tests of equal/unequal variance, and their nonparametric permutation alternatives [17]. The two groups can be assembled from any type of files, for example, two conditions within a subject, two conditions across subjects or two subjects for the same conditions, and so forth. These operations are generic in Brainstorm and can be applied to any type of data in the database: MEG/EEG recordings, source maps, and time-frequency decompositions. Furthermore, analysis of variance (ANOVA) tests are also supported up to 4 factors. Figure 15 displays the use of a Student t-test to compare two conditions, "GM" and "GMM," across 16 subjects.

We specifically address here how to perform multisubject data analysis using Brainstorm. In multisubject studies, measurement variance has two sources: the within-subject variance and the between-subject variance. Using collectively all trials from every subject simultaneously for comparisons is fixed-effects analysis [54] and does not consider the multiple sources of variance. Random-effects analysis [54, 55], which properly takes into account all sources of variance, is available in Brainstorm in its simplest and most commonly used form of the summary statistic approach [56, 57]. Based on this approach, analysis occurs at two levels. At the first level, trials from each subject are used to calculate statistics of interest separately for each subject, and at the second level, the different subjects are combined into an overall statistic.

Consider the example of investigating experimental effects, where prestimulus data are compared against post-stimulus data. The first level analysis averages all trials from each subject to yield prestimulus and post-stimulus responses. The second-level analysis can be a paired t-test between the resulting N prestimulus maps versus the N post-stimulus maps, where N is the number of subjects. Brainstorm processes and statistics include averaging trials and paired t-tests, making such analysis possible. Also, the procedure described above assumes equal within-subject variance, but the subjects can be weighted accordingly if this is not the case.

Brainstorm also supports statistical thresholding of the resulting activation maps, which takes into account the multiple hypotheses testing problem. The available methods include Bonferroni, false discovery rate [58], which controls the expected portion of false positives among the rejected hypotheses, and familywise error rate [59], which controls the probability of at least one false positive under the null hypothesis of no experimental effect. The latter is controlled with a permutation test and the maximum statistic approach, as detailed in [17].

In order to compare multiple subjects at the source level, an intermediate step is required if the sources were originally mapped on the individual subject anatomies. The sources estimated on individual brains are first projected on the cortical surface of the MNI-Colin27 brain. In the current implementation, the surface-to-surface registration is performed hemisphere by hemisphere using the following procedure: (1) alignment along the anterior commissure/posterior commissure axis, (2) spatial smoothing to preserve only the main features of the surfaces onto which the registration will be performed, (3) deformation of the



(a) (b) (c)

Figure 15: Student t-test between two conditions. (a) selection of the files. (b) selection of the test. (c) options tab for the visualization of statistical maps, including the selection of the thresholding method.

individual surface to match the MNI surface with an iterative closest point algorithm (ICP) [60], and (4) interpolation of the source amplitudes using Shepard's method [61]. Figure 16 shows the sources on the individual anatomy (left), and its reprojection on the MNI brain (right). This simple approach will eventually be replaced by cortical surface registration and surface-constrained volume registration methods developed at the University of Southern California as described in [62]. We will also add functionality to use the common coordinate system used in FreeSurfer for intersubject surface registration.

16. Future Developments

Brainstorm is a project under constant development, and the current version provides an environment where new features are readily implemented and adapted to the interface. There are several recurrent requests from users for new features, as well as plans for future developments. Examples of forthcoming developments in the next two years include:

– expanding the preprocessing operations with the most popular techniques for noise reduction and automatic artifact detection,

– integration of methods for functional connectivity analysis and multivariate statistical analysis [16, 63],

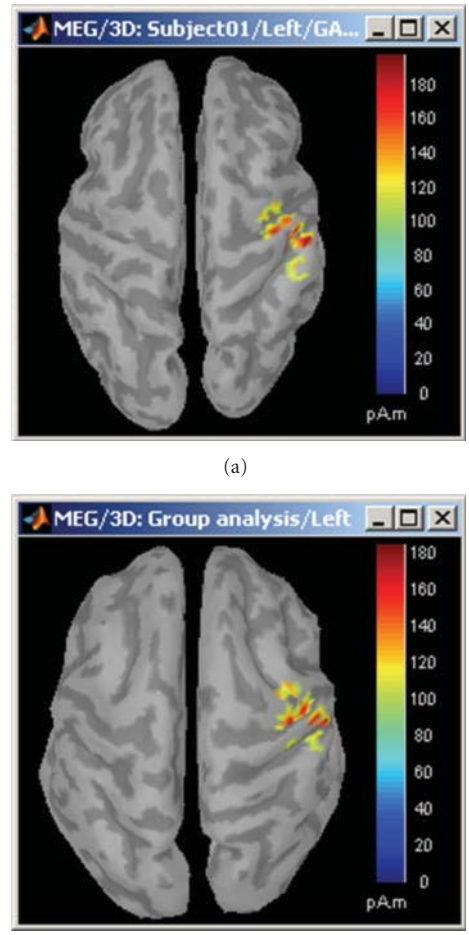
– expanding forward and inverse calculations to include BEM and multiple dipole fitting methods,

– interface for simulating MEG/EEG recordings using simulated sources and realistic anatomy,

– segmentation of MEG/EEG recordings in functional micro-states, using optical flow models [64].

17. Brainstorm in the Software Development Landscape

Several commercial solutions to visualize and process MEG/EEG data are available. Most are developed for specific



(a)

(b)

Figure 16: Cortical activations 46 ms after the electric stimulation of the left median nerve on the subject's brain (a) and their projection in the MNI brain (b).

acquisition systems and are often designed by the manufacturers of these systems. They are typically unsuitable for research for several reasons: they are mainly driven by the requirements of clinical environment and FDA and CE certifications; their all-graphical interface seldom provides information about the underlying data analysis, file formats, are sometimes proprietary and undocumented; source code and description of the algorithms are not accessible to the user, and they are expensive. The research community needs solutions that are completely open, with the possibility of directly manipulating the code, data, and parameters.

As a result, many laboratories have developed their own tools for MEG and EEG data analysis. However, these tools are often not shared either because of the lack of interest or because of the required effort to support the software, develop documentation, and create and maintain a distribution website. However, the approach of developing individual tools is very limiting because of the limited availability of human resources assigned to software development in most research groups and the breadth of expertise that is required

(electrophysiology, electromagnetic modeling, signal processing, statistics, classification, software optimization, real-time processing, human-machine interfaces ergonomics, etc.).

In the past two decades, many projects have been developed to offer open and free alternatives to the wide range of commercial solutions. Common among these projects is the support by a large community of developers around the world, who produce free and reusable source code. For this purpose, the free software community equipped itself with tools to facilitate collaborative work, such as version managers, forums, wikis, and discussion lists. This approach to collaborative software development has not only reached a high level of maturity, but also proved its efficiency. The best example is probably the Linux operating system, whose stability matches or exceeds that of commercially produced operating systems.

In the realm of functional brain mapping, open-source tools such as SPM [65] and EEGLab [35] have been broadly adopted in many research labs throughout the world. Providing open access to source code in combination with a willingness to accept additions and modifications from other sites clearly appeals both to users in clinical and neuroscientific research and others involved in methodology development. A variety of public licenses also allows developers to choose whether all or part of the code remains in the public domain. Importantly for software developed in academic and nonprofit labs, which are dependent on externally funded research support, recent experience indicates that open-source distribution is valued by the resesarch community and credit for this distribution is attributed to the original developers.

Free software packages with similar features to Brainstorm (general purpose software for MEG/EEG) are EEGLab, FieldTrip, and MNE. The first two are written under the Matlab environment, with noncompiled scripts, and are supported by large communities of users connected with active forums and diffusion lists. EEGLab offers a simple but functional interface, and its target application is oriented towards the preprocessing of recordings and ICA analysis. FieldTrip is a rich and powerful toolbox that offers the widest range of functionalities, but without a graphic interface; its usage requires good skills in Matlab programming. MNE is also organized as a set of independent functions, easily scriptable and mostly oriented towards the preprocessing of the recordings and the source estimation using minimum norm technique, but written in C++ and compiled for Linux and MacOSX platforms.

Brainstorm, in contrast, is an integrated application rather than a toolbox. At the present time, it offers fewer features than FieldTrip; but on the other hand, its intuitive interface, its powerful visualization tools, and the structure of its database allow the user to work at a higher level. It is possible to complete in a few minutes, and within a few mouse clicks, what would take hours otherwise: there is no need to write any scripts, and no need to think about where data files are stored on hard drives; the data is directly accessible, and a simple mouse click is sufficient to open a wide variety of display windows. It enables the researcher to concentrate on exploring his or her data. When visual exploration is complete and group analysis needs to be performed, Brainstorm offers a very high level scripting system, based on the interface and the database. The resulting code is easy to understand, and with few arguments: all the contextual information is gathered automatically from the database when needed, in contrast to FieldTrip, for example, where this information has to be specifically passed in arguments to each function.

To conclude, Brainstorm now represents a potentially highly-productive option for researchers using MEG or EEG; however, it is a work in progress and some key features are still missing. In the spirit of other open source developments, to the extent possible, we will reuse functions developed by other groups, which will then jointly maintain. Similarly, other developers are welcome to use code from Brainstorm in their software.

Acknowledgment

This software was generated primarily with support from the National Institutes of Health under Grants nos. R01-EB002010, R01-EB009048, and R01-EB000473. Primary support also includes permanent site support from the Centre National de la Recherche Scientifique (CNRS, France) for the Cognitive Neuroscience and Brain Imaging Laboratory (La Salpêtrière Hospital and Pierre and Marie Curie University, Paris, France). Additional support was provided by two grants from the French National Research Agency (ANR) to the Cognitive Neuroscience Unit (Inserm/CEA, Neurospin, France) and to the ViMAGINE project (ANR-08-BLAN-0250), and by the Epilepsy Center in the Cleveland Clinic Neurological Institute. The authors are grateful to all the people who contributed to the conception, the development, or the validation of specific Brainstorm functionalities. In alphabetical order: Charles Aissani, Syed Ashrafulla, Elizabeth Bock, Lucie Charles, Felix Darvas, Ghislaine Dehaene-Lambertz, Claude Delpuech, Belma Dogdas, Antoine Ducorps, Guillaume Dumas, John Ermer, Line Garnero, Alexandre Gramfort, Matti Hämäläinen, Louis Hovasse, Esen Kucukaltun-Yildirim, Etienne Labyt, Karim N'Diaye, Alexei Ossadtchi, Rey Ramirez, Denis Schwartz, Darren Weber, and Lydia Yahia-Cherif. The software, extensive documentation, tutorial data, user forum, and reference publications are available at http://neuroimage.usc.edu/brainstorm.

References

[1] S. Baillet, J. C. Mosher, and R. M. Leahy, "Electromagnetic brain mapping," *IEEE Signal Processing Magazine*, vol. 18, no. 6, pp. 14–30, 2001.

[2] M. X. Huang, J. C. Mosher, and R. M. Leahy, "A sensor-weighted overlapping-sphere head model and exhaustive head model comparison for MEG," *Physics in Medicine and Biology*, vol. 44, no. 2, pp. 423–440, 1999.

[3] F. Darvas, J. J. Ermer, J. C. Mosher, and R. M. Leahy, "Generic head models for atlas-based EEG source analysis," *Human Brain Mapping*, vol. 27, no. 2, pp. 129–143, 2006.

[4] J. C. Mosher, P. S. Lewis, and R. M. Leahy, "Multiple dipole modeling and localization from spatio-temporal MEG data," *IEEE Transactions on Biomedical Engineering*, vol. 39, no. 5, pp. 541–557, 1992.

[5] J. W. Phillips, R. M. Leahy, and J. C. Mosher, "MEG-Based imaging of focal neuronal current sources," *IEEE Transactions on Medical Imaging*, vol. 16, no. 3, pp. 338–348, 1997.

[6] S. Baillet and L. Garnero, "A Bayesian approach to introducing anatomo-functional priors in the EEG/MEG inverse problem," *IEEE Transactions on Biomedical Engineering*, vol. 44, no. 5, pp. 374–385, 1997.

[7] D. M. Schmidt, J. S. George, and C. C. Wood, "Bayesian inference applied to the electromagnetic inverse problem," *Human Brain Mapping*, vol. 7, no. 3, pp. 195–212, 1999.

[8] G. L. Barkley and C. Baumgartner, "MEG and EEG in epilepsy," *Journal of Clinical Neurophysiology*, vol. 20, no. 3, pp. 163–178, 2003.

[9] A. Arieli, A. Sterkin, A. Grinvald, and A. Aertsen, "Dynamics of ongoing activity: explanation of the large variability in evoked cortical responses," *Science*, vol. 273, no. 5283, pp. 1868–1871, 1996.

[10] C. Tallon-Baudry and O. Bertrand, "Oscillatory gamma activity in humans and its role in object representation," *Trends in Cognitive Sciences*, vol. 3, no. 4, pp. 151–162, 1999.

[11] G. Pfurtscheller and F. H. Lopes Da Silva, "Event-related EEG/MEG synchronization and desynchronization: basic principles," *Clinical Neurophysiology*, vol. 110, no. 11, pp. 1842–1857, 1999.

[12] P. Tass, M. G. Rosenblum, J. Weule et al., "Detection of n:m phase locking from noisy data: application to magnetoencephalography," *Physical Review Letters*, vol. 81, no. 15, pp. 3291–3294, 1998.

[13] C. W. J. Granger, B. N. Huangb, and C. W. Yang, "A bivariate causality between stock prices and exchange rates: evidence from recent Asianflu," *Quarterly Review of Economics and Finance*, vol. 40, no. 3, pp. 337–354, 2000.

[14] W. Hesse, E. Möller, M. Arnold, and B. Schack, "The use of time-variant EEG Granger causality for inspecting directed interdependencies of neural assemblies," *Journal of Neuroscience Methods*, vol. 124, no. 1, pp. 27–44, 2003.

[15] H. B. Hui, D. Pantazis, S. L. Bressler, and R. M. Leahy, "Identifying true cortical interactions in MEG using the nulling beamformer," *NeuroImage*, vol. 49, no. 4, pp. 3161–3174, 2010.

[16] J. L. P. Soto, D. Pantazis, K. Jerbi, S. Baillet, and R. M. Leahy, "Canonical correlation analysis applied to functional connectivity in MEG," in *Proceedings of the 7th IEEE International Symposium on Biomedical Imaging: From Nano to Macro (ISBI '10)*, pp. 113–116, April 2010.

[17] D. Pantazis, T. E. Nichols, S. Baillet, and R. M. Leahy, "A comparison of random field theory and permutation methods for the statistical analysis of MEG data," *NeuroImage*, vol. 25, no. 2, pp. 383–394, 2005.

[18] S. Baillet, J. C. Mosher, and R. M. Leahy, "BrainStorm beta release: a Matlab software package for MEG signal processing and source localization and visualization," *NeuroImage*, vol. 11, no. 5, p. S915, 2000.

[19] J. C. Mosher, S. Baillet, F. Darvas et al., "Electromagnetic brain imaging using brainstorm," vol. 7, no. 2, pp. 189–190.

[20] A. Tzelepi, N. Laskaris, A. Amditis, and Z. Kapoula, "Cortical activity preceding vertical saccades: a MEG study," *Brain Research*, vol. 1321, pp. 105–116, 2010.

[21] F. Amor, S. Baillet, V. Navarro, C. Adam, J. Martinerie, and M. Le Van Quyen, "Cortical local and long-range synchronization interplay in human absence seizure initiation," *NeuroImage*, vol. 45, no. 3, pp. 950–962, 2009.

[22] T. A. Bekinschtein, S. Dehaene, B. Rohaut, F. Tadel, L. Cohen, and L. Naccache, "Neural signature of the conscious processing of auditory regularities," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 106, no. 5, pp. 1672–1677, 2009.

[23] F. Carota, A. Posada, S. Harquel, C. Delpuech, O. Bertrand, and A. Sirigu, "Neural dynamics of the intention to speak," *Cerebral Cortex*, vol. 20, no. 8, pp. 1891–1897, 2010.

[24] M. Chaumon, D. Hasboun, M. Baulac, C. Adam, and C. Tallon-Baudry, "Unconscious contextual memory affects early responses in the anterior temporal lobe," *Brain Research*, vol. 1285, pp. 77–87, 2009.

[25] S. Moratti and A. Keil, "Not what you expect: experience but not expectancy predicts conditioned responses in human visual and supplementary cortex," *Cerebral Cortex*, vol. 19, no. 12, pp. 2803–2809, 2009.

[26] D. Pantazis, G. V. Simpson, D. L. Weber, C. L. Dale, T. E. Nichols, and R. M. Leahy, "A novel ANCOVA design for analysis of MEG data with application to a visual attention study," *NeuroImage*, vol. 44, no. 1, pp. 164–174, 2009.

[27] P. Hansen, M. Kringelbach, and R. Salmelin, Eds., *Meg: An Introduction to Methods*, Oxford University Press, Oxford, UK, 2010.

[28] R. Salmelin and S. Baillet, "Electromagnetic brain imaging," *Human Brain Mapping*, vol. 30, no. 6, pp. 1753–1757, 2009.

[29] The MoinMoin Wiki Engine, http://moinmo.in/.

[30] vBulleti, http://www.vbulletin.com/.

[31] D. W. Shattuck and R. M. Leahy, "Brainsuite: an automated cortical surface identification tool," *Medical Image Analysis*, vol. 8, no. 2, pp. 129–142, 2002.

[32] Y. Cointepas, J.-F. Mangin, L. Garnero, J.-B. Poline, and H. Benali, "BrainVISA: software platform for visualization and analysis of multi-modality brain data," *Neuroimage*, vol. 13, no. 6, p. S98, 2001.

[33] FreeSurfer, http://surfer.nmr.mgh.harvard.edu/.

[34] FieldTrip, Donders Institute for Brain, Cognition and Behaviour, http://fieldtrip.fcdonders.nl/.

[35] A. Delorme and S. Makeig, "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis," *Journal of Neuroscience Methods*, vol. 134, no. 1, pp. 9–21, 2004.

[36] M. S. Hämäläinen, MNE software, http://www.nmr.mgh.harvard.edu/martinos/userInfo/data/sofMNE.php.

[37] D. L. Collins, A. P. Zijdenbos, V. Kollokian et al., "Design and construction of a realistic digital brain phantom," *IEEE Transactions on Medical Imaging*, vol. 17, no. 3, pp. 463–468, 1998.

[38] R. M. Leahy, J. C. Mosher, M. E. Spencer, M. X. Huang, and J. D. Lewine, "A study of dipole localization accuracy for MEG and EEG using a human skull phantom," *Electroencephalography and Clinical Neurophysiology*, vol. 107, no. 2, pp. 159–173, 1998.

[39] F. Darvas, D. Pantazis, E. Kucukaltun-Yildirim, and R. M. Leahy, "Mapping human brain function with MEG and EEG: methods and validation," *NeuroImage*, vol. 23, no. 1, pp. S289–S299, 2004.

[40] P. Berg and M. Scherg, "A fast method for forward computation of multiple-shell spherical head models," *Electroencephalography and Clinical Neurophysiology*, vol. 90, no. 1, pp. 58–64, 1994.

[41] A. Gramfort, T. Papadopoulo, E. Olivi, and M. Clerc, "OpenMEEG: opensource software for quasistatic bioelectromagnetics," *BioMedical Engineering Online*, vol. 9, article 45, 2010.

[42] M. S. Hämäläinen and R. J. Ilmoniemi, "Interpreting magnetic fields of the brain: minimum norm estimates," *Medical and Biological Engineering and Computing*, vol. 32, no. 1, pp. 35–42, 1994.

[43] A. M. Dale, A. K. Liu, B. R. Fischl et al., "Dynamic statistical parametric mapping: combining fMRI and MEG for high-resolution imaging of cortical activity," *Neuron*, vol. 26, no. 1, pp. 55–67, 2000.

[44] R. D. Pascual-Marqui, "Standardized low-resolution brain electromagnetic tomography (sLORETA): technical details," *Methods and Findings in Experimental and Clinical Pharmacology*, vol. 24, no. D, pp. 5–12, 2002.

[45] B. D. Van Veen and K. M. Buckley, "Beamforming: a versatile approach to spatial filtering," *IEEE ASSP Magazine*, vol. 5, no. 2, pp. 4–24, 1988.

[46] R. O. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Transactions on Antennas and Propagation*, vol. 34, no. 3, pp. 276–280, 1986, Reprint of the original 1979 paper from the RADC Spectrum Estimation Workshop.

[47] N. Tzourio-Mazoyer, B. Landeau, D. Papathanassiou et al., "Automated anatomical labeling of activations in SPM using a macroscopic anatomical parcellation of the MNI MRI single-subject brain," *NeuroImage*, vol. 15, no. 1, pp. 273–289, 2002.

[48] C. Tallon-Baudry, O. Bertrand, C. Wienbruch, B. Ross, and C. Pantev, "Combined EEG and MEG recordings of visual 40 Hz responses to illusory triangles in human," *NeuroReport*, vol. 8, no. 5, pp. 1103–1107, 1997.

[49] M. S. Worden, J. J. Foxe, N. Wang, and G. V. Simpson, "Anticipatory biasing of visuospatial attention indexed by retinotopically specific alpha-band electroencephalography increases over occipital cortex," *The Journal of Neuroscience*, vol. 20, no. 6, p. RC63, 2000.

[50] W. Klimesch, M. Doppelmayr, T. Pachinger, and B. Ripper, "Brain oscillations and human memory: EEG correlates in the upper alpha and theta band," *Neuroscience Letters*, vol. 238, no. 1-2, pp. 9–12, 1997.

[51] O. Jensen and C. D. Tesche, "Frontal theta activity in humans increases with memory load in a working memory task," *European Journal of Neuroscience*, vol. 15, no. 8, pp. 1395–1399, 2002.

[52] W. Klimesch, M. Doppelmayr, H. Russegger, T. Pachinger, and J. Schwaiger, "Induced alpha band power changes in the human EEG and attention," *Neuroscience Letters*, vol. 244, no. 2, pp. 73–76, 1998.

[53] C. S. Herrmann, M. Grigutsch, and N. A. Busch, "EEG oscillations and wavelet analysis," in *Event-Related Potentials-A Methods Handbook*, pp. 229–259, MIT Press, Cambridge, Mass, USA, 2005.

[54] C. E. McCulloch and S. R. Searle, *Generalized, Linear, and Mixed Model*, John Wiley & Sons, New York, NY, USA, 2001.

[55] W. D. Penny, A. P. Holmes, and K. J. Friston, "Random effects analysis," *Human Brain Function*, vol. 2, pp. 843–850, 2004.

[56] D. Pantazis and R. M. Leahy, "Meg: an introduction to methods," in *Statistical Inference in MEG Distributed Source Imaging*, chapter 10, Oxford University Press, Oxford, UK, 2010.

[57] J. A. Mumford and T. Nichols, "Modeling and inference of multisubject fMRI data," *IEEE Engineering in Medicine and Biology Magazine*, vol. 25, no. 2, pp. 42–51, 2006.

[58] Y. Benjamini and Y. Hochberg, "Controlling the false discovery rate: a practical and powerful approach to multiple testing," *Journal of the Royal Statistical Society Series B*, vol. 57, no. 1, pp. 289–300, 1995.

[59] T. Nichols and S. Hayasaka, "Controlling the familywise error rate in functional neuroimaging: a comparative review," *Statistical Methods in Medical Research*, vol. 12, no. 5, pp. 419–446, 2003.

[60] D. J. Kroon, "Iterative Closest Point using finite difference optimization to register 3D point clouds affine," http://www.mathworks.com/matlabcentral/fileexchange/24301-finite-iterative-closest-point.

[61] D. Shepard, "Two- dimensional interpolation function for irregularly- spaced data," in *Proceedings of the ACM National Conference*, pp. 517–524, 1968.

[62] A. A. Joshi, D. W. Shattuck, P. M. Thompson, and R. M. Leahy, "Surface-constrained volumetric brain registration using harmonic mappings," *IEEE Transactions on Medical Imaging*, vol. 26, no. 12, pp. 1657–1668, 2007.

[63] J. L. P. Soto, D. Pantazis, K. Jerbi, J. P. Lachaux, L. Garnero, and R. M. Leahy, "Detection of event-related modulations of oscillatory brain activity with multivariate statistical analysis of MEG data," *Human Brain Mapping*, vol. 30, no. 6, pp. 1922–1934, 2009.

[64] J. Lefèvre and S. Baillet, "Optical flow approaches to the identification of brain dynamics," *Human Brain Mapping*, vol. 30, no. 6, pp. 1887–1897, 2009.

[65] SPM, http://www.fil.ion.ucl.ac.uk/spm/.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 813870, 15 pages
doi:10.1155/2011/813870

Review Article

Spatiotemporal Analysis of Multichannel EEG: CARTOOL

Denis Brunet,[1,2] Micah M. Murray,[2,3] and Christoph M. Michel[1,2]

[1] *Functional Brain Mapping Laboratory, Departments of Fundamental and Clinical Neurosciences, University Medical School, University of Geneva, 1 rue Michel-Servet, 1211 Geneva, Switzerland*
[2] *EEG Brain Mapping Core, Center for Biomedical Imaging (CIBM), 1211 Geneva, Switzerland*
[3] *The Functional Electrical Neuroimaging Laboratory, Department of Clinical Neurosciences and Department of Radiology, Vaudois University Hospital Center, University of Lausanne, 1011 Lausanne, Switzerland*

Correspondence should be addressed to Denis Brunet, denis.brunet@unige.ch

Received 7 September 2010; Accepted 10 November 2010

Academic Editor: Sylvain Baillet

This paper describes methods to analyze the brain's electric fields recorded with multichannel Electroencephalogram (EEG) and demonstrates their implementation in the software CARTOOL. It focuses on the analysis of the spatial properties of these fields and on quantitative assessment of changes of field topographies across time, experimental conditions, or populations. Topographic analyses are advantageous because they are reference independents and thus render statistically unambiguous results. Neurophysiologically, differences in topography directly indicate changes in the configuration of the active neuronal sources in the brain. We describe global measures of field strength and field similarities, temporal segmentation based on topographic variations, topographic analysis in the frequency domain, topographic statistical analysis, and source imaging based on distributed inverse solutions. All analysis methods are implemented in a freely available academic software package called CARTOOL. Besides providing these analysis tools, CARTOOL is particularly designed to visualize the data and the analysis results using 3-dimensional display routines that allow rapid manipulation and animation of 3D images. CARTOOL therefore is a helpful tool for researchers as well as for clinicians to interpret multichannel EEG and evoked potentials in a global, comprehensive, and unambiguous way.

1. Introduction

The traditional analysis of the electroencephalogram (EEG) and event-related potentials (ERPs) focuses on waveform morphology over time at certain electrode positions. Scalp sites of interest are selected and the time course of the potential recorded at any site is analyzed using a variety of signal processing tools in the time and frequency domains. While this approach has provided many important insights into normal and pathological neuronal activity, it disregards another important dimension that multichannel EEG offers: the spatial characteristics of the electric fields at the scalp and the temporal dynamics of these fields. Any distribution and orientation of the active neurons at a given moment in time will generate a certain electric field on the scalp surface due to volume conduction [1]. While different generator configurations can lead to the same scalp fields, the inverse is not true: different scalp fields must have been generated by different configurations of generators in the brain [2, 3]. Consequently, analyzing the electric field topography, that is, the configuration of the potential isocontour maps on the scalp, and looking for topographical differences allows to detect moments when different neuronal populations were active in the brain, being it in time, between experimental conditions or under given pathological circumstances. Besides this direct neurophysiological interpretability, the analysis of the topography of the electric fields has another important advantage as compared to the analysis of waveforms: it is completely reference independent. The recording reference does not influence the topography of the scalp electric field and thus does not influence global topographic measures [4–6]. This is not true for methods that analyze the EEG or evoked potential waveforms. There is no point that records zero potential over time [4]. Consequently, changing the reference electrode changes the waveform shapes at each recording electrode. Therefore, any statistical comparison of amplitudes at a given electrode between conditions will change when the reference has changed, making the results

ambiguous [7–9]. By contrast, the map topography does not change when changing the reference. Only the zero line is shifted, but not the landscape of the potential map [3]. Consequently, all analysis methods based on map topography are reference independent and unambiguous. This important argument for the spatial analysis of the EEG is illustrated in Figure 1.

With the program CARTOOL, which exists now since over 14 years with constantly increasing capabilities, we wanted to provide an analysis tool for those researchers and clinicians who are interested in such reference-free EEG mapping techniques. It started with dynamic displays of EEG maps and calculation of some basic quantitative topographic measures. Soon after, standard data preprocessing tools were implemented, such as interpolation of electrodes, filtering, averaging, rereferencing. The next versions implemented spatial analysis methods for spontaneous and event-related EEG, most importantly the spatial microstate segmentation, initially proposed by Lehmann and collaborators [11], and subsequently advanced by cluster analysis and fitting methods. Also, other statistical topographic analysis methods were implemented. With the development of distributed inverse solutions and the advancement of computer power, source estimations in realistic head models were integrated. Important aspects of the software are the 3-dimensional visualization of the data as well as the fast display of the temporal dynamics of the scalp electric fields and the corresponding estimated sources. For that, the software puts particular emphasis on interactive manipulation and synchronization of the different windows by the user, using mouse and keyboard commands.

In the following we will describe some of the main methods for the spatial analysis of the scalp electric fields and how they are implemented in CARTOOL. Not all methods will be covered here, but they will give an impression of how multichannel EEG and ERP data can be analyzed in a comprehensive way. More details regarding the different spatial analysis methods can be found in the book "Electrical Neuroimaging" [12].

2. Data Preprocessing

Topographic analysis of the electric field at the scalp very crucially depends on the quality of the data at each channel. Contamination by bad or noisy electrodes can lead to steep local gradients that have no neurophysiologic basis, which can in turn obfuscate interpretability of results (particularly those of source localization). Artifacts on one particular channel are not always easy to detect if only EEG waveforms are displayed. In contrast, contaminated channels are readily seen on time series of EEG maps because they behave differently from the neighboring channels and appear as isolated "spots" in the maps (Figure 2; see also [13]).

CARTOOL provides various options to display EEG maps in 2D and 3D and with dynamic color scaling (Figure 3). Maps can be displayed in series over time or as animated movies. Electrode positions and electrode names can be displayed and marked both on the maps and on the waveforms. After clicking on bad channels, an interpolation tool is available that interpolates these channels using any of several different types of interpolation methods (surface spline, spherical spline or 3D spline) [14]. The 3D spline interpolation accounts for the real geometry of the head and is recommended if the real position of the electrodes is available. The interpolation toolbox also allows transforming the individual data of different subjects with different electrode positions to a common coordinate system for further statistical processing across subjects.

Once the data are cleaned from bad electrodes and interpolated, several standard preprocessing tools are available in CARTOOL, such as:

(i) filtering using 2nd-order Butterworth filters with – 12 dB/octave roll-off,

(ii) DC removal and Notch filter, envelope filter by rectifying absolute or squared values.

(iii) downsampling by a Cascaded Integrator Comb filter followed by a high-pass FIR and decimation,

(iv) recalculation against any type of single or combined electrode reference including current density (i.e., 2nd spatial derivative),

(v) exporting single or multiple tracks before or after applying the above preprocessing steps.

For evoked potential analysis, single or averaged epochs can be calculated for any combination of triggers or markers with or without baseline correction. Automatic artifact detection and epoch rejection using amplitude windows is available. CARTOOL puts a lot of emphasis in a flexible visualization of the EEG tracks during evoked potential analysis for manual determination of artifacts. During the evoked potential analysis, a trigger validation file is generated that can later be used to more rapidly redo the averaging with the same epochs.

In addition to the epoching according to defined triggers, CARTOOL allows to set markers according to specific characteristics in certain channels. This allows, for example, one to set markers at the onset of a motor response recorded with the EMG or at the peak of epileptic spikes.

Finally, a file calculator tool has been implemented in CARTOOL that allows applying preset as well as user-defined mathematical operations to different files in a batch-mode processing.

3. Global Topographic Measures

In the default display mode, CARTOOL always shows two global topographic measures together with the waveforms and the maps (Figure 3). These two global measures are the Global Field Power and the Global Map Dissimilarity [5]. They are considered as additional measures and can be treated in the same way as the different tracks of the electrodes. The Global Field Power (GFP) is the standard deviation of the potentials at all electrodes of an average-reference map. It is defined as

$$\mathrm{GFP} = \sqrt{\frac{\sum_{i=1}^{N} (u_i - \overline{u})^2}{N}}, \tag{1}$$

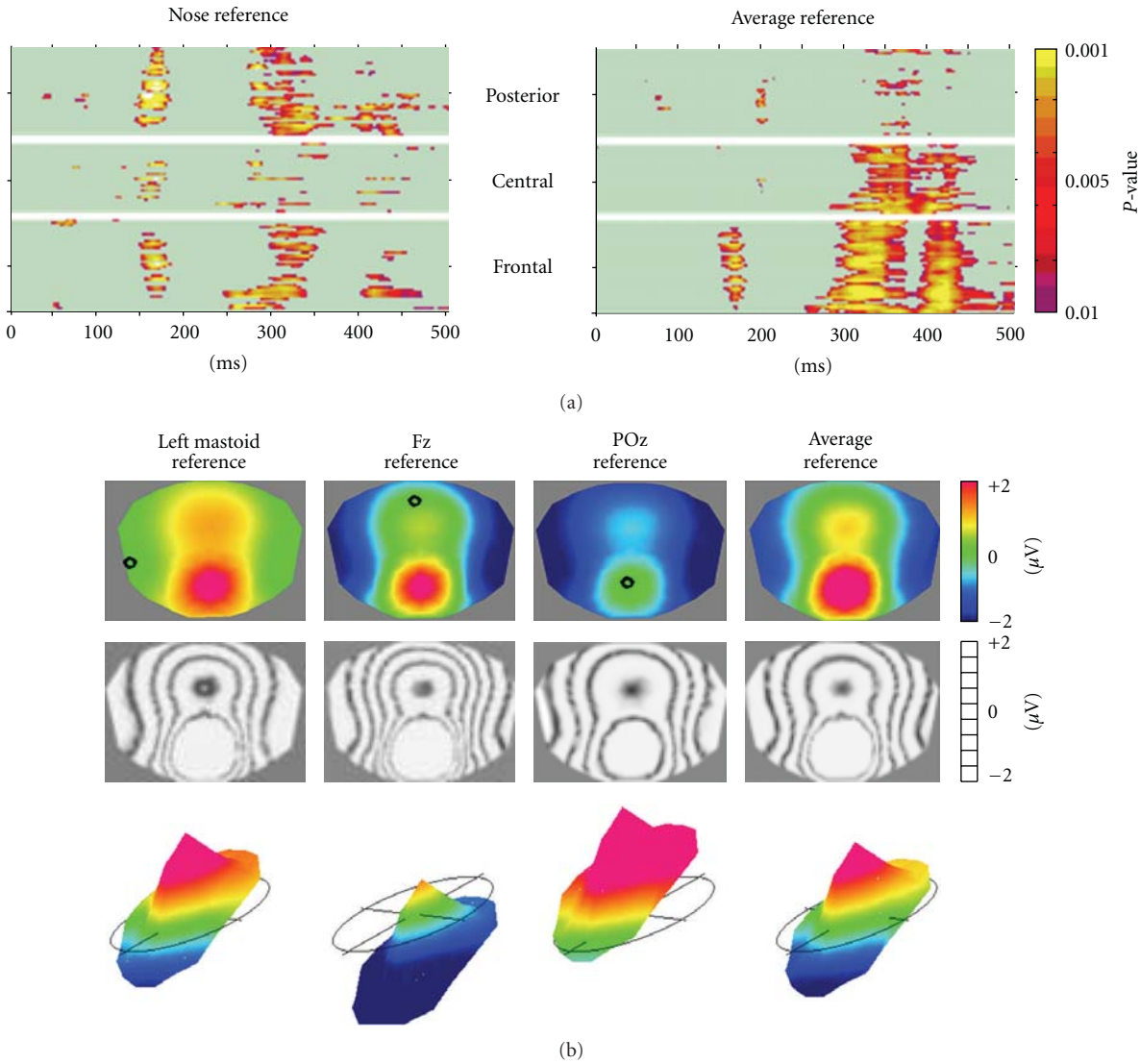


Figure 1: The reference issue in EEG: EEG waveform analysis is reference-dependent while topographical analysis is not. (a) Statistical comparison of evoked potential waveforms of two conditions (illusory contours versus no contours from [10]) for each electrode and each time point. Left: All electrodes referenced to the nose, right: all electrodes referenced to the Average Reference. Note the change of the results in time as well as in space. (b) Scalp potential map (seen from top) referenced to different electrodes. Top: color maps. Middle: same maps displayed with isopotential lines. Bottom: same maps displayed in relief with the zero level indicated. Note that the topography of the maps does not change, only the zero line and thus the color codes change.

where u_i is the voltage of the map u at the electrode i, $\overline{u}$ is the average voltage of all electrodes of the map u and N is the number of electrodes of the map u. Scalp potential fields with pronounced peaks and troughs and steep gradients will result in high GFP, while GFP is low in maps which have a "flat" appearance with shallow gradients. GFP is a one-number measure of the map at each moment in time. Displaying this measure over time allows to identify moments of high signal-to-noise ratio, corresponding to moments of high global neuronal synchronization [15, 16]. GFP can also be used to normalize data across subjects by dividing each map (i.e., the voltage at each electrode) by the mean GFP over time. General intra individual differences of the surface potential, for example due to difference in skull conductivity, can thereby be adjusted.

The Global Map Dissimilarity measure (GMD) is a measure of topographic differences of scalp potential maps. It is defined as

$$\mathrm{GMD} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left\{\frac{u_i-\overline{u}}{\sqrt{\sum_{i=1}^{N}(u_i-\overline{u})^2/N}}-\frac{v_i-\overline{v}}{\sqrt{\sum_{i=1}^{N}(v_i-\overline{v})^2/N}}\right\}^2}, \quad (2)$$

where u_i is the voltage of map u at the electrode i, v_i is the voltage of map v at the electrode i, $\overline{u}$ is the average voltage

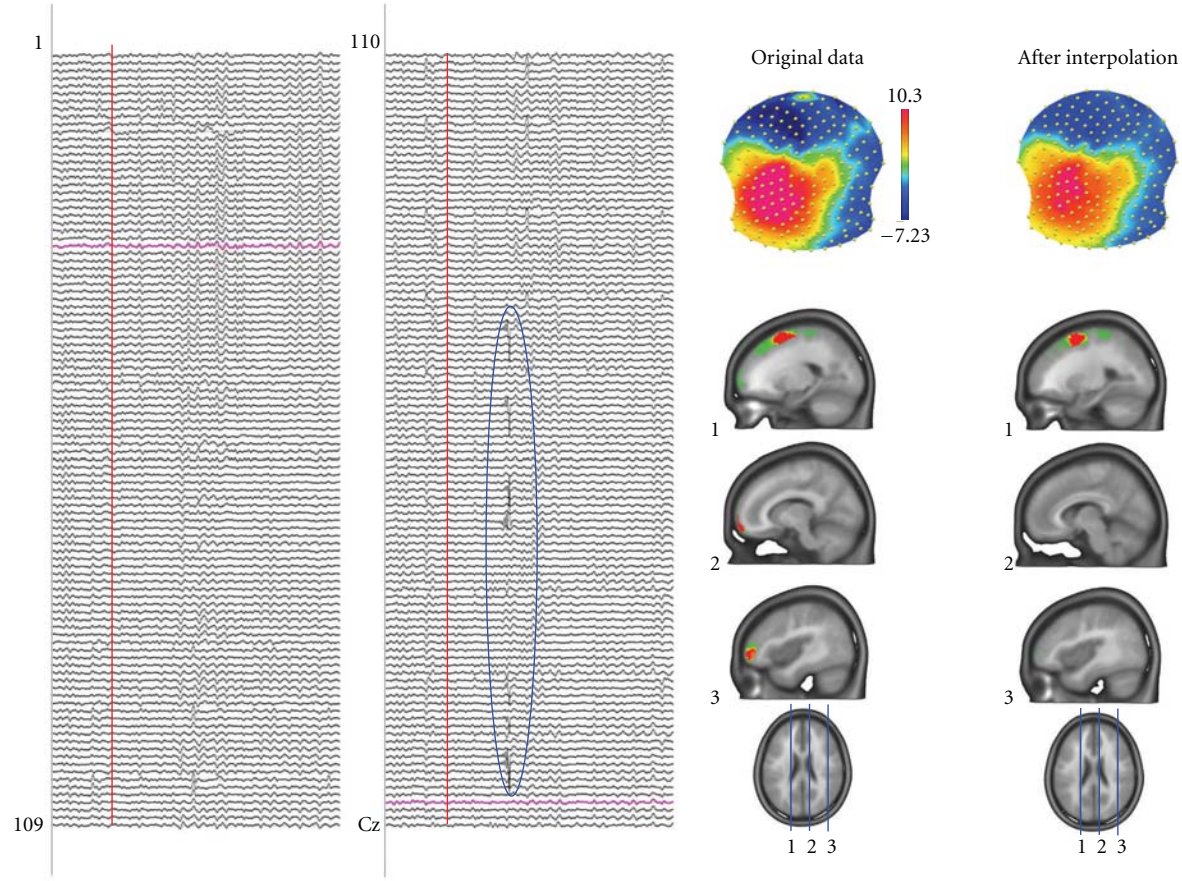


Figure 2: Artifact detection by inspecting the potential maps. The left panel shows spontaneous EEG recorded from 204 electrodes. Some artifacts, like the one encircled, are easy to see in the traces and such epochs can be eliminated. However, other artifacts are not easy to see in the traces but are readily detected in the maps by isolated "islands" of potential of a certain electrode. In this example a mid-frontal and a right frontal electrode are artifact contaminated. They generate steep gradients in the electric field and consequently produce strong sources in the inverse solution (here LAURA). Interpolating these electrodes using spline interpolation eliminates the bad electrodes and the sources caused by these artifacts disappear.

of all electrodes of map u, $\bar{v}$ is the average voltage of all electrodes of map v, and N is the total number of electrodes. In order to assure that only topography differences are taken into account, the two maps that are compared are first normalized by dividing the potential values at each electrode of a given map by its GFP. The GMD is 0 when two maps are equal, and maximally reaches 2 for the case where the two maps have the same topography with reversed polarity. Figure 2 in [17] illustrates the definition of the GMD.

The GMD is equivalent to the spatial Pearson's product-moment correlation coefficient between the potentials of the two maps to compare [18]. The calculation of the GMD is a first step for defining whether different sources are involved in generating the electrical activity at the scalp for the two processes/populations being compared. If two maps differ in topography independently of their strength, it directly indicates that the two maps were generated by a different configuration of sources in the brain. As will be described later, all statistical topographic analysis methods in CARTOOL that compare topographies between conditions or groups use GMD (or the spatial correlation) as the basic measure of map similarity. GMD can also be used to compare topographies between successive time points. The display of the GMD across time then allows defining periods of map stability and moments of map changes. It is generally observed (particularly in evoked potentials) that GMD is inversely correlated with the GFP: GMD is high when GFP is low [19]. This observation indicates that maps tend to remain rather stable in topography during high GFP and change the configuration when GFP is low.

4. Microstate Segmentation

The display of the GMD across time has a very characteristic behavior which is similar for spontaneous EEG and for evoked potentials: the topography of the maps remains stable for several tens of milliseconds and then abruptly switches to a new configuration in which it remains stable again. This leads to periods of low GMD interrupted by sharp GMD peaks (Figure 3). This highly reproducible observation of periods of stable map topography has led to the concept of *functional microstates* first described by

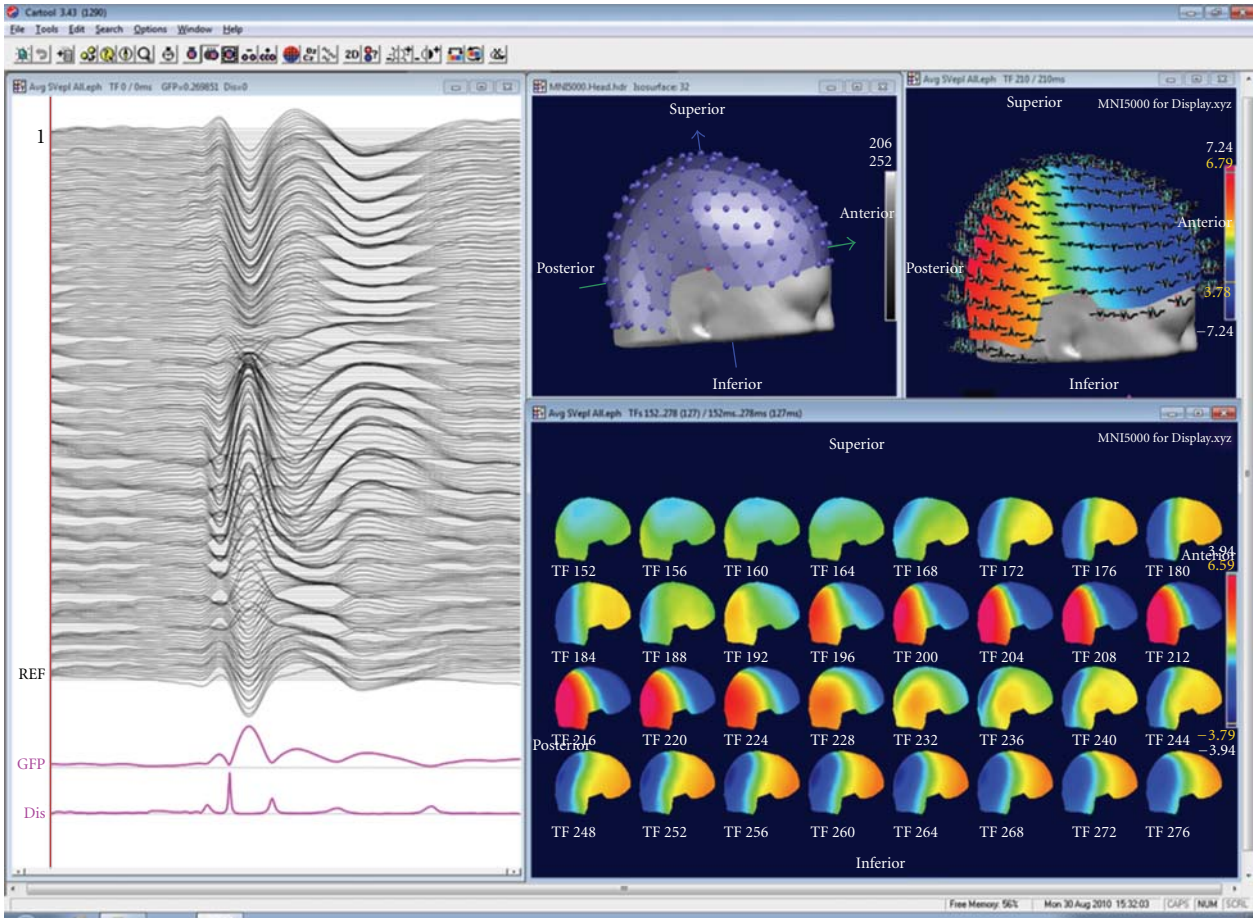


Figure 3: Example of a basic display window in CARTOOL. The individual tracks are displayed together with the Global Field Power and the Global Map Dissimilarity on the left. Tracks can also be displayed in 3D on the head surface as shown on the top right. Electrode positions and maps can be displayed in 3D or 2D. Maps can be shown at a single time point at cursor position, as animation over time or as time series of maps within a selected window.

Lehmann et al. [11, 20]. The microstates correspond to a period of coherent synchronized activation of a large-scale neuronal network. Lehmann et al. proposed that the functional microstates represent the basic building blocks of information processing, the "atoms of thought", being it spontaneous or evoked by a stimulus [21]. This corresponds well to the proposal that neurocognitive networks evolve through a sequence of quasistable coordination states, rather than a continuous flow of neuronal activity [22–25]. With respect to the ERPs, each successive microstate represents a certain information processing step that leads from perception to action [26]. While several parallel activations are possible and are most likely occurring in each step, there nonetheless seems to be a certain sequence of information processing, probably related to the integration of the information at different complexity levels [27].

In light of this interpretation of the observed sequential periods of map stability, different methods have been proposed to objectively and automatically define the different microstates and to statistically evaluate the specificity of certain microstates under given experimental conditions. CARTOOL has implemented these methods in the "microstate segmentation" and "map fitting" modules. The microstate segmentation is based on cluster analysis using either a modified k-means cluster analysis [28] or an atomize and agglomerate hierarchical cluster analysis with or without GFP normalization [17], followed by some temporal post-processing steps. The k-means cluster analysis is a classical pattern recognition method used in many applications in different fields. It is an iterative procedure, starting with an initial guess of maps and terminating when successive iterations differ negligibly. Because of these iterations the result of the k-means cluster analysis can slightly vary from one run to the other. In contrast, the hierarchical cluster analysis that we devised specifically for microstate segmentation does not iterate and thus gives unique results. It is a modified agglomerative hierarchical clustering in a way that clusters that greatly contribute to the global explained variance are retained even if they are present for a short period of time only. More detailed explanation of the two methods can be found in [9].

The cluster analysis can be applied to one data file or to different files of different experimental conditions and/or populations (Figure 4). The result is a certain number

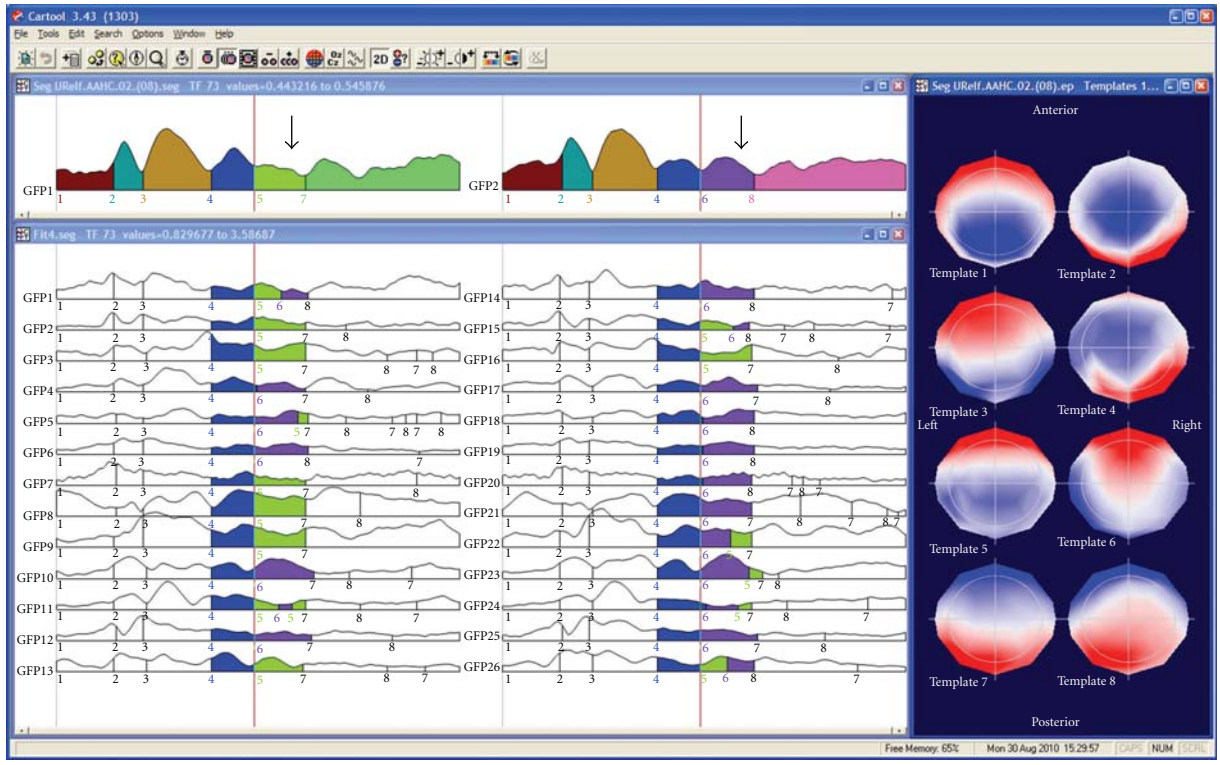


Figure 4: Illustration of the microstate segmentation in CARTOOL. The two windows on the top show the segments resulting from the k-means cluster analysis of the grand-mean ERP of two conditions. The segments are marked under the Global Field Power curves. Different colors indicate different segments. The cluster maps of these segments are displayed on the right. Note that in the beginning the same segments are found for the two conditions, while different segments explain the later components. Fitting the cluster maps to each single subject ERP statistically tests this finding. This is illustrated here by showing that more subjects have map number 5 (green) in condition 1 and map number 6 (purple) in condition 2. Duration, explained variance and other parameters are computed for each segment and can then be statistically compared using CARTOOL or any other statistical package.

of prototype maps (also called cluster maps) that best represent the whole data set. For defining the optimal number of cluster maps, CARTOOL proposes two criteria: a cross-validation criterion and the Krzanovski-Lai criterion [17]. The cross-validation is derived by dividing the global explained variance by the degrees of freedom, the latter depending on the number of electrodes. The Krzanovski-Lai criterion is determined by the L-corner of the dispersion curve, which is a quality measure of the clustering, meaning the optimal clustering is set when an additional cluster does not lead to a significant gain of the global quality (for details see [9]).

The cluster maps are finally fitted back to the original data and each time point is labeled with the cluster maps it correlated best with (in terms of GMD). In order to ensure a certain degree of continuity in time of the different a final relabeling step is performed which satisfies two requirements: (1) the correlation between the measurement and the cluster map should be high, and (2) the majority of the neighboring measurements should belong to the same microstate. Standard smoothing techniques, well-known in statistics, are used to fulfill this compromise between goodness of fit and smoothness [28]. CARTOOL allows adjusting these smoothness parameters. In addition, small segments can optionally be rejected. The result of the microstate segmentation is displayed color-coded under the GFP curve with each color representing a different cluster map. Additional options are available to sequentialize clusters and to merge highly correlated clusters.

Another method that has been proposed to define the most dominant evoked component topographies in a dataset is based on an independent component analysis (ICA, [29]). It has been shown that both ICA and cluster analysis lead to rather similar results and thus have compatible underlying assumptions [30, 31]. However, the main limitation of ICA is that it assumes that global brain activity is generated by a superimposition of a number of independent processes. While this assumption might be valid in the case of artifacts such as eye movements or cardiac activity, it is difficult to accept for brain activity, where the principal organization relies on distributed neural networks with tightly linked cross-talk between the different areas. In such systems, the different components are dynamically coupled and cannot be separated in independent components. ICA would fail to uncover such processes, while the cluster analysis does not require such independence.

The cluster analysis of ERPs is usually applied to the group-averaged files. All experimental conditions/populations are entered into the cluster analysis, and the optimal number of clusters for the whole data set is determined [27, 32]. The cluster maps are then fitted to the data by calculating the spatial correlation (the GMD, see above) between each cluster map and each time point of the data. Each time point is then labeled with the cluster map with which it correlates best. It is interesting to note that this fitting procedure results in stable periods that are represented by the same cluster map even though no temporal constraint is a priori imposed, thus directly and empirically confirming the microstate model. The labeling procedure and the display of the results of this labeling as color-coded segments under the GFP curves allows the experimenter to generate hypotheses about the specificity of certain microstate maps for certain experimental conditions/populations (Figure 4).

It is important to emphasize that the microstate segmentation on the grand mean data allows hypothesis generation and is not the final result. Changing the number of clusters might change the results at this level by proposing more or less map differences across time or between conditions. A second statistical step is needed to confirm these hypotheses and define those microstates that remain statistically significant. The "microstate fitting" module of CARTOOL allows to perform this test. The fitting procedure is the same as for the grand mean, but now the cluster maps are fitted to the individual ERPs of each subject and each condition/population [32] (Figure 4). Several different parameters are then computed that describe the goodness of fit, the number of maps that each cluster explained, the onset and the offset of each cluster map, and so forth [17]. Spreadsheets are generated with these values that can directly be read into any statistical software package as well as into Excel, but that can also directly be used in the statistical analysis module in CARTOOL. Only microstates that are significantly different after this statistical fitting procedure are considered as stable. Users will realize that in most cases increasing the number of clusters beyond the one proposed by the cross-validation or other optimization criterion will not lead to new microstates that survive the statistical tests.

The microstate segmentation using the cluster analysis can also be applied to spontaneous EEG. It leads to a reduction of the data to a stream of microstates of certain durations, on average around 80–100 msec [33]. It is important to note that in the spontaneous EEG polarity inversion caused by the intrinsic oscillatory activity of the generator processes is ignored. Numerous studies in healthy subjects as well as in patients with different pathologies have shown that a very limited number of map topographies are needed to explain extended periods of spontaneous EEG, and that these few configurations follow each other according to certain rules [34]. We have shown that these different microstates are correlated with well-known fMRI resting states [35]. Analysis of the temporal structure of the microstate transitions showed that the microstates have fractal properties, that is, that their temporal structure is scale invariant over a large time scale [36].

5. Statistical Analysis Using CARTOOL

CARTOOL offers a variety of parametric and non-parametric statistical EEG mapping analysis procedures (Figure 5). Non-parametric tests are based on Monte-Carlo bootstrapping methods, while the parametric tests use paired or non-paired t-tests. At present, only univariate statistics are implemented in CARTOOL, but multivariate analysis procedures are currently under evaluation before formal inclusion. The univariate analysis can be applied to the potential at each electrode and each time point as a comprehensive exploratory analysis of the data [10, 37, 38]. It is important to note, however, that this analysis is reference-dependent and does not tell us whether topographic or amplitude differences underlie the observed effects (Figure 1). Also the problem for corrections of multiple testing is ill posed. CARTOOL offers Bonferroni corrections as well as the application of the restriction that effects last a certain minimal duration [39]. In order to separately assess strength and topographic differences, CARTOOL proposes statistical analyses using the global variables described above, that is, the GFP and the GMD.

Testing for differences in GFP at each time point is straightforward using the parametric or nonparametric tests. In order to test for differences in topography, CARTOOL implemented what has been called a "topographic ANOVA", or TANOVA [17, 40]. Since GMD is a single measure of difference between the maps of two conditions, mean and standard error of topography for each condition/population cannot be calculated. The way to overcome this problem is to perform a non-parametric randomization test based on the GMD values. This is done in the following way: (1) assigning the maps of the single subject in a randomized fashion to different experimental conditions, (2) recalculating the group-average ERPs, and (3) recalculating the resulting GMD value for these "new" group-average ERPs. The number of permutations that can be made with a group-average ERP based on n participants is 2^n. The GMD value from the actual group-average ERPs is then compared with the values from the empirical distribution to determine the likelihood that the empirical distribution has a value higher than the GMD from the actual group-average ERPs. In a within-subject design, the permutation of the maps is done within the subjects, while the permutation is done across subjects in group comparisons. In the example shown in Figure 5, the TANOVA is performed between two conditions for each time point. The figure illustrates the simplicity of the analysis of the global parameters GFP and GMD compared to the electrode-wise statistics (besides the fact that the latter are reference independent). However, it is important to note here that a latency shift of one condition with respect to the other could lead to strong and long topographic differences that are in fact not due to different areas being activated by the two conditions, but by activations of the same areas at different moments in time. The comparison of the TANOVA result with the microstate segmentation is therefore important.

The statistical analysis can also be performed on the values that result from the microstate fitting procedure. Again, only univariate statistics are currently implemented.

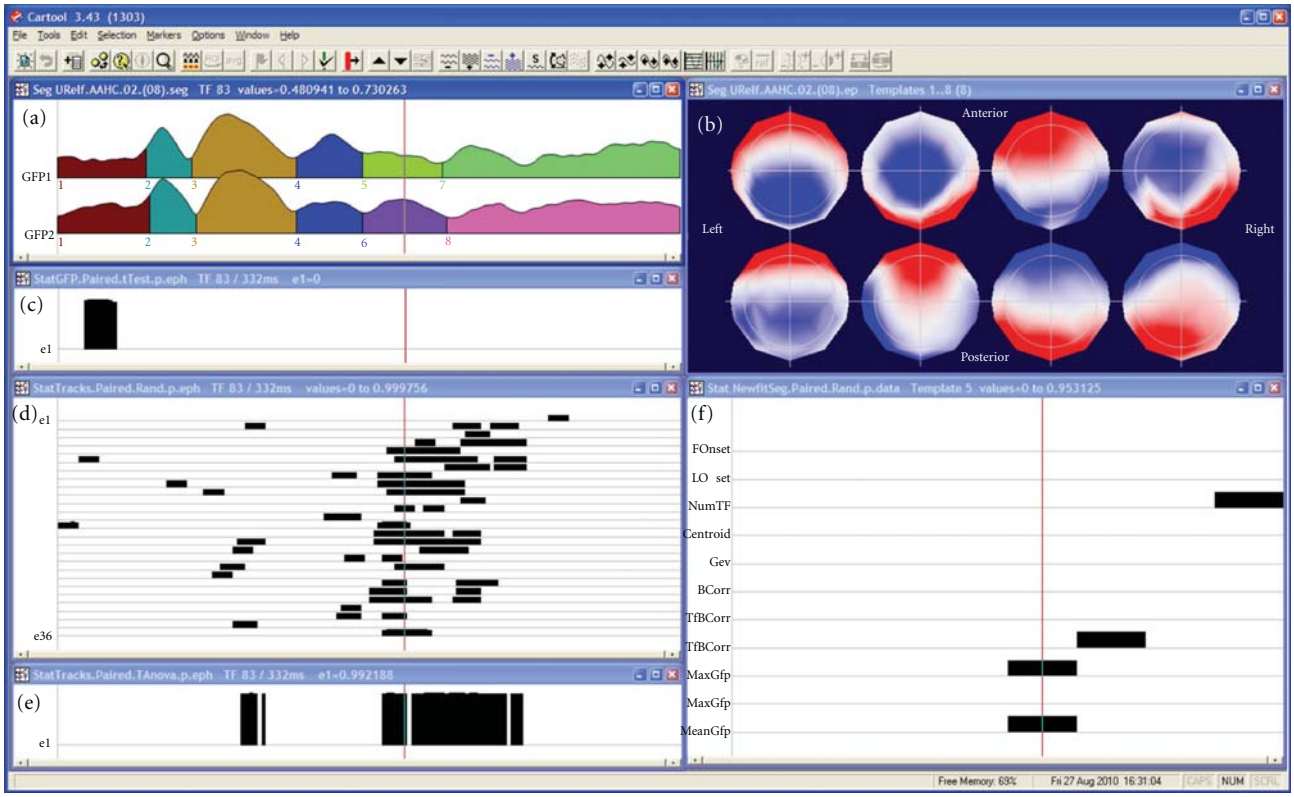


Figure 5: Illustration of the statistical analysis in CARTOOL. The same data as in Figure 2 are used and the segmentation result and the maps of the eight microstates (labeled consecutively) are again shown in the window on the top. The second window on the left shows the test of the Global Field Power (GFP). Black bars indicate time points with $P < .05$. The third window on the left shows the t-test for each electrode and each time point. The bottom window shows the test of topographic differences using the TANOVA method. Finally, the window on the right shows the different parameters from the map fitting methods, showing which segments have significant differences in subjects. Note that in this case the topographic analysis corresponds to the two time periods that were significant in the complex electrode-wise t-tests, while the GFP test reveals an early effect that was not significant at the single electrode level.

For multivariate analysis the spreadsheets have to be read into other statistical packages. Ongoing developments are underway for the implementation of multivariate statistics [41, 42] as well as other topographic analysis methods such as the test for ERP component stability recently proposed by Koenig and Melie-García [16].

6. Frequency Analysis of Multichannel EEG

Quantitative analysis of spontaneous EEG has traditionally relied on Fourier-transformation based spectral analysis. Thereby, the power of the different frequencies or frequency bands is compared between different conditions/populations. In multichannel data, power maps are often used and statistical maps of power differences are calculated. This approach has been very successful and helped to characterize vigilance changes, sleep stages, drug effects and various neurological and psychiatric disorders [43]. More recently, time resolved frequency analysis has been applied to spontaneous EEG using wavelet procedures [44]. CARTOOL has implemented these frequency analysis methods using FFT as well the S-transform. Windows can flexibly be defined with variable amount of overlap.

However, frequency power maps have two important problems. First, they are reference-dependent. In contrast to potential maps in the time domain, power maps in the frequency domain change when the position of the recording reference changes [45, 46]. Second, power maps ignore the phase differences between the electrodes. Only the amplitude is considered. Therefore, source localization of power maps is not possible [47]. Ignoring the phase relationship between electrodes is unfortunate, because they are determined by the configuration and interaction of the intercerebral sources. In order to perform source localization in the frequency domain, the inverse solutions have to be calculated for the complex data derived from the FFT. Consequently, inverse solutions in the frequency domain are initially complex [48]. An efficient way to perform distributed source localizations of a large number of EEG epochs in the frequency domain is to first compute average cross-spectral matrices [49]. Another, more simplified way is to approximate the power maps by maps where all electrodes have a common phase. This method has been called FFT-approximation [47]. It is based on the calculation of the first principal component of the data in the complex plane. The results are single-phase approximated potential map for each frequency that can be

subjected to source estimation methods. It has initially been developed for single equivalent dipole localization methods, because a single dipole cannot account for phase differences between electrodes. However, distributed source estimation algorithms can also be applied to these maps in order to localize the distribution of all sources that are oscillating in phase. The FFT-Approximation method is implemented in CARTOOL.

7. Source Localization

While the analysis of the scalp potential maps as described up to now has the advantage, as compared to waveform analysis, to be reference independent and considers the whole brain electrical activity, it does not provide any direct conclusions about the number, location and orientation of the intracranial generators [50]. Inverse solution methods are required to estimate these sources.

A major breakthrough in the spatial analysis of multichannel EEG/MEG was the development of distributed inverse solution methods that allow the estimation of the 3-dimensional distribution of neuronal activity in the whole brain at each moment in time [3, 51, 52]. The stability and reliability of these methods are impressive, and they have been validated by several direct comparisons with intracranial recordings, lesion studies and other neuroimaging methods [53]. The advancement in this field has tremendously boosted the use of electrophysiological methods in experimental and clinical studies because of the major advantages that the high temporal resolution provides. CARTOOL has implemented some of the major distributed linear inverse solutions, namely the weighted minimum norm solution (WMN) [54], the low resolution electromagnetic tomography (LORETA) [55] and the local autoregressive average (LAURA) [56] and EPIFOCUS [57]. CARTOOL calculates the inverse matrices for these different source models. It is well known that the regularization parameter can strongly influence the inverse solutions. It cannot only eliminate, but also create "ghost sources" in case of overfitting the data [58]. CARTOOL uses the L-curve method [59] to find the optimal regularization value for a given data file. This optimal value is used as default display, but the user has the possibility to toggle through stronger and weaker regularizations that are also stored in the inverse matrix. The inverse matrices are multiplied online with the EEG data and displayed for each time point using different display options. Currently, the so-called SMAC (Spherical Model with Anatomical Constraints) head model [60] as well as a more complex head models based on local spheres are available, both applied either to the individual MRI (if available) or to template MRIs. In the SMAC model, the full head is transformed to a sphere through a non-linear warping function based on the surface of the scalp. The mean radius of the scalp, skull and brain are then used for a 3-shell model. Depending on brain size, between 3000 and 5000 solution points are then defined in regular distances within the gray matter. This also includes deeper brain structures such as the amygdala, hippocampus and thalamic structures, as long as they are recognized as grey matter. The forward problem is then solved with an analytical solution using this "realistic" (i.e., individual) head model without any constraints on dipole orientation. The LSMAC model (Locally Spherical Model with Anatomical Constrains), on the contrary, does not need this initial spherization step. Instead, at each electrode locus, an adaptive local spherical model is used. To do so, and sequentially under each electrode, the thicknesses of the scalp, skull and brain are estimated. These thicknesses are then used in a 3-shell spherical model with the local radiuses, allowing the real geometry between solution points and electrodes to be accounted for. The SMAC and LSMAC methods are illustrated in Figure 6.

The results of the inverse solutions (norm or vectors) are displayed 3-dimensionally in the real (i.e., untransformed) MRI (Figure 7). Slices in all orientations can be shown as well as the solutions on surface-rendered images. The inverse solution results can be stored as matrices for further statistical processing (source waveform analysis), or as volumes for fusioning with other neuroimaging results or for importing into other image analysis tools such as SPM.

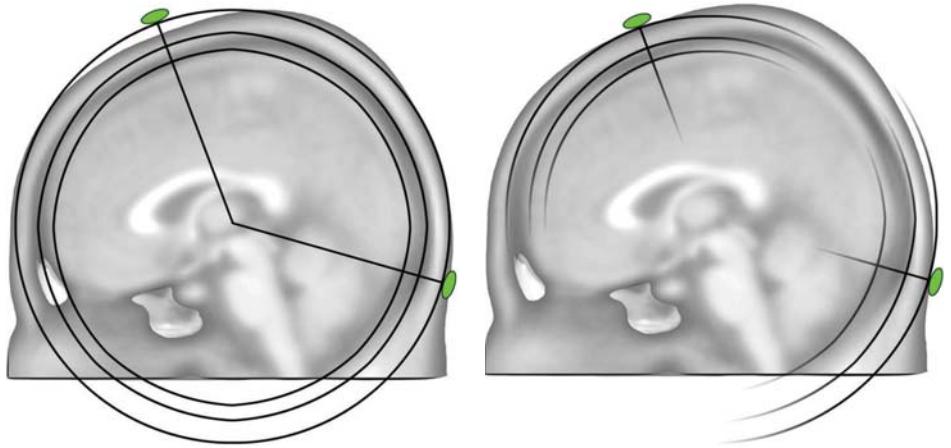
It is worthwhile to note that the segmentation of the brain surface and the grey matter is implemented in CARTOOL including manual correction tools to exclude incorrect classification of grey matter or exclude structures as brainstem and cerebellum if desired. Alternatively, already segmented brains can be read into CARTOOL if preferred, such as standard template brains. Also the warping transformations as well as the distribution of the solution points are done within CARTOOL. Thus, the whole source localization process can be performed within CARTOOL, starting with the original EEG and the original MRI.

Besides source reconstruction in the individual MRI, CARTOOL can also use template brains such as the MNI brain. In this case the solution space remains the same for all subjects, allowing group studies on the source level. In the case of the MNI brain, all solution points are labeled with their Talairach coordinates as well as their anatomical labels. Time periods of interest (e.g., the microstates) or regions of interest (ROI) can be created and the mean current density within these segments or ROI can be stored for further statistical analysis [61]. Finally, the source waveforms can be stored and further treated in CARTOOL like scalp electrode traces.

8. Additional Implementations

Several other analysis tools are available in CARTOOL. In addition to the analysis of scalp EEG, CARTOOL also allows to visualize and analyze intracranial EEG, to read and fusion 3D images of different imaging modalities, to create and work with 3D regions of interest, and to convert and manipulate MR images.

The analysis of intracranial recordings includes the color-coded mapping of grids as well as along depth electrodes (Figure 8). Intracranial electrodes are directly fused with the patient's MRI if the exact positions are available. Depth shifting tricks allow the experimenter to see the potentials of the depth electrodes within the MRI.



(a) (b)

Figure 6: Illustration of the two head models used for the inverse solution calculation in CARTOOL. The SMAC model (a) uses 3-shell of constant radiuses for the scalp and skull, which is in average a good approximation, but can be locally inaccurate for some electrodes. Due to the spherization step, the geometrical relationship between the inverse space and the electrodes is also slightly incorrect. The LSMAC model (b) uses the local radiuses of the scalp and skull, under each electrode locus, to generate different sets of 3 shells spherical model. Therefore, the forward problem is geometrically correct for each electrode.

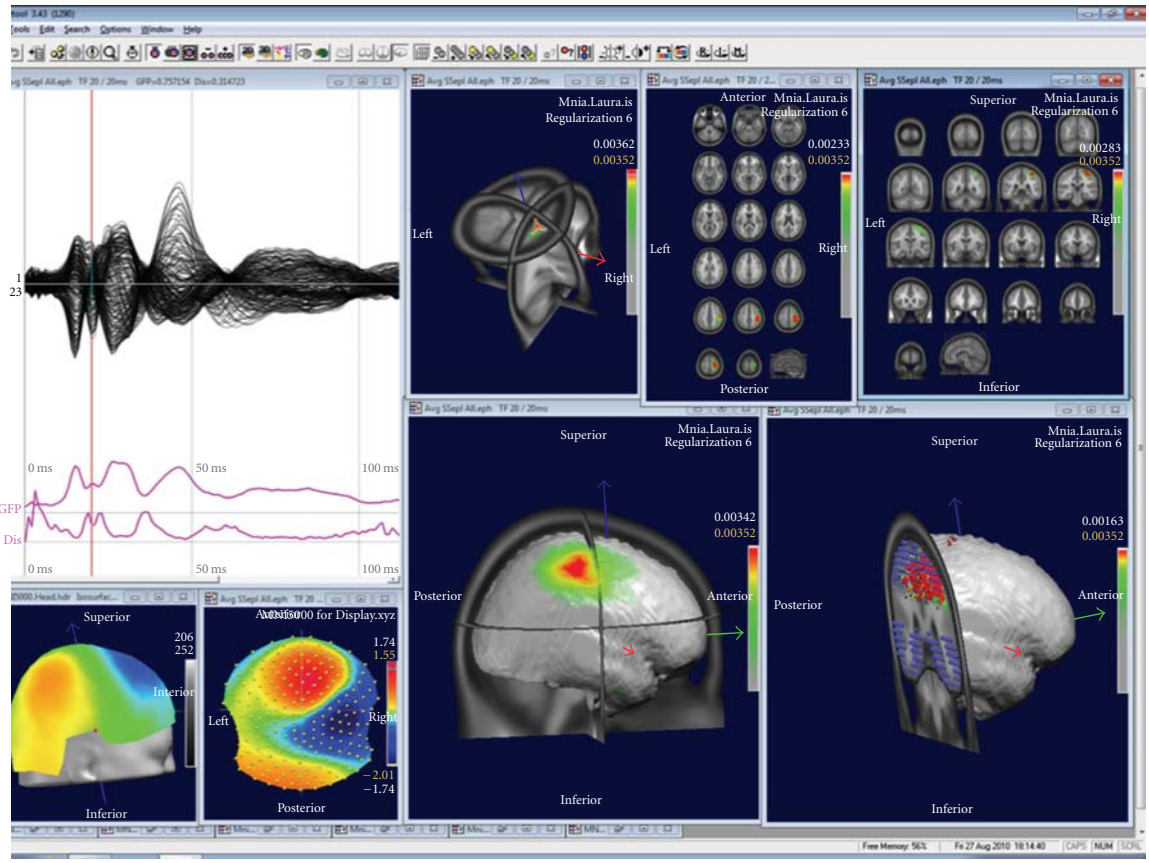


Figure 7: Examples of displays of the source localization results in CARTOOL. The data show a grand-mean somatosensory evoked potential after left median nerve stimulation. Overlapped waveforms and the map at 20 ms are displayed on the left. The right windows show different types of displays of the sources at this time point.

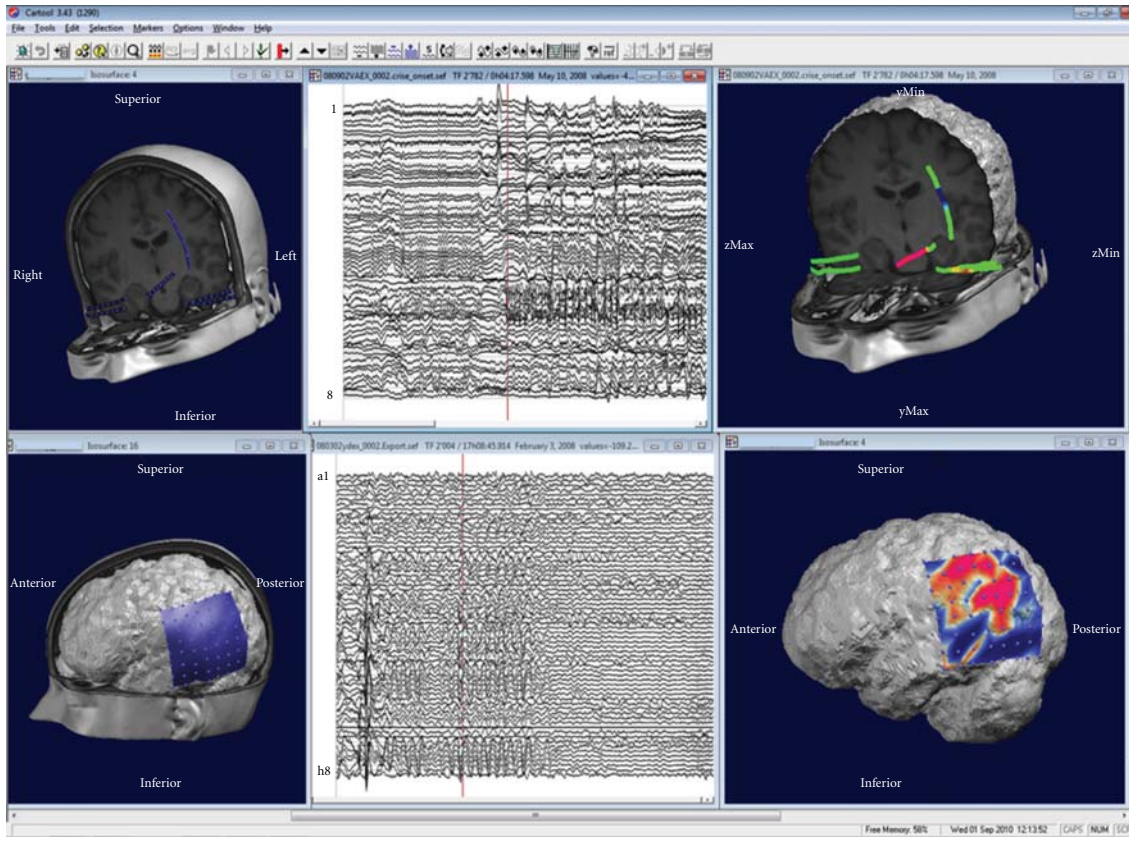


Figure 8: Illustration of the display of intracranial recordings using CARTOOL. Top: recordings from depth electrodes: Left: display of the electrode positions. Middle: waveform display of the beginning of a seizure. Right: potential mapped on the electrodes. Bottom: recordings from a 8 × 8 subdural grid. Left: display of the grid position on the patient's MRI. Middle: Traces from the 64 electrodes. Right: potential mapped on the grid for one moment in time.

CARTOOL can read any type of 3D volumes in Analyze format. In case of functional images (fMRI or PET) the activation areas are displayed as colored blobs within the MRI. Several different volumes can be overlapped and thus results of different imaging modalities (including the inverse solutions) can be displayed within the same MRI. MRIs can also be manipulated and converted in various ways and stored as new volumes.

A large palette of 3D display tools is available in CARTOOL that allows flexible visualization of the data. Any 3D object can be inserted into another one by a single click, allowing the merging of different information instantly, while retaining the spatial coherence of the objects. Different windows can easily be synchronized, allowing the user to visualize the dynamic behavior of the data on traces, maps and inverse solutions simultaneously (Figure 9).

9. Ongoing Developments

CARTOOL is constantly implementing new analysis methods that appear as useful and have been published. Concerning the statistical analysis, multivariate methods will soon be implemented. The TANOVA described above can easily be extended to multivariate measures [42]. Also tests of topographic consistency across subjects based on randomization tests of GFP are promising and will be implemented soon [16].

Concerning microstate segmentation, De Lucia et al. [31, 62] proposed a variation of the cluster analysis described above to apply to single-trial ERP data. The method proposes to model the overall electrical response, that is, the event-related and the ongoing activity, as a mixture of Gaussians, in an N-dimensional space, where N is the number of the electrodes. The computation is initialized by a K-means algorithm, which iteratively improves the estimation of means, covariances and priors of the Q Gaussians until the likelihood reaches a plateau. For each time point and trial it then provides Q conditional probabilities that relate the topographies to the clusters. Like in the labeling procedure described above, each time point and trial is then labeled with the cluster with which it has highest conditional probability. The method has been shown to reliably identify specific component maps in the single trial ERPs despite the clear dominance of the ongoing spontaneous EEG activity.

An important new development in the field of EEG/MEG analysis is the measure of connectivity between different brain areas as a way to understand the organized behavior of different brain regions [63]. The use of EEG data to examine

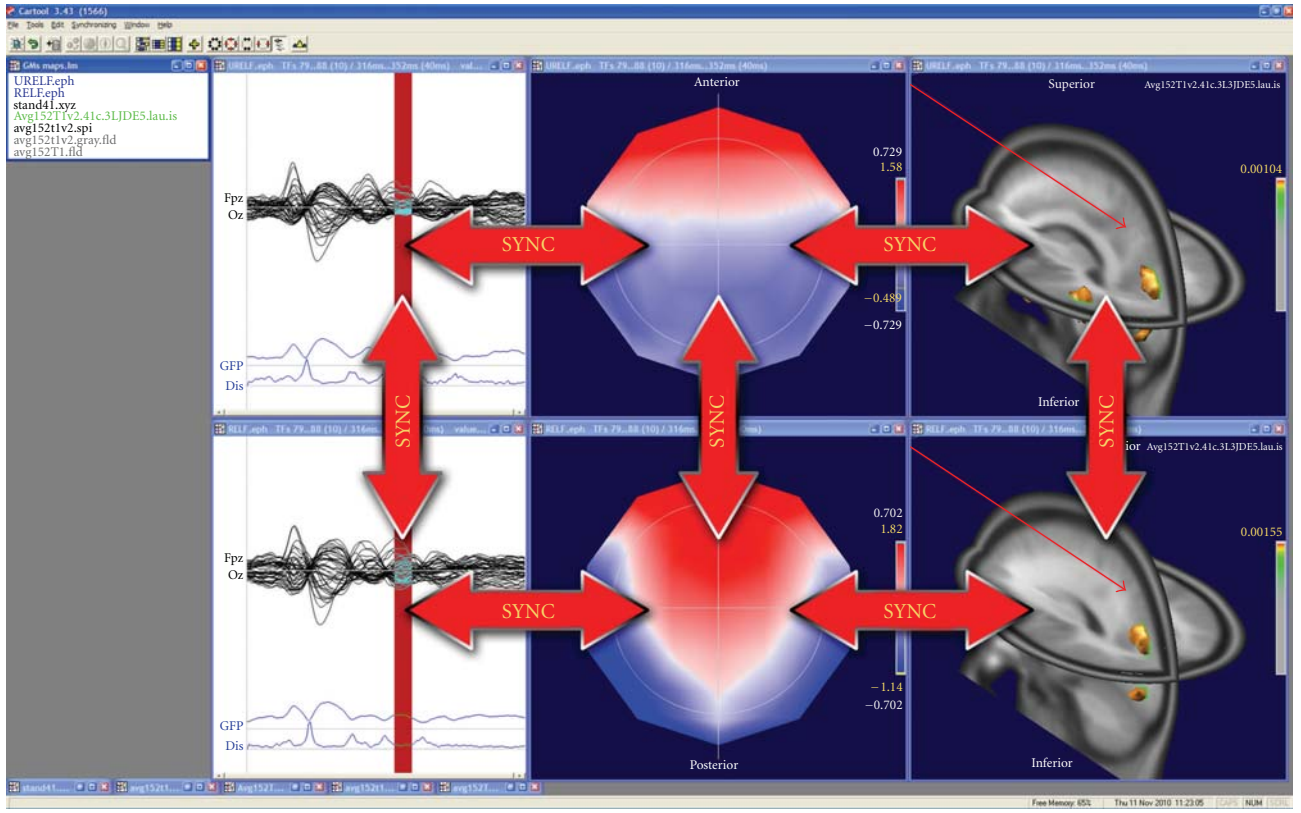


Figure 9: Illustration of the synchronization of windows in CARTOOL. Synchronization within the same dataset allows dynamic animations in time. Synchronization across datasets allows comparison of different conditions in time and in space.

the functional connectivity has a long history [64, 65] and a variety of techniques have been used, most prominently the calculation of cross-correlation or phase synchronization between pairs of scalp electrodes or sensors [66]. Also graph-theory-based tools from the study of complex network have been proposed [67]. The problem with such analysis on the scalp surface is that the interpretation with respect to the sources that generated the connectivity between electrodes is ambiguous because of the spreading of electromagnetic signals from the cortex to the sensors. More appropriate is the use of connectivity measures in the inverse space. A popular method that is often applied to MEG data (but can also be used for EEG) is the so-called dynamic imaging of coherent sources (DICS), proposed by [68]. This method uses a beamformer spatial filter to identify coherent sources in the brain for specific frequency bands. Like other spectral coherence methods, DICS does not give information of the direction of information flow. Several alternative methods have therefore been proposed that are based on the Granger causality theory and multivariate autoregressive models such as Partial Directed Coherence [69] or the Directed Transfer Function [70]. Applying these methods to the data in the inverse space after applying the distributed linear inverse solutions described above will allow estimating the flow of electrical information in large-scale neuronal networks in real time. Such methods will be implemented in future versions of CARTOOL together with other promising distributed inverse solutions and head models that currently emerge in the literature.

10. Software Details

CARTOOL is not an Open Source project; however the program is distributed freely to any nonprofit research group. Users need to register only once and will then be informed of any updates of the software. They are asked to cite the use of CARTOOL in the *Methods* and *Acknowledgements* sections of their papers. Currently about 650 users from all over the world have registered and downloaded CARTOOL.

CARTOOL runs only on Windows platforms (ranging from Windows 95 up to Windows 7) as a standalone compiled executable program, included with its complete documentation within a single installation program. It is fully written in C++ in order to attain the highest speed and compactness in memory use. The advanced display is completely done in *OpenGL*, so it needs an *OpenGL* accelerated graphic card, and as much memory as possible for the most demanding operations. Matlab is not needed to run CARTOOL.

Interoperability is achieved mainly by exchanging intermediate results through files. Considerable efforts have been put in reading and writing standard files with a maximum of convenience for the user. For example, most of the operations can be done by *Drag & Drop* in CARTOOL, the landing of

the drop usually conditioning the actions to be undertaken on the files.

Help is provided at different levels. A thorough *Reference Guide*, describing all the options and technical details for all processes, is included in the distribution in the form of compiled HTML (10 MB *chm* file). In addition a CARTOOL *Community* group (Google Sites) has been built in order to offer a central access for the users, which includes a *User's Guide*, collaboratively edited as a Wiki, a *Discussion Forum* for all questions and announcements, some *FAQs*, and a few shared files. Finally, it is worth noting that CARTOOL updates, including Beta releases, can be easily assessed online through a Google Docs repository.

Here are the main internet addresses for CARTOOL:

http://brainmapping.unige.ch/cartool

http://cartoolcommunity.unige.ch/

Files formats read by CARTOOL include formats produced by the EEG systems form the companies Biologic, Biosemi, Brain Products, Deltamed, EGI, and Neuroscan. Also EDF format and standard text files can be read. Concerning MRI, Analyze and AVS formats are read. A complete list of file formats is included in the Reference Guide of Cartrool.

Acknowledgments

The development of CARTOOL is supported by the Center for Biomedical Imaging (CIBM) of the Universities of Geneva and Lausanne, The Swiss Federal Institute of Technology in Lausanne (EPFL), and the University Hospitals of Geneva and Lausanne, Switzerland. This work has been supported by the Swiss National Science Foundation (Grant no. 33CM30-124089, and no. 310030-132952 to C. M. Michel and 310030B-133136 to M. M. Murray).

References

[1] P. L. Nunez and R. Srinivasan, *Electric Fields of the Brain: The Neurophysics of EEG*, Oxford University Press, New York, NY, USA, 2nd edition, 2006.

[2] H. G. Vaughan, "The neural origins of human event-related potentials," *Annals of the New York Academy of Sciences*, vol. 388, pp. 125–138, 1982.

[3] C. M. Michel, M. M. Murray, G. Lantz, S. Gonzalez, L. Spinelli, and R. Grave De Peralta, "EEG source imaging," *Clinical Neurophysiology*, vol. 115, no. 10, pp. 2195–2222, 2004.

[4] D. B. Geselowitz, "The zero of potential," *IEEE Engineering in Medicine and Biology Magazine*, vol. 17, no. 1, pp. 128–136, 1998.

[5] D. Lehmann and W. Skrandies, "Reference-free identification of components of checkerboard-evoked multichannel potential fields," *Electroencephalography and Clinical Neurophysiology*, vol. 48, no. 6, pp. 609–621, 1980.

[6] R. D. Pascual-Marqui and D. Lehmann, "Topographic maps, source localization inference, and the reference electrode: comments on a paper by Desmedt et al," *Electroencephalography and Clinical Neurophysiology*, vol. 88, no. 6, pp. 532–533, 1993.

[7] J. Dien, "Issues in the application of the average reference: review, critiques, and recommendations," *Behavior Research Methods, Instruments, and Computers*, vol. 30, no. 1, pp. 34–43, 1998.

[8] T. Koenig and L. R. R. Gianotti, "Scalp field maps and their characterization," in *Electrical Neuroimaging*, C. M. Michel et al., Ed., pp. 25–48, Cambridge University Press, Cambridge, UK, 2009.

[9] M. M. Murray et al., "Principles of topographic analyses for electrical neuroimaging," in *Brain Signal Analysis*, T. C. Handy, Ed., pp. 21–54, The MIT Press, Cambridge, Mass, USA, 2009.

[10] M. M. Murray, G. R. Wylie, B. A. Higgins, D. C. Javitt, C. E. Schroeder, and J. J. Foxe, "The spatiotemporal dynamics of illusory contour processing: combined high-density electrical mapping, source analysis, and functional magnetic resonance imaging," *Journal of Neuroscience*, vol. 22, no. 12, pp. 5055–5073, 2002.

[11] D. Lehmann, "Multichannel topography of human alpha EEG fields," *Electroencephalography and Clinical Neurophysiology*, vol. 31, no. 5, pp. 439–449, 1971.

[12] C. M. Michel et al., Ed., *Electrical Neuroimaging*, Cambridge University Press, Cambridge, UK, 2009.

[13] C. M. Michel and D. Brandeis, "Data acquisition and preprocessing standards for electrical neuroimaging," in *Electrical Neuroimaging*, C. M. Michel, T. Koenig, D. Brandeis, L. R. R. Gianotti, and J. Wackermann, Eds., pp. 79–92, Cambridge University Press, Cambridge, UK, 2009.

[14] F. Perrin, J. Pernier, O. Bertrand, and J. F. Echallier, "Spherical splines for scalp potential and current density mapping," *Electroencephalography and Clinical Neurophysiology*, vol. 72, no. 2, pp. 184–187, 1989.

[15] W. Skrandies, "The effect of stimulation frequency and retinal stimulus location on visual evoked potential topography," *Brain Topography*, vol. 20, no. 1, pp. 15–20, 2007.

[16] T. Koenig and L. Melie-García, "A method to determine the presence of averaged event-related fields using randomization tests," *Brain Topography*, vol. 23, no. 3, pp. 233–242, 2010.

[17] M. M. Murray, D. Brunet, and C. M. Michel, "Topographic ERP analyses: a step-by-step tutorial review," *Brain Topography*, vol. 20, no. 4, pp. 249–264, 2008.

[18] D. Brandeis, H. Naylor, R. Halliday, E. Callaway, and L. Yano, "Scopolamine effects on visual information processing, attention, and event-related potential map latencies," *Psychophysiology*, vol. 29, no. 3, pp. 315–336, 1992.

[19] D. Lehmann, "Principles of spatial analysis," in *Methods of Analysis of Brain Electrical and Magnetic Signals*, A. S. Gevins and A. Remont, Eds., pp. 309–354, Elsevier, Amsterdam, The Netherlands, 1987.

[20] D. Lehmann, H. Ozaki, and I. Pal, "EEG alpha map series: brain micro-states by space-oriented adaptive segmentation," *Electroencephalography and Clinical Neurophysiology*, vol. 67, no. 3, pp. 271–288, 1987.

[21] D. Lehmann, R. Pascual-Marqui, and C. M. Michel, "EEG microstates," *Scholarpedia*, vol. 4, no. 3, p. 7632, 2009.

[22] B. J. Baars, "The conscious access hypothesis: origins and recent evidence," *Trends in Cognitive Sciences*, vol. 6, no. 1, pp. 47–52, 2002.

[23] S. L. Bressler and E. Tognoli, "Operational principles of neurocognitive networks," *International Journal of Psychophysiology*, vol. 60, no. 2, pp. 139–148, 2006.

[24] S. Grossberg, "The complementary brain: unifying brain

dynamics and modularity," *Trends in Cognitive Sciences*, vol. 4, no. 6, pp. 233–246, 2000.

[25] J.-P. Changeux and C. M. Michel, "Mechanism of neural integration at the brain-scale level," in *Microcircuits*, S. Grillner and A. M. Graybiel, Eds., pp. 347–370, The MIT Press, Cambridge, Mass, USA, 2004.

[26] M. M. Murray et al., "Principles of topographic analyses for electrical neuroimaging," in *Brain Signal Analysis: Advances in Neuroelecric and Neuromagnetic Methods*, T. C. Handy, Ed., pp. 21–54, The MIT Press, London, UK, 2009.

[27] C. M. Michel, M. Seeck, and T. Landis, "Spatiotemporal dynamics of human cognition," *News in Physiological Sciences*, vol. 14, no. 5, pp. 206–214, 1999.

[28] R. D. Pascual-Marqui, C. M. Michel, and D. Lehmann, "Segmentation of brain electrical activity into microstates: model estimation and validation," *IEEE Transactions on Biomedical Engineering*, vol. 42, no. 7, pp. 658–665, 1995.

[29] A. J. Bell and T. J. Sejnowski, "An information-maximization approach to blind separation and blind deconvolution," *Neural Computation*, vol. 7, no. 6, pp. 1129–1159, 1995.

[30] G. Pourtois, S. Delplanque, C. Michel, and P. Vuilleumier, "Beyond conventional event-related brain potential (ERP): exploring the time-course of visual emotion processing using topographic and principal component analyses," *Brain Topography*, vol. 20, no. 4, pp. 265–277, 2008.

[31] M. De Lucia, C. M. Michel, and M. M. Murray, "Comparing ICA-based and single-trial topographic ERP analyses," *Brain Topography*, vol. 23, pp. 119–127, 2010.

[32] A. J. Pegna, A. Khateb, L. Spinelli, M. Seeck, T. Landis, and C. M. Michel, "Unraveling the cerebral dynamics of mental imagery," *Human Brain Mapping*, vol. 5, no. 6, pp. 410–421, 1997.

[33] T. Koenig, L. Prichep, D. Lehmann et al., "Millisecond by millisecond, year by year: normative EEG microstates and developmental stages," *NeuroImage*, vol. 16, no. 1, pp. 41–48, 2002.

[34] D. Lehmann, P. L. Faber, S. Galderisi et al., "EEG microstate duration and syntax in acute, medication-naïve, first-episode schizophrenia: a multi-center study," *Psychiatry Research*, vol. 138, no. 2, pp. 141–156, 2005.

[35] J. Britz, D. van de Ville, and C. M. Michel, "BOLD correlates of EEG topography reveal rapid resting-state network dynamics," *NeuroImage*, vol. 52, no. 4, pp. 1162–1170, 2010.

[36] D. van de Ville, J. Britz, and C. M. Michel, "EEG microstate sequences in healthy humans at rest reveal scale-free dynamics," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 107, no. 42, pp. 18179–18184, 2010.

[37] M. M. Murray, D. M. Foxe, D. C. Javitt, and J. J. Foxe, "Setting boundaries: brain dynamics of modal and amodal illusory shape completion in humans," *Journal of Neuroscience*, vol. 24, no. 31, pp. 6898–6903, 2004.

[38] A. Khateb, A. J. Pegna, T. Landis, M. S. Mouthon, and J. M. Annoni, "On the origin of the N400 effects: an ERP waveform and source localization analysis in three matching tasks," *Brain Topography*, vol. 23, no. 3, pp. 311–320, 2010.

[39] D. Guthrie and J. S. Buchwald, "Significance testing of difference potentials," *Psychophysiology*, vol. 28, no. 2, pp. 240–244, 1991.

[40] I. Kondakor, R. D. Pascual-Marqui, C. M. Michel, and D. Lehmann, "Event-related potential map differences depend on the prestimulus microstates," *Journal of Medical Engineering and Technology*, vol. 19, no. 2-3, pp. 66–69, 1995.

[41] M. Wirth, H. Horn, T. Koenig et al., "The early context effect reflects activity in the temporo-prefrontal semantic system: evidence from electrical neuroimaging of abstract and concrete word reading," *NeuroImage*, vol. 42, no. 1, pp. 423–436, 2008.

[42] T. Koenig and L. Melie-Garcia, "Statistical analysis of multichannel scalp field data," in *Electrical Neuroimaging*, C. M. Michel et al., Ed., pp. 169–189, Cambridge University Press, Cambridge, UK, 2009.

[43] E. John, L. Prichep, and P. Easton, "Normative data banks and neurometrics. Basic concepts, methods and results of norm constructions," in *Handbook of Electroencephalography and Clinical Neurophysiology. Vol.1: Methods of Analysis of Brain Electrical an Magnetic Signals*, A. S. Gevins and A. Remond, Eds., pp. 449–496, Elsevier, Amsterdam, The Netherlands, 1987.

[44] M. Le Van Quyen, J. Foucher, J. P. Lachaux et al., "Comparison of Hilbert transform and wavelet methods for the analysis of neuronal synchrony," *Journal of Neuroscience Methods*, vol. 111, no. 2, pp. 83–98, 2001.

[45] D. Lehmann, H. Ozaki, and I. Pal, "Averaging of spectral power and phase via vector diagram best fits without reference electrode or reference channel," *Electroencephalography and Clinical Neurophysiology*, vol. 64, no. 4, pp. 350–363, 1986.

[46] T. Koenig and R. D. Pascual-Marqui, "Multichannel frequency and time-frequency analysis," in *Electrical Neuroimaging*, C. M. Michel et al., Ed., pp. 145–168, Cambridge University Press, Cambridge, UK, 2009.

[47] D. Lehmann and C. M. Michel, "Intracerebral dipole source localization for FFT power maps," *Electroencephalography and Clinical Neurophysiology*, vol. 76, no. 3, pp. 271–276, 1990.

[48] T. Koenig, F. Marti-Lopez, and P. Valdes-Sosa, "Topographic time-frequency decomposition of the EEG," *NeuroImage*, vol. 14, no. 2, pp. 383–390, 2001.

[49] E. Frei, A. Gamma, R. Pascual-Marqui, D. Lehmann, D. Hell, and F. X. Vollenweider, "Localization of MDMA-induced brain activity in healthy volunteers using low resolution brain electromagnetic tomography (LORETA)," *Human Brain Mapping*, vol. 14, no. 3, pp. 152–165, 2001.

[50] D. H. Fender, "Source localization of brain electrical activity," in *Methods of Analysis of Brain Electrical and Magnetic Signals*, A. S. Gevins and A. Remont, Eds., pp. 355–403, Elsevier, Amsterdam, The Netherlands, 1987.

[51] M. S. Hämälainen, "Interpreting measured magnetic fields of the brain: estimates of current distributions," Tech. Rep. TKK-F-A559, Helsinki University of Technology, Espoo, Finland, 1984.

[52] B. He and J. Lian, "Electrophysiological neuroimaging: solving the EEG inverse problem," in *Neuroal Engineering*, B. He, Ed., pp. 221–261, Kluwer Academic Publishers, Norwell, Mass, USA, 2005.

[53] R. D. Pascual-Marqui et al., "Imaging the electrical neuronal generators of EEG/MEG," in *Electrical Neuroimaging*, C. M. Michel et al., Ed., Cambridge University Press, Cambridge, UK, 2009.

[54] F. H. Lin, T. Witzel, S. P. Ahlfors, S. M. Stufflebeam, J. W. Belliveau, and M. S. Hämäläinen, "Assessing and improving the spatial accuracy in MEG source localization by depth-weighted minimum-norm estimates," *NeuroImage*, vol. 31, no. 1, pp. 160–171, 2006.

[55] R. D. Pascual-Marqui, C. M. Michel, and D. Lehmann, "Low resolution electromagnetic tomography: a new method for localizing electrical activity in the brain," *International Journal of Psychophysiology*, vol. 18, no. 1, pp. 49–65, 1994.

[56] R. Grave de Peralta Menendez, M. M. Murray, C. M. Michel, R. Martuzzi, and S. L. Gonzalez Andino, "Electrical neuroimaging based on biophysical constraints," *NeuroImage*, vol. 21, no. 2, pp. 527–539, 2004.

[57] R. Grave de Peralta Menendez, S. G. Andino, G. Lantz, C. M. Michel, and T. Landis, "Noninvasive localization of electromagnetic epileptic activity. I. Method descriptions and simulations," *Brain Topography*, vol. 14, no. 2, pp. 131–137, 2001.

[58] M. Fuchs, M. Wagner, T. Köhler, and H. A. Wischmann, "Linear and nonlinear current density reconstructions," *Journal of Clinical Neurophysiology*, vol. 16, no. 3, pp. 267–295, 1999.

[59] P. Hansen, "Analysis of discrete Ill-posed problems by means of the L-curve," *SIAM Review*, vol. 34, pp. 561–580, 1992.

[60] L. Spinelli, S. G. Andino, G. Lantz, M. Seeck, and C. M. Michel, "Electromagnetic inverse solutions in anatomically constrained spherical head models," *Brain Topography*, vol. 13, no. 2, pp. 115–125, 2000.

[61] C. E. James, J. Britz, P. Vuilleumier, C. A. Hauert, and C. M. Michel, "Early neuronal responses in right limbic structures mediate harmony incongruity processing in musical experts," *NeuroImage*, vol. 42, no. 4, pp. 1597–1608, 2008.

[62] M. De Lucia et al., "Single subject EEG analysis based on topographic information," *International Journal of Bioelectromagnetism*, vol. 9, pp. 168–171, 2007.

[63] A. A. Ioannides, "Dynamic functional connectivity," *Current Opinion in Neurobiology*, vol. 17, no. 2, pp. 161–170, 2007.

[64] W. R. Adey, D. O. Walter, and C. E. Hendrix, "Computer techniques in correlation and spectral analyses of cerebral slow waves during discriminative behavior," *Experimental Neurology*, vol. 3, no. 6, pp. 501–524, 1961.

[65] A. S. Gevins, B. A. Cutillo, S. L. Bressler et al., "Event-related covariances during a bimanual visuomotor task. II. Preparation and feedback," *Electroencephalography and Clinical Neurophysiology*, vol. 74, no. 2, pp. 147–160, 1989.

[66] J. P. Lachaux, E. Rodriguez, J. Martinerie, and F. J. Varela, "Measuring phase synchrony in brain signals," *Human Brain Mapping*, vol. 8, no. 4, pp. 194–208, 1999.

[67] S. H. Strogatz, "Exploring complex networks," *Nature*, vol. 410, no. 6825, pp. 268–276, 2001.

[68] J. Gross, J. Kujala, M. Hämäläinen, L. Timmermann, A. Schnitzler, and R. Salmelin, "Dynamic imaging of coherent sources: studying neural interactions in the human brain," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 98, no. 2, pp. 694–699, 2001.

[69] L. Astolfi, H. Bakardjian, F. Cincotti et al., "Estimate of causality between independent cortical spatial patterns during movement volition in spinal cord injured patients," *Brain Topography*, vol. 19, no. 3, pp. 107–123, 2007.

[70] C. Wilke, L. Ding, and B. He, "Estimation of time-varying connectivity patterns through the use of an adaptive directed transfer function," *IEEE Transactions on Biomedical Engineering*, vol. 55, no. 11, pp. 2557–2564, 2008.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 130714, 12 pages
doi:10.1155/2011/130714

Research Article

EEGLAB, SIFT, NFT, BCILAB, and ERICA: New Tools for Advanced EEG Processing

Arnaud Delorme,[1,2,3] Tim Mullen,[1,4] Christian Kothe,[1] Zeynep Akalin Acar,[1] Nima Bigdely-Shamlo,[1] Andrey Vankov,[1] and Scott Makeig[1,5]

[1] *Swartz Center for Computational Neuroscience, Institute for Neural Computation, University of California San Diego, La Jolla, 92093 CA, USA*
[2] *Université de Toulouse, UPS, Centre de Recherche Cerveau et Cognition, 31062 Toulouse, France*
[3] *CNRS, CerCo, 31062 Toulouse, France*
[4] *Department of Cognitive Science, University of California San Diego, La Jolla, 92093 CA, USA*
[5] *Department of Neurosciences, School of Medicine, University of California San Diego, La Jolla, 92093 CA, USA*

Correspondence should be addressed to Arnaud Delorme, arno@ucsd.edu

Received 5 October 2010; Revised 5 January 2011; Accepted 10 February 2011

Academic Editor: Sylvain Baillet

We describe a set of complementary EEG data collection and processing tools recently developed at the Swartz Center for Computational Neuroscience (SCCN) that connect to and extend the EEGLAB software environment, a freely available and readily extensible processing environment running under Matlab. The new tools include (1) a new and flexible EEGLAB STUDY design facility for framing and performing statistical analyses on data from multiple subjects; (2) a neuroelectromagnetic forward head modeling toolbox (NFT) for building realistic electrical head models from available data; (3) a source information flow toolbox (SIFT) for modeling ongoing or event-related effective connectivity between cortical areas; (4) a BCILAB toolbox for building online brain-computer interface (BCI) models from available data, and (5) an experimental real-time interactive control and analysis (ERICA) environment for real-time production and coordination of interactive, multimodal experiments.

1. Introduction

A variety of new signal processing methods have been applied to EEG signal processing over the past fifteen years [1]. These new methods require new tools to allow routine processing of EEG data, and also make possible the analysis of multimodal data collected using more complex experimental designs than previous analysis methods allowed. Here we summarize a collection of new tools designed to be made freely available for nonprofit use and which integrate with the well-established EEGLAB software environment [2], an interactive, graphic interface menu and command line script-based environment for processing electrophysiological data. Since its introduction in 2001, EEGLAB has become a widely used platform for processing of biophysical data and for sharing of new signal processing approaches. Recently, we have introduced a number of new EEGLAB-associated toolboxes: NFT, a neuroelectromagnetic forward head modeling toolbox [3] is a new toolbox for electrical head modeling, an essential first step in electrophysiological source localization. SIFT, a source information flow toolbox, allows users to apply a wide range of recently published methods for assessing effective connectivity between EEG signals including quasi-independent sources of EEG activity. Finally, the ERICA framework, composed of the Datariver, Matriver, and Producer toolboxes, and the interoperable BCILAB toolbox manage real-time synchronization and online processing of EEG and other multimodal data streams. ERICA also handles feedback and delivery of appropriate sensory stimuli to participant(s) and/or to a control system they are operating [4].

Figure 1 depicts how the new toolboxes interact and may connect to a distributed data archiving environment (here, the proposed HeadIT data and tools resource [5]). Table 1 lists the components of the Swartz Center for

Computational Neuroscience (SCCN) software suite. Note that we designate by "EEGLAB plug-in" any function, toolkit, or more organized and ambitious projects such as fully operational and standalone toolboxes or signal processing toolboxes that use the EEGLAB data structure and conventions. In this paper, we designate by "framework" any grouping of tools or toolboxes in which common code providing generic functionality can be selectively overridden or specialized by user code to provide custom functionality. For instance the ERICA is a framework centered around the concept of a "Data River" and including the clients and server implementing this concept.

2. EEGLAB

EEGLAB is an interactive menu-based and scripting software for processing electrophysiological data based under the Matlab interpreted programming script environment [2]. EEGLAB provides an interactive graphical user interface allowing users to flexibly and interactively process their high-density electrophysiological data (of up to several hundreds of channels) and/or other dynamic brain time series data. EEGLAB implements common methods of electroencephalographic data analysis including independent component analysis (ICA) and time/frequency analysis. EEGLAB has become a widely used platform for applying and sharing new techniques for biophysical signal processing. At least 28 plug-ins have been implemented and released by user groups. Here we describe recent developments in EEG software interoperative with EEGLAB. Several of the new tools are Matlab applications that conveniently plug in to the EEGLAB menu (or may also be run as stand-alone applications).

Key EEGLAB features include

(1) an event structure and functions for importing, editing, and manipulating event information. Users can select (sub)epochs time-locked to classes of events and can sort trials for visualization based on values in any event field (e.g., subjects' reaction time),

(2) independent component analysis (ICA) decomposition of electroencephalographic data [6]. Though ICA data analysis methods have now been incorporated into most commercial software processing EEG data (BrainVision, Neuroscan, BESA), EEGLAB has the most extensive repertoire of processing and data evaluation tools for ICA-based data analysis,

(3) ready adaptability to users with different levels of programming sophistication. EEGLAB unique "history" features build scripts as users navigate through menus, allowing users to "replay", vary, or extend their data processing through easily constructed Matlab scripts. Users can either interact only with the EEGLAB graphic interface, call EEGLAB functions directly from the Matlab command line, or write their own Matlab scripts using modular EEGLAB functions and documented data structures,

(4) a truly open source philosophy, allowing any researcher to build and distribute plug-in functions or toolboxes that appear automatically in the EEGLAB menu windows of their users. This structure ensures stability of core code that a handful of expert users modify while, at the same time, allows easy inclusion of new algorithms and methods by other users.

EEGLAB comprises more than 400 Matlab functions totaling more than 50,000 lines of programming. First developed under Matlab v5.3 on Linux, EEGLAB currently runs under all versions of Matlab v7 running on Linux, Unix, Windows, and Mac OSX. Since the Matlab program is not free itself, we have also used the Matlab compiler to compile EEGLAB for those users who do not have access to Matlab. To our knowledge, 28 user-initiated EEGLAB plug-ins have been developed and made available. The online EEGLAB tutorial comprises more than 300 pages of documentation. In addition, each of the 400 stand-alone modular EEGLAB functions contains its own documentation. EEGLAB has been downloaded more than 65,000 times from 88 country domains since 2003. As of April 2010, 9,218 unique opt-in users are currently on the EEGLAB mailing lists.

3. The EEGLAB STUDY.Design Framework

The EEGLAB STUDY.design concept was introduced in June, 2010 in EEGLAB v9. Complex event-related experiments typically include a number of different types of events. Statistical contrasts between EEG activities time locked to different subsets of these event types require researchers to be able to define custom sets of independent variables for different statistical treatments of the same data. The new STUDY.design framework in EEGLAB allows users to freely define independent and dependent variables and to analyze data channel or independent component (IC) activities across subjects using mean event-related potential (ERP), power spectrum, event-related spectral perturbation (ERSP) [7], and intertrial coherence (ITC) [8] measures for any number of sets of event-related data trials time locked to different sets of events, each set of trials termed a STUDY "condition".

For example, a STUDY might contain data sets for two conditions from two groups of subjects (a 2×2 (condition, group) statistical design). Statistical comparisons might be targeted to look at main effects and interactions of condition and group in this design, or at contrasts between selected (1×2) pairs of conditions or groups. Figure 2 shows the EEGLAB STUDY.design graphic interface by means of which users can create new designs and select independent variables to include in them.

Building a STUDY design involves multiple steps. Users begin by preprocessing binary EEG data files generated by proprietary EEG recording software; for each subject, this involves importing raw data, re-referencing, filtering and removing artifacts. Once these data sets have been preprocessed, users then have to import the subject data sets into a STUDY. Creating a STUDY design for analysis then allows statistical group comparison of data measures for

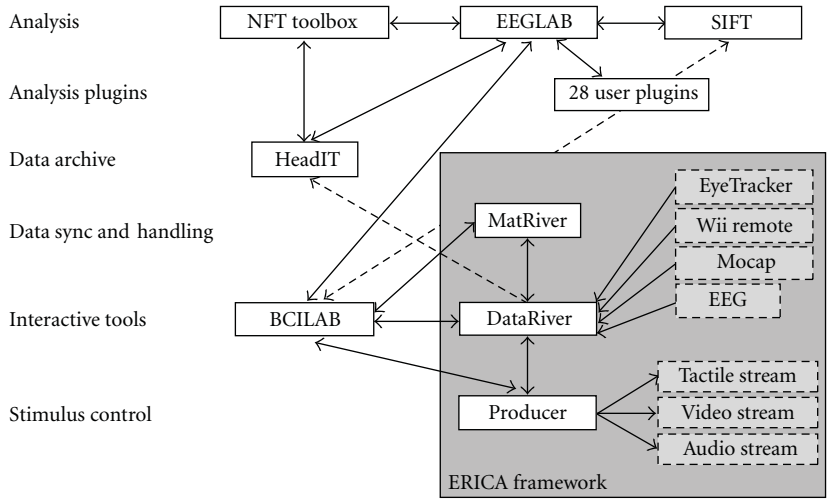


Figure 1: Complete electrophysiological experiment control, data collection, analysis, archiving, and meta-analysis suite: the EEGLAB environment for data analysis; the ERICA framework for data recording, online analysis, and stimulus control; the BCILAB toolbox for online and offline classification and BCI; the SIFT toolbox for information flow modeling; HeadIT, an archival data and tools resource under development for laboratory or archival data storage, retrieval and meta-analysis; dashed lines indicates planned interfaces under construction.

Table 1: Components of the extended SCCN software suite.

Software	Since	Vers.	Licence	Open Src.	Platform	Web link
EEGLAB	2002	10.0	GNU GPL	Yes	Matlab	http://sccn.ucsd.edu/wiki/EEGLAB
NFT toolbox	2009	2.0	GNU GPL	Yes†	Matlab†	http://sccn.ucsd.edu/wiki/NFT
SIFT	2010	0.1a	GNU GPL	Yes	Matlab	http://sccn.ucsd.edu/wiki/SIFT
BCILAB	2010	0.9	GNU GPL	Yes	Matlab	http://sccn.ucsd.edu/wiki/BCILAB
ERICA	2009	1.0	Mixed	Mixed	Windows††	http://sccn.ucsd.edu/wiki/ERICA

DataRiver, a central compiled C++ ERICA component, is free for noncommercial use. It is not open source.
† Contains a large number of precompiled C and C++ routines, all of them being open source.
†† Many components also run under Linux and Mac OSX.

different conditions (e.g., time locked to specific event types) for each subject. For example, in an oddball paradigm comprised of trials time locked to target, distractor, and standard stimuli, users might want to contrast these three types of trials using a 3×1 design. Alternatively, they might want to contrast distractor and target stimulus-locked trials, considered together, with responses to standard stimuli. The STUDY design feature of EEGLAB allows users to easily investigate such contrasts. In a STUDY with N subject groups, the STUDY design scheme also allows users to look at group effects for each condition using a $2 \times N$ design.

All of the above design concepts may be implemented within a single STUDY using multiple STUDY.design specifications. Finally, use of multiple designs may also be useful for testing different signal processing options. For instance, one might create two identical STUDY designs, one computing time/frequency measures using fast fourier transforms (FFT) and the other using wavelets. Once computed, the user can toggle between designs to compare results using the two types of time/frequency decomposition.

EEGLAB uses statistical tools including surrogate and parametric statistics to perform hypothesis testing on STUDY designs. Surrogate tests involve bootstrap or permutation methods. Depending on the design type, statistical hypothesis testing using t-test, one-way ANOVA or two-way ANOVA—or their surrogate-data equivalents—are performed for paired data or unpaired data designs. Finally, the False Discovery Rate (FDR) algorithm is applied to correct for multiple comparisons [9]. Using these simple yet powerful statistical tools, EEGLAB allows comparison of multiple experimental designs applied to a given data STUDY.

When working with data from multiple subjects using the STUDY design framework, users may analyse either IC, scalp channel, or other types of component activities associated with individual subjects. Decomposition of the data into ICs allows inclusion of source localization information, since many ICs strongly resemble the projection of a single equivalent current dipole, presumably reflecting their origin in a single locally synchronized cortical patch. The neuroelectromagnetic forward head modeling toolbox (NFT) thus allows for more precise source localization of IC processes for each subject using subject-adapted forward electrical head models.

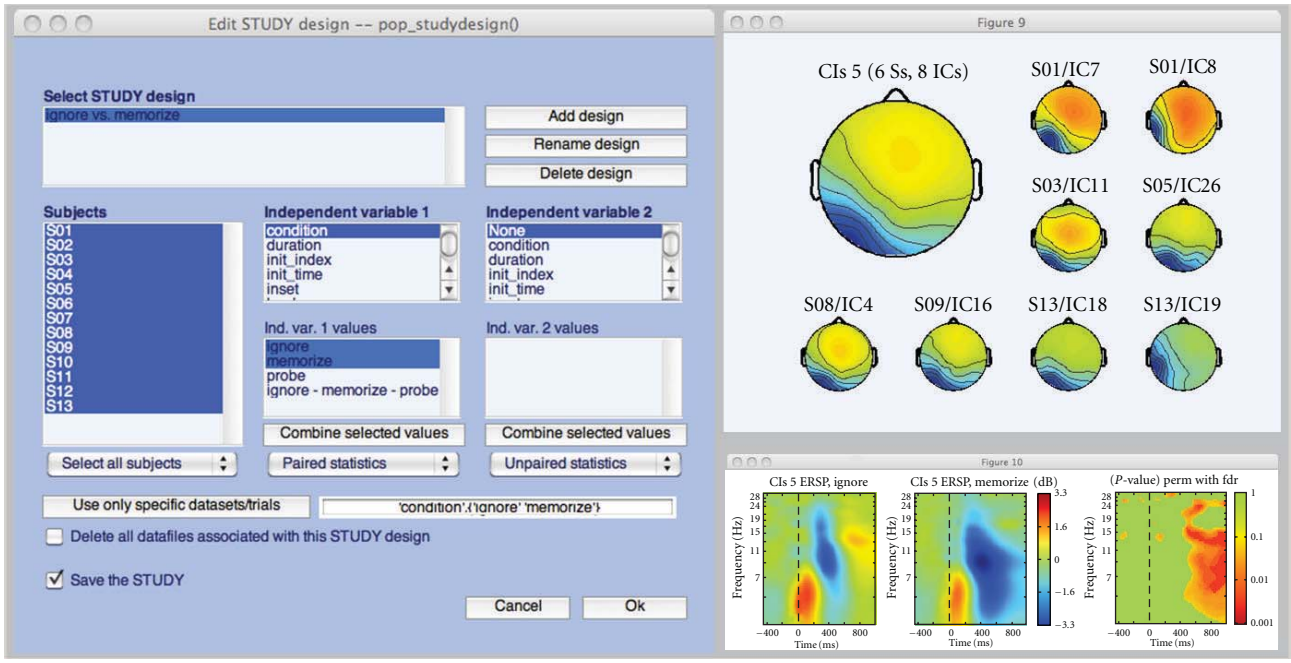


Figure 2: EEGLAB STUDY design interface using the tutorial STUDY data available via the EEGLAB wiki (http://sccn.ucsd.edu/wiki/eeglab). The three push buttons at the top may be used to add a new design ("Add design"), rename a design ("Rename design"), or delete a design ("Delete design"). The "Independent variable 1" list helps define independent variables. The list of independent variables is automatically generated based on the STUDY definition information and individual data set event types. For a given independent variable, it is also possible to select a subset of its values or to combine some of its values. For instance, in this example the user has selected "ignore" and "memorize" stimuli as values for the independent variable "condition". The "Subject" list contains the subjects to include in a specific design. Unselecting a given subject from the list excludes him/her from further data analysis within the design. Once a design is selected, measures including ERPs, mean spectra or event-related spectral perturbations (ERSP) may be plotted. Here, we have plotted the event-related spectral perturbations of an independent component (IC) cluster in the selected STUDY.design. In the top right panel, the scalp maps of one IC cluster are shown—the large map representing the average scalp map. In the bottom right panel, mean cluster ERSPs are shown for Ignore versus Memorize letter trials, and their significant differences are assessed using permutation-based statistics and a false discovery rate method to correct for multiple comparisons.

4. The Neuroelectromagnetic Forward Head Modeling Toolbox (NFT)

Our previous work has shown that some ICA component scalp topographies are highly compatible with compact cortical domains of local field synchrony that may be localized in the brain [1, 10, 11] using a four-shell spherical model or the standard boundary element method (BEM) head model included in the EEGLAB Dipfit plug-in (http://sccn.ucsd.edu/wiki/A08:_DIPFIT). When additional subject information is available, more precise localization approaches are possible. To obtain accurate source localization one needs to use a realistic electrical head model that reflects the actual electrical and geometric properties of the head. NFT adds a realistic head modeling framework to the spherical and MNI head models already provided by Dipfit within EEGLAB. The NFT framework automates most of the tasks needed to generate a realistic head model from magnetic resonance (MR) images and/or from measured EEG sensor coordinates, and provides advanced boundary element method (BEM) and Finite Element Method (FEM) solvers for estimating the projected scalp fields for a given set of possible brain source areas, thus estimating solutions to the "forward" EEG modeling problem [3].

NFT is accessible from the EEGLAB graphic interface as an EEGLAB plug-in. The toolbox provides both a Matlab command line and graphical user interface for generating realistic head models from available subject information, and for solving the forward problem numerically to provide a lead-field-matrix for a given source space and sensor distribution. This makes it easy to integrate a forward head model produced by NFT into any inverse source localization approach.

NFT performs the following steps:

(1) Segmentation of MR images: If a 3-D whole-head structural T-1 MR image of the subject's head is available, the toolbox can segment the scalp, skull, CSF, and brain tissues.

(2) High-quality head models: The accuracy of numerical solutions to an inverse source localization problem depends on the quality of the underlying meshes that model conductance changes at tissue boundaries. NFT can create high-quality surface meshes from segmented MR images for use in BEM head model. FEM meshes may be generated from the BEM surface meshes using the open source Tetgen

tool [12]. Two examples of FEM and BEM meshes generated using NFT are shown in Figure 3.

(3) Warping a template head model: While use of a subject whole-head MR image is the preferred way to generate a realistic head model, such an image may not always be available. NFT can generate a semirealistic head model of the subjects' head by warping a standard template head model to the digitized 3-D electrode coordinates, when these are available.

(4) Coregistration of electrode positions with the head mesh: NFT has a two-step (manual and automatic) coregistration function for aligning the digitized electrode locations to the scalp mesh.

(5) Accurate and efficient forward problem solution: The NFT uses high-performance BEM and FEM implementations from the open source METU-FP Toolkit (http://www.eee.metu.edu.tr/metu-fp) [13, 14] for bioelectromagnetic field computations.

We have successfully used NFT to model realistic cortical source spaces comprising a large number of dipolar elements that we assume are oriented perpendicular to the local cortical surface which was extracted from subject MR head images using tessellated FreeSurfer gray and white matter surfaces [15]. We created a multiscale cortical patch basis on this surface by selecting seed points (single voxel dipoles), then extended each patch conformally to a set of Gaussian-tapered patches with areas in the range 50–200 mm^2 [16]. NFT thus may allow precise source localization of IC processes based on accurately modeled electrical current flow consistent with the individual subject head anatomy.

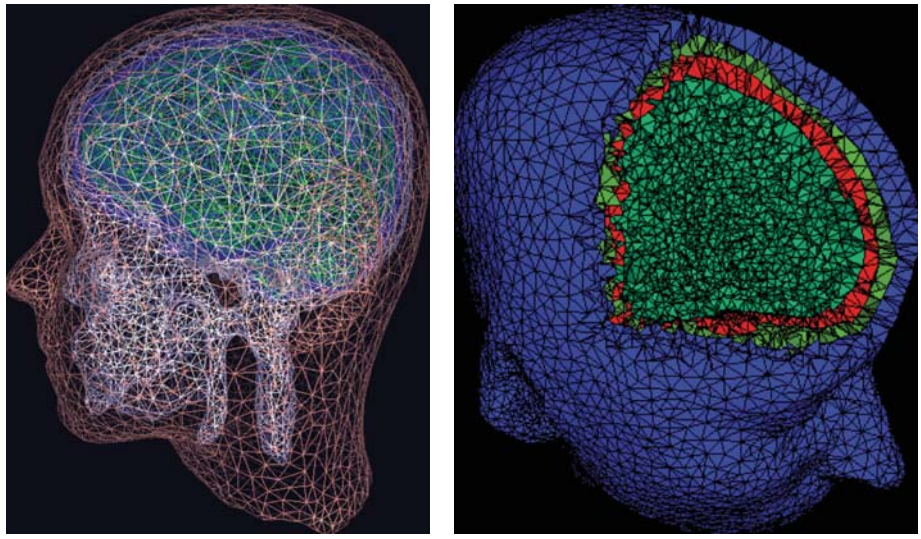
FEM modeling is a recent addition to NFT (NFT 2.0) and patch-based source space generation will be integrated into NFT in 2011. In the future, the NFT model will also be able to incorporate models of current anisotropy based on white-matter distribution information extracted from diffusion tensor/weighted imaging (DTI/DWI) head images co-registered with structural MR images.

5. Analyzing Source Information Flow Dynamics Using SIFT

Once activity in specific brain areas have been identified using source separation (e.g., ICA), and localized (e.g., using NFT), it is possible to look for transient changes in the independence of these different brain source processes. Advanced methods for noninvasively detecting and modeling distributed network events contained in high-density scalp EEG data are desirable for basic and clinical studies of distributed brain activity supporting behavior and experience. In recent years, Granger Causality (GC) and its extensions have increasingly been used to explore "effective" connectivity (directed information flow, or causality) in the brain based primarily on observed ongoing or event-related relationships between channel waveforms. While many landmark studies have applied GC to invasively recorded local field potentials and spike trains, a growing number of studies have successfully applied GC to noninvasively recorded human EEG and MEG data (as reviewed by Bressler and Seth [17]).

Based on the prediction error of autoregressive (AR) models, a process (A) is said to *Granger-cause* another process (B) if past values of process A, in addition to past values of process B, help to linearly predict future values of process B beyond what can be achieved by using past values of process B alone [18]. Using multivariate autoregressive (MVAR also referred to in the literature as VAR or MAR) models, the GC concept has been extended to an arbitrary number of signals, which may include a collection of source activities in the brain. Using this approach, through Fourier-transformation of the MVAR coefficient matrices, we can obtain the transfer and spectral density matrices (power), and ordinary, multiple, and partial coherences, where the latter quantity expresses the amount of phase coherence between two channels after subtracting out the part of the interaction which can be explained by a linear combination of all other channels. From these quantities, we can derive a frequency-domain representation of bivariate GC as well as several frequency-domain measures of directed conditional (multivariate) dependence closely related to Granger's definition of causality such as the (direct) directed transfer function (dDTF, DTF) and partial directed coherence (PDC). These and related estimators describe different aspects of network dynamics and thus comprise a complementary set of tools for MVAR-based connectivity analysis within the well-established and interpretable framework of GC [19]. To study transient causal dynamics of nonstationary phenomena, adaptive MVAR (AMVAR) approaches may be applied using locally-stationary sliding windows [20], Kalman filtering, or spectral matrix factorization. These approaches can be used to explore finely-resolved time- and frequency-dependent dynamics of directed information flow or causality between neuronal sources during cognitive information processing. Baseline significance levels for causal influence are typically obtained by a modification of a surrogate "phase randomization" algorithm [21]. This and other bootstrap, permutation, and analytical tests can be used to establish rigorous confidence intervals on estimated connectivity. Additional details on all aforementioned methods can be found in [19].

SIFT is a toolbox for modeling and visualizing information flow between sources of EEG data, possibly after separating the data into (instantaneously) maximally independent processes using ICA. The toolbox currently consists of four modules, (1) data preprocessing, (2) model fitting and connectivity estimation, (3) statistical analysis, and (4) visualization. The first module contains routines for normalization, downsampling, detrending, and other standard preprocessing steps. The second module currently includes support for several adaptive MVAR modeling approaches. From the fitted model, the user can chose to estimate spectral power, coherence, and frequency-domain connectivity, selecting from over fifteen measures published to date. The third module includes routines for surrogate statistics (phase-randomization and bootstrap statistics) for



(a) (b)

Figure 3: Two examples of (a) a set of subject head BEM meshes (modeling scalp, skull, cerebrospinal fluid (CSF), and cortex tissue boundaries) and (b) a FEM head volume for the same subject with 3-D voxels for scalp, skull, and brain tissues shown in different colors.

all measures, and analytic statistics for partial directed coherence and directed transfer function measures. The fourth module contains novel routines for interactive visualization of information flow dynamics and graph-theoretic measures across time, frequency, and anatomical source location. A graphical user interface allows easy access to the SIFT data processing pipeline.

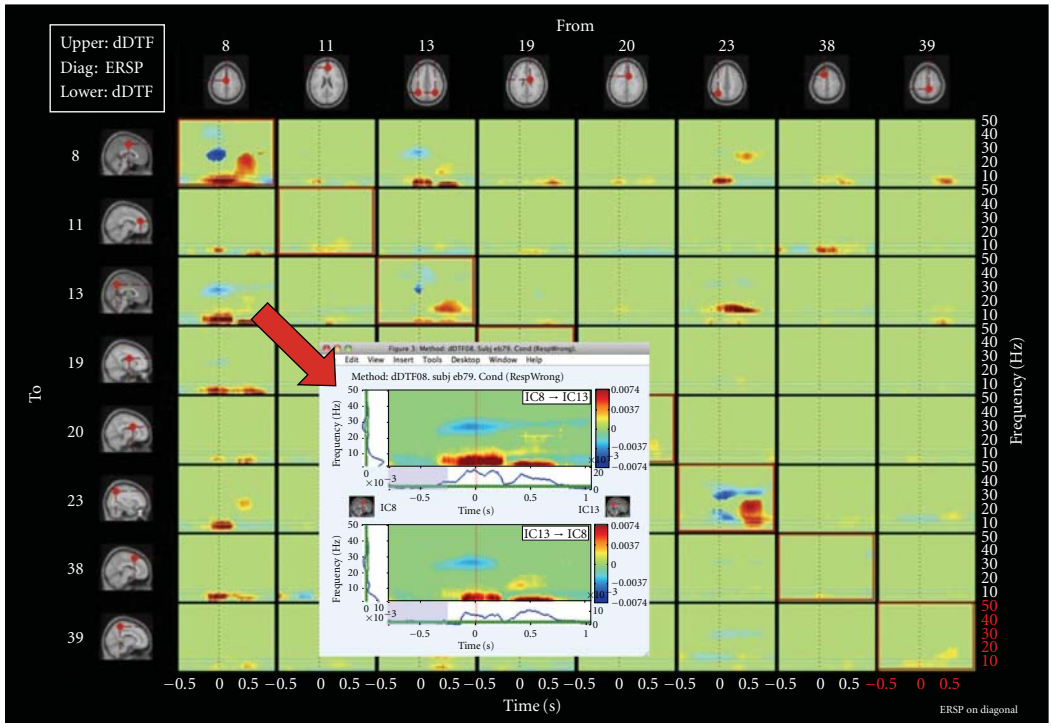
A key aspect of SIFT is that it focuses on estimating and visualizing multivariate effective connectivity in the source domain rather than between scalp electrode signals. This should allow us to achieve finer spatial localization of the network components while minimizing the challenging signal processing confounds produced by broad volume conduction from cortical sources (as well as nonbrain sources) to the scalp electrodes. SIFT may help find transient, dynamic network events that link spatially static component processes (Figure 4). The toolbox may also be used for effective connectivity analysis and visualization of phenomena in electrocorticographic (ECoG) data, for example, to identify sources and directions of information flow at onsets of and during epileptic seizures.

While the first test release of SIFT contains a number of popular MVAR-based effective connectivity measures, we are working on incorporating additional phase-amplitude coupling and transfer entropy measures. In the EEGLAB tradition, the architecture of the toolbox is also designed to allow easy addition of new methods from the user community. Another group analysis module, in development, will also be included in the upcoming second test release. This will afford clustering-based and Bayesian techniques for obtaining estimates of source-domain connectivity with confidence intervals over a subject population. The analysis framework described above allows exploration of EEG source-domain connectivity following the use of EEGLAB and NFT routines for ICA-based source separation and localization. We are currently evaluating the relative suitability of different source separation algorithms when combined with MVAR-based connectivity algorithms, and will further develop the toolbox accordingly. In the near future, we plan to interface SIFT with the BCILAB toolbox, discussed below, with the hope of applying these methods online in advanced brain-machine interfaces for real-time EEG processing, cognitive monitoring, and feedback applications.

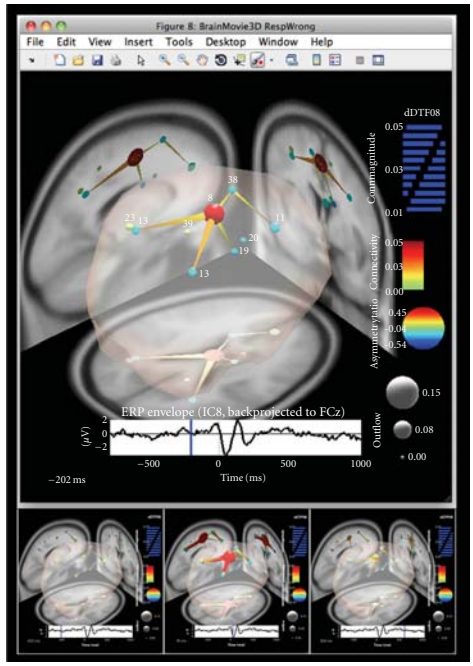
6. The Experimental Real-Time Interactive Control and Analysis (ERICA) Framework

For the purpose of real-time data acquisition and processing, we have developed an online EEG and multimodal data collection, processing, and interactive feedback environment, ERICA. Processing of EEG data in real-time software applications requires, first, organized handling of data controlling its streaming into online data processing (e.g., data-adaptive BCI or other feedback) routines whose outputs, combined into synchronized data streams (figuratively a "data river"), can be used to control or adapt ongoing stimulation processes. Synchronization of different asynchronous streams in real time over a local network may prove difficult; the originality of the ERICA framework comes from solving these issues in an efficient and elegant manner.

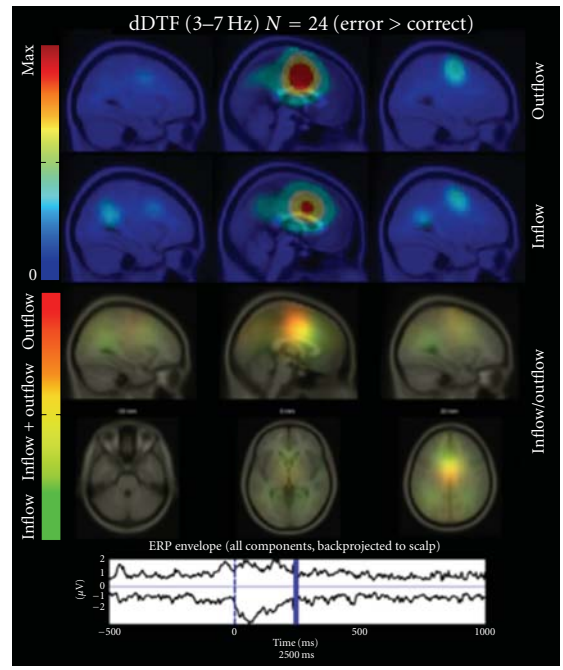
The ERICA framework is based on a unique streaming data management and real-time cross-platform synchronization application called DataRiver developed from an ADAPT data acquisition and stimulation control environment [22]. The Producer software is a DataRiver client that controls stimulus presentation in a flexible way using Variéte, an original scripting language. MatRiver, another DataRiver client,



(a)



(b)



(c)

Figure 4: EEG-based brain connectivity analysis and visualization using SIFT. (a) An interactive time-frequency grid demonstrating transient bursts of theta (3–7 Hz) and delta (1–3 Hz) band information flow during error commission, estimated using the direct directed transfer function (dDTF), between a subset of independent component (IC) sources. Dashed vertical line denotes time of erroneous button press. Callout shows an expanded view of information flow to/from sources 8 and 13, obtained by clicking on the respective grid cell. (b) Several frames from an interactive BrainMovie3D animation showing an event-related causal relationship in the theta band between these sources (200 ms (top) and − 520, 40, and 600 ms (bottom) relative to an erroneous button press). Ball (node) color and size denotes asymmetry ratio (red: causal source, blue: causal sink) and outflow strength, respectively, for that IC. Cylinder (edge) color and size denote connectivity strength. The event-related potential of IC8 (red, medial), back-projected to a superior electrode is superimposed below each frame (blue bar denotes frame index). This shows a network interpretation of the classic "error-related negativity" (ERN) phenomenon observed during error-processing. (c) A frame from a causal projection movie showing mean net causal inflow (green) and causal outflow (red) in the theta band at each brain location during error commission across 24 subjects. Note the significant causal outflow from or near anterior cingulate cortex, thought to be critically involved in error-processing, during and following the negative peak of the ERN.

allows direct read/write access to DataRiver data streams from within Matlab processes.

The central application driving development of ERICA is the development of mobile brain/body imaging (MoBI) data acquisition and analysis methods [23]—the simultaneous study of what the brain is doing (assessed via distributed EEG source dynamics), what the brain is sensing (via audiovisual scene recording), and what the brain is controlling (the totality of our behavior assessed by body motion capture, eye tracking, etc.) in performing naturally motivated actions in ordinary 3-D task environments.

To allow real-time analysis, data streams acquired by separate devices first need to be synchronized. Such streams are, by definition, asynchronous, even when they are acquired at the same nominal sampling frequency because independent clocks are used for data acquisition in each device. In addition, the sampling rates for different data sources may differ significantly: while EEG is usually sampled between 250 Hz and 2,000 Hz, video, body motion capture or subject behavioral responses may be acquired at a much lower sampling rate, and audio data streams at still higher sampling rates. For synchronization purposes, another important challenge is dealing with sporadic delays introduced by equipment acquisition, network, and operating system buffers that ensure overall regularity of data samples at the cost of ms-level time delays. For data acquired through an IP socket connection, network delays may be significant and constantly varying. Finally, Windows or any other multitasking operating system introduces variable delays in the processing of asynchronous flows—in a multitasking system, data are most often processed only when the corresponding task or program is activated and not when the data first becomes available.

DataRiver was developed in an attempt to solve these synchronization problems. DataRiver is a flexible and universal high-precision synchronization engine, providing a strong and near real-time synchronization of simultaneous data streams. It has been designed and tested with accuracy of better than 2 ms, even when synchronizing data acquisition streams from different computers (running Windows, Unix, Linux, or Mac OSX) over a local area network or the internet subnet. The DataRiver application interfaces several hardware and is typically seen as a server to DataRiver Clients that display or process data. However, each DataRiver client can also add output data to the "data river," so the strict concept of server and client does not apply.

The flexibility of the ERICA framework stems from its modular design—data output from a variety of devices are managed by specialized device drivers that convert each data stream into a device-independent stream. These streams are then merged in real time and combined into a "river" (hence the name DataRiver). DataRiver device drivers are currently available for several types of input devices and data systems including Biosemi EEG, PhaseSpace and OptiTrack motion capture systems, eye trackers, and the Wii remote (Nintendo, Inc.). This enables the rapid development of a wide range of experimental paradigms that can be tailored for a variety of multimodal experimental or application environments. Data from incoming DataRiver data streams may be used in real time by clients for recording, online data processing, and/or to provide feedback to the subject(s) being monitored. DataRiver has integrated support for data exchange in real time between one or more remote computers connected to a local area network (LAN), enabling distributed and cooperative experiments (Figure 5). New drivers and online data processing applications can easily be added to DataRiver to meet evolving research needs.

MatRiver is a MATLAB DataRiver client optimized for real-time EEG data processing, buffering and visualization using the OpenGL-based Simulink 3-D toolbox (The MathWorks, Inc.). MatRiver communicates with DataRiver by calling a binary library of functions under Windows OS. MatRiver allows online performance of common EEG preprocessing steps such as channel selection, channel re-referencing, frequency filtering and linear spatial filtering using a pre-defined ICA source signal unmixing matrix [6]. Most often, these steps may be accomplished in near real time by directly calling relevant EEGLAB functions. MatRiver also includes routines to dynamically detect "bad" channels and compensate for them by taking into account a linear ICA source propagation model. Preprocessed channel or independent component (IC) signals are accumulated and can subsequently be used for classification using MATLAB tools such as BCILAB (see following). MatRiver uses Matlab "timers" to run in the background allowing real-time processing in a nonblocking manner, even including near real-time interactive exploration of the incoming data from the Matlab command line. Continuous visualizations of data characteristics such as alpha band energy are also possible. In short, Matriver functions provide an elegant and straightforward pipeline for EEG preprocessing and classification using the rich tool set and programming simplicity of MATLAB.

7. Designing Brain-Computer Interfaces with BCILAB

After results of data stream synchronization and preprocessing have been accomplished within the ERICA framework, one may use BCILAB, an open-source MATLAB toolbox and EEGLAB [2] plug-in, to support brain-computer interface (BCI) research, and more generally, the design, learning (or adaptation), use, and evaluation of real-time predictive models operating on signals. The main objects of study in BCILAB are Brain-Computer Interface (BCI) models [24], generally defined as systems that take human biosignals as input and output estimates of some aspect of the subject's cognitive state. The signals processed by BCIs are traditionally restricted to EEG signals, but may include other modalities, such as motion-capture data or skin conductance (plus context parameters such as vehicle state, previous events, etc.). These data can be processed either using BCILAB running as a data processing node in a real-time experimentation environment (e.g., ERICA), or offline simulated real-time applications to existing data. The classifier outputs of a BCI can be streamed to a real-time application to effect stimulus or prosthetic control, or may

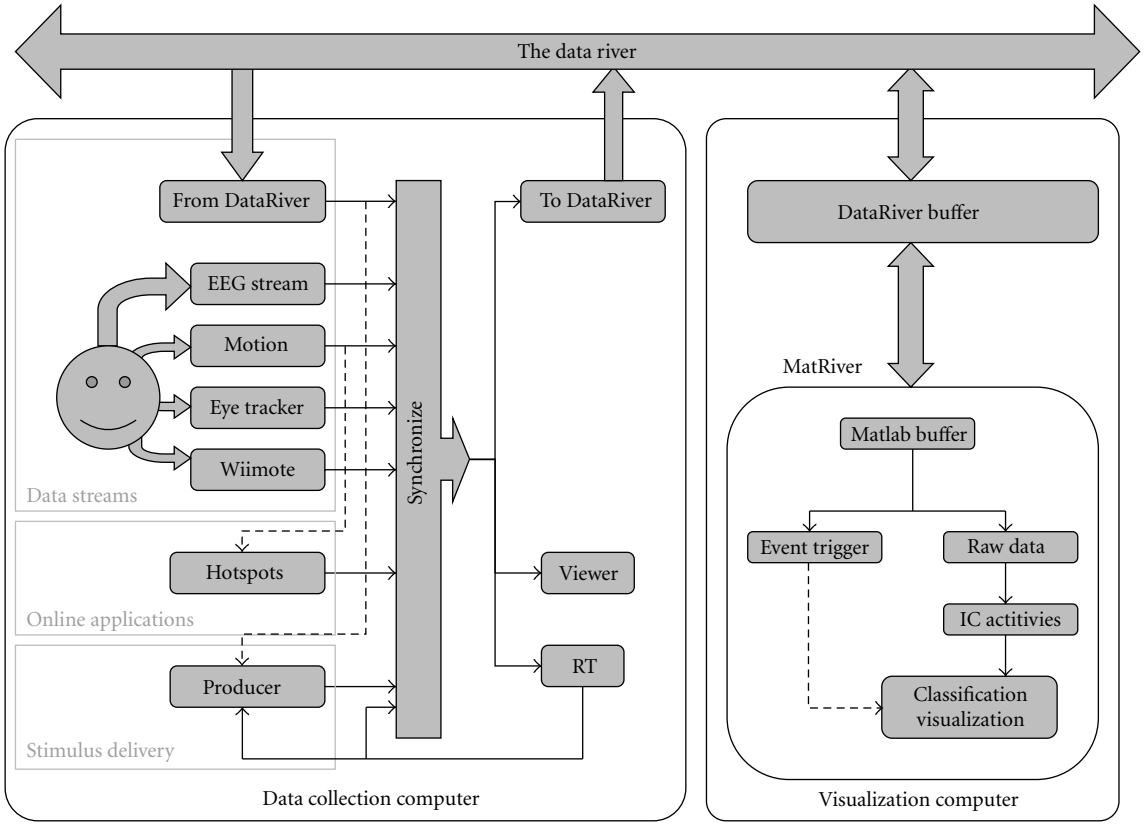


Figure 5: An ERICA data flow involving two separate computers each running an instance of the DataRiver application. Dashed lines indicate control signals. Here, computer visualization is performed using the Matlab DataRiver client MatRiver.

be derived post hoc from recorded data, for example for statistical analysis of the model's prediction accuracy when applied to a database of previously recorded data. BCILAB is highly flexible and most accessible cognitive states can be investigated, for example imagined movements (affecting in sensorimotor mu rhythms), surprise (provoking, e.g., the oddball P3), or indicators of drowsiness.

The tools provided by BCILAB facilitate most steps in BCI research, including the design, implementation, learning, evaluation, and on- or off-line application of BCI (or other) models. Further tasks, including the exploration of recorded data and visualization of model parameters may be supported using EEGLAB tools. BCILAB has several layers, the top layer including a graphic interface, a scripting interface, and a real-time application interface, with a second layer including core model learning, model execution, and model evaluation functions. These core facilities in turn rely on a framework of "BCI paradigms", which can be understood as prototypical template-like approaches to designing a BCI model. Pre-defined paradigms include common spatial patterns (CSP), logarithmic band-power estimates, and the approach proposed in the dual augmented lagrange framework [25]. A BCI "paradigm" defines the entire approach as it would be described in a publication, from raw input data to final output, and usually involves both a learning and a prediction stage, because sufficient performance can often only be achieved after a model is learned (or calibrated) based on sample data from a given session, subject, or task. BCI paradigms can be fully customized by the user, including removal or addition of entire components, but come with defaults for all their parameters, both to keep the learning curve gentle as well as to minimize the amount of information that must be specified.

At lower levels, BCILAB provides additional frameworks designed to be extensible and flexible and to have low implementation overhead. In particular, most BCI paradigms are defined within a "data flow" scheme wherein information is passed through several stages that are themselves plug-in frameworks: filters (signal processing), feature maps (feature extraction), and model learners as well as predictors/estimators (using machine learning). These frameworks are general enough to cover a wealth of implementations, such as adaptive/statistical epoched-signal processing, adaptive feature extraction, and classification/regression/density estimation, with general (discrete/continuous, multivariate, point-estimate/full-posterior) outputs. We are currently working to explore additional concepts including hierarchical Bayesian models spanning sessions, subjects and (related) tasks.

A simple use case of BCILAB is for the offline reanalysis of a BCI study. For example, given a collection of data sets, one per subject, containing imagined movements of either the left or the right hand in random order, with events "SL"

Table 2: Signal processing, feature extraction, and machine learning algorithms included in the BCILAB/EEGLAB framework.

Signal processing	Feature extraction	Machine learning algorithms
(i) Channel selection	(i) Multiwindow averages [26, 27]	(i) Linear discriminant Analysis (LDA) [28]
(ii) Resampling	(ii) Common Spatial Patterns (CSP) [29]	(ii) Quadratic discriminant analysis (QDA) [30]
(iii) Artifact rejection (spike detection, bad window detection, bad channel detection, local peak detection)	(iii) Spectrally-weighted common spatial patterns [31]	(iii) Regularized and analytically regularized LDA and QDA [30, 32]
(iv) Envelope extraction	(iv) Adaptive autoregressive modeling, from BioSig [33]	(iv) Linear SVM [34] (LIBLINEAR/CVX)
(v) Epoch extraction	(1) Dual-agumented lagrange (DAL) [25]	(v) Kernel SVM [34]
(1) Time-frequency window selection	(2) Frequency-domain DAL (FDAL)	(vi) Gaussian mixture models (GMM), 9 methods [35–37])
(2) Spectral transformation	(3) Independent Modulators [38]	(vii) Regularized and variational Bayesian logistic regression and sparse Bayesian logistic regression [39, 40]
(vi) Baseline filtering	(4) Multiband-CSP [41]	(1) Hierarchical kernel learning [42]
(vii) Resampling	(5) Multi-Model Independent component features	(viii) Relevance vector machines (RVM) [43]
(viii) Re-referencing		(1) group-sparse/rank-sparse linear and logistic regression [25]
(ix) Surface Laplacian filtering [44]		(2) high-dimensional Gaussian Bayes density estimator/classifier
(x) ICA methods (Infomax, FastICA, AMICA) [6, 45]		(3) Voting metalearner
(xi) Spectral filters (FIR, IIR)		
(xii) Spherical spline interpolation [46]		
(1) Signal normalization		
(2) Sparse signal reconstruction (NESTA, SBL [47], FOCUSS, l1; currently offline only)		
(3) Linear projection		

and "SR" indicating the timing and type of the respective cue stimuli, a user of the Matlab-based BCILAB scripting interface may proceed as follows: For each subject,

(1) Load a data set

```
eeg = io_loadset('session1.eeg');
```

(2) Define an analysis approach (customizing parts of a standard paradigm)

```
approach = {'SpecCSP', 'events',
{'SL','SR' }, learner', 'logreg'};
```

(3) Apply the approach to the data, to get an estimate of its performance on the given data

```
[performance,model,statistics] =
bci_train({'data',eeg,...'approach',
approach});
```

This analysis gives the prediction accuracy results that are the key ingredient of most BCI publications (along with visualizations). Step (3) above also produces a calibrated predictive model which can be loaded into one of the provided real-time plug-ins (for ERICA, BCI2000 [48], and OpenViBE [49] real-time environments, with others forthcoming) for online testing.

A major focus of the BCILAB toolbox is to allow, as much as possible, that competitive BCI estimation performance may be obtained using simply stated procedures (as above). For this purpose, a large collection of state-of-the-art methods have been provided and are listed in Table 2. A second, complementary focus is to provide rigorous analyses (e.g., for performance estimation) by default. For this purpose, a framework for automated cross-validation, systematic parameter search, and nested cross-validation is provided,

and a suitable evaluation method is automatically chosen depending on the supplied data (though the evaluation method may also be customized). For example, if a single data set and at least one unknown parameter is provided by the user, nested block-wise cross-validation with safety margins is chosen by default. In a similar vein, to rule out common BCI research errors such as accidental non-causal signal processing, offline and online processing uses identical code.

BCILAB aims to be not just a collection of off-the-shelf tools to enable BCI experiments, but is designed to be a development platform for new BCI technology, facilitating the creation of new methods, approaches (e.g., combining existing methods), and paradigms. For this purpose, the toolbox provides extensive infrastructure, including, among others, the frameworks mentioned above, a small Mathematica-inspired symbolic expression system, an Adobe ASL-inspired declarative graphic interface property model, a decentralized distributed computing infrastructure (not dependent on MATLAB toolboxes), a generic dependency loader, a transparent multi-level cache for results, as well as bundled toolboxes for convenience. All BCILAB code is thoroughly documented, with additional citation-rich documentation for user-facing functions. Backwards compatibility to MATLAB 7.1 is attempted (and reached for most functionality except the graphic interface, which requires Matlab 2008a+, due to the use of objects).

8. Conclusion

The extended SCCN software suite centered on EEGLAB data structures and processing functions is an ongoing product of a coordinated effort to develop and test new methods for observing and modeling the dynamics of noninvasively observed electrophysiological activity in human cortex during a wide range of behavioral task performance, both *post hoc* and in real time. The tools we have developed towards this end include software for online data streaming and storage, advanced offline and online EEG analysis and prediction, source localization, and multivariate connectivity analysis and visualization. These build on and integrate with our well-established EEGLAB software suite that is now in use by thousands of researchers around the world. We plan to continue to extend and further coordinate these modular toolboxes with the hope that they will facilitate development of novel 21st century EEG analysis and data mining techniques which in turn will lead to transformative gains in our understanding of human neuroscience, cognition and behavior, facilitating a broad range of practical and clinical applications.

Contributions and Acknowledgments

EEGLAB was mainly developed by A. Delorme and S. Makeig [2] from the ICA electrophysiology toolbox (1997–2001) of Makeig et al., with functions and design input from many dozens of colleagues and EEGLAB users. The neuroelectromagnetic forward head modeling toolbox (NFT) was developed by Z. Akalin Acar [3]. SIFT (source information flow toolbox) was developed by T. Mullen. BCILAB has been developed by C. Kothe inspired by the preceding PhyPA BCI toolbox created by C. Kothe and T. Zander at the Berlin Technical University. The ERICA framework was mainly developed by A. Vankov and its Matlab elements by N. Bigdely-Shamlo. Gifts from The Swartz Foundation (Old Field NY), and grant support from the National Institutes of Health (USA), the National Science Foundation (USA), the Office of Naval Research (US), and the Army Research Laboratories (US) are gratefully acknowledged.

References

[1] S. Makeig, S. Debener, J. Onton, and A. Delorme, "Mining event-related brain dynamics," *Trends in Cognitive Sciences*, vol. 8, no. 5, pp. 204–210, 2004.

[2] A. Delorme and S. Makeig, "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis," *Journal of Neuroscience Methods*, vol. 134, no. 1, pp. 9–21, 2004.

[3] Z. A. Acar and S. Makeig, "Neuroelectromagnetic forward head modeling toolbox," *Journal of Neuroscience Methods*, vol. 190, no. 2, pp. 258–270, 2010.

[4] A. Delorme, "Matlab tools for BCI research?" in *Human-Computer Interaction and Brain-Computer Interfaces*, D. Tan and A. Nijholt, Eds., Springer, 2009.

[5] S. Makeig, A. Delorme, and J. Grethe, "HeadIT: a human electrophysiology data and integrated tools resource," in *Society for Neuroscience*, San Diego, Calif, USA, 2010.

[6] S. Makeig et al., "Independent component analysis of electroencephalographic data," in *Advances in Neural Information Processing Systems*, D. Touretzky, M. Mozer, and M. Hasselmo, Eds., pp. 145–151, 1996.

[7] S. Makeig and M. Inlow, "Lapses in alertness: coherence of fluctuations in performance and EEg spectrum," *Electroencephalography and Clinical Neurophysiology*, vol. 86, no. 1, pp. 23–35, 1993.

[8] S. Makeig, M. Westerfield, T. P. Jung et al., "Dynamic brain sources of visual evoked responses," *Science*, vol. 295, no. 5555, pp. 690–694, 2002.

[9] Y. Benjamini and D. Yekutieli, "The control of the false discovery rate in multiple testing under dependency," *Annals of Statistics*, vol. 29, no. 4, pp. 1165–1188, 2001.

[10] J. Onton, A. Delorme, and S. Makeig, "Frontal midline EEG dynamics during working memory," *NeuroImage*, vol. 27, no. 2, pp. 341–356, 2005.

[11] A. Delorme, M. Westerfield, and S. Makeig, "Medial prefrontal theta bursts precede rapid motor responses during visual selective attention," *Journal of Neuroscience*, vol. 27, no. 44, pp. 11949–11959, 2007.

[12] H. Si, "Adaptive tetrahedral Mesh generation by constrained Delaunay refinement," *International Journal for Numerical Methods in Engineering*, vol. 75, no. 7, pp. 856–880, 2008.

[13] Z. Akalin-Acar and N. G. Gencer, "An advanced boundary element method (BEM) implementation for the forward problem of electromagnetic source imaging," *Physics in Medicine and Biology*, vol. 49, no. 21, pp. 5011–5028, 2004.

[14] N. G. Gencer and C. E. Acar, "Sensitivity of EEG and MEG measurements to tissue conductivity," *Physics in Medicine and Biology*, vol. 49, no. 5, pp. 701–717, 2004.

[15] A. M. Dale, B. Fischl, and M. I. Sereno, "Cortical surface-based analysis—I. Segmentation and surface reconstruction," *NeuroImage*, vol. 9, no. 2, pp. 179–194, 1999.

[16] Z. Akalin Acar, G. Worrell, and S. Makeig, "Patch-based cortical source localization in epilepsy," in *Proceedings of the IEEE EMBC*, Minneapolis, Minn, USA, 2009.

[17] S. L. Bressler and A. K. Seth, "Wiener-Granger Causality: a well established methodology," *Neuroimage*. In press.

[18] C. Granger, "Investigating causal relations by econometric models and cross-spectral methods," *Econometrica*, vol. 37, pp. 424–438, 1969.

[19] M. Kaminski, "Multichannel data analysis in biomedical research," in *Understanding Complex Systems*, V. K. Jirsa and A. R. McIntosh, Eds., Handbook of Brain Connectivity series, pp. 327–355, Springer, Berlin, Germany, 2007.

[20] M. Ding, S. L. Bressler, W. Yang, and H. Liang, "Short-window spectral analysis of cortical event-related potentials by adaptive multivariate autoregressive modeling: data preprocessing, model validation, and variability assessment," *Biological Cybernetics*, vol. 83, no. 1, pp. 35–45, 2000.

[21] J. Theiler, S. Eubank, A. Longtin, B. Galdrikian, and J. Doyne Farmer, "Testing for nonlinearity in time series: the method of surrogate data," *Physica D*, vol. 58, no. 1–4, pp. 77–94, 1992.

[22] A. Vankov, Adapt © 1987–2003 and Varieté , © 2000, 2001 are property of EEG Solutions LLC, and are used under free license for scientific non-profit research, 1987.

[23] S. Makeig, K. Gramann, T. P. Jung, T. J. Sejnowski, and H. Poizner, "Linking brain, mind and behavior," *International Journal of Psychophysiology*, vol. 73, no. 2, pp. 95–100, 2009.

[24] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan, "Brain-computer interfaces for communication and control," *Clinical Neurophysiology*, vol. 113, no. 6, pp. 767–791, 2002.

[25] R. Tomioka and K. R. Müller, "A regularized discriminative framework for EEG analysis with application to brain-computer interface," *NeuroImage*, vol. 49, no. 1, pp. 415–432, 2010.

[26] B. Blankertz, G. Curio, and K. Müller, "Classifying single trial EEG: towards brain computer interfacing," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS '01)*, T. Diettrich, S. Becker, and Z. Ghahramani, Eds., pp. 157–164, 2002.

[27] B. Blankertz, C. Schäfer, G. Dornhege, and G. Curio, "Single trial detection of EEG error potentials: A tool for increasing BCI transmission rates," in *Proceedings of the Artificial Neural Networks (ICANN '02)*, 2002.

[28] R. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of Eugenics*, vol. 7, pp. 179–188, 1936.

[29] H. Ramoser, J. Müller-Gerking, and G. Pfurtscheller, "Optimal spatial filtering of single trial EEG during imagined hand movement," *IEEE Transactions on Rehabilitation Engineering*, vol. 8, no. 4, pp. 441–446, 2000.

[30] J. Friedman, "Regularized discriminant analysis," *Journal of the American Statistical Association*, vol. 84, no. 405, pp. 165–175, 1989.

[31] R. Tomioka, G. Dornhege, K. Aihara, and K. R. Müller, "An iterative algorithm for spatio-temporal filter optimization," in *Proceedings of the 3rd International BCI Workshop and Training Course*, Verlag der Technischen Universität Graz, Graz, Austria, 2006.

[32] O. Ledoit and M. Wolf, "A well-conditioned estimator for large-dimensional covariance matrices," *Journal of Multivariate Analysis*, vol. 88, no. 2, pp. 365–411, 2004.

[33] A. Schlögl, *The Electroencephalogram and the Adaptive Autoregressive Model: Theory and Applications*, Shaker, Aachen, Germany, 2000.

[34] B. Schölkopf and A. Smola, *Learning with Kernels*, MIT Press, Cambridge, Mass, USA, 2002.

[35] J. Bilmes, *Gentle Tutorial of the EM Algorithm and Its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models*, International Computer Science Institute, 1998.

[36] N. Vlassis and A. Likas, "A greedy EM algorithm for Gaussian mixture learning," in *Neural Processing Letters*, vol. 15, Kluwer Academic Publishers, 2002.

[37] M. A. T. Figueiredo and A. K. Jain, "Unsupervised learning of finite mixture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 3, pp. 381–396, 2002.

[38] J. Onton and S. Makeig, "High-frequency broadband modulations of electroencephalographic spectra," *Frontiers in Neuroscience*, vol. 159, pp. 99–120, 2009.

[39] T. Jaakkola and M. Jordan, "A variational approach to bayesian logistic regression models and their extensions," in *Proceedings of the 6th International Workshop on Artificial Intelligence and Statistics*, 1997.

[40] D. Wipf, S. Nagarajan, J. Platt, D. Koller, Y. Singer, and S. Roweis, *A New View of Automatic Relevance Determination*, MIT Press, 2008.

[41] K. K. Ang, Z. Y. Chin, H. Zhang, and C. Guan, "Filter Bank Common Spatial Pattern (FBCSP) in brain-computer interface," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '08)*, pp. 2390–2397, June 2008.

[42] F. Bach, "Exploring large feature spaces with hierarchical multiple Kernel learning," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS '08)*, 2008.

[43] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *Journal of Machine Learning Research*, vol. 1, no. 3, pp. 211–244, 2001.

[44] F. Babiloni, C. Babiloni, L. Fattorini, F. Carducci, P. Onorati, and A. Urbano, "Performances of surface Laplacian estimators: a study of simulated and real scalp potential distributions," *Brain Topography*, vol. 8, no. 1, pp. 35–45, 1995.

[45] J. A. Palmer, K. Kreutz-Delgado, B. D. Rao, and S. Makeig, "Modeling and estimation of dependent subspaces with non-radially symmetric and skewed densities," in *Proceedings of the 7th International Conference on Independent Component Analysis and Signal Separation*, London, UK, 2007.

[46] F. Perrin, J. Pernier, and O. Bertrand, "Mapping of scalp potentials by surface spline interpolation," *Electroencephalography and Clinical Neurophysiology*, vol. 66, no. 1, pp. 75–81, 1987.

[47] D. P. Wipf, J. P. Owen, H. T. Attias, K. Sekihara, and S. S. Nagarajan, "Robust Bayesian estimation of the location, orientation, and time course of multiple correlated neural sources using MEG," *NeuroImage*, vol. 49, no. 1, pp. 641–655, 2010.

[48] G. Schalk, D. J. McFarland, T. Hinterberger, N. Birbaumer, and J. R. Wolpaw, "BCI2000: a general-purpose brain-computer interface (BCI) system," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 6, pp. 1034–1043, 2004.

[49] Y. Renard, F. Lotte, G. Gibert et al., "OpenViBE: an open-source software platform to design, test, and use brain-computer interfaces in real and virtual environments," *Presence*, vol. 19, no. 1, pp. 35–53, 2010.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 158970, 11 pages
doi:10.1155/2011/158970

Research Article

ELAN: A Software Package for Analysis and Visualization of MEG, EEG, and LFP Signals

Pierre-Emmanuel Aguera,[1,2] Karim Jerbi,[1,2] Anne Caclin,[1,2] and Olivier Bertrand[1,2]

[1] *INSERM U1028, CNRS UMR5292, Lyon Neuroscience Research Center, Brain Dynamics and Cognition Team, Centre Hospitalier Le Vinatier, Bâtiment 452, 95 Boulevard Pinel, 69500 Bron, France*
[2] *University Lyon 1, 69000 Lyon, France*

Correspondence should be addressed to Pierre-Emmanuel Aguera, pe.aguera@inserm.fr

Received 4 October 2010; Revised 13 December 2010; Accepted 27 January 2011

Academic Editor: Sylvain Baillet

The recent surge in computational power has led to extensive methodological developments and advanced signal processing techniques that play a pivotal role in neuroscience. In particular, the field of brain signal analysis has witnessed a strong trend towards multidimensional analysis of large data sets, for example, single-trial time-frequency analysis of high spatiotemporal resolution recordings. Here, we describe the freely available ELAN software package which provides a wide range of signal analysis tools for electrophysiological data including scalp electroencephalography (EEG), magnetoencephalography (MEG), intracranial EEG, and local field potentials (LFPs). The ELAN toolbox is based on 25 years of methodological developments at the Brain Dynamics and Cognition Laboratory in Lyon and was used in many papers including the very first studies of time-frequency analysis of EEG data exploring evoked and induced oscillatory activities in humans. This paper provides an overview of the concepts and functionalities of ELAN, highlights its specificities, and describes its complementarity and interoperability with other toolboxes.

1. Introduction

The unprecedented increase in computational power over the last 20 years has led to extensive developments of advanced data processing and visualization methods. The steady stream of novel methods that continue to flourish in the field of electrophysiological data processing has substantial implications in numerous fields including basic neurophysiology, cognitive brain research, and clinical neuroscience. Indeed, in recent years, noninvasive recordings techniques such as electroencephalography (EEG) or magnetoencephalography (MEG), and invasive methods, such as intracranial EEG (iEEG) or microelectrode recordings (measuring local field potentials, LFPs) increasingly rely on sophisticated analysis of high-resolution data sets, across multiple spatial, temporal, and spectral dimensions. Yet, beyond the attractive perspective of applying emerging methods to one's data, many researchers and clinicians face the question of method development and software selection. In this context, sharing existing methods via freely available software packages and toolboxes is poised to play an important role, especially when combined with a user-friendly environment and computationally efficient code.

The ELAN software package provides a wide range of preprocessing and signal analysis tools for many types of electrophysiological data. Data types typically analyzed with ELAN includes macroscopic surface-level recordings such as scalp electroencephalography (EEG) or magnetoencephalography (MEG), invasive recordings such as intracranial EEG (iEEG) in humans (electrocorticographic, ECoG, and stereotactic EEG) as well as local field potentials (LFPs) in animals. In addition to many basic signal processing tools for electrophysiological data, the main features of ELAN include topographical mapping, time-frequency analysis, along with adapted statistical routines, and interactive user-friendly visualization tools to navigate in the data sets, all implemented in C for fast computations.

The first version of the ELAN toolbox was released in 2001, integrating 15 years of methodological developments at the INSERM Brain Dynamics and Cognition laboratory in Lyon and has served as the standard in-house software for the analysis of all types of electrophysiological data. So far ELAN has been used by direct collaborators and was licensed to approximately 20 national and international research labs. To date, the methods incorporated in ELAN have been used and reported in over 100 publications across various fields in neuroscience and electrophysiology. With the publication of this paper, the ELAN software package will become available for free download. The developers of ELAN aim to create a community of users and a knowledge base for exchange of expertise and continued development and improvement of the ELAN software package in the future.

The goal of this paper is to provide an overview of the concepts and functionalities that build the structural backbone and strengths of ELAN. We also highlight the specificities of this software and describe its complementarity and interoperability with other existing toolboxes. Up-to-date information about downloads, user guide, tutorials and additional resources are available online at the ELAN website (http://elan.lyon.inserm.fr/).

2. ELAN Software: Concepts and Functionalities

In this section we describe the main features of the ELAN package including system requirements, overall data workflow, basic ELAN analysis tools and visualization modules, as well as compatibility and interoperability issues.

2.1. System Requirements. ELAN is available for 32-bit and 64-bit Linux operating systems (Debian, Fedora, and OpenSuse distributions are currently supported and therefore most of their derivative implementations as well, including Ubuntu and CentOS). Mac and Windows users can also run ELAN on their system by first installing a virtual machine that runs a Linux guest in their preferred host (examples of such virtualization software include VirtualBox and VMWare). The minimal RAM memory requirement is 128 MB, but 4 GB is highly recommended. The ELAN toolbox can import data from many commercially available electrophysiological data acquisition devices, including research and clinical systems. These include: Alpha Omega AlphaMap (Alpha Omega, Nazareth, Israel), Wavemetrics IGOR Pro (WaveMetrics Inc., Lake Oswego, OR, USA), Biosemi (BioSemi, Amsterdam, The Netherlands), Brain Products Recorder (BrainProducts GmbH, Munich, Germany), CTF Inc. (VSM Med Tech Ltd., Coquitlam, BC, Canada), Elekta/Neuromag (Helsinki, Finland), EGI Geodesic (Electrical Geodesics Inc., Eugene, OR, USA), InstEP System (InstEP systems, Ottawa, Canada), Micromed (Micromed, Treviso, Italy), Neuroscan (Neuroscan Inc., VA, USA). In addition, beyond standard data formats from hardware devices, ELAN can also import data by conversion from many standard file format such as ASCII (text), MATLAB (mat file), EDF (European Data Format), GDF (General Data Format for Biosignals). New data conversion and import tools are constantly added to the ELAN package.

2.2. ELAN's Workflow. ELAN consists of compiled C programs that make up a bundle of data analysis and visualization functions for various electrophysiological data including EEG, MEG, iEEG, LFP, reconstructed source time series, or any type of continuous signals. All ELAN functions can be run in command-line or batch-mode. Note that some routines (e.g., interfacing with EEGlab ICA routines) are implemented and provided as MATLAB functions (.m files). In the following, we describe the main data formats and visualization tools available in ELAN.

2.2.1. Data and Event File Types. ELAN uses three main data formats: continuous data (.eeg, see Figure 1), event-related data (.p, see Figure 2), and time-frequency data (.tf, see Figure 3). First, the continuous data format (.eeg) stores any continuous data with event codes corresponding to stimulations and behavioral events such as experiment triggers and subject's responses. The continuous data can be EEG, MEG, iEEG, LFP (for examples of animal datasets analysis, see [1, 2]), reconstructed source time series, ICA components, EEG Laplacians (see Section 2.3.2 below), and so forth. Second, ELAN event-related data format (.p) typically stores data that has been averaged with respect to a specific event. However, this same format can also be used to process temporal or frequency profiles of various types. Finally, the time-frequency (.tf) data format files store the output of all time-frequency analysis functions available in ELAN.

In addition to the three main data types, ELAN uses two other types of files: event files and parameter files. Event files are in standard text format (.pos) and store all information related to the events (e.g., triggers, responses, etc.) that occur in a given continuous data set (.eeg). The event file can easily be modified in order to reject, group, recode, or create new events.

Furthermore, a typical call to an ELAN function requires the specification of a number of parameters. These computing parameters are stored and passed to the function via a parameter text file (.par). A parameter file may, for example, contain flags indicating which channels to use, time window parameters (with respect to specified events), parameters for the time-frequency analysis or for the statistical analysis, and so forth.

An important feature of the ELAN toolbox is that the result of any analysis performed with an ELAN function is always stored in one of the three main data formats (.eeg,.p, or.tf), so that it can still be processed and visualized using ELAN tools. For example, the time profile of a particular frequency band in a time-frequency file (.tf) can be extracted and then visualized and/or analyzed as an event-related data file (.p). Similarly, reconstructed time-courses of neural generators obtained after source modeling with another software and converted to an ELAN file format can be

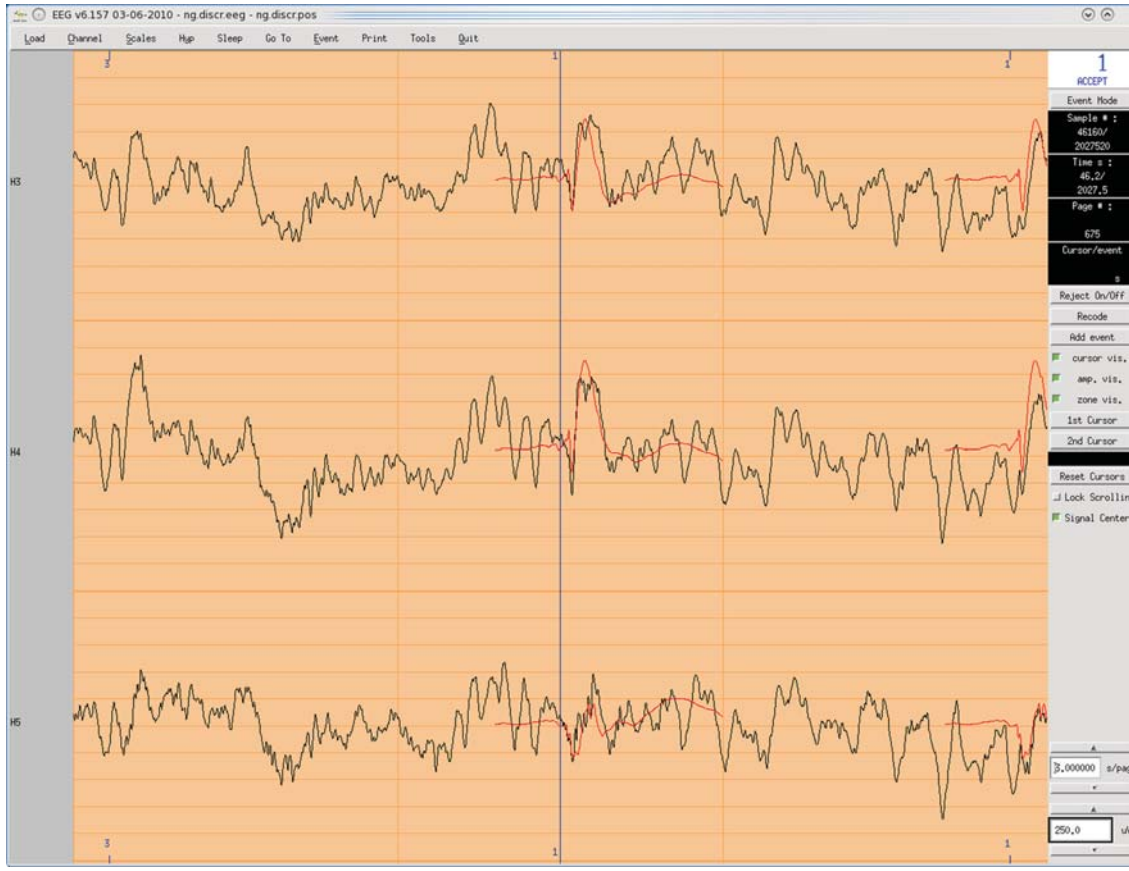


Figure 1: *EEG* vizualisation tool for continuous data (.eeg file), with an example of intracranial auditory EEG signals. X-axis: time, Y-axis: signal amplitude, one row per channel. Events are displayed above and below the continuous EEG signals, and the averaged event-related response is superimposed in red. Various plotting tools and online analysis functions are available through the menus (top and right of the window).

analyzed in the time-frequency domain just as original scalp recordings.

2.2.2. Data Visualization. ELAN contains three distinct data visualization graphical user interfaces (GUIs). In principle data from each one of the three ELAN file formats (see above) is visualized with its dedicated tool. The *EEG* tool allows for the visualization of continuous data (Figure 1), *ERPA* is the tool used for event-related data files (Figure 2), while *TFVIZ* is the tool used to display the results of the time-frequency analysis (Figure 3). These three viewers share numerous common functions, such as curve plotting, cursor-based measures, topographical mapping, and so forth.

2.3. Overview of ELAN Tools

2.3.1. Data Preprocessing. A number of ELAN tools allows for preprocessing of continuous (.eeg) data, typically performed just after importation of raw data and prior to the main analysis, to handle artifacts, filter the data, organize event codes, or even score sleep stages.

Artifact Rejection. Artifacts can either be manually detected and rejected through the *EEG* visualization tool, or automatically detected and rejected, using either a fixed threshold that should not be exceeded during a trial (e.g., $\pm 75\,\mu$V), or a range of variation that should not be exceeded within a time window (e.g., 2000 fT of variation within 500 ms). Thresholds can be adjusted separately for each channel. Whenever the threshold is exceeded in at least one channel, the full trial is rejected. Information regarding whether a given trial has been kept or rejected is stored in the event file (.pos) associated with the continuous data file (.eeg) and can be visualized with *EEG* (Figure 4).

Artifact Correction. Ocular artifacts can be corrected for instance using ICA. This is currently possible via a provided MATLAB function that applies ICA functions to the ELAN data file using the freely available EEGLab toolbox [3]. In addition, data from bad or missing channels can be reconstructed by interpolation using spline functions.

Data Filtering. Continuous and concatenated epoched data (.eeg), as well as event-related data (.p), can be filtered using dedicated ELAN functions that apply Butterworth low-pass, high-pass, band-pass, or stop-band filters. In addition, any other type of filter can be applied if the user can supply its coefficients (computed with external tools such as MATLAB, Octave, Scilab, or SciPy).

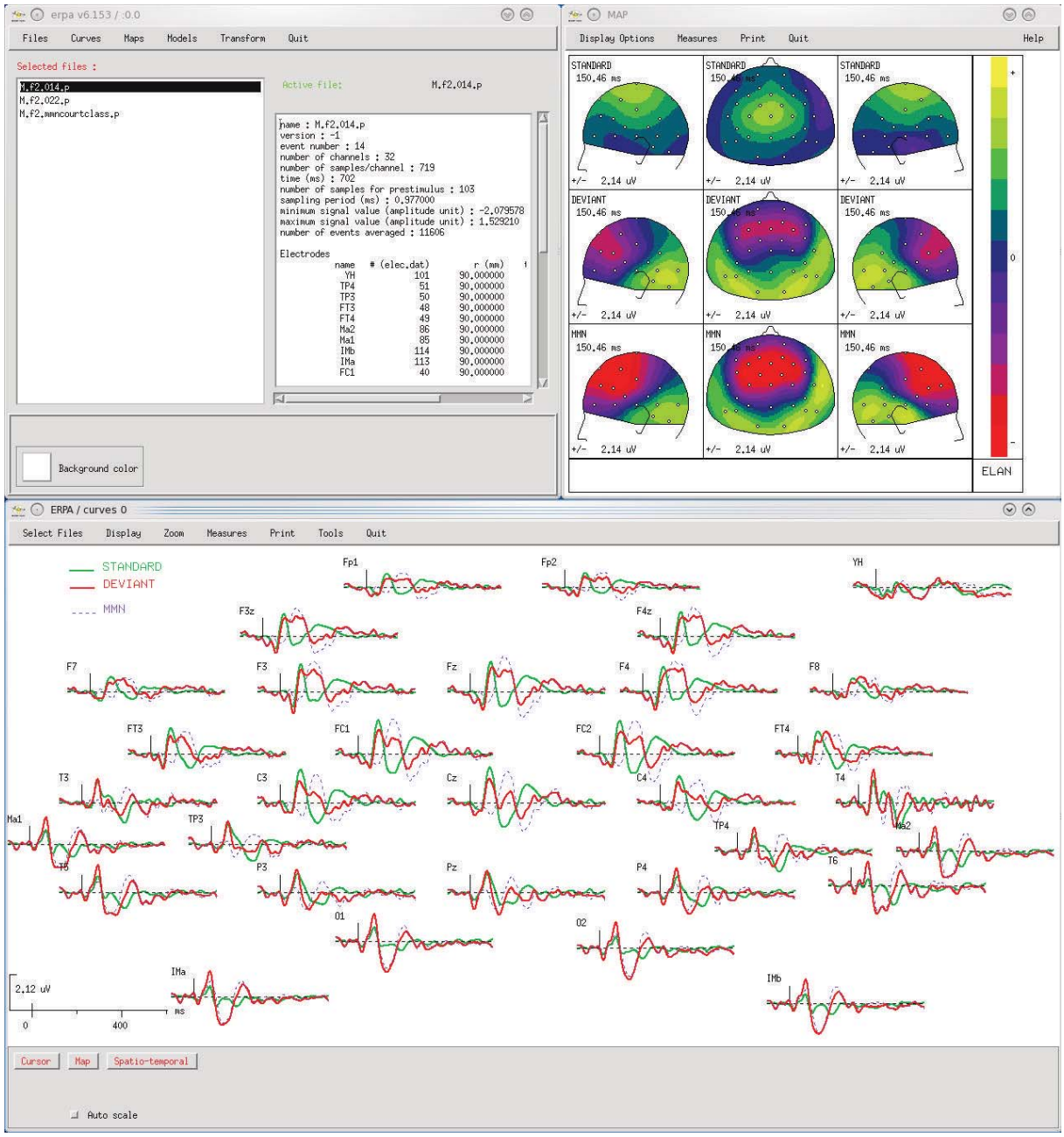


Figure 2: *ERPA* visualization tool for averaged signals (.p file), with an example of scalp auditory ERPs (MMN protocol). Top left window: main ERPA window to select files and launch visualization tools. Bottom: time courses of event-related responses (positivity is up). Top right: mapping of event-related responses, here with several views of three different ERPs at the same latency. In the windows where data are visualized, plotting functions and tools for on-the-fly signal processing or measurements are available through the menus at the top.

Sleep Stage Scoring. For polysomnographic data (containing EEG, EMG, and EOG signals), ELAN allows for a convenient manual scoring of sleep stages, implemented within the *EEG* visualization tool resulting in a hypnogram file (.hyp, text file). The hypnogram can then be combined with an event file (.pos) using a MATLAB function, in order to analyze events depending on the sleep stage (e.g.,[4]).

Event File (.pos) Preprocessing. Prior to data averaging, a number of simple operations can be performed on event files (.pos) using ELAN functions, for example to change or regroup event codes, compute reaction times, and so forth. Furthermore, event files are simple text file that can easily be edited and processed, so that more sophisticated event preprocessing can be performed outside of ELAN, for example, using MATLAB or simply a spreadsheet.

2.3.2. Time-Domain Analysis and Topographical Mapping. Event-Related Averaging and Data Analysis. Continuous

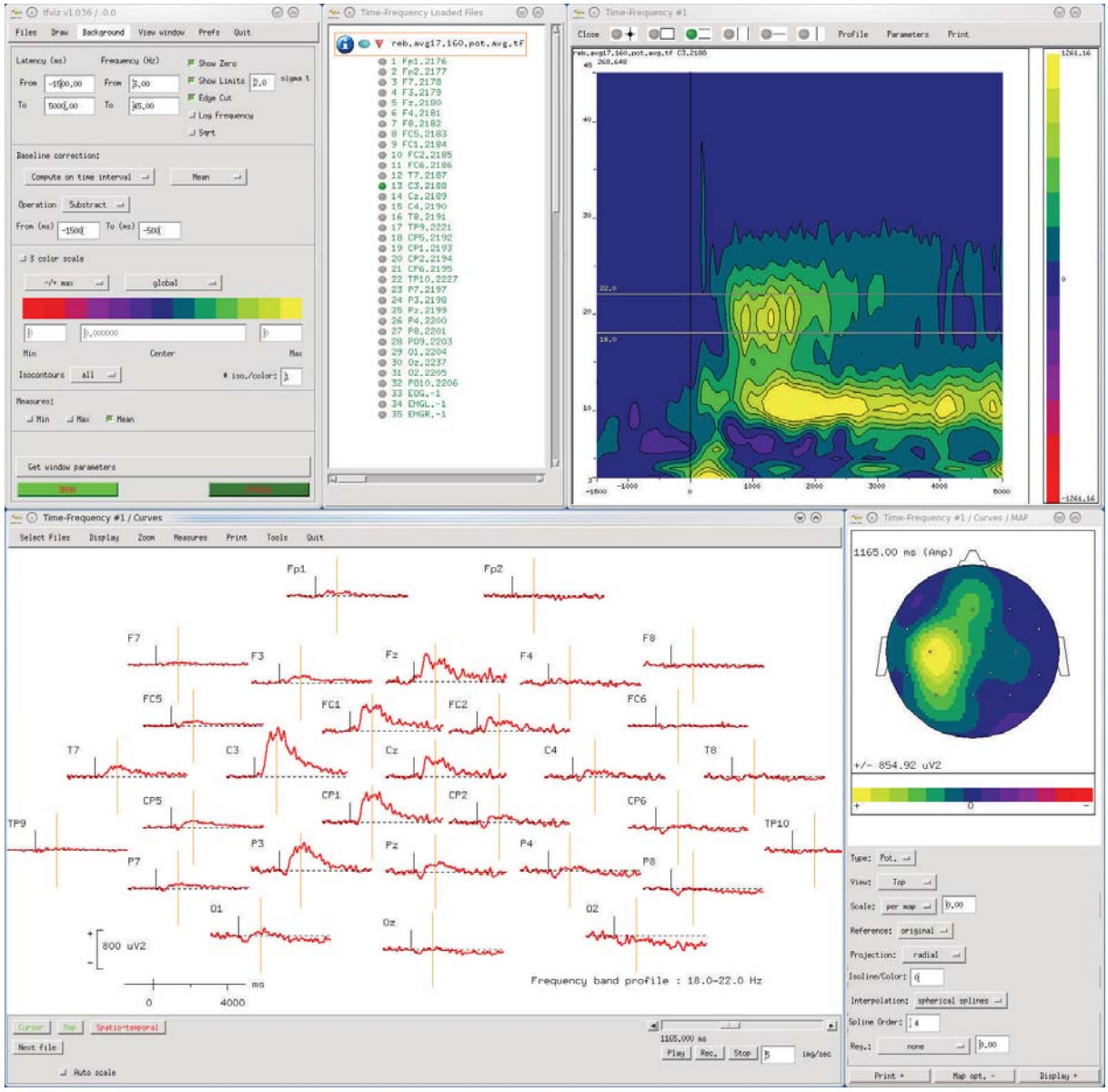


Figure 3: *TFVIZ* visualization tool for time-frequency data (.tf file), with an example of scalp EEG in a motor finger tapping protocol. Top left: main *TFVIZ* window, to select files, set vizualisation parameters, apply baseline correction, and so forth. Top center: window to select the channels to visualize. Top right: time-frequency plot, here for one channel in one TF file (X-axis: time, Y-axis: frequency, amplitude is color-coded). Bottom left: visualization of the time-course of signal amplitude in one frequency band defined by two cursors on the TF plot (this display offers the same functionalities as in *ERPA*, see Figure 2). Bottom right: topography for one frequency band at the latency defined by the cursor on the curves. This figure illustrates the possibility to navigate with *TFVIZ* in this multidimensional data set (in the frequency, time, and spatial dimensions).

(possibly preprocessed) data (.eeg) are averaged according to event positions stored in an event file (.pos), resulting in an event-related data file (.p) for each event code, which can be visualized using *ERPA*. Baseline correction can be performed by subtracting the average signal in a time-window either relative to the event of interest or relative to another event (e.g., averaging can be done relative to the subject's response, using a baseline taken prior to stimulus onset). Numerous operations can be performed on event-related data files (.p) using ELAN functions: filtering, (see Section 2.3.1), subtraction of two files, average of several files to obtain, for example, the grand average across subjects, and so forth. When visualizing the event-related time-courses with *ERPA*, a variety of amplitude and latency measurements can be performed and saved as text files for further analysis with a statistics software.

Topographical Mapping and Scalp Current Density Analysis of Event-Related Signals. The topography of event-related EEG or MEG data (ERP, ERF) or other topographical data (ICA components, etc.) is computed through spherical spline

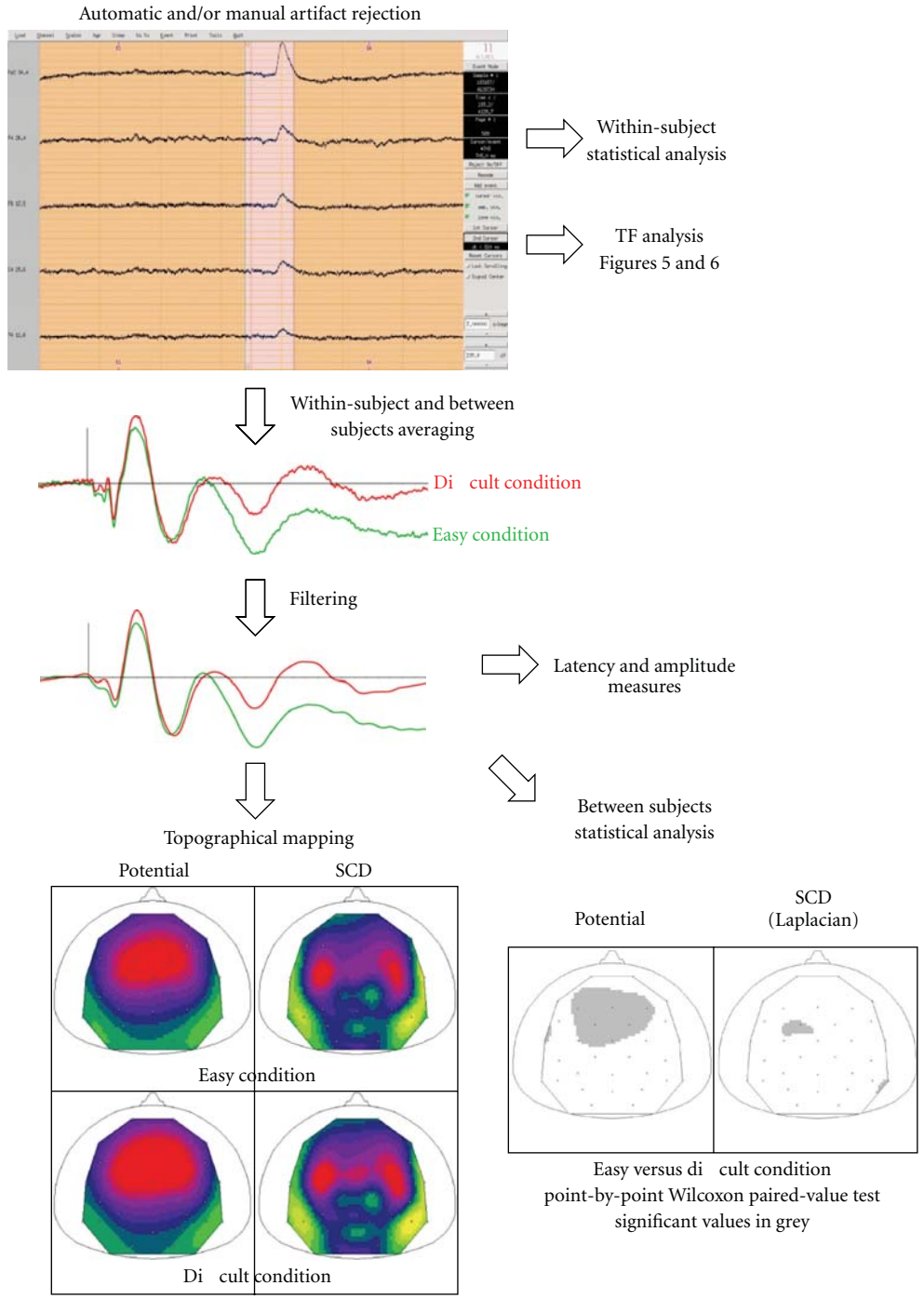


Figure 4: Overview of ELAN analysis workflow, with an example of scalp auditory EEG data. Artifact rejection is done automatically or manually and visualized with *EEG* (top). Signal averaging is then performed within and across subjects (middle), with a possibility to filter the data. Topography (including SCDs) can then be assessed with *ERPA* (bottom left, example of auditory N1). Note that additional frontal current sinks are identified more easily with SCDs than with potential maps. Bottom right: statistical map obtained by comparing the amplitude of the responses in two conditions across subjects at the latency of the auditory N1 (adapted from [5]).

interpolation [7] for each time point (or each component) and visualized with *ERPA* using orthogonal or radial projections. ERP mapping is done on the reference sphere of the 10–20 system. To further analyze the topography of ERP components, Scalp Current Densities (SCDs), corresponding to the surface Laplacian of the potential distribution, can be computed [8–10] and visualized with *ERPA*. SCD maps have several advantages over potential maps: they allow dissociating more easily multiple generators (see Figure 4) and are reference-free. They favor superficial generators

relative to deep ones. In ELAN, SCDs can be computed on event-related data (.p) or continuous data (.eeg). In the latter case, they can be used as a preprocessing step before time-frequency analysis to dissociate the generators of the oscillatory response (e.g., [11]). It is possible to map event-related signals derived from scalp-EEG, from a whole-head magnetometer, or even from grids of electrodes placed on the cortical surface in human or animals. Finally, ELAN also incorporates specific tools to assess the spatiotemporal topography of event-related data for intracranial EEG (iEEG) acquired with depth electrodes having several contiguous contacts (e.g., see spatiotemporal mapping in [12]).

2.3.3. Time-Frequency Analysis. To study oscillatory brain activities, ELAN computes time-frequency (TF) representations of the signals using wavelet analysis. Typically Morlet wavelets are recommended as they provide an ideal compromise between time and frequency resolution [13]. In addition, ELAN also allows for TF analysis using Gabor transform of the signals. Generally, time-frequency power is computed for each single trial independently (using a .eeg and a .pos files) and then averaged across trials yielding TF maps of both stimulus-evoked and-induced activities (stored in a .tf file). To dissociate these two components of the response, ELAN provides a number of useful functions (Figure 5). For instance, ELAN can compute a measure known as stimulus phase-locking factor (or intertrial coherence) in the TF domain (see, e.g., [14]). Besides, ELAN allows for the subtraction of the evoked response (.p files) on each single trial prior to TF transform. Furthermore, TF representations can be computed directly on the evoked response.

2.3.4. Interchannel Synchrony Analysis. ELAN can also be used to compute coupling between pairs of channels using phase synchrony measures in the time-frequency domain [15] (see, e.g., [6] and Figure 6). One advantage of phase synchrony is that it is independent of signal amplitude. However, it is sensitive to signal-to-noise ratio (SNR) of the data. Further coupling measures such as coherence and cross-frequency analysis will be added to the toolbox in future developments. Users should keep in mind the general concerns and limitations that apply to all measures of coupling, including spurious connectivity that can result from field spread at the sensor level [16].

2.4. Individual and Group-Level Statistics. As mentioned above, it is possible to extract the amplitudes and latencies of event-related responses (see Section 2.3.2) for statistical analysis with custom-designed routines or other commercial software. In addition, ELAN itself offers numerous possibilities to perform point-by-point statistical analysis (i.e., one statistic is computed for each sample of each channel in the time domain, and for each sample and each frequency of each channel in the time-frequency domain), both within and between subjects. The currently available statistics are mostly nonparametric and randomization tests, which have larger domains of validity than their parametric counterparts. Specific tools using randomization methods have been developed in ELAN to test multisensory interactions using an additive model [12, 17, 18].

To handle the multiple comparison problem several strategies are possible, depending on the particular test performed: Bonferroni correction, False Detection Rate (FDR) correction [19] implemented in ELAN, or applying for each channel a threshold on the minimum consecutive numbers of samples where a P-value lower than .05 (or any other preset alpha level) is observed to consider an effect to be significant (for approximate values see [20]; such maximum statistics are computed in ELAN when using randomization tests). Outputs (e.g., P-values) of the statistical analysis programs in ELAN are always stored in an ELAN file format, and can thus be visualized using *ERPA*, or *TFVIZ*, in the temporal, spatial, and time-frequency domains.

2.4.1. Across-Trial Statistics for Within-Subject Analysis. Single-trial data are extracted from continuous data (.eeg file) using an event file (.pos) for one subject. In the time domain, emergence of responses relative to a baseline can be performed using Wilcoxon paired-value tests (or parametric t-tests), and conditions can be compared using Kruskal-Wallis tests or randomization tests. Similar computations can be made in the time-frequency domain (with tests performed for each sample and each frequency for each channel), with the addition of randomization statistics to test inter-channel synchrony (example in Figure 6).

2.4.2. Across-Subject Statistics for Group-Level Analysis. Event-related responses (.p file) in the time domain can be compared between conditions for a group of subjects using Wilcoxon paired-value tests (example in Figure 4), Quade tests, or randomization tests, and compared between groups of subjects using Kruskal-Wallis tests. In the time-frequency domain, paired conditions can be compared for a group of subjects using Wilcoxon or Quade tests.

2.5. Interoperability and MATLAB Compatibility. ELAN is fully compatible with MATLAB. Input and output functions (read/write) for data import and export from and into MATLAB are available for all three ELAN data formats (e.g., eeg2mat.m, mat2tf.m, mat2ep.m, etc.). In addition, ELAN can also export data to several MATLAB-based toolboxes such as SPM, FieldTrip, and Nutmeg. Data export functions for other toolboxes such as EEGLab and BrainStorm as well as a range of additional import routines will be available soon.

3. Discussion

3.1. Advantages and Specificities of ELAN. ELAN is a toolbox which allows to analyze virtually any kind of electrophysiological continuous signal (EEG, MEG, iEEG, ECoG, LFP, etc.). ELAN results from 25 years of interactions between methodologists, software developers, and users, which has resulted in a robust and useful set of functions covering a wide range of applications in neurophysiology. Regarding the type of data processing and analysis that ELAN can

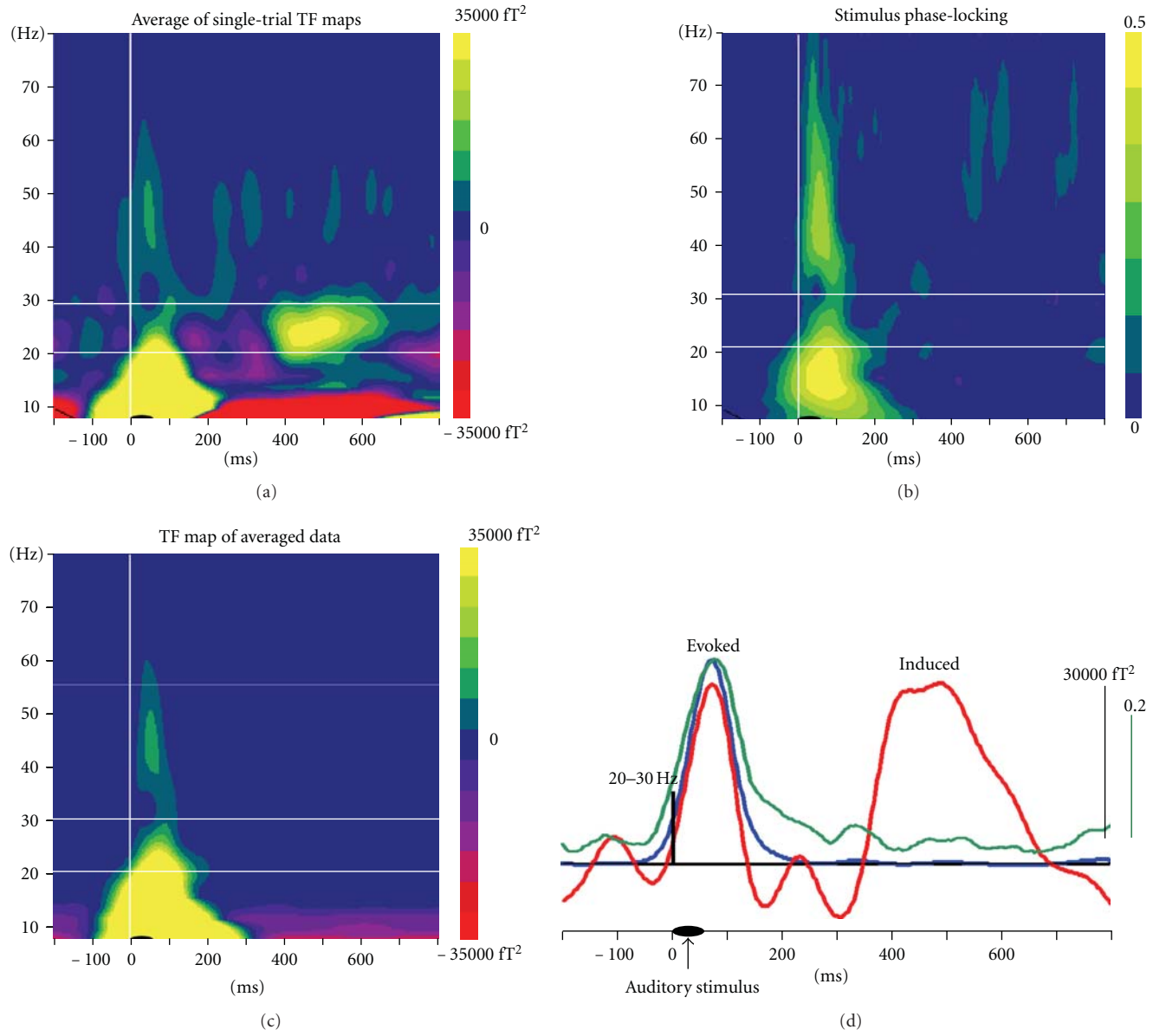


Figure 5: Time-Frequency analysis: dissociation of evoked and induced oscillatory activities, in an example of MEG responses to auditory stimuli. Top left: TF map obtained after averaging of wavelet transforms of single trials, which contains both evoked and induced responses. Top right: stimulus phase locking factor. For each single trial the phase of the response relative to an event (here the auditory stimulus) is computed, and the stability of the phase across trials is represented. Phase locking is close to 0 for activities not phase-locked to the stimulus also called induced activity, and larger stimulus phase locking values (up to 1) characterize stimulus phase-locked activities (evoked responses). Bottom left: TF map obtained after wavelet transform of the averaged evoked response, reflecting mostly evoked activities. Bottom right: time courses in the 20–30 Hz frequency band of the averaged oscillatory power (red), of evoked oscillatory power (blue), and of the stimulus phase-locking factor (green). The first peak of oscillatory responses at 100 ms could be identified as evoked, whereas the second one around 500 ms is induced.

perform, its main strengths are an easy preprocessing of the data, advanced scalp-level topographical mapping, time-frequency analysis, and adequate nonparametric and randomization statistics. Computations are run with command-line programs which are easily scriptable. Two of the main advantages of ELAN are the user-friendly navigation in the acquired and processed data which is made possible through three interactive viewers (*EEG*, *ERPA*, and *TFVIZ*) on one hand and the fast computations on the other hand. The high processing speed is achieved through optimized algorithms and compiled C code. This is particularly important for time-frequency analysis and related statistics. Several computation-intensive procedures can further be run using distributed computing according to the MPI standard. In addition to importation routines to read data from many commercial EEG or MEG systems, I/O MATLAB functions

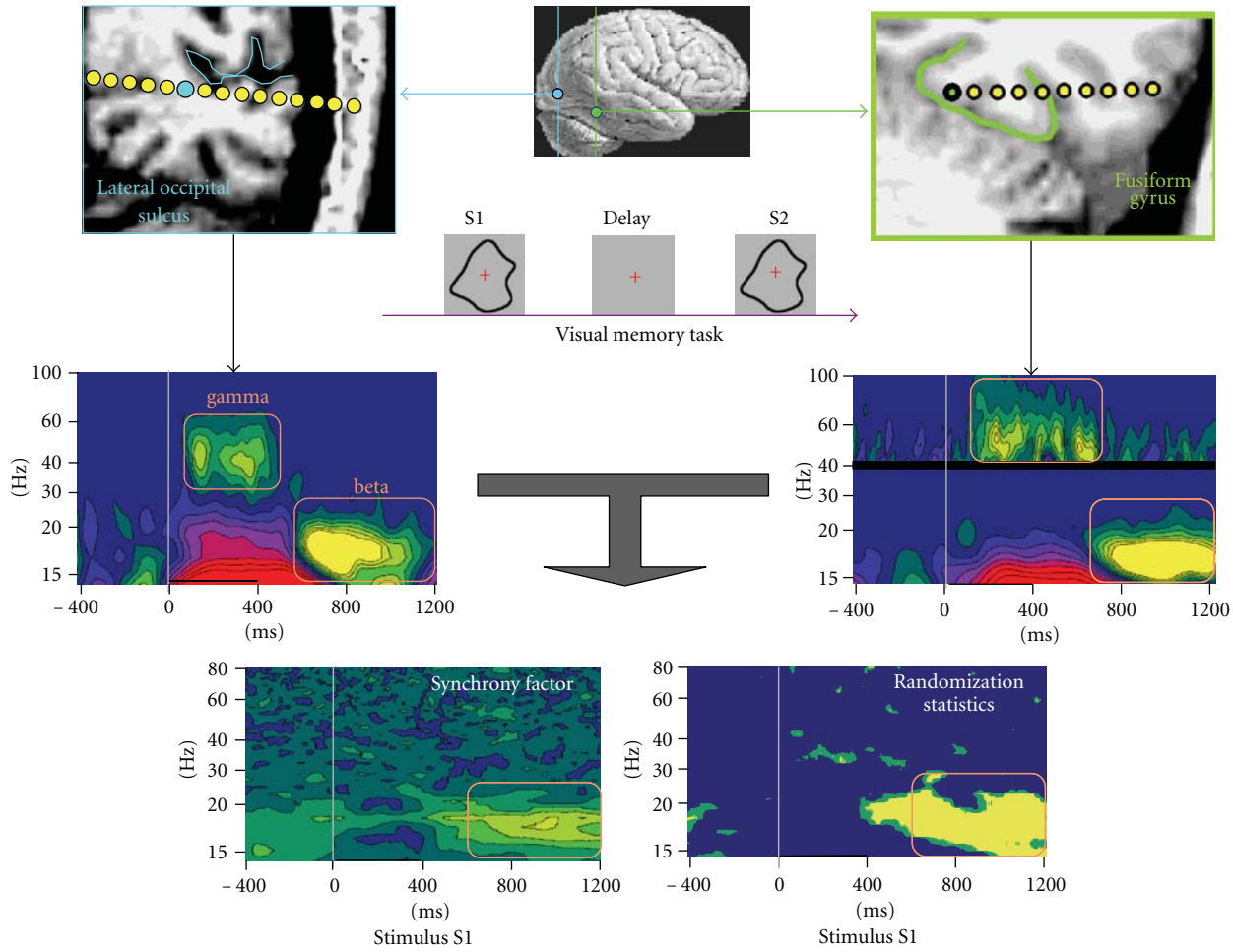


Figure 6: TF synchrony analysis, with an example of intracranial EEG activities after visual stimulation. Middle panels represent the TF power for two recording sites in the lateral occipital sulcus (left) and fusiform gyrus (right). Phase synchrony between the two sites was computed for each trial and averaged (bottom left). Significance of the synchrony values was assessed using randomization statistics (bottom right). Significant coupling was observed in the beta band (adapted from [6]).

are available for all ELAN data formats, and we are currently adding I/O functions for other analysis toolboxes which offer complementary functionalities such as source modeling (SPM, BrainStorm, Fieldtrip, etc.). Future implementations of additional functions optimized for real-time analysis are currently under development. The speed of ELAN code written in C would be ideal for real-time applications that require online processing of electrophysiological signals, such as brain-computer interface (BCI) systems. Furthermore, by contrast to a number of alternative freely available toolboxes, ELAN does not require the purchase of a MATLAB license. In addition to being a significant argument for some laboratories with limited financial resources, the fact that ELAN is independent of MATLAB also means that it is not affected by changes occurring between different MATLAB releases. In fact, backward compatibility within ELAN is an important aspect of our development policy. Finally, as mentioned above, a further specificity of ELAN is the availability of a simple and straight-forward tool for manual sleep stage scoring based on polysomnographic data (i.e., combination of EEG, EMG and EOG signals acquired during sleep).

3.2. Current Limitations. In terms of data analysis, one current limitation of ELAN is an absence of sourcereconstruction programs for EEG/MEG. However as mentioned above, it is possible to export data which have been pre-processed with ELAN to other software packages that have source-modelling procedures. Reciprocally, the resulting source time series obtained with other toolboxes can be imported into ELAN for further analysis (a template MATLAB function for conversion from .mat to .eeg file format is provided with ELAN). For example, source-level time series estimated using BrainStorm or SPM can easily be imported into ELAN for time-frequency analysis and statistical testing. Note also that ELAN libraries and header files are made available within the ELAN distribution package, with limited documentation. Moreover, upon user request, specific conversion files for specific source-level data formats (obtained with other toolboxes) could be made available. ELAN is currently not

open-source because the readability of the source code needs to be improved (i.e., cleaning up variable names, improving interpretability and architecture of the code). Nevertheless, it is already easy to achieve a reasonably flexible level of customization of the toolbox based on MATLAB interfacing or independent C routines, reading and writing ELAN file formats.

3.3. Software Access and Support. The ELAN software package is available for download free of charge through the ELAN web platform at http://elan.lyon.inserm.fr/. A detailed documentation is available for all ELAN routines, either on the website or as a pdf file. Although only limited online support is currently provided, we aim to enhance support especially through the creation of a community of users that exchange information through a dedicated forum on the ELAN website. We expect that the combination of continued in-house development and a growing knowledge base will enhance the quality, flexibility, and user-friendliness of ELAN. A user mailing list and ELAN newsletter also provides ELAN users with up-to-date information on software updates, availability of additional plug-ins, and bug reports. One of the future plans is to release ELAN source code in order to promote new developments and exchange of expertise in the community. Most importantly, we will continue to enhance the interoperability of ELAN with other free toolboxes in order to strengthen interactions and foster scientific cross-fertilization. We hope that ELAN will be an active contributor to the growing common effort within the community of method developers, programmers, and neuroscientists to substantially improve the power and reliability of neurophysiological data analysis tools over many years to come.

Acknowledgments

The development of ELAN over the years has been the result of an intense collaborative work undertaken by many members of our INSERM Laboratory. In addition to the authors, it involved François Perrin(†), Jacques Pernier, Marc Thevenet, Jean-François Echallier, Marie-Hélène Giard, Franck Peronnet, and Julien Besle for ERP topographical mapping and related statistics; Catherine Tallon-Baudry and Claude Delpuech for time-frequency and single-trial analysis; Blaise Yvert, Anne Cheylus, Emmanuel Maby, Dominique Morlet, and Gilles Bailly for the implementation of numerous signal processing and visualization methods.

References

[1] N. Ravel, P. Chabaud, C. Martin et al., "Olfactory learning modifies the expression of odour-induced oscillatory responses in the gamma (60–90 Hz) and beta (15–40 Hz) bands in the rat olfactory bulb," *European Journal of Neuroscience*, vol. 17, no. 2, pp. 350–358, 2003.

[2] G. Rols, C. Tallon-Baudry, P. Girard, O. Bertrand, and J. Bullier, "Cortical mapping of gamma oscillations in areas V1 and V4 of the macaque monkey," *Visual Neuroscience*, vol. 18, no. 4, pp. 527–540, 2001.

[3] A. Delorme and S. Makeig, "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis," *Journal of Neuroscience Methods*, vol. 134, no. 1, pp. 9–21, 2004.

[4] P. Ruby, A. Caclin, S. Boulet, C. Delpuech, and D. Morlet, "Odd sound processing in the sleeping brain," *Journal of Cognitive Neuroscience*, vol. 20, no. 2, pp. 296–311, 2008.

[5] A. Caclin, S. McAdams, B. K. Smith, and M. H. Giard, "Interactive processing of timbre dimensions: an exploration with event-related potentials," *Journal of Cognitive Neuroscience*, vol. 20, no. 1, pp. 49–64, 2008.

[6] C. Tallon-Baudry, O. Bertrand, and C. Fischer, "Oscillatory synchrony between human extrastriate areas during visual short-term memory maintenance," *The Journal of Neuroscience*, vol. 21, no. 20, p. RC177, 2001.

[7] F. Perrin, J. Pernier, O. Bertrand, M. H. Giard, and J. F. Echallier, "Mapping of scalp potentials by surface spline interpolation," *Electroencephalography and Clinical Neurophysiology*, vol. 66, no. 1, pp. 75–81, 1987.

[8] J. Pernier, F. Perrin, and O. Bertrand, "Scalp current density fields: concept and properties," *Electroencephalography and Clinical Neurophysiology*, vol. 69, no. 4, pp. 385–389, 1988.

[9] F. Perrin, O. Bertrand, and J. Pernier, "Scalp current density mapping: value and estimation from potential data," *IEEE Transactions on Biomedical Engineering*, vol. 34, no. 4, pp. 283–288, 1987.

[10] F. Perrin, J. Pernier, O. Bertrand, and J. F. Echallier, "Spherical splines for scalp potential and current density mapping," *Electroencephalography and Clinical Neurophysiology*, vol. 72, no. 2, pp. 184–187, 1989.

[11] M. P. Deiber, P. Missonnier, O. Bertrand et al., "Distinction between perceptual and attentional processing in working memory tasks: a study of phase-locked and induced oscillatory brain dynamics," *Journal of Cognitive Neuroscience*, vol. 19, no. 1, pp. 158–172, 2007.

[12] J. Besle, C. Fischer, A. Bidet-Caulet, F. Lecaignard, O. Bertrand, and M. H. Giard, "Visual activation and audiovisual interactions in the auditory cortex during speech perception: intracranial recordings in humans," *Journal of Neuroscience*, vol. 28, no. 52, pp. 14301–14310, 2008.

[13] C. Tallon-Baudry and O. Bertrand, "Oscillatory gamma activity in humans and its role in object representation," *Trends in Cognitive Sciences*, vol. 3, no. 4, pp. 151–162, 1999.

[14] C. Tallon-Baudry, O. Bertrand, C. Delpuech, and J. Pernier, "Stimulus specificity of phase-locked and non-phase-locked 40 Hz visual responses in human," *Journal of Neuroscience*, vol. 16, no. 13, pp. 4240–4249, 1996.

[15] J. P. Lachaux, E. Rodriguez, J. Martinerie, and F. J. Varela, "Measuring phase synchrony in brain signals," *Human Brain Mapping*, vol. 8, no. 4, pp. 194–208, 1999.

[16] J. M. Schoffelen and J. Gross, "Source connectivity analysis with MEG and EEG," *Human Brain Mapping*, vol. 30, no. 6, pp. 1857–1865, 2009.

[17] J. Besle, A. Fort, and M. H. Giard, "Interest and validity of the additive model in electrophysiological studies of multisensory interactions," *Cognitive Processing*, vol. 5, pp. 189–192, 2004.

[18] M. H. Giard and J. Besle, "Methodological considerations: electrophysiology of multisensory interactions in humans," in *Multisensory Object Perception in the Primate Brain*, J. Kaiser and M. J. Naumer, Eds., Springer, New York, 2010.

[19] C. R. Genovese, N. A. Lazar, and T. Nichols, "Thresholding of statistical maps in functional neuroimaging using the false discovery rate," *NeuroImage*, vol. 15, no. 4, pp. 870–878, 2002.

[20] D. Guthrie and J. S. Buchwald, "Significance testing of difference potentials," *Psychophysiology*, vol. 28, no. 2, pp. 240–244, 1991.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 861705, 10 pages
doi:10.1155/2011/861705

Research Article

ElectroMagnetoEncephalography Software: Overview and Integration with Other EEG/MEG Toolboxes

Peter Peyk,[1] Andrea De Cesarei,[2] and Markus Junghöfer[3]

[1] *Department of Clinical Psychology and Psychotherapy, Saarland University, Campus, 66123 Saarbrücken, Germany*
[2] *Department of Psychology, University of Bologna, 40127 Bologna, Italy*
[3] *Institute for Biomagnetism and Biosignalanalysis, University of Münster, 48149 Münster, Germany*

Correspondence should be addressed to Peter Peyk, p.peyk@mx.uni-saarland.de

Received 31 August 2010; Revised 30 November 2010; Accepted 18 January 2011

Academic Editor: Sylvain Baillet

EMEGS (electromagnetic encephalography software) is a MATLAB toolbox designed to provide novice as well as expert users in the field of neuroscience with a variety of functions to perform analysis of EEG and MEG data. The software consists of a set of graphical interfaces devoted to preprocessing, analysis, and visualization of electromagnetic data. Moreover, it can be extended using a plug-in interface. Here, an overview of the capabilities of the toolbox is provided, together with a simple tutorial for both a standard ERP analysis and a time-frequency analysis. Latest features and future directions of the software development are presented in the final section.

1. Introduction

EMEGS (electromagnetic encephalography software) is an open-source software written in MATLAB and developed for the analysis of data collected with high density, whole head electroencephalography (EEG), and magnetoencephalography (MEG). It comprises batch functions to segment and filter continuous data, statistically exclude artifacts, correct eye movement artifacts, average across trials, interpolate for missing or noisy sensors, and average across participants. It offers a range of visualization modules for event-related potentials/event-related fields (ERP/ERF) curve plotting, wavelet spectrograms, data projection onto 3D models (sphere, realistic, or brain), and statistical coloring of P, t, or F values. It allows to directly perform a variety of analyses and statistical tests on ERP/ERF data, such as inverse source estimations, current source density functions, MEG sensor position coregistration, fast Fourier transformation, principal component analysis, t-tests, repeated measures analysis of variance (ANOVA), permutation tests, regression, and correlation. It offers integration with *R* environment for statistical computing, allowing the calculation of advanced factorial designs directly on EEG and MEG data.

This presentation of the software will be divided into three sections, which will provide an overview of the EMEGS project (Section 1), describe its analysis and visualization features (Section 2), and provide a perspective on future directions (Section 3). Additionally, two appendixes give simple tutorials on an ERP and on a wavelet analysis (Appendix A) and describe the use of plug-ins to extend EMEGS (Appendix B).

1.1. Main Field of Software Application. Having been developed in a university environment, one main requirement of EMEGS is its applicability for both teaching and science—and thus for bachelor, diploma, master, and Ph.D. students, with not more than basic electrophysiological background knowledge and limited experience with EEG/MEG data analysis, as well as for experienced researchers demanding more than basic analysis functions. While some EEG/MEG analysis methods do not necessarily depend on a deep comprehension of their underlying functions and can be applied rather automatically, successful applications of various methods (e.g., EEG/MEG combination, realistic head modeling, integration of a priori knowledge in inverse modeling, Independent Component Analysis, etc.) strongly

depend on the users experience. In fact, usage of more sophisticated analysis methods, in the absence of convergent knowledge and experience, does often result in less reliable study results compared to the application of more basic methods. This might be commonplace for all research areas but appears, at least from our experience, of particular importance for the field of EEG/MEG data analysis. Therefore, a "basic" interface and an "expert" interface have been developed in EMEGS, in order to fulfill the user-specific needs.

In its "basic mode," the program allows novice users to thoroughly and autonomously analyze electrophysiological group studies, within the usually rather short amount of time given to prepare a thesis. It has thus been designed to be fast, easy to use, and easy to learn. It is completely based on graphical user interfaces, allows storage/loading of all relevant study parameter settings, and provides log files of all analysis steps and settings for final revision. The program tries to prevent typical novice user errors by giving warning messages and suggestions in case of rather unusual parameter settings or atypical combinations of methods (e.g., baseline correction of statistical values). It provides a pipeline for all major data analysis steps beginning with data preprocessing up to the final statistical analysis and visualization of results integrated within one program, avoiding the time consuming and, especially for beginners, often error-prone application of different analysis software components. All processing steps can be performed in batch mode, that is, a multitude of listed files can be processed identically and automatically. This does not only fasten but also secures the analysis against user errors (e.g., typos of parameter settings, which may have serious consequences especially in group studies). However, preventing a "black-box-" like usage, visual data inspection remains mandatory during the data artifact detection procedure and can be bypassed in "expert mode" only.

In the "expert mode," the program provides a wider spectrum of functions and combinations thereof. For all analysis steps, EMEGS offers state of the art methods. However, the variety of methods is limited—for example, the L2-Minimum-Norm is provided as fundamental inverse method [1] but other inverse functions like LORETA, SWARM, Beamformer, or others are not. If a specific method is missing, such as estimations of functional and effective connectivity or boundary or finite element conductor models, EMEGS offers interfaces for data exchange with other commercial and noncommercial programs like Curry, Besa, BrainVision, SPM, FieldTrip and R (see below).

EMEGS has been developed and optimized for the analysis of group studies investigating distributed neural network activity with limited a priori knowledge of its spatial and temporal characteristics. EMEGS is not recommended if integration of very detailed a priori knowledge, such as number and location of underlying neural sources, is desired. EMEGS does not provide detailed localization of neural activities in individuals, such as localization of epileptic spikes. As it comes without any warranty (see below) EMEGS should not be used for clinical purposes.

1.2. Project History and People. The development of EMEGS was initiated in 1997 by Markus Junghöfer, who at that time was situated at Konstanz University (Germany), in order to analyze data collected with a 128-sensor Electrical Geodesics Incorporated (EGI) EEG device and a 4D-Neuroimaging/BTi 148 sensor MEG magnetometer system, but it has evolved since to support more data formats and analysis types. After Peter Peyk (Saarland University, Germany) joined the developer team in 2003, EMEGS was initially published under the terms of the GNU General Public License (GPL) in 2004 [2] and has since continued to appeal to a small but growing set of users. Substantial programming contributions have also come from Andrea De Cesarei (Bologna, Italy), Andreas Keil (Gainesville, USA), and Andreas Wollbrink (Münster, Germany), and the package implements subroutines from a number of other authors, namely Thomas Gruber (Osnabrück, Germany), Olaf Hauk (Cambridge, Great Britain), and Nathan Weisz (Konstanz, Germany). As all software developers were and are actively doing research in various fields of affective and cognitive neuroscience with varying methodological core areas, EMEGS has strongly been shaped in the direction of applicability in these fields.

1.3. Availability, License, and Support. EMEGS is available for download free of charge at http://www.emegs.org/ (Figure 1(a)) and is managed as a CVS (Concurrent Version System) repository by a server located at Saarland University. Interested developers can apply informally for CVS access by email to the corresponding author. EMEGS is published under the terms of the GNU General Public License (GPL) v3. Documentation and help are provided in several manners. The program itself contains a documentation of the most frequent functions, while supplementary information is provided by online documentation. An email discussion list and an email archive exist at https://lists.sourceforge.net/lists/listinfo/emegs-user to allow users and developers to discuss problems and suggestions concerning the software, report bugs, and provide help. Furthermore, a revised chronological manual, written and continuously updated by novice users, will soon be available online. It will provide a standard operation procedure for the most common line of data analysis and give answers to frequently asked novice questions.

1.4. Supported Data Formats. EMEGS offers import/export or data conversion functions for averaged event-related potential or event-related field data sets from BESA, Vision Analyzer, Curry, EGI, Neuroscan, Biosemi, CTF, BTI, and European data format. All analysis tools on averaged data (see Section 2.3) are supported for EEG and MEG data equally.

Full data preprocessing (i.e., filtering, epoching, artifact detection/extraction/correction, and averaging; see Sections 2.1 and 2.2) of continuous EEG data is supported for Electrical Geodesics Incorporated (EGI), Neuroscan continuous (CNT), European data format (EDF), and Biosemi data format (BDF). The import of data epoched in foreign software packages (epoch file formats by EGI, Neuroscan, etc.) for

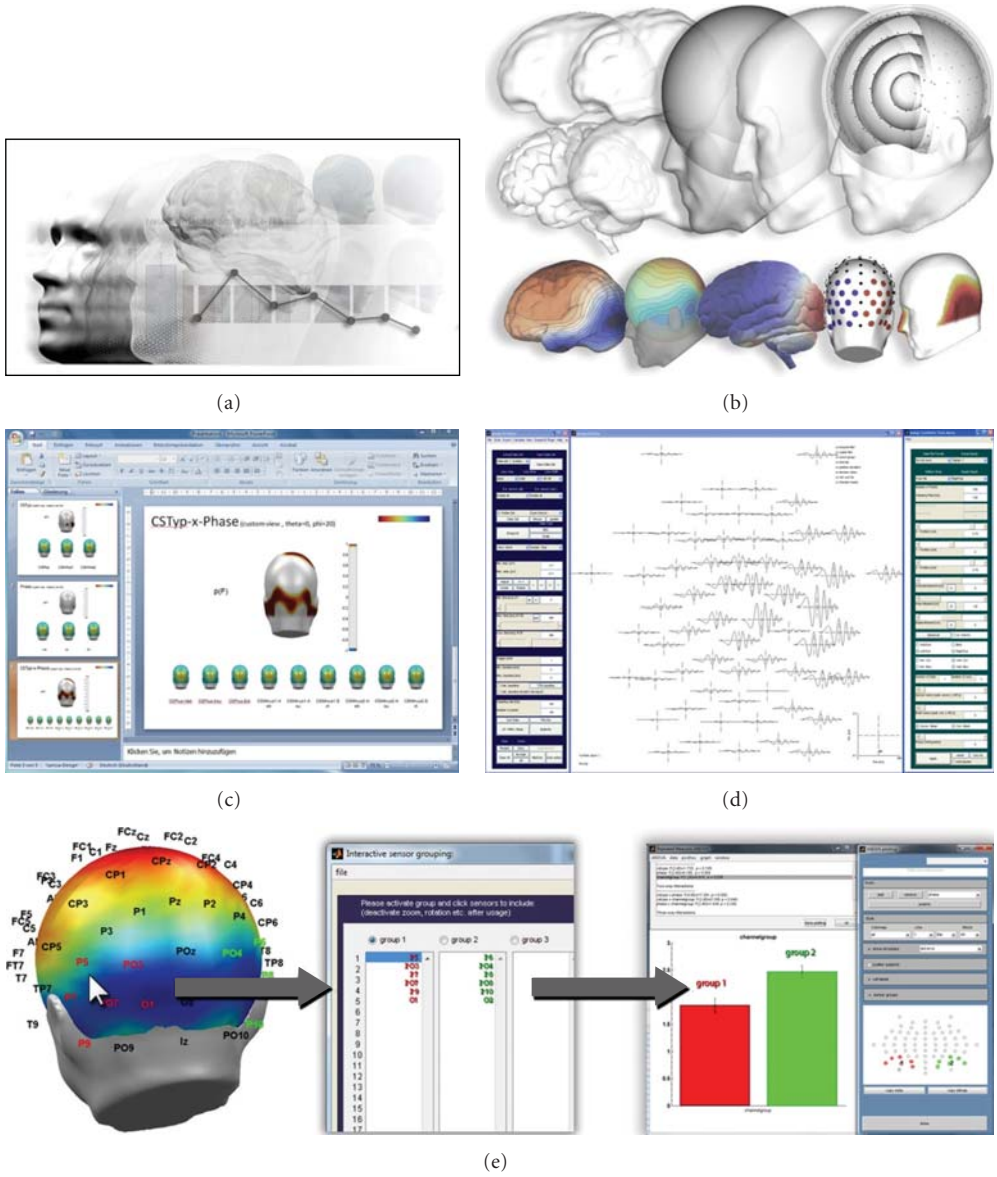


Figure 1: EMEGS snapshots of typical applications: (a) logo from the EMEGS web site at http://www.emegs.org/. (b) Various examples for the 3-dimensional visualization of results. (c) An automatically created Microsoft PowerPoint presentation as statistical report on a pointwise ANOVA. (d) The "Synthetic Data" graphical user interface was used to simulate a chirp signal. (e) Interactive sensor selection for statistical analysis with the built-in repeated measures ANOVA.

continued preprocessing in EMEGS (artifact detection and averaging) is not supported.

Preprocessing of MEG data (continuous or epoched in foreign software packages) is not supported.

1.5. System Requirements. EMEGS is written in MATLAB and cannot run without the MATLAB environment (7.1 or higher). Moreover, for full functionality, it requires the MATLAB Signal Processing and Statistics Toolbox. EMEGS runs on almost all platforms that MATLAB can be installed on but has been tested most thoroughly on Windows and Linux. For statistical analyzes or group studies, with many participants, experimental conditions, channels, and time points, a large amount of RAM is required (>2 GB). Hardware accelerated graphics are helpful for 3D displays.

2. Features and Implementation

Once raw EEG or MEG data have been collected, several preprocessing steps have to be carried out. In particular, for each trial a data interval has to be selected in the time domain (segmentation around an event of interest) and in the frequency domain (high, low, or band-pass filtering). Additionally, data which have been contaminated by undesired events, such as muscular activity, electrical noise, or bad electrode contact, have to be detected. The

original uncontaminated signal may be recovered using data correction techniques (e.g., eye movement correction) or sensor interpolation based on sensors containing uncontaminated data (as in the case of one single noisy sensor). Finally, experimental condition averages, across single and groups of participants, are calculated. With clean brain responses available, second-order analysis is usually sought after, such as source localization, wavelet and exploratory and inferential statistical analysis. For method testing and educational purposes, it can be useful to generate synthetic EEG/MEG data.

The following sections will give an overview of how these different tasks are implemented in EMEGS. Additionally, Appendix A provides a tutorial showing how a typical EMEGS analysis session is structured.

2.1. Preprocessing. Data preprocessing in EMEGS is optimized for statistical control of artifacts. It guarantees optimal signal-to-noise ratios with objective and identical parameter settings for all participants and experimental conditions within group studies.

First, continuous interleaved data are converted to a demultiplexed format, to allow a fast and optimal filtering of channels by avoiding edge filter artifacts and allowing high-pass filtering with low cutoff frequencies. Biosemi data are re-referenced to Cz (Data in a Biosemi recording are written to file, referenced to the common mode sense (CMS) electrode and need to be re-referenced to remove the CM signal (c.f. http://www.biosemi.com/)), non-EEG/MEG channels are excluded, and the status (trigger) channel is analyzed for value changes that are written to a text file for faster access.

Second, continuous data are filtered as specified by the user. Visualization tools to investigate transfer functions of a variety of filters (Butterworth, Kaiser, firls, etc.) and identify reasonable filter settings are provided.

Third, trigger-based trials are extracted and stored in an epoch file, which contains trials in chronological sequence. An additional condition file in text format stores the trigger information of each epoch. Editing this file allows for reordering of experimental conditions. Special editing programs provide typical recoding types, such as balancing of trial numbers and odd-even split.

Fourth, if selected by the user, epochs are corrected for eye movement and eye blink artifacts, by either using a built-in implementation of the Gratton et al. regression approach [3] or by calling on the MATLAB-based toolbox BIOSIG [4], which contains a similar routine.

2.2. Artifact Detection and Averaging. After the preprocessing steps described above, parameters for the statistical editing of artifacts are calculated, saved to file, and applied. Typical parameters are the absolute maximum amplitude, the standard deviation, and the absolute maximum temporal gradient of potential/fields at individual trials, sensors and within or across predefined time intervals. Further parameters, such as amplitudes for specific frequency bands (e.g., alpha waves), may be added. These parameters serve to detect artifacts that are either global or specific to individual channels. Artificial sensors will be interpolated if the residual sensor distribution allows a reasonable approximation. The goodness of sensor interpolation is tested by interpolating a multitude (number of sensors) of sensor specific synthetic potentials or field topographies. If many to be interpolated sensors fall into one region or if many noisy sensors are positioned at the edge of the sensor coverage, a larger number of these test topographies are not interpolated with a sufficient decency and the corresponding trials get rejected from further analysis. For MEG or single-reference EEG data, statistical parameters are calculated and applied only once. For average reference EEG data however, this process is done by first using the recording reference and, in a second loop, using average reference [5]. The first pass avoids contamination of all channels by sensor-specific artifacts when transforming EEG data to average reference. The second pass, based on the average reference, detects global artifacts more clearly because the reference bias has been removed. The user interface to perform this statistical artifact detection is resented in more detail in the ERP tutorial section (Appendix A).

Following the artifact detection stage, data from each individual trial may fall in one of three scenarios. (1) If data from all sensors are clean, then all data will be averaged together to obtain the corresponding ERP/ERF. (2) If data from too many channels in one specific trial are too noisy and the above described interpolation check would indicate an insufficient goodness of interpolation, the trial will be discarded. (3) With a positive interpolation check, artifact-contaminated sensors within individual trials will be replaced by spherical spline interpolation, statistically weighted on the basis of all remaining sensors [6]. In this way, clean and approximated epochs are averaged by experimental condition in time or frequency domains and stored trial-by-trial on request for an optional second-order analysis (e.g., time-frequency analysis, wavelet, and single-trial inverse modeling).

2.3. Interpolation, Current Source Density, and Source Localization. To estimate brain signal values between sensor positions for mapping of surface models, EMEGS uses an inverse-forward source estimation [7], which can also be used to calculate the current source density (CSD) or Laplacian. Thus, the stiffness of the spherical spline functions, used for interpolation and CSD, is based on physiological parameters.

For source localization, EMEGS uses the L2-Minimum-Norm-Pseudoinverse (L2MNP), an inverse modeling technique, which estimates cortical generator structures without any a priori assumptions regarding the location and/or number of current sources [8, 9]. The classical minimum norm solution is a highly recommended inverse method, especially when no reliable a priori information about source generators is available [1]. As source model, a 4-shell (with radii of 2, 4, 6, and 8 cm) sphere model is used comprising 3 (azimuthal, polar, and radial) × 655 (EEG) or 2 (azimuthal, polar) × 655 (MEG) equidistant dipoles with a user adjustable Tikhonov regularization parameter λ. The outer two shells can also be used as separate full source models. The 4-shell source model is illustrated in Figure 1(b).

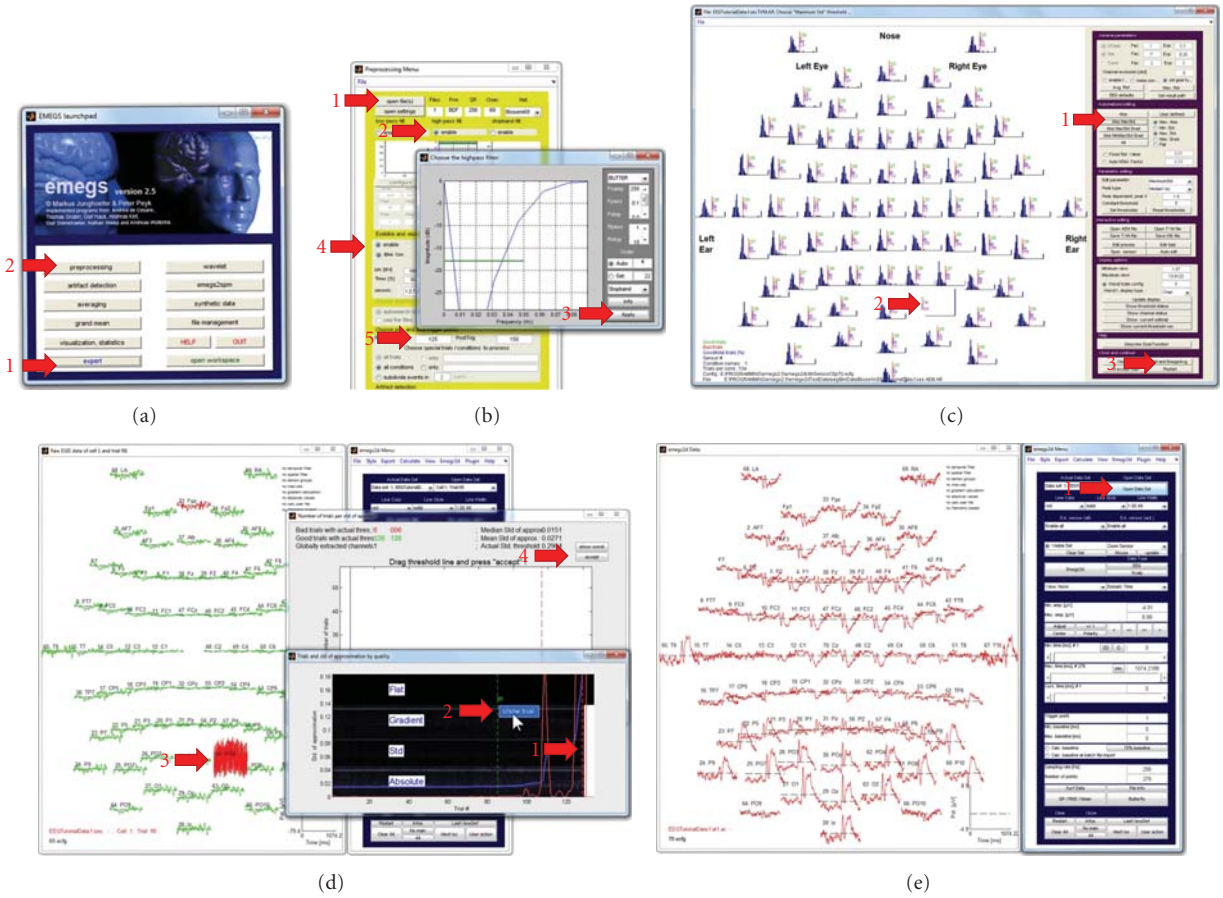


Figure 2: ERP tutorial illustration. Red arrows indicate processing steps of the tutorial. (a) The EMEGS launch pad, which can be used to start all major EMEGS programs. (b) The EMEGS preprocessing program, with an additional dialog to configure a high-pass filter. (c) The EMEGS artifact detection program, which is based on statistical parameter distributions. (d) Detailed display of single trial data during artifact threshold editing. (e) The EMEGS program for 2D ERP/ERF data display.

While visualization tools allow the representation of activations over realistic head or brain models, it should be emphasized that EMEGS does not provide inverse modeling routines based on realistic MRT- or CT-based head modeling. The 3D visualization tools, which are presented later on, are spherical projections from the underlying sphere-fit-based solutions on a realistic shaped head or brain surface model and are only used to illustrate the approximate localization of inverse solutions. In fact, although a sphere is a good first approximation of the human brain, there are quite strong deviations from the sphere model, especially at prefrontal cortex regions. Thus, the sphere-based localization of estimated neural activity deviates stronger from realistic head model-based estimations within these compared to other areas, providing a reasonable spherical fit [10]. It has to be noted though that the usage of realistic head models in inverse source estimations exhibits its own problems. As one example, radial sources do not result in measurable magnetic fields (MEG) outside of a sphere and can thus be ignored in spherical head models. Quasiradial sources in realistic models, however ask for cautious and user-dependent regularization, as they might otherwise provoke ghost effects as a mere consequence of inappropriate regularization. Again, there is no doubt that the usage of additional information, such as a MRI-, CT-, or DTI-based realistic volume conductor modeling, is advantageous with regard to the accuracy of inverse modeling. But usage of this additional information is still user dependent, that is, nonautomatic, and based on a deep understanding of the underlying algorithms—an understanding that, from our point of view, cannot be required from all software users.

2.4. Statistical and Exploratory Analysis of Evoked Brain Signals. A number of statistical and exploratory calculations on evoked or induced potentials/fields, as well as on their estimated underlying neural sources, can be run directly within EMEGS, without the need to export data to external statistical applications. Available calculations include various utility functions to calculate potential/field/source differences, averages, and EEG re-referencing, in addition to methods like principal component analysis, t-tests, correlation and regression analysis, repeated measures ANOVA, post hoc contrasts, permutation tests, global dissimilarity functions, spatial and temporal filtering, and so forth.

Most tests can either be done for defined sensor/source groups and selected time intervals or on all individual sensors/sources and all sample points. A graphical user interface for a fast and objective definition of sensor groups or source regions of interest and automatic identification of corresponding mirror groups/regions is provided, as is an interface for the interactive definition of analysis intervals. A useful application of sensor groups is the regional mean square calculation for a number of sensors, in order to obtain a regional power (e.g., the regional power of all occipital sensors). An application of interval means is, for instance, the creation of a statistical surface plot corresponding to a single region-of-interest ANOVA (e.g., the average potential from 80 to 120 ms after stimulus onset for the P1 component).

In addition to the repeated-measures statistical functions provided by MATLAB, EMEGS includes a built-in and graphical user interface-based repeated-measures ANOVA that can handle up to six within factors—including the two within-factors sensor/source groups and time intervals of interest—and up to three between factors. This ANOVA can be run as a single region-of-interest analysis with bar graphs and post hoc testing (illustrated in Figure 1(e)) or as a point-by-point analysis, resulting in 4d statistical parametric maps, that are stored in standard SCADS files and can be displayed like any other brain signal. All statistical methods allow various online data transformations during data import, such as averaging across time intervals or sensor groups, rectification, temporal and spatial filtering, or further custom made transformations. This allows a fast exploration on the impact of different parameter settings (e.g., spatial and temporal filtering, strength of regularization) or alternative data transformation methods (e.g., baseline correction versus high-pass filtering) on the final statistical results.

To control for the accumulation of type I errors, EMEGS provides post hoc filtering routines to scan significant time ranges for a sufficient sequence length (number of time points) and spatial spread (number of adjacent significant sensors) and remove results that do not match these criteria. The nonparametric permutation (rerandomization or exact) test is also provided.

2.5. Data Display and Visualization. EMEGS allows the visualizing of potentials, fields or estimated sources as well as statistics thereof (F value or t value distributions) as curves with planar projection of the sensor arrays, single sensor zoom, global power (GP), and global root-mean-square (RMS), global mean or corresponding time curves for sensor-groups or regions of interest definition. Global or regional power/RMS/mean values on a trial-by-trial basis can be visualized by time × trial color surface plots.

Data may also be visualized as 3D projections onto several models, including a simple sphere model, a realistic head shape, a realistic brain shape, and spatially smoothed versions of the former two (Figure 1(b)). For illustrative purposes, a brain shape comprised of independently selectable brain lobes has been implemented. Coloring can be selected from a large set of linear, exponential, logarithmic, or customized color maps and configured as smooth transitions or as contour plot type. Sensor/source positions or groups thereof can be displayed and colored at leisure. Visualization parameters such as line or surface coloring and line options can be stored and loaded for repetitive usage.

The temporal development of any 3-dimensional projection can be illustrated on a column- and rowwise subplot figure, using either default equidistant and consecutive time intervals or customized intervals with variable length and onsets. These four-dimensional illustrations can also be presented as movies and can be stored in various movie file formats.

Overview functions to compare corresponding averages of trial-by-trial data curves (2D) or data topographies (3D) for all participants and experimental conditions are provided (e.g., for identification of spurious effects in participants and/or experimental conditions). Additionally, several other data types may be visualized, including horizontal scrollable raw data display, spectrogram wavelet display, and magnetic resonance imaging (MRI) data.

Statistical results may be exported as continuous waveforms (4d-SPMs) and projected on a 3D head model. Statistical graphs may be created in parametric mapping mode, highlighting significant regions and time ranges, and hiding areas where nonsignificant results were observed.

In addition, EMEGS offers a number of functions to automate the creation of surface plots, for instance for the purpose of visual inspection across an entire sample of experimental conditions and/or subjects or for statistical reports. It can automatically create a Microsoft PowerPoint presentation after a point-by-point ANOVA has been run—with cell averages and P value maps brought together on one slide per main effect and interaction (illustrated in Figure 1(c)).

2.6. Generation of Synthetic Data. For educational purposes, as well as for method development testing, EMEGS also provides a generation of synthetic EEG/MEG data based on default or customized sensor distributions. Location and direction of single or multiple synthetic neural sources with various predefined or customized waveforms can be defined. Various combinations of temporally uncorrelated (white), temporally correlated (Gaussian), spatially uncorrelated (sensor noise), and spatially correlated noise (brain noise), as well as simulated ocular artifacts, are provided. Simulated data can be stored in various averaged or single-trial formats. Signal and noise amplitudes can be systematically varied across simulated trials or simulated averaged participant data. Figure 1(d) visualizes the corresponding "Synthetic Data" graphical user interface (left and right frame) used here to simulate sweep or chirp signals (central frame). Such signals can for instance be used to illustrate or test the application of time-frequency methods, such as wavelet analysis.

2.7. Extending EMEGS Capabilities: Exemplified Time Frequency Analysis Using FieldTrip. The modulation of brain activity, which is related to an event of interest, can be investigated using event-related potentials or fields. However, this technique reflects modulations of neural correlates with

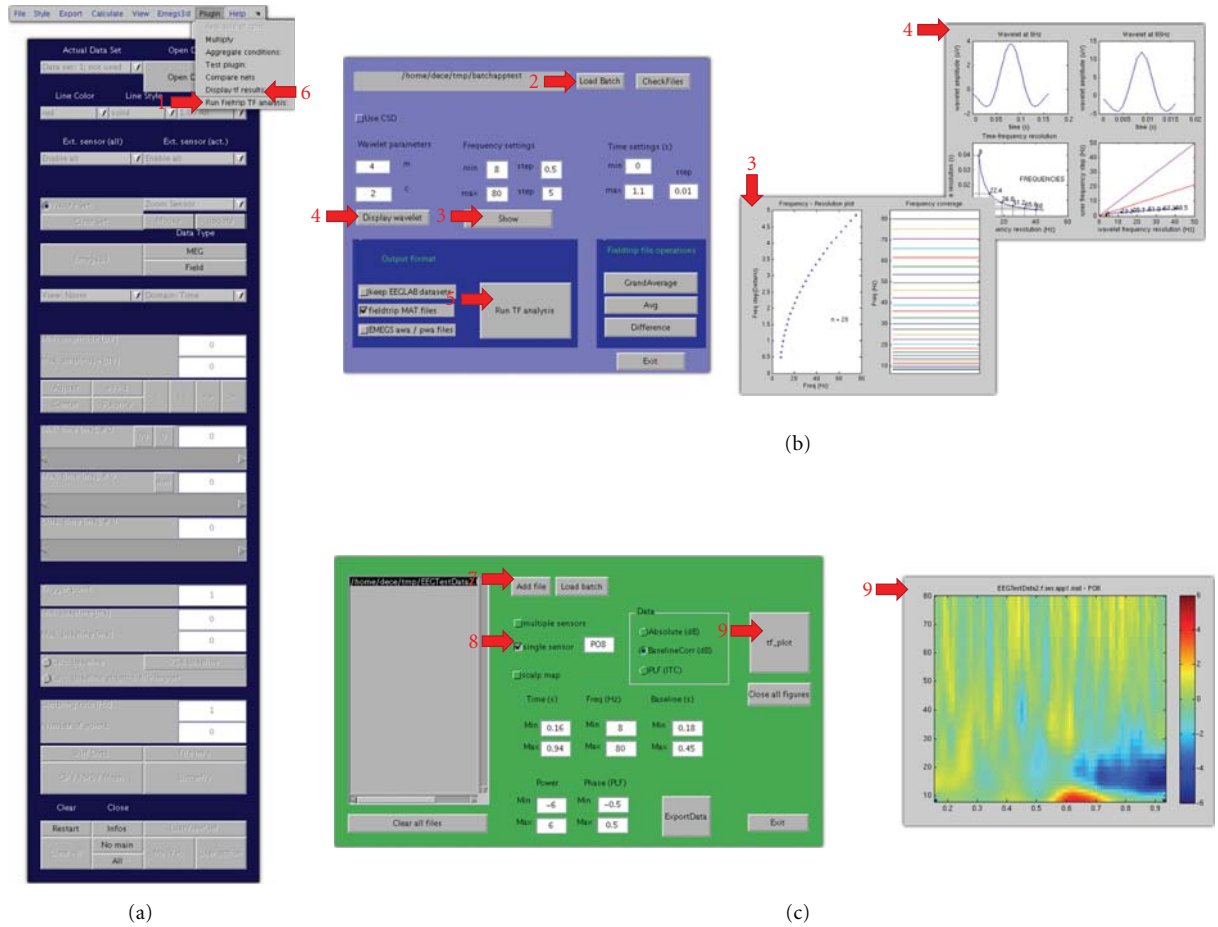


FIGURE 3: Time-frequency tutorial illustration. Red arrows indicate processing steps of the tutorial. (a) The EMEGS program for 2D ERP/ERF data display, illustrating the plug-in menu. (b) The plug-in analysis interface, including diagnostic plots. (c) The plug-in visualization interface, including analysis results from sensor PO8.

rather strong phase coupling to an event of interest [11]. Therefore, a growing interest has arisen towards techniques that allow the examination of signals with rather weak phase coupling. Analysis of time-frequency characteristic EEG/MEG data allows one to observe the extent to which specific rhythms (e.g., alpha or gamma) are modulated by experimental events, regardless of their phase orientation [12]. This analysis separates phase and power information [13], and EEG/MEG spectral changes can be classified as phase locked (*evoked*) or nonphase locked (*induced*).

To allow time-frequency analysis of electrophysiological data demanding more than the basic functions provided by EMEGS, an EMEGS plug-in (see Appendix B) has been recently added, which integrates EMEGS preprocessing with functions provided by the FieldTrip EEG processing toolbox [14]. At a basic level, EMEGS and FieldTrip serve the same purpose: to review EEG/MEG data, analyze them, and visualize and export the results. At a more specific level however, each application offers its users a selected choice of analysis algorithms and graphical user interfaces (GUIs). Here, a plug-in is described, which calls on FieldTrip routines to perform a time-frequency analysis on trial-by-trial EMEGS data, running through EEGLAB [15] for data conversion.

All analysis steps (data conversion, parameter selection, and analysis) are handled by the plug-in GUI, which provides settings for a reasonable set of parameters allowing to perform a Morlet wavelet analysis. GUIs for the setting of further wavelet parameters could be added easily. The plug-in interface allows to adjust wavelet parameters (width and length) and calculate time-frequency changes in power and in the phase-locking factor. The Laplacian or Current Source Density (CSD) may be applied as additional deblurring methods [7, 16, 17] for a better identification of regional EEG/MEG spectral changes. In addition, a variable frequency resolution can be used, and results can be written in a FieldTrip- or EMEGS-readable format. Finally, simple operations (grand average, average, and difference) can be performed on the result files. Results can be visually displayed using the provided visualization plug-in, or using the standard EMEGS visualization functions. The use of the EMEGS time-frequency plug-in is illustrated in Appendix A.

3. Future Directions

Here, an overview of the EMEGS processing capabilities was given in terms of data preprocessing, statistical analysis, and data visualization. As an example of the EMEGS plug-in architecture, an integration with the FieldTrip software to perform time-frequency analysis was presented.

Analyzing electromagnetic brain signals represents a special challenge, as EEG/MEG modulations reflect brain processes with spatial, temporal, and functional properties that are largely unknown. New analysis approaches are suggested as research advances, and they allow the uncovering of specific characteristics of neurophysiological signals. It is common to all developers of EEG/MEG data analysis software to implement these new methods in their applications or to provide convenient interfaces to other software that provides specific functions.

EMEGS is continuously being developed, and new functions are added on a regular basis. For instance, work on an EEG/MEG beamformer approach for source modeling is in progress (see applications in [10]). Statistical capabilities are being extended by integrating EMEGS with the R software for statistical computing [18]. (For the analysis of between-group designs with unequal group sizes, EMEGS calls the R software for statistical computing (available at http://www.r-project.org/) via the R-(D)COM Interface *Statconn* (available at http://www.statconn.com/) on Windows and the R-package *Rserve* and Java R client *JRClient* (available at http://rosuda.org/Rserve/) on UNIX-type operating systems.) Additionally, a stand-alone version of EMEGS (http://www.emegs.org/wiki/emegs qt version) is under construction (preliminary releases of the Qt version are already available for download on the EMEGS website (http://www.emegs.org/)), programmed in C++ and built with the open-source cross-platform GUI library Qt (http://qt.nokia.com/). This version is meant to provide the conveniences of modern graphical user interfaces, which cannot be realized within MATLAB, and allows for faster openGL-based graphics. Furthermore, analysis modules for peripheral psychophysiological measures, such as electrocardiography (ECG) and electrodermal activity (EDA), are under construction.

As an open-source project, EMEGS welcomes participation of new developers. To this end, the plug-in architecture provides an easy way to extend EMEGS analysis capabilities, while maintaining an easy to use graphical interface. Here, a plug-in devoted to run a Morlet wavelet analysis was described. It is important to note that, using the mechanisms and low-level functions described here, users can quite easily create new plug-ins, for instance to access the analysis functions provided by EEGLAB or FieldTrip directly from the EMEGS interface.

4. Conclusion

In sum, we recommend the usage of EMEGS as a fast and secure pipeline for EEG/MEG data analysis. EMEGS provides all relevant data analysis modules from preprocessing to statistics and final visualization of results within one integrative package. Users with only basic methodological knowledge may particularly benefit as EMEGS allows them to investigate typical neuroscientific research questions in a rather easy, fast, secure, and autonomous way. As such, EMEGS especially recommends itself for researchers with quite limited temporal resources and restricted assistance from advanced users to learn and apply software packages offering a greater variety of analysis tools. However, EEG/MEG experts also benefit from the highly automated and standardized analysis pipeline. For specific nonprovided functions, EMEGS offers interfaces for data exchange with other commercial and noncommercial EEG/MEG analysis software packages.

Appendices

A. EMEGS Analysis Tutorial

In this section, two brief analysis tutorials will be given to illustrate the EMEGS processing facilities described above. A 70-sensor EEG data file acquired at 256 Hz with a simple passive viewing paradigm was used in this tutorial. It includes 44 trials in each of three experimental conditions (3 different neutral faces), presented in random order. The processed data is included in the EMEGS download (at .../emegs2.5/emegs2dTestData/eeg/BinData/Biosemi/EEGTutorialData.7z) as is a more detailed version of this tutorial, which interested users are encouraged to consult for following along (http://www.emegs.org/tutorial.html). To run this tutorial, EMEGS 2.5 must be extracted in the Matlab path. To run the wavelet analysis, FieldTrip and EEGLAB must also be downloaded and extracted in the Matlab path.

A.1. ERP Analysis Tutorial

A.1.1. Preprocessing. Open the EMEGS start dialog (called "EMEGS launch pad", Figure 2(a)) by typing "emegs" in the MATLAB command window. Activate the "expert" mode (the expert mode is needed to run the eye movement correction using the BIOSIG toolbox files) on the bottom of the launch pad (Figure 2(a)-1), then start the "Preprocessing" tool (Figure 2(a)-2). A yellow interface will open (Figure 2(b)), push the "open data file" button at the top (Figure 2(b)-1) and select the demo datafile in the file dialog. Enable a 0.1 Hz high-pass filter (Figure 2(b)-2) by clicking on the "enable" button under "high pass filt" and then selecting "Apply" (Figure 2(b)-3). To perform an automatized eye movement correction, activate the eye blink/eye movement correction button (Figure 2(b)-4), then activate the checkbox on the bottom of the preprocessing interface reading "eyecorr with biosig" (not shown). To select the size of the data epochs to be extracted, set the number of points to be extracted before each trigger to 125 (Figure 2(b)-5, "PreTrig") and the number of points to be extracted after each trigger to 150 (Figure 2(b)-5, "PostTrig"). Finally, push the "Run" button on the bottom of the preprocessing window (not shown) to start the analysis. EMEGS will start the preprocessing and

```
(1) function [PlugTag, PlugLabel]=PlugHelloWorld(PlugAction)
(2) if nargin<1; PlugAction="Exec"; end
(3) if strcmp(PlugAction,"Init")
(4)         PlugTag="HelloWorld";
(5)         PlugLabel="Hello World";
(6) else
(7)          msgbox("hello world!");
(8) end
```

Algorithm 1

will write a log file in the folder where the data file resides (called "PrePro.log").

A.1.2. Artifact Detection. Continue with the artifact detection by pushing the "Edit processed files" button on the bottom of the preprocessing window (not shown). To start the artifact detection, push the "Abs MaxStd" button (Figure 2(c)-1), which has EMEGS performance statistical control of artifacts based on the maximum absolute amplitude (Abs) and the standard deviation excluding outliers on the maximum side of the trial distribution (MaxStd). The program will then display histograms of the Abs and MaxStd parameters for each channel (Figure 2(c)). Narrow unimodal distributions in nearly all cases indicate good data quality, whereas multimodal and broad distributions nearly always indicate a substantial amount of artifacts in the data. EMEGS prompts you to set the trial threshold separating good from bad trials manually (Figure 2(d)) by dragging the dotted red line (Figure 2(d)-1) from left to right to the proper position. Each column of the gray background graph codes the parameters of one trial, and columns (trials) are sorted by data quality with the best trials on the far left and the worst trials on the far right. The dotted red line you need to adjust marks the threshold between good and bad trials. For the present data file, data quality is very good, so a far right position indicating mostly good trials and very few bad trials is appropriate. To check the data of a specific trial, right click and select "show trial" from the context menu (Figure 2(d)-2). The data display program comes up and shows the current trial, highlighting good channels in green and bad channels in red (Figure 2(d)-3). When you have set the threshold properly, push the "accept" button in the window named "Number of trials per std of approx.:" (Figure 2(d)-4). A graphical display of the chronological sequence of good and bad trials is displayed next (not shown), and global and trial- and channel-specific thresholds are written to disk.

A.1.3. Event-related Averaging, Sensor Interpolation, and Simple Data Display. Push the "Close all and EmegsAvg" button on the artifact detection main window (Figure 2(c)-3) to directly start the averaging window for the edited data. The data format is set automatically, and you just need to push the "Run average" button to start the averaging and approximation of missing sensors.

Now, you should be able to find the result files in the data folder. Namely, the files "EEGTutorialData.f.ses. app1," "EEGTutorialData.f.ses.app2," and "EEGTutorialData.f.ses. app3" containing the clean data epochs for the conditions 1, 2, and 3, and the files "EEGTutorialData.f.at1.ar," "EEGTutorialData.f.at2.ar," and "EEGTutorialData.f.at2.ar" containing the ERPs. Single-trial and averaged data was converted to average reference.

Push the "Close All and Emegs2d" button on the bottom right corner of the averaging window, and the data display and analysis window will open (Figure 2(e)). Push the "Open data file" button (Figure 2(e)-1) and select one of the ERP-files listed above to display its data.

A.1.4. Wavelet Analysis. Run the time-frequency analysis plug-in from Emegs2d -> Plug-in -> Run Fieldtrip TF analysis, or run *Plugtfwltgui* from the Matlab prompt (Figure 3(a)-1). In the plug-in interface (Figure 3(b)-2), push the "load .app " button and select any of the three newly created .app files. Now enter the wavelet analysis parameters: under the "frequency settings" label, set min = 8, step = 0.5, max = 80, and step = 5, to focus on data from 8 Hz to 80 Hz. Steps between successive frequencies now vary linearly from 0.5 Hz for the lowest frequencies to 5 Hz for the highest frequencies. Which and how many frequencies will be analyzed can be displayed by pushing the "show" button (Figure 3(b)-3). Since the pretrigger and the posttrigger interval is rather short in the present data, wavelet parameters will be set to m = 4 and c = 2, to obtain a short wavelet which can be repeated several times in the data segment, even at the lowest frequencies. In general, longer baseline and data segments allow for more accurate time-frequency analyses. To display the resulting wavelet, push the "display wavelet" button (Figure 3(b)-4). In the "time settings" section, specify min = 0, max = 1.1, and step = 0.01 to analyze the entire 1100 ms epoch in steps of 10 ms each. Push "Run TF analysis" to start the analysis (Figure 3(b)-5). Data will be automatically converted in the EEGLAB format, then imported in Fieldtrip, and analyzed using the chosen parameters. As a result, a .mat file containing the time-frequency results, in terms of power and phase-locking factor (PLF), will be created in the same folder as the original data.

To display results, choose Emegs2d -> Plug-in -> Display TF results (Figure 3(a)-6) or run *Plugtfgui* from the MATLAB command window. Load the newly created .mat files by selecting "Add file" (Figure 3(c)-7), then activate "single sensor" from the visualization options and "PO8" as sensor of interest (Figure 3(c)-8). In the time section, enter

min = 0.16, max = 0.94, with baseline correction from 0.18 to 0.45 (as described in the ERP tutorial, data epochs include 500 ms of baseline, thus stimulus onset is at 0.5 s). Change the power scale to − 6 to +6, and select tf_plot to display a Fieldtrip plot of the chosen sensor (Figure 3(c)-9). In a similar fashion, the whole-head topographies and multiple sensor plots can be easily obtained for both power and PLF data.

B. EMEGS Plug-ins

Since EMEGS version 2.0, the facility to create plug-ins has been added. A plug-in is a MATLAB script that is placed in the "… emegsX.X/emegs2dUtil/Plug-ins" directory. This directory is automatically analyzed at EMEGS startup, to create menu entries for each plug-in. Plug-ins can also be directly executed as MATLAB commands, while EMEGS is not running. Algorithm 1 is the basic structure of a plug-in.

At startup, EMEGS scans the plug-in directory and executes each plug-in with the optional argument PlugAction = "Init". The plug-in interprets this as an EMEGS request and outputs a tag and a file menu descriptor, which will be used by EMEGS to create the corresponding menu entries (lines 3–5). On the other hand, when the corresponding menu entry is selected, or when the plug-in is executed from the MATLAB prompt with no arguments, the code indicated at line 7 is executed (in the example, a message box with the text "hello world!" is displayed). EMEGS plug-ins offer an easy way to integrate ones own analysis functions into the EMEGS GUI. Therefore, they were chosen here to integrate EMEGS, EEGLAB, and FieldTrip analysis functions.

Acknowledgments

The paper is supported by the DFG (SFB-TRR58-C1, JU445/5-1) and the Academy of Science, Heidelberg Germany. P. Peyk and A. De Cesarei contributed equally to this work.

References

[1] O. Hauk, "Keep it simple: a case for using classical minimum norm estimation in the analysis of EEG and MEG data," *NeuroImage*, vol. 21, no. 4, pp. 1612–1621, 2004.

[2] M. Junghöfer and P. Peyk, "Analyzing electrical activity and magnetic fields in the brain," *Matlab News & Notes*, vol. 2/04, pp. 14–15, 2004.

[3] G. Gratton, M. G. H. Coles, and E. Donchin, "A new method for off-line removal of ocular artifact," *Electroencephalography and Clinical Neurophysiology*, vol. 55, no. 4, pp. 468–484, 1983.

[4] R Development Core Team (2011), "R: A language and environment for statistical computing. R Foundation for Statistical Computing," Vienna, Austria, ISBN 3-900051-07-0, URL http://www.R-project.org/.

[5] M. Junghöfer, T. Elbert, D. M. Tucker, and C. Braun, "The polar average reference effect: a bias in estimating the head surface integral in EEG recording," *Clinical Neurophysiology*, vol. 110, no. 6, pp. 1149–1155, 1999.

[6] M. Junghöfer, T. Elbert, D. M. Tucker, and B. Rockstroh, "Statistical control of artifacts in dense array EEG/MEG studies," *Psychophysiology*, vol. 37, no. 4, pp. 523–532, 2000.

[7] M. Junghöfer, T. Elbert, P. Leiderer, P. Berg, and B. Rockstroh, "Mapping EEG-potentials on the surface of the brain: a strategy for uncovering cortical sources," *Brain Topography*, vol. 9, no. 3, pp. 203–217, 1997.

[8] M. S. Hamalainen and R. J. Ilmoniemi, "Interpreting magnetic fields of the brain: minimum norm estimates," *Medical and Biological Engineering and Computing*, vol. 32, no. 1, pp. 35–42, 1994.

[9] S. Baillet, J. C. Mosher, and R. M. Leahy, "Electromagnetic brain mapping," *IEEE Signal Processing Magazine*, vol. 18, no. 6, pp. 14–30, 2001.

[10] O. Steinsträter, S. Sillekens, M. Junghöfer, M. Burger, and C. H. Wolters, "Sensitivity of beamformer source analysis to deficiencies in forward modeling," *Human Brain Mapping*, vol. 31, no. 12, pp. 1907–1927, 2010.

[11] C. S. Herrmann, M. Grigutsch, and N. A. Busch, "EEG oscillations and wavelet analysis," in *Event-Related Potentials: A Methods Handbook*, MIT Press, Cambridge, Mass, USA, 2005.

[12] C. Tallon-Baudry, O. Bertrand, C. Delpuech, and J. Pernier, "Oscillatory γ-band (30–70 Hz) activity induced by a visual search task in humans," *Journal of Neuroscience*, vol. 17, no. 2, pp. 722–734, 1997.

[13] B. J. Roach and D. H. Mathalon, "Event-related EEG time-frequency analysis: an overview of measures and an analysis of early gamma band phase locking in schizophrenia," *Schizophrenia Bulletin*, vol. 34, no. 5, pp. 907–926, 2008.

[14] Donders Institute for Brain, Cognition and Behavior, "FieldTrip: a Matlab software toolbox for MEG and EEG analysis," 2010, http://www.ru.nl/neuroimaging/fieldtrip.

[15] A. Delorme and S. Makeig, "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis," *Journal of Neuroscience Methods*, vol. 134, no. 1, pp. 9–21, 2004.

[16] F. Perrin, J. Pernier, O. Bertrand, and J. F. Echallier, "Spherical splines for scalp potential and current density mapping," *Electroencephalography and Clinical Neurophysiology*, vol. 72, no. 2, pp. 184–187, 1989.

[17] G. Huiskamp, "Difference formulas for the surface Laplacian on a triangulated surface," *Journal of Computational Physics*, vol. 95, no. 2, pp. 477–496, 1991.

[18] R Development Core Team (2011), "R: A language and environment for statistical computing," in *R Foundation for Statistical Computing*, Vienna, Austria, ISBN 3-900051-07-0, URL http://www.R-project.org/.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 156869, 9 pages
doi:10.1155/2011/156869

Research Article

FieldTrip: Open Source Software for Advanced Analysis of MEG, EEG, and Invasive Electrophysiological Data

Robert Oostenveld,[1] Pascal Fries,[1,2] Eric Maris,[1] and Jan-Mathijs Schoffelen[1]

[1] *Donders Institute for Brain, Cognition and Behaviour, Centre for Cognitive Neuroimaging, Radboud University Nijmegen, 6500 HB Nijmegen, The Netherlands*
[2] *Ernst Strüngmann Institute and Max Planck Society, D-60528 Frankfurt, Germany*

Correspondence should be addressed to Robert Oostenveld, r.oostenveld@donders.ru.nl

Received 26 August 2010; Accepted 18 October 2010

Academic Editor: Sylvain Baillet

This paper describes FieldTrip, an open source software package that we developed for the analysis of MEG, EEG, and other electrophysiological data. The software is implemented as a MATLAB toolbox and includes a complete set of consistent and user-friendly high-level functions that allow experimental neuroscientists to analyze experimental data. It includes algorithms for simple and advanced analysis, such as time-frequency analysis using multitapers, source reconstruction using dipoles, distributed sources and beamformers, connectivity analysis, and nonparametric statistical permutation tests at the channel and source level. The implementation as toolbox allows the user to perform elaborate and structured analyses of large data sets using the MATLAB command line and batch scripting. Furthermore, users and developers can easily extend the functionality and implement new algorithms. The modular design facilitates the reuse in other software packages.

1. General Overview

FieldTrip is a MATLAB-toolbox for the analysis of MEG, EEG, and other electrophysiological data, which is freely available from http://www.ru.nl/neuroimaging/fieldtrip under the GNU public license. The development of FieldTrip started in 2003 at the F.C. Donders Centre for Cognitive Neuroimaging and up to today it continues to be actively developed at the Donders Institute for Brain, Cognition and Behaviour of the Radboud University Nijmegen, the Netherlands, together with collaborating researchers and institutes.

The software is fully implemented in MATLAB, a high-level technical computing language and interactive environment for algorithm development, data analysis, and visualization, which is available for all commonly used computer platforms (http://www.mathworks.com). MATLAB is widely known and used in the neuroimaging community. Although MATLAB is relatively expensive, the investment is easily compensated by the rich feature set and flexibility it provides.

The FieldTrip toolbox consists of approximately 108 high-level and 858 low-level functions with in total 103227 lines of code. The main focus is on the analysis of noninvasive and invasive electrophysiological data, including spike recordings, but in theory any time series data (e.g., BOLD or NIRS time courses) can be analysed. The toolbox supports reading data from a large number of different file formats (Table 1). Supported functionality includes algorithms for data preprocessing, event-related field/response analysis, parametric and nonparametric spectral analysis, forward and inverse source modelling, connectivity analysis, classification, real-time data processing, and statistical inference. Finally, the toolbox contains a module allowing for peer-to-peer distributed computing. The structure of the toolbox with its modules is shown schematically in Figure 1.

An important goal of the FieldTrip project is to provide a common platform for experimental scientists and methods developers. The FieldTrip toolbox allows experimental scientists to have access to state-of-the-art data analysis algorithms. For methods developers it facilitates their algorithms to be applied to a large variety of experimental data.

The organization of the FieldTrip project facilitates a highly dynamic development model with a rapid availability of software updates to the user. This is realized by a daily

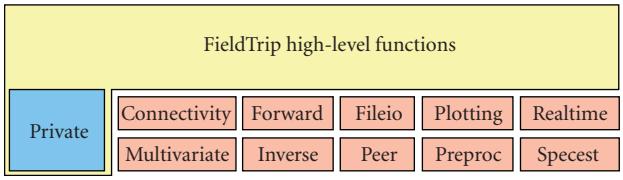


Figure 1: The structure of the toolbox.

release of the latest version on an FTP-server. Next to this, the documentation is fully available online as a wiki (http://www.ru.nl/neuroimaging/fieldtrip), which promotes active contributions of both users and methods developers. The FieldTrip wiki contains a large amount of documentation for facilitating the use of the toolbox, including tutorial documentation, answers to frequently asked questions and example MATLAB code. Finally, there is an active e-mail discussion list, with approximately 650 subscribers (state of August 2010).

The focus of this paper is on features that discriminate FieldTrip from other (publicly available) toolboxes as described elsewhere in this issue. We will first describe FieldTrip from the user's perspective, followed by a description from the developer's perspective. Both sections highlight important features relevant for the specific target group. Subsequently, specific features of the different modules are summarized. This paper ends with some concluding remarks on the FieldTrip project as a whole.

2. The User Perspective

2.1. No Graphical User Interface. An important feature of the FieldTrip toolbox is that it does not have a Graphical User Interface (GUI). Instead, the user is interacting directly with the functions on the MATLAB command line or in scripts. Consequently, users need to have some basic knowledge of MATLAB in order to fruitfully use the toolbox. Although this requires an initial investment from the side of the user, it allows for very flexible combination of the functions to suit specific analysis needs.

The FieldTrip toolbox consists of high- and low-level functions. The high-level functions provide a consistent and easy-to-use interface of the functionality to the users, enabling them to do the analysis in well-defined steps. The low-level functions implement the core functionality, but are not designed to be used by the common neuroscience researcher and do not provide an easy programming structure to implement a complete analysis of the data. The low-level functions are largely hidden from the regular end users in private directories.

2.2. Analysis Scripts to Mix and Match. Practically, users start by writing an analysis script, in which they mix and match the FieldTrip high-level functions according to the experimental research question. A script consists of a sequence of FieldTrip function calls, each of which performing a specific part of the analysis pipeline. If required, the users extend the analyses with their own code. The content and style of analysis scripts highly depend on the expertise and programming skills of the user. In general the resulting scripts can be thought of as (parts of) analysis protocols. The scripts can be easily used for batch processing, allowing for a convenient application of the same analysis protocol to multiple subjects or experiments. Also, scripts can be exchanged between users, and between students and their supervisors, facilitating collaboration and knowledge transfer.

Table 1: File formats supported by FieldTrip. Less common file formats are excluded from this listing but can be found on the website.

Class of data	Manufacturer/file format
MEG file formats	CTF/VSM
	Neuromag/Elekta
	BTi/4D Neuroimaging
	Yokogawa/KIT
	Chieti ITAB system
EEG file formats	BrainProducts/BrainVision
	NeuroScan
	Electrical Geodesics, Inc.
	Megis software/BESA research
	Biosemi
	BCI2000
	ANT/EEProbe
	Curry
	Micromed
	Nexstim
	European data format
	Generic standard formats
Anatomical MRI formats	Dicom
	NIfTI
	Analyze
	MINC
	AFNI
Animal electrophysiology file formats	Neuralynx
	Plexon
	Tucker Davis Technology
	Cambridge Electronic Design

2.3. A Typical FieldTrip Function Call. High-level FieldTrip functions have a well-defined function-call interface. The input to a particular FieldTrip function consists of one or more MATLAB structures: a configuration structure, optionally followed by one or more data structures. The input configuration structure contains the options or parameters that specify how the data will be processed by the function and/or how the algorithm will behave in detail. The input

data structure is usually the output of a FieldTrip function that was called earlier in the analysis pipeline (see below).

2.4. Configuration Structure. The specification of the parameters in the configuration structure follows the user's perspective: channels are for example indicated with their label and physical quantities are expressed in SI-units (e.g., frequency in Hz). Configuration parameters are stored in fields that express their meaning in human readable names. If possible, default values will be assigned to parameters that have not been specified by the end user.

2.5. Output of the FieldTrip Function. The output of a FieldTrip function is a MATLAB structure containing the processed data. This data structure also includes the configuration field that was used for the computations inside the function, allowing the user to inspect the details of the analysis, for example, the default configuration settings that were used. Some FieldTrip functions do not produce output data, but rather a figure displaying the data. A small set of FieldTrip functions generates neither data, nor figures, but extends the input configuration structure.

2.6. Definition of Data Structures. FieldTrip makes use of a number of well-defined data structures which are designed to be parsimonious, yet complete. They contain the numeric representation of the data in combination with the information necessary to interpret this numeric data. There are certain types of data structures for the different representations of the data. For example, segmented sensor-level time domain data is stored in a structure of data type "raw". Structures of this data type consist of a cell-array "trial", in which each cell contains a Channels × Timepoints matrix, a cell-array "label", referring to the label of each of the channels, and a cell-array "time", in which each cell contains the 1 × Timepoints vector, providing temporal information for each of the samples in each of the trials (Figure 2(a)). Figure 2(b) shows an example of a structure of data type "freq".

2.7. Analysis Scripts for Step-by-Step Analysis Are Protocollike. As mentioned before, analysis scripts usually contain a sequence of FieldTrip function calls. Each analysis step is usually performed by a single high-level FieldTrip function. To illustrate this, the following paragraphs and Figure 3 describe an analysis pipeline, showing the one-to-one mapping between a conceptual analysis step, and a high-level FieldTrip function. Figure 4 gives an impression of the corresponding analysis script.

2.7.1. Define Data Segments of Interest. A typical analysis starts with reading and segmenting the data such that the experimental conditions are represented as trials in a data structure. For simple experimental designs, segmenting the data can be done using a standard function that is included. For complex experimental designs, the user can provide his or her own function that decodes the sequence of triggers. Specific to FieldTrip is the possibility to create and analyze

```
data =
      trial:  {1x100 cell}
       time:  {1x100 cell}
      label:  {275x1 cell}
        hdr:  [1x1 struct]
       grad:  [1x1 struct]
        cfg:  [1x1 struct]
>> data.time (1)
ans =
    [1x600 double]
>> data.trial (1)
ans =
    [275x600 double]
```

(a)

```
freq =
     powspctrm:  [275x10x50 double]
       dimord:  'chan_freq_time'
        label:  {275x1 cell}
         freq:  [1x10 double]
         time:  [1x50 double]
          cfg:  [1x1 struct]
```

(b)

Figure 2: Data representation examples. (a) Epoched time domain, sensor-level data. (b) Time-frequency representation of sensor-level data.

segments of variable length. One can think of segmenting the data as inverting the implementation of the experimental design in the stimulus presentation software. The definition of the boundaries of the relevant data segments is generated by ft_definetrial.

2.7.2. Identify and Remove Artifacts. Once the interesting segments of data have been identified, one may want to identify artifacts in the data that would affect the quality of the analysis results. Subsequently, the user can either remove the affected segments from the data altogether, or remove the artifact from the data by applying a linear projection.

The function ft_rejectartifact allows for semiautomatic detection of well-defined artifacts such as eye blinks, muscle contractions, or MEG SQUID jumps. With a minimum of user interaction artifacts are identified by thresholding the data after processing the data to increase the sensitivity to pick up the characteristics of the specific artifact. For example, MEG SQUID jumps are easily detected after applying a median filter to the data. Alternatively, users can use the ft_databrowser function, allowing them to browse through the data and manually identify data segments containing artifacts (Figure 5).

To project out artifacts with a characteristic spatial topography, such as eye blinks or cardiac activity, the ft_componentanalysis function can be used. This implements a variety of blind source separation methods, such as

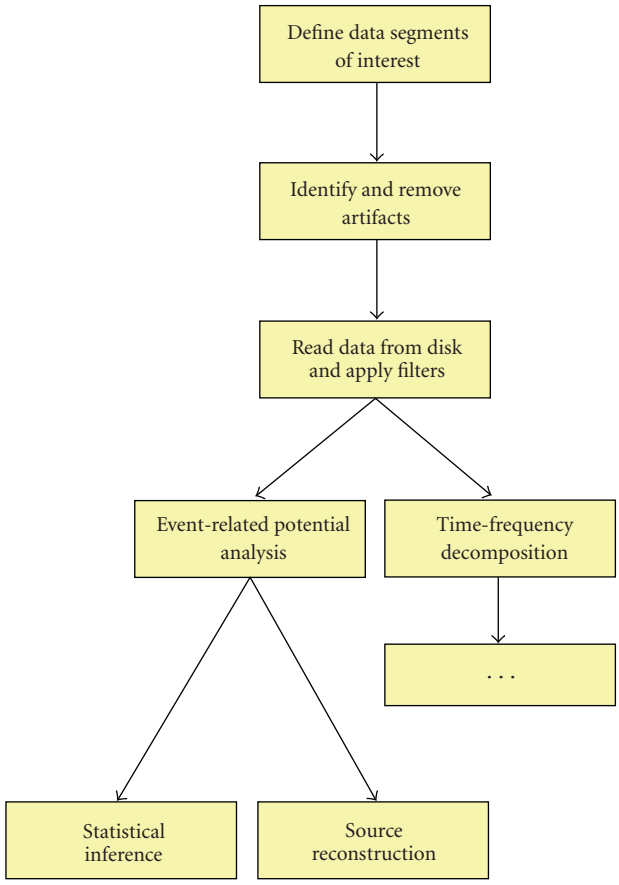


Figure 3: Example analysis pipeline.

```
% Define data segments of interest
cfg = [];
cfg.dataset                 = Subject01.dat ;
cfg.trialdef.eventtype      = TRIGGER ;
cfg.trialdef.eventvalue     = 4;
cfg.trialdef.prestim        = 0.5;
cfg.trialdef.poststim       = 1.0;
cfg = ft_definetrial(cfg);

% Identify and remove artifacts
cfg.artfctdef.eog.channel = { EOGv  EOGh };
cfg = ft_rejectartifact(cfg);

% Read data from disk and apply filters
cfg.dftfilter  = yes ;
cfg.dftfreq    = [50 100 150];
cfg.blc        = yes ;
cfg.blcwindow  = [-0.5 1];
data = ft_preprocessing(cfg);

% Time-frequency decomposition
cfg = [];
cfg.method     = mtmconvol ;
cfg.output     = pow ;
cfg.foi        = [8:2:26];
cfg.toi        = -0.25:0.05:0.75;
cfg.t_ftimwin  = 0.5 ones(1,10);
cfg.tapsmofrq  = 4 ones(1,10);
cfg.taper      = dpss ;
freq = ft_freqanalysis(cfg, data);

...
```

Figure 4: Example analysis script.

independent component analysis (ICA, based on code from the EEGLAB toolbox [1]), and principal component analysis (PCA).

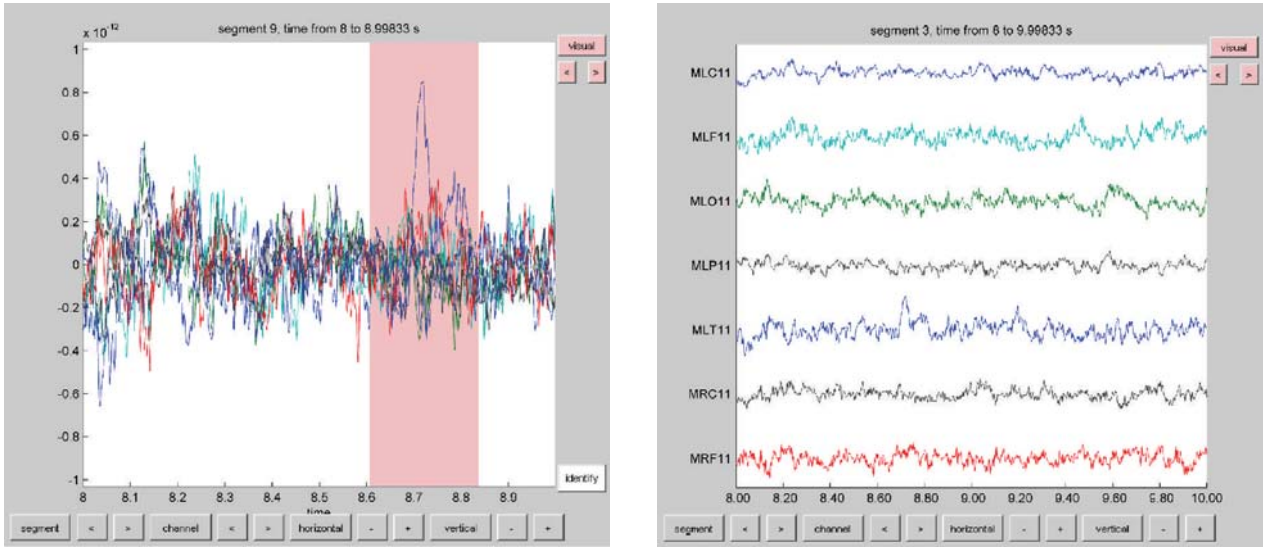
2.7.3. Read Data from Disk and Apply Filters. The ft_preprocessing function is used to read the interesting segments of data from disk into the MATLAB workspace and to apply various processing steps to the raw data such as filtering, rereferencing and baseline correction. The input to ft_preprocessing is a single configuration structure, specifying the filename of the raw data file, as well as the definition of the segments of interest. In addition, the configuration contains the instructions for the various optional processing steps.

2.7.4. Event-Related Potential Analysis. Once the segmented data are in the MATLAB workspace, the next step could be to average across trials of a particular experimental condition to obtain the event-related field (ERF) using the function ft_timelockanalysis. Figure 7 gives some examples of visualizing the ERFs.

2.7.5. Time-Frequency Decomposition. Alternative to the analysis of event-related fields, the experimental question may warrant the data to be analysed in the frequency domain. The transformation from the time domain into the frequency domain is achieved by ft_freqanalysis.

2.7.6. Source Reconstruction. Reconstructing the location and the strength of the underlying neuronal activity can either be done with ft_sourceanalysis or ft_dipolefitting. The latter function implements a nonlinear optimization algorithm that fits a prespecified number of dipoles to the data [2]. The ft_sourceanalysis function implements distributed source models, such as minimum norm estimates (MNE) [3], and beamformers for time-domain and frequency-domain data [4, 5]. The source space can be defined either as a three-dimensional regular grid, or can be based on a triangulation of the cortical sheet. There is no functionality in FieldTrip for the creation of cortical meshes, since excellent and freely available toolboxes such as FreeSurfer [6] already exist. Distributed source data and beamformer maps as well as statistically transformed derivations from these maps can be readily visualized in combination with anatomical information using the ft_sourceplot function (Figure 6). Functional data can be statistically thresholded and plotted on top of the anatomy, using opacity mapping. Threedimensional volumetric data can be rendered onto a template or individual cortical mesh (Figure 6(b)).

2.7.7. Statistical Inference. Usually the final step in a particular analysis stream is the assessment of statistical significance of the observed experimental effect, either at the level of a single subject or across subjects. At this point in the analysis,



(a) (b)

Figure 5: Visualization of data with the ft_databrowser function. (a) Display of a set of sensors in the "butterfly" mode, showing the possibility to select segments of interest, for example, to identify artifacts (pink box). (b) Display of a set of sensors in the "vertical" mode.

the data can be expressed in the time domain, frequency domain, or time-frequency domain. Furthermore, the data can either be represented at the sensor or at the source level. Independent of the data representation, FieldTrip uses the same underlying code to assess significance using parametric or nonparametric algorithms [7] for statistical inference.

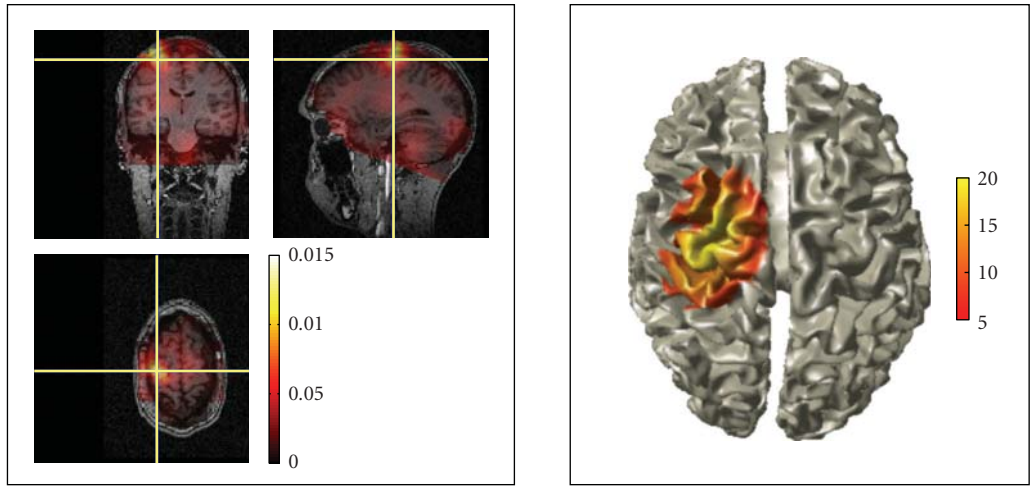
2.8. Handling the Data. An important concept in FieldTrip is that the data are in the hands of the end users. The data flows through the different FieldTrip functions and is transformed along the way. After each transformation the data corresponds to a variable, that is, a data structure in the MATLAB workspace, and the user can optionally save it to disk. The data serving as the input to a particular FieldTrip function is not included in its output. The user has to explicitly manage the data, that is, assign meaningful variable names and save the variables to disk as a MATLAB mat file, especially if intermediate analysis results need to be revisited. In addition to saving workspace variables to mat files, analysis results can be exported to a number of non-MATLAB file formats that are supported by external software. Sensor-level electrophysiological data can, for example, be saved in EDF and BrainVision Analyzer format, source reconstructed volumetric data can be saved in NIfTI format.

Although the data at the different levels of the analysis are not kept within a single structure, the history of the analysis is still present at any stage. Each output data structure of a FieldTrip function contains a nested configuration field. This field not only holds the parameters used to generate the data at the present level, but also contains the parameters used to process the data at all previous levels. In this way, information about the processing history is present at any level of the analysis pipeline.

2.9. Batch Processing. For most cognitive experiments the data from many subjects is required. Recent technological developments allow for recordings with more channels, leading to larger data sets. Therefore, despite advancements in computational power, the analysis of MEG/EEG data remains computationally demanding and takes a significant amount of time. To analyze experiments with a large number of subjects, batch processing is convenient and often necessary to systematically explore the outcome of the analysis given a particular set of parameters.

One of the preferred ways of using batch processing in a FieldTrip analysis is to start with the construction of a single script containing the full analysis pipeline for a single subject. During the implementation of this script, the user extends the script in the MATLAB editor and uses copy-and-paste to execute segments of the script. Once the user is satisfied with the sequence of analysis steps and with the parameter settings, this script can be converted into batch analysis. This can be implemented by breaking the single script into separate components, each of which resulting in an intermediate result that the user wants to inspect and/or save to disk. By adding a for loop around each of these components, the whole analysis pipeline can easily be executed for all subjects. Parameters that are specific to the individual subjects can be put in an additional subject-specific script, which is evaluated inside the batch component scripts. The whole batch can be easily reevaluated with different parameter settings.

2.10. Visualizing Analysis Results. MATLAB contains a variety of high-quality and multipurpose plotting tools and visualization of (intermediate) analysis results is often possible using these standard MATLAB plotting functions. Important for this is that the numeric representation of



(a) (b)

Figure 6: Visualization of source-reconstructed data. (a) Three-dimensional orthographic rendering of corticomuscular coherence with opacity mapping. (b) Surface rendering of statistically thresholded corticomuscular coherence after Z-transformation.

the data is easily accessible in the FieldTrip datastructures. However, complex analysis results and the multidimensional nature of some data representations sometimes prohibit easy visualization or exploration. To this end FieldTrip contains several high-level plotting functions for channel and source level data representations. For example ft_multiplorTFR allows the user to visualize the spatiotemporal spectral data and to interactively explore all three dimensions by making selections of channels and along the time and/or frequency axis.

Examples of the graphical output of some plotting functions are shown in Figures 6 and 7. Sensor-level data can be visualized by interpolating it on a two-dimensional projection of the sensor positions, for example, to look at the spatial distribution of specific ERF components (Figure 7(a)), or the specific oscillatory components in the time-frequency representation (TFR) of the data. Another method to visualize sensor-level data is to plot the complete ERF or TFR at each sensor position (Figures 7(c) and 7(d)) or to plot the averaged ERF or TFR over a subset of channels (Figure 7(b)). Source-reconstructed data can be visualized in combination with anatomical information using the ft_sourceplot function using orthogonal MRI slices or a 3D surface rendering of the cortical sheet (Figure 6). Relevant for exploring the data is the interactive option, enabling the user to select with the mouse regions-of-interest in time, frequency, and/or space. Finally, the ft_databrowser (Figure 5) can be used to interactively explore time-domain sensor-level data, at the same time allowing for the visually guided specification of artifacts.

3. The Developer's Perspective

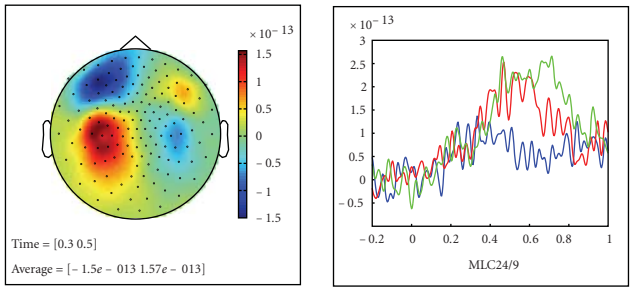
Contrary to other software where the GUI provides the central structure for the end user and, consequently, in which the developer has to adhere to the GUI structure, FieldTrip is specifically targeted at being a toolbox rather than an application. The functions in the toolbox are implemented at a level that allows them to be used in (batch) scripts, but at the same time to be called from other MATLAB-based software.

Working in a high profile scientific setting requires the experimental scientists to keep up with the latest methodological developments. Therefore, the distinction between user and developer is often not so clearcut. As already mentioned, the FieldTrip project aims at providing a common platform for end users and for methods developers, but also tries to be useful for researchers in between.

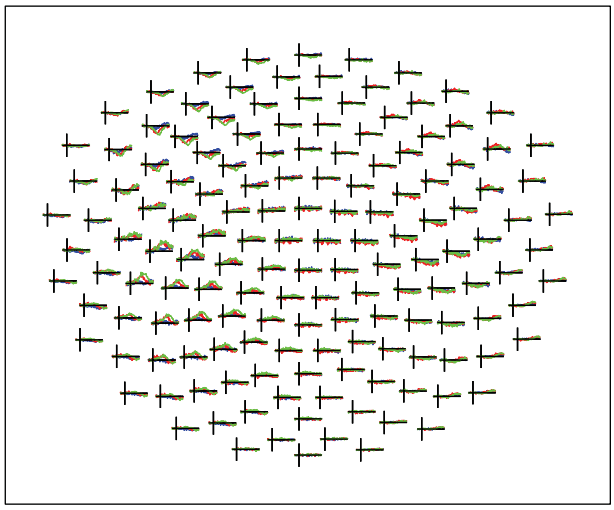
FieldTrip started to be developed in the context of a young and rapidly growing neuroimaging centre. As a consequence, the researchers involved were constantly pushing for improved and extended functionality. This led to a development model that is still in use to date. Characteristic is the continuously evolving codebase as opposed to fixed-point releases. The users rely on the latest daily released version. The developers take great care in ensuring continuity by providing backward compatibility in the many, but small, steps that the codebase takes. This is facilitated by the separation of FieldTrip into high-level functions with a stable function-call interface and well-defined data structures, and low-level functions that can easily be modified and extended to meet the evolving requirements.

3.1. Contributing Code to FieldTrip. In general, methods developers may want to contribute code to FieldTrip because it offers a unique opportunity to get innovative analysis methods applied to a large variety of real-world experimental data. Furthermore, contributing new methods results in feedback from the user community, which can result in new insights into the methods themselves.

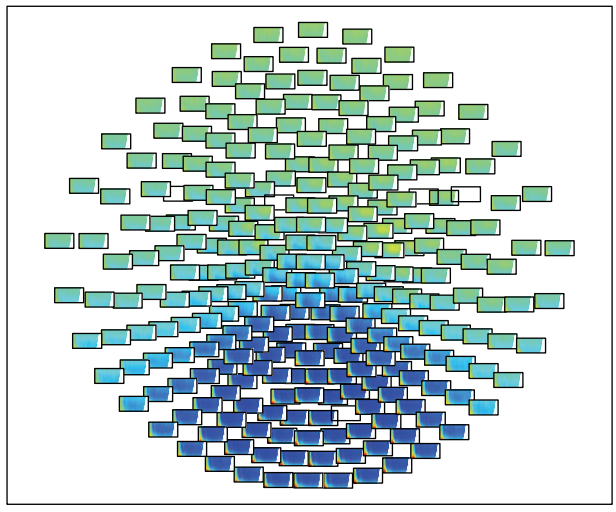
For a methods developer it is easy to add a new high-level function, because of the parsimonious data representation,



(a) (b)



(c)



(d)

Figure 7: Visualization of multidimensional data. (a) Topographical representation of a specific temporal component of the ERF. (b) Single sensor display of an ERF. (c) Topographical display of sensor-level ERFs in three experimental conditions. (d) Topographical display of sensor-level TRFs.

and because of the one-to-one relation between analysis steps and single FieldTrip functions. By utilizing an existing FieldTrip data structure as input to the function, and by remaining close to another data structure as output format for the function, the method developer does not have to be concerned with specific data formats and subsequent processing and visualisation of the analysis results. By combining the separate functions, the FieldTrip user has an exponential increase in possible combinations of functionality.

3.2. Using FieldTrip Code Elsewhere. During the last two years the code has been modularised to clarify the layered organisation of some of the low-level functionality. Having a well-defined structure for the modules simplifies the maintenance of the code. Furthermore, it facilitates the collaboration with methods developers and with developers of other software. Each module contains intermediate- and low-level functions related to a particular type of computation, for example, the forward module contains functions for the computation of lead fields, and the fileio module contains functions for reading in raw data of various file formats. The function-call interface for the intermediate-level module functions works with key-value pairs to specify the behaviour, rather than with a configuration structure. Compared to the use of a single configuration structure with parameters in the high-level FieldTrip functions, key-value pairs represent a more widely used programming style.

The modular design facilitates reuse of source code in other software. For example, SPM8 and FieldTrip share the same fileio module, which is clearly beneficial for both the end users and the code developers: code does not need to be written twice, and both the SPM and FieldTrip users can access a comprehensive set of data formats. Apart from sharing the fileio, forward and preproc modules, SPM uses FieldTrip for various MEG and EEG analysis algorithms, whereas FieldTrip uses SPM for processing anatomical MRI data for the purpose of spatial normalisation and segmentation. Besides the active collaboration with the SPM developers, FieldTrip shares code with other noncommercial and commercial software such as EEGLAB, BESA, BCI2000 and others.

4. Specific Details Related to the Different Modules

The following part provides some additional details related to the functionality of the different modules, without the aim of being exhaustive.

4.1. Fileio. The fileio module implements a consistent interface to electrophysiological sensor level data from many acquisition systems by separating the information in the datasets into header information, events (such as triggers), and the actual recorded signals. The intermediate-level reading functions perform file format detection and automatically select the appropriate low-level reading functions. The different file formats supported by this module are shown in Table 1.

4.2. Preproc. The preproc module contains algorithms for time domain filtering, rereferencing, baseline correction, detrending, and other functions that are usually associated with the preprocessing of raw data.

4.3. Specest. Nonparametric (Fourier transform based) and parametric spectral analysis methods are implemented in the specest module. It contains algorithms for estimating the power and/or phase of oscillatory components using (time-) frequency decomposition, wavelets, and multitapers [8].

4.4. Connectivity. This module provides functionality to compute measures of bivariate and multivariate connectivity. Originally, FieldTrip evolved from code that was written to analyze sensor and source-level coherence in MEG. Past years have witnessed an increased interest in studying connectivity, which has led to the emergence of open source toolboxes specifically designed for this purpose [9]. The FieldTrip connectivity module focuses on the analysis of connectivity in the frequency domain. Various connectivity metrics are available, such as coherence, phase locking value [10], imaginary part of coherency [11], phase slope index [12], partial directed coherence [13], directed transfer function [14], and Geweke's extension of Granger causality to the frequency domain [15].

4.5. Forward. This module contains functions to compute lead fields, that is, the solutions to the forward problem. Various algorithms are implemented, including for MEG single sphere [16], overlapping spheres [17], and a spherical harmonics approximation of realistic geometries [18]. For EEG, single and multiple concentric sphere models and the boundary element method (BEM) are available [19, 20].

4.6. Inverse. Different source reconstruction algorithms are available for the estimation of the location and strength of neuronal activity, including dipole fitting based on nonlinear optimization, [2] scanning methods such as minimum variance beamformers in the time and frequency domain [4, 5], and linear estimation of distributed source models [3].

4.7. Multivariate. The multivariate module contains a wide range of machine learning algorithms for classification, such as linear discriminant analysis, support vector machines, Bayesian networks, Gaussian mixture models, and groupwise logistic regression [21]. The classification algorithms can be used for offline single trial analysis, and for online brain-computer interface (BCI) applications.

4.8. Plotting. This module contains intermediate-level functions that facilitate the visualization of complex data such as multichannel time-frequency decompositions or source reconstructions.

4.9. Realtime. Although MATLAB itself is a largely single-threaded application that provides an interpreted programming environment, it is highly suited for rapid application development and is sufficiently fast for real-time analysis of multichannel EEG and MEG data. The core functionality of the real-time module is provided by the FieldTrip buffer, a multithreaded network transparent TCP server that allows the acquisition client to stream data in small blocks, while at the same time allowing analysis of the data in MATLAB. This module allows the user to build BCI systems.

4.10. Peer. Efficient use of available computation resources speeds up the often time-consuming analysis of electrophysiological data. The peer module boasts a zero configuration, dynamically adjusting peer-to-peer network that allows for sharing computational resources among multiple users. It allows the user to distribute multiple computational jobs in parallel over multiple MATLAB sessions running on a single computer, or, just as easily, running on different computers in the network.

5. Concluding Remarks

The features of the FieldTrip software that set it apart from the other free and commercially available EEG/MEG analysis software are that it allows for analyzing experimental designs in which the trial duration varies, the elaborate implementation of spectral analysis using multitapers, statistical inference using nonparametric permutation tests and source reconstruction with beamformer methods. There is an active involvement of the users through the e-mail discussion list and online wiki documentation system.

The open source development model of FieldTrip has proven to be very effective, on the one hand creating a large and well-tested collection of MATLAB functions, on the other hand resulting in a large contribution to experimental neuroscience. The latter is exemplified both by the large number of publications in which FieldTrip is used (>100) and by the high impact factor of those publications, among others in Science, Nature, Neuron, Current Biology, PNAS, and the Journal of Neuroscience. The open nature of the FieldTrip project has resulted in a community with an active exchange of scientific ideas between users and developers.

Besides the impact that FieldTrip itself has on experimental neuroscience, just as important is the contribution of the open source model to scientific research. There is a healthy competition between different EEG/MEG software packages, which results in an ongoing drive for improved methods and improved usability of the software. The open source development model fits very well with the scientific approach of providing the information required for obtaining reproducible findings. Sharing the source code pushes forward both the fields of neuroscience methods and experimental neuroscience.

When we embarked on our journey to create, use, and share new ideas for the analysis of electrophysiological data, we did not yet realise the full potential of FieldTrip. Looking back over the short, but intense history of the project, the most rewarding are the scientific and personal fulfilment resulting from the interaction with all the great researchers that we got to know through this project.

Acknowledgments

Ever since its inception in 2003, the FieldTrip project has been developed by a constantly changing, but highly motivated team. Over the years many people have contributed in one way or another, ranging from code development, writing documentation and answering questions on the e-mail discussion list, providing funding, reporting bugs in the code, and generating ideas for further improvements. Without the help of the people at the Donders Centre for Cognitive Neuroimaging, and all external people that were bold enough to give it a try, FieldTrip would not have been what it is today. The authors would explicitly like to thank Alexandre Gramfort, Ana Todorovic, Andre Bastos, Arjen Stolk, Arnaud Delorme, Christian Hesse, Cristiano Micheli, Dennis Pasveer, Gavin Paterson, Geerten Kramer, Guido Nolte, Guillaume Flandin, Hanneke van Dijk, Ingrid Nieuwenhuis, Jens Schwarzbach, Joseph Dien, Marcel van Gerven, Mark Lalancette, Markus Bauer, Markus Siegel, Martin Vinck, Michael Wibral, Nathan Weisz, Nienke Hoogenboom, Ole Jensen, Roemer van der Meij, Sandra van Aalderen-Smeets, Saskia Haegens, Stefan Klanke, Stephen Whitmarsh, Thilo Womelsdorf, Tilmann Sander-Thömmes, Tim Engelkes, Tineke Snijders, Tom Holroyd, Vladimir Litvak, and all others whom we forgot to mention. The authors gratefully acknowledge the support of the BrainGain Smart Mix Programme of the Netherlands Ministry of Economic Affairs and the Netherlands Ministry of Education, Culture and Science.

References

[1] A. Delorme and S. Makeig, "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis," *Journal of Neuroscience Methods*, vol. 134, no. 1, pp. 9–21, 2004.

[2] M. Scherg, "Fundamentals of dipole source potential analysis," in *Advances in Audiology. Auditory Evoked Magnetic Fields and Electric Potentials*, vol. 6, pp. 40–69, Karger, Basel, Switzerland, 1990.

[3] M. S. Hämäläinen and R. J. Ilmoniemi, "Interpreting magnetic fields of the brain: minimum norm estimates," *Medical and Biological Engineering and Computing*, vol. 32, no. 1, pp. 35–42, 1994.

[4] B. D. Van Veen, W. Van Drongelen, M. Yuchtman, and A. Suzuki, "Localization of brain electrical activity via linearly constrained minimum variance spatial filtering," *IEEE Transactions on Biomedical Engineering*, vol. 44, no. 9, pp. 867–880, 1997.

[5] J. Gross, J. Kujala, M. Hämäläinen, L. Timmermann, A. Schnitzler, and R. Salmelin, "Dynamic imaging of coherent sources: studying neural interactions in the human brain," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 98, no. 2, pp. 694–699, 2001.

[6] A. M. Dale, B. Fischl, and M. I. Sereno, "Cortical surface-based analysis—I. Segmentation and surface reconstruction," *NeuroImage*, vol. 9, no. 2, pp. 179–194, 1999.

[7] E. Maris and R. Oostenveld, "Nonparametric statistical testing of EEG- and MEG-data," *Journal of Neuroscience Methods*, vol. 164, no. 1, pp. 177–190, 2007.

[8] P. P. Mitra and B. Pesaran, "Analysis of dynamic brain imaging data," *Biophysical Journal*, vol. 76, no. 2, pp. 691–708, 1999.

[9] J. Cui, L. Xu, S. L. Bressler, M. Ding, and H. Liang, "BSMART: a Matlab/C toolbox for analysis of multichannel neural time series," *Neural Networks*, vol. 21, no. 8, pp. 1094–1104, 2008.

[10] J.-P. Lachaux, E. Rodriguez, J. Martinerie, and F. J. Varela, "Measuring phase synchrony in brain signals," *Human Brain Mapping*, vol. 8, no. 4, pp. 194–208, 1999.

[11] G. Nolte, O. Bai, L. Wheaton, Z. Mari, S. Vorbach, and M. Hallett, "Identifying true brain interaction from EEG data using the imaginary part of coherency," *Clinical Neurophysiology*, vol. 115, no. 10, pp. 2292–2307, 2004.

[12] C. J. Stam, G. Nolte, and A. Daffertshofer, "Phase lag index: assessment of functional connectivity from multi channel EEG and MEG with diminished bias from common sources," *Human Brain Mapping*, vol. 28, no. 11, pp. 1178–1193, 2007.

[13] L. A. Baccalá and K. Sameshima, "Partial directed coherence: a new concept in neural structure determination," *Biological Cybernetics*, vol. 84, no. 6, pp. 463–474, 2001.

[14] M. Kamiński, M. Ding, W. A. Truccolo, and S. L. Bressler, "Evaluating causal relations in neural systems: granger causality, directed transfer function and statistical assessment of significance," *Biological Cybernetics*, vol. 85, no. 2, pp. 145–157, 2001.

[15] A. Brovelli, M. Ding, A. Ledberg, Y. Chen, R. Nakamura, and S. L. Bressler, "Beta oscillations in a large-scale sensorimotor cortical network: directional influences revealed by Granger causality," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 26, pp. 9849–9854, 2004.

[16] B. N. Cuffin and D. Cohen, "Magnetic fields of a dipole in special volume conductor shapes," *IEEE Transactions on Biomedical Engineering*, vol. 24, no. 4, pp. 372–381, 1977.

[17] M. X. Huang, J. C. Mosher, and R. M. Leahy, "A sensor-weighted overlapping-sphere head model and exhaustive head model comparison for MEG," *Physics in Medicine and Biology*, vol. 44, no. 2, pp. 423–440, 1999.

[18] G. Nolte, "The magnetic lead field theorem in the quasi-static approximation and its use for magnetoenchephalography forward calculation in realistic volume conductors," *Physics in Medicine and Biology*, vol. 48, no. 22, pp. 3637–3652, 2003.

[19] G. Adde, M. Clerc, O. Faugeras, R. Keriven, J. Kybic, and T. Papadopoulo, "Symmetric BEM formulation for the M/EEG forward problem," *Information Processing in Medical Imaging*, vol. 18, pp. 524–535, 2003.

[20] T. F. Oostendorp and A. Van Oosterom, "Source parameter estimation in inhomogeneous volume conductors of arbitrary shape," *IEEE Transactions on Biomedical Engineering*, vol. 36, no. 3, pp. 382–391, 1989.

[21] M. van Gerven, C. Hesse, O. Jensen, and T. Heskes, "Interpreting single trial data using groupwise regularisation," *NeuroImage*, vol. 46, no. 3, pp. 665–676, 2009.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 758973, 17 pages
doi:10.1155/2011/758973

Research Article

MEG/EEG Source Reconstruction, Statistical Evaluation, and Visualization with NUTMEG

Sarang S. Dalal,[1,2,3] Johanna M. Zumer,[4,5] Adrian G. Guggisberg,[6] Michael Trumpis,[7] Daniel D. E. Wong,[8] Kensuke Sekihara,[9] and Srikantan S. Nagarajan[7]

[1] *Department of Psychology, Zukunftskolleg, University of Konstanz, 78457 Konstanz, Germany*
[2] *MEG Department, CERMEP, 69500 Lyon, France*
[3] *INSERM U1028, CNRS UMR5292, Lyon Neuroscience Research Center, Brain Dynamics and Cognition Team, 69500 Lyon, France*
[4] *Donders Institute for Brain, Cognition and Behaviour, Centre for Cognitive Neuroimaging, Radboud University Nijmegen, 6500 HB Nijmegen, The Netherlands*
[5] *Sir Peter Mansfield Magnetic Resonance Centre, University of Nottingham, Nottingham NG7 2RD, UK*
[6] *Division of Neurorehabilitation, Department of Clinical Neurosciences, University Hospital of Geneva, 1211 Geneva, Switzerland*
[7] *Biomagnetic Imaging Laboratory, Department of Radiology and Biomedical Imaging, University of California, San Francisco, CA 94143, USA*
[8] *Institute of Biomaterials and Biomedical Engineering, University of Toronto, Canada M5S 3G9*
[9] *Department of Systems Design and Engineering, Tokyo Metropolitan University, Tokyo 191-0065, Japan*

Correspondence should be addressed to Sarang S. Dalal, sarang.dalal@uni-konstanz.de

Received 4 October 2010; Revised 30 November 2010; Accepted 17 January 2011

Academic Editor: Sylvain Baillet

NUTMEG is a source analysis toolbox geared towards cognitive neuroscience researchers using MEG and EEG, including intracranial recordings. Evoked and unaveraged data can be imported to the toolbox for source analysis in either the time or time-frequency domains. NUTMEG offers several variants of adaptive beamformers, probabilistic reconstruction algorithms, as well as minimum-norm techniques to generate functional maps of spatiotemporal neural source activity. Lead fields can be calculated from single and overlapping sphere head models or imported from other software. Group averages and statistics can be calculated as well. In addition to data analysis tools, NUTMEG provides a unique and intuitive graphical interface for visualization of results. Source analyses can be superimposed onto a structural MRI or headshape to provide a convenient visual correspondence to anatomy. These results can also be navigated interactively, with the spatial maps and source time series or spectrogram linked accordingly. Animations can be generated to view the evolution of neural activity over time. NUTMEG can also display brain renderings and perform spatial normalization of functional maps using SPM's engine. As a MATLAB package, the end user may easily link with other toolboxes or add customized functions.

1. Introduction

As exemplified by this special issue on open-source analysis toolboxes, many software solutions exist to suit a variety of experimental goals and level of end-user programming experience, including the option to mix and match toolboxes for different stages of processing. However, a decade ago, few options existed for analyzing magnetoencephalography (MEG) data with noncommercial open-source software, especially for more sophisticated inverse algorithms or with a graphical interface to navigate results.

Electroencephalography (EEG) analysis and corresponding software packages are dominated by sensor level processing, such as topography, evoked responses, and ICA. Source localization is more feasible with MEG data; however, many commercial packages offer only one of several basic inverse methods (dipole fitting, beamforming, and minimum-norm). Within open-source options available at present, BrainStorm (http://neuroimage.usc.edu/brainstorm) and MNESuite (http://www.nmr.mgh.harvard.edu/martinos/userInfo/data/sofMNE.php) offer similar source localization options; FieldTrip (http://fieldtrip.fcdonders.nl/)

additionally offers beamforming, and SPM8 (http://www.fil.ion.ucl.ac.uk/spm/) offers an advanced Bayesian source estimation method. However, to date, other packages do not provide a whole suite of reconstruction algorithms ranging from the simple to the complex powerful ones that have been recently published.

In 2003, the seeds of NUTMEG (Neurodynamic Utility Toolbox for Magnetoencephalo- and Electroencephalo-Graphy) were planted at the University of California, San Francisco (UCSF), with the motivation to meet several research goals, including implementation of experimental source localization algorithms and general independence from commercially provided software, as well as user extensibility for custom analyses [1]. Specific strengths of NUTMEG include: (1) choice of several inverse algorithms, including variants of popular beamforming, minimum-norm, and Bayesian inference techniques, (2) intuitive viewing and navigation of results, (3) both GUI and command-line batch use, and (4) several methods of source space functional connectivity analysis.

NUTMEG can be downloaded from http://nutmeg.berkeley.edu/. Documentation and a user's wiki are also located at this website, and users can subscribe to a mailing list which is intended as a general forum for questions related to the software itself or analysis procedures.

NUTMEG is primarily written in MATLAB (MathWorks, Natick, MA, USA). The MATLAB Signal Processing Toolbox is required for digital filter operations, and the Image Processing Toolbox is needed for (optional) graphical volume-of-interest (VOI) selection. A link with SPM8 allows activations to be overlaid onto standard orthogonal magnetic resonance imaging (MRI) slices or a rendered 3D brain volume; at present, SPM8's data analysis engine is not used. Via SPM8, activations may also be spatially normalized and displayed on an MNI template brain [2, 3]. Visualization tools in Python (http://www.python.org/) are under development and will be made available in future versions.

NUTMEG is also interoperable with other software for, for example, scrolling through and artifact rejection of sensor data (FieldTrip and Brainstorm), BEM forward models (OpenMEEG, http://openmeeg.gforge.inria.fr/), FieldTrip, SPM8, Helsinki BEM Toolbox), preprocessing (CTF MEG software, MEG International Services, Coquitlam, Canada), and ELAN (see [4]).

2. Philosophy

2.1. Need for Open Source. Releasing analysis software as open source provides a fair and effective means to distribute methodological developments made possible by public research funds, as well as to promote the spirit of academic scientific cooperation. Additionally, the open-source model allows the same analysis methods to be easily used with nearly any type of input data, regardless of equipment manufacturer. The source code, being open to the end user and the academic community at large, also becomes a transparent tool, removing any mystery as to how data is being manipulated and allowing custom modifications; any errors can be found and corrected more efficiently as well. Furthermore, both the theory and practical implementation of analysis methods for MEG/EEG data have been rapidly developing in the past two decades. Whether one is a methods researcher comparing algorithms or a cognitive scientist eager to use the latest methods, neither should have to wait the several years that it can sometimes take for a method to be released as part of a proprietary software package.

2.2. Types of Data/Experiments/Paradigms. Other functional neuroimaging modalities such as fMRI benefit from a relatively established stream of standard processing steps that facilitate learning by beginners and batch processing by more experienced users. However, with MEG and EEG, it sometimes seems that there can be as many ways of analyzing the data as there are datasets, as appropriate analyses can vary considerably according to the paradigm and the types of responses. A useful software package needs to be flexible enough to introduce the various analysis streams as they are developed and straightforward to use for routine analysis.

Experiment types that have been successfully processed with NUTMEG include (1) evoked paradigms, for example, auditory stimulation in healthy subjects [5], verbal stimulation compared between healthy subjects and schizophrenia patients [6], perturbation of self-speech perception [7], and somatosensory stimulation in humans and monkeys [8], (2) time-frequency analysis, for example, finger movements [9], visual stimulation [10], decision making [11], discrimination of tone rate modulation [12], and visually guided behavior [13], and (3) resting state and task-induced connectivity [14, 15]. Data types supported in NUTMEG include MEG, EEG [16], and intracranial EEG [17].

2.3. Integration with Other Toolboxes. It is logical for certain basic software components, such as data import/export, to be shared between different toolboxes. Nevertheless, a particular software package may excel for certain processing or analysis procedures; it would benefit other software packages to be able to call the code for such components transparently from within their own package. Throughout the description of processing steps for NUTMEG, we will describe which procedures are specifically implemented in NUTMEG and which ones take advantage of links to other software.

3. NUTMEG Processing Steps

The first step of the NUTMEG workflow (illustrated in Figure 1) involves loading in the MEG/EEG data and, if available, MRI and coregistration information. A forward lead field is computed within NUTMEG, or imported from external software. This information is all stored in a MATLAB structure that advanced users may access from the command line or with user-created scripts, facilitating links with other software. Translators between NUTMEGs structure and the FieldTrip and ELAN formats are included in the standard NUTMEG distribution. Likewise, results are stored in a separate structure, so that derived outputs such as "virtual electrode" time series can be further analyzed in MATLAB with the user's preferred tools. Results from

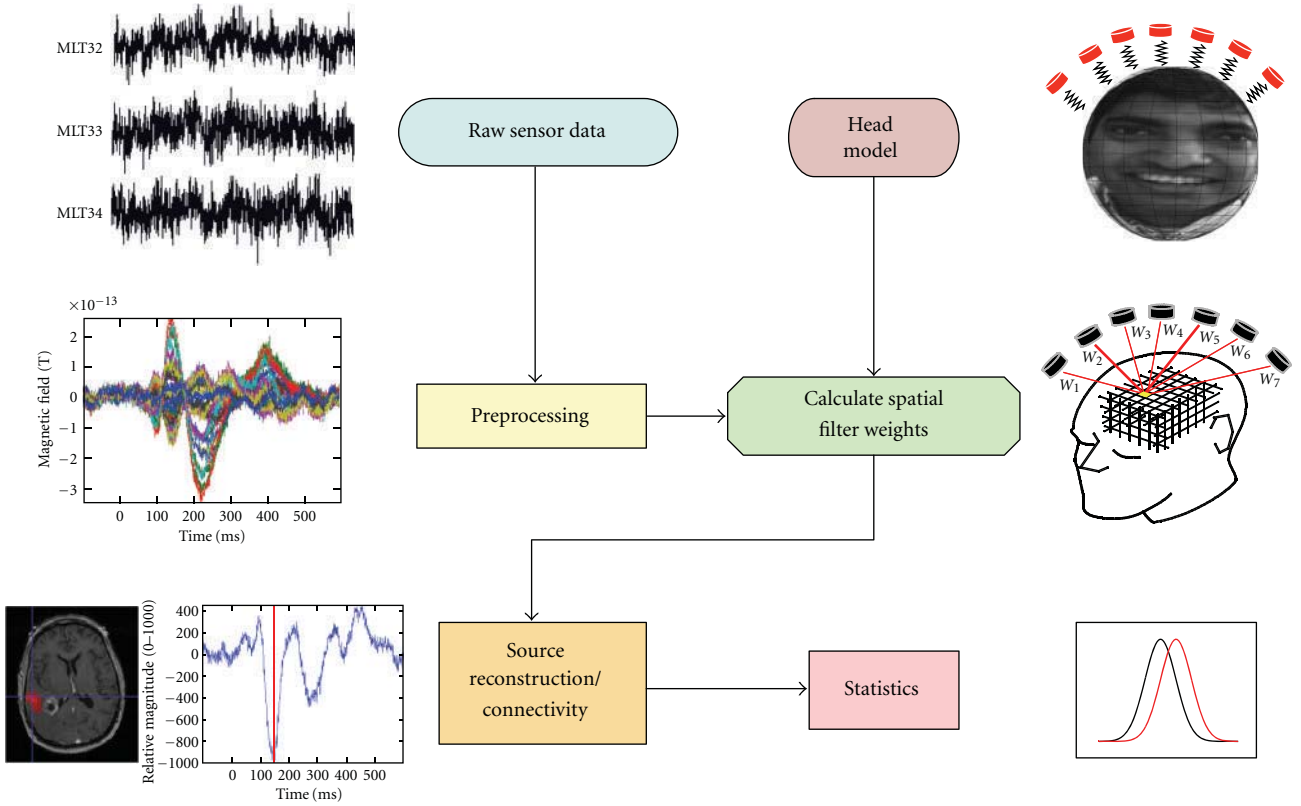


Figure 1: NUTMEG workflow. After preprocessing, sensor data is combined with head coregistration information to create a forward lead field model, which is then used to compute a source estimate either for time-series or time-frequency domains. Source space data can be further processed with single- or multisubject statistics, or for functional connectivity.

external programs can also be reformatted to allow viewing with NUTMEGs results navigator.

The main GUI (Figure 2) guides a new user through the processing steps, greying out the boxes that cannot yet be completed given the information currently provided. An advanced user may bypass the GUI and operate all steps from the command-line.

3.1. Loading Different Data Types. NUTMEG can import MEG, EEG, and intracranial EEG data from various manufacturer's systems. At present, this includes CTF, 4D/BTi, KIT/Yokogawa, and Elekta Neuromag MEG systems, as well as EEG data from BrainProducts and Micromed. Several other formats are also supported via a link with the *fileio* module of FieldTrip. Data may comprise unaveraged multiple trials, an average across trials, or continuous data.

3.2. Sensor Preprocessing. After loading the data into NUTMEG, the user may click on "View/Select MEG Channels" from the main GUI (Figure 2), which opens a new window (Figure 3). After selecting a time window of interest, the root mean square of the sensors is displayed on a 2D sensor map. Sensors can be (de-)selected for further processing. The effects of baseline removal and filtering on the sensor map can also be examined.

Preprocessing components from other software packages may optionally be used as well and imported into NUTMEG. SPM8 has especially useful tools for automated artifact rejection. For more advanced sensor or trial selection, the graphical interface from FieldTrip could be used.

3.3. Forward Methods. NUTMEG includes a built-in single sphere [18] and multisphere model for MEG [19]. The individual subject's structural MRI or digitized headshape can be loaded via the Coregistration Tool GUI (Figure 4). Here, additional information such as a spatially normalized MRI or rendered brain surface (created via SPM8) can be loaded, and fiducial positions can be imported or manually set. Furthermore, a head surface mesh can be generated within NUTMEG, which can aid with fiducial coregistration if digitized headshape measurements have been made, for example, with a Polhemus FASTRAK device (Colchester, VT, USA). If no individual subject MRI or headshape is available, the MNI template brain may be used in their place. Cortical segmentation is not used to constrain either the source locations or orientations computed within NUTMEG, as slight errors in coregistration may lead to larger errors in source estimation. Lead fields for scalp and intracranial EEG can be computed within NUTMEG, currently implemented as a simple semi-infinite homogeneous volume conductor.

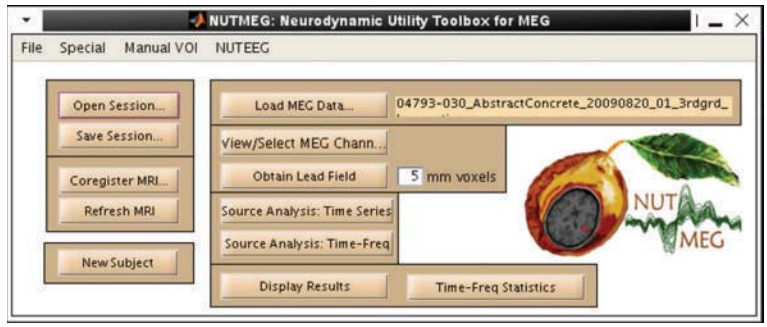


Figure 2: NUTMEG session main interface for guiding the user through processing steps and calling other GUIs for loading data and coregistration and computing forward lead field and source localizations.

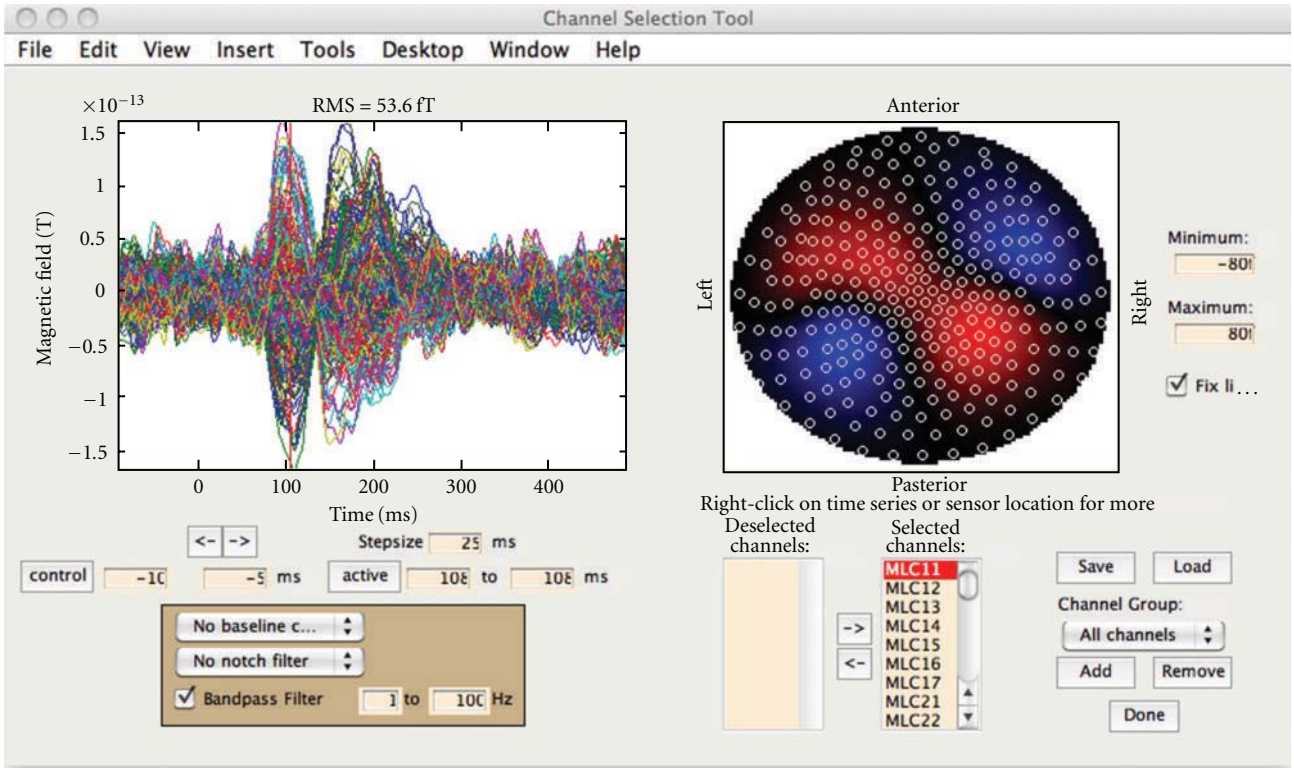


Figure 3: Sensor Preprocessing GUI. The left subplot shows the averaged time series for each sensor overlaid. The right subplot shows the RMS over a selected time window. The effects of filtering and time window selection on both the sensor time series and RMS spatial plots can be viewed here. Channels may be (de-)selected for further processing, either as individual noisy channels or groups of channels by spatial region.

Additionally, boundary element model (BEM) and finite element model (FEM) head models can be generated externally and imported for use with NUTMEG for either MEG or EEG. Currently supported external models include OpenMEEG (BEM), MNE (BEM), FieldTrip (BEM), and SMAC [20] (spherical model with anatomical constraints). A link to generate and import FEM from SimBio/NeuroFEM (https://www.mrt.uni-jena.de/simbio) is planned. Imported lead fields may be specified either with free orientations in vector form or orientation-constrained in scalar form.

After the data and coregistration information are loaded and lead field obtained, the NUTMEG main GUI (Figure 2) will make available the buttons for source estimation.

The coregistration from MEG sensors to MNI coordinates can also be used independently of NUTMEGs source localization tools in order to obtain MNI coordinates of dipole fits computed elsewhere, as shown in Zhu et al. [21].

3.4. Inverse Methods. NUTMEG can be used to localize evoked (averaged) data or induced (nonphase-locked) data.

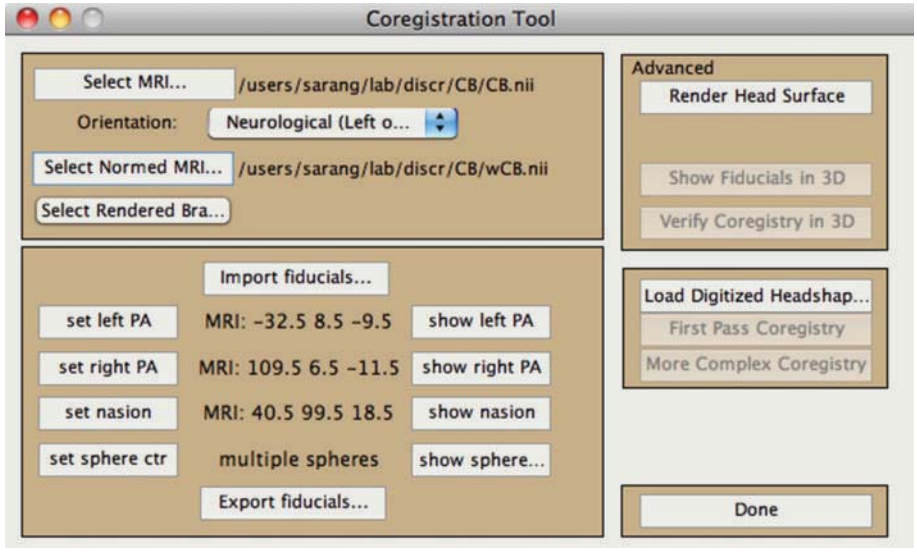


Figure 4: Coregistration Toolbox GUI allows inclusion of subject's MRI, fiducials to be loaded or set manually, spatially normalized MRI or rendered surface (created via SPM8), surface mesh, and digitized headshape points.

Certain methods are better tuned to each type of analysis. When the user clicks the button from the main GUI (Figure 2) called "Source Analysis: Time-Series," a new window appears (Figure 5). Included in this window are drop-down menus for choice of inverse method and regularization. The user has a choice about whether to use the covariance from averaged or single-trial data, for those inverse algorithms that use data covariance. The main GUI has tick-boxes for this averaging choice, for whether to create a contrast with a control time window, and for whether to send the computations to the "qsub" distributed job manager.

3.4.1. Beamformers. The most commonly used and developed class of inverse method within NUTMEG is the beamformer. This is an adaptive method which minimizes the variance at a given source location while suppressing noise from other locations [22]. It takes as inputs both the sensor data covariance and the forward lead field, represented by the basic formula

$$\mathbf{W}_r^T = \frac{\mathbf{L}_r{}^T \mathbf{R}_{yy}^{\ 1}}{\mathbf{L}_r{}^T \mathbf{R}_{yy}^{-1} \mathbf{L}_r}, \quad (1)$$

where $\mathbf{W}_r$ comprises the computed sensor weights to derive the activity at brain location r, $\mathbf{L}_r$ contains the gain at each sensor (forward model) for a source at location r, and $\mathbf{R}_{yy}$ is the sample covariance for the chosen data segment.

Many flavors of beamforming are created by the many ways to compute the data covariance estimate and lead field. These choices can be dictated by the experimental paradigm or by tradeoffs of computational intensiveness versus accuracy (in the case of a lead field). The data covariance estimate needs to be optimally tuned to the effect of interest (e.g., time window length and filter parameters) while maintaining invertibility. The sample data covariance may be computed either from an averaged evoked response or by averages of the sample covariance of each trial, as selected by the user with the tick-box on the Beamforming Tool GUI (Figure 5); also see section Regularized Beamformer for Evoked Data.

3.4.2. Eigenspace Beamformer for Evoked Data. Sekihara et al. [23] proposed the eigenspace beamformer to improve stability of reconstructions using averaged evoked responses. Based on the singular value decomposition (SVD) of the covariance matrix of the averaged data, the user defines the *signal space* from the largest few eigenvalues, rejecting eigenvalues from the remaining *noise space*. A signal space data covariance is then computed and inverted, replacing the data covariance of the weight formula (1) in the numerator while maintaining the original evoked response covariance in the denominator. This method has the advantage of improving weight computation for averaged data, focusing the result on the eigenvectors of interest, and allows for effective removal of large-amplitude phase-locked artifacts. To aid selection of desired signal space eigenvalues, the right-hand plot within the "Source Analysis: Time-Series" GUI (Figure 5) displays the relative magnitudes of the eigenvalues and the time course of the top selected eigenvectors.

3.4.3. Time-Frequency Beamformer. Dalal et al. [9] developed a method for optimized time-frequency beamforming (TFBF). NUTMEG implements this algorithm to be computed easily over a grid of many time-frequency windows, assembling the results for intuitive interactive navigation (Figures 6 and 7(b)). TFBF is based on the LCMV beamformer [22] and contrasts each active time-frequency window with a common control window. The user is encouraged to select time windows as short as possible to focus on transient and frequency-specific power changes, within the

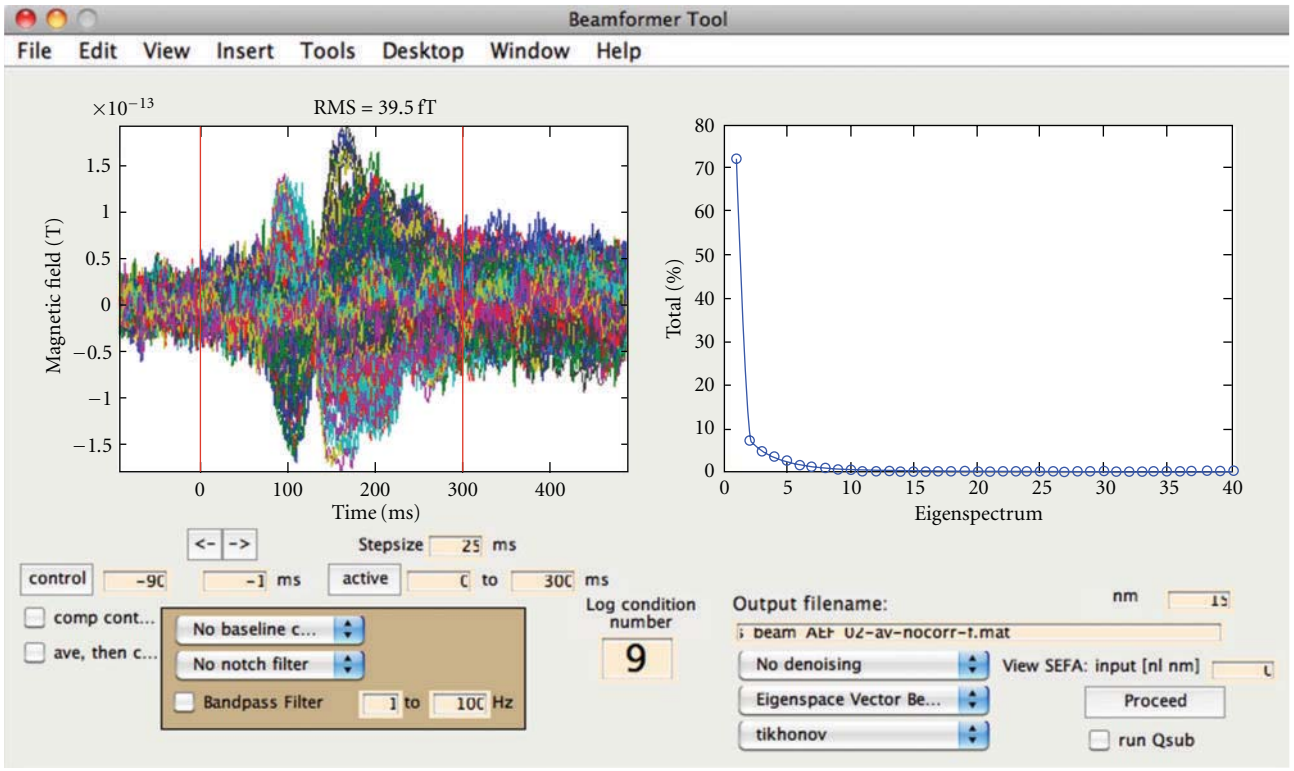


Figure 5: Time-series source estimation GUI. The left subplot shows the averaged sensor time series for each sensor. The right subplot changes depending on user selection. The eigenvalues can be plotted (as shown) for assisting the eigenspace beamformer; alternatively, cleaned sensor data and factors can be plotted if SEFA denoising is selected. The condition number of the sensor data covariance is displayed to help prevent meaningless results computed with overly high condition numbers. Other options here include filtering, time window selection, averaging before/after data covariance computation, method of source inversion, other denoising methods, sensor covariance regularization, and the option to submit the computation to a grid cluster for site-specific setups.

confines of SNR and period of oscillation for the given frequency band. Data covariance for TFBF is estimated by averaging the sample covariance from each trial.

The TFBF GUI (Figure 6) is opened from the main GUI by clicking "Source Analysis: Time-Freq" and guides the user through selecting options. These parameters can be saved and called again or run as a batch process on a single computer or a high performance computing grid.

3.4.4. Regularized Beamformer for Evoked Data. SAMerf and erSAM [24] use weights derived from the data covariance of unaveraged data and applied to evoked averaged data; however, these may not be optimally tuned for phase-locked activity, especially if the nonphase-locked activity is stronger. To overcome the difficulty of inverting the ill-conditioned matrix obtained from the covariance of averaged data, Brookes et al. [25] proposed to regularize with the minimum eigenvalue of the unaveraged data covariance. This option is included in the "regularization-type" drop-down menu on the Source Analysis: Time-Series GUI (Figure 5) and can be used with the Scalar LCMV Beamformer applied to averaged data. This option works especially well for stimulus-driven phase-locked effects, such as from a flashing checkerboard.

3.4.5. Coherent Source Suppression. An occasional point of failure with beamformer techniques occurs when two sources are highly temporally correlated, as might occur, for example, in some subjects with bilateral auditory evoked responses. We have implemented a *coherent source suppression* technique in NUTMEG that can overcome such a correlated source failure [5]. A zone containing an expected interfering source must be defined, and this can be accomplished interactively with the MRI viewer. The algorithm has been independently shown to improve upon standard beamformer performance whether using the "partial sensor coverage" strategy [26] or using whole head coverage as usual [27]. The method has also been successfully applied to suppress cochlear implant artifacts in EEG data [16].

3.4.6. Source Stability Index for Evoked Data. Another method to bypass the problem of beamformers with temporally correlated evoked sources is proposed by Prendergast et al. [28], termed the *Source Stability Index*. First, one obtains a source localization estimate from data covariance of unaveraged trials, then the corresponding weights are applied to an average of two separate halves the trials. The correlation at each voxel between the source estimates

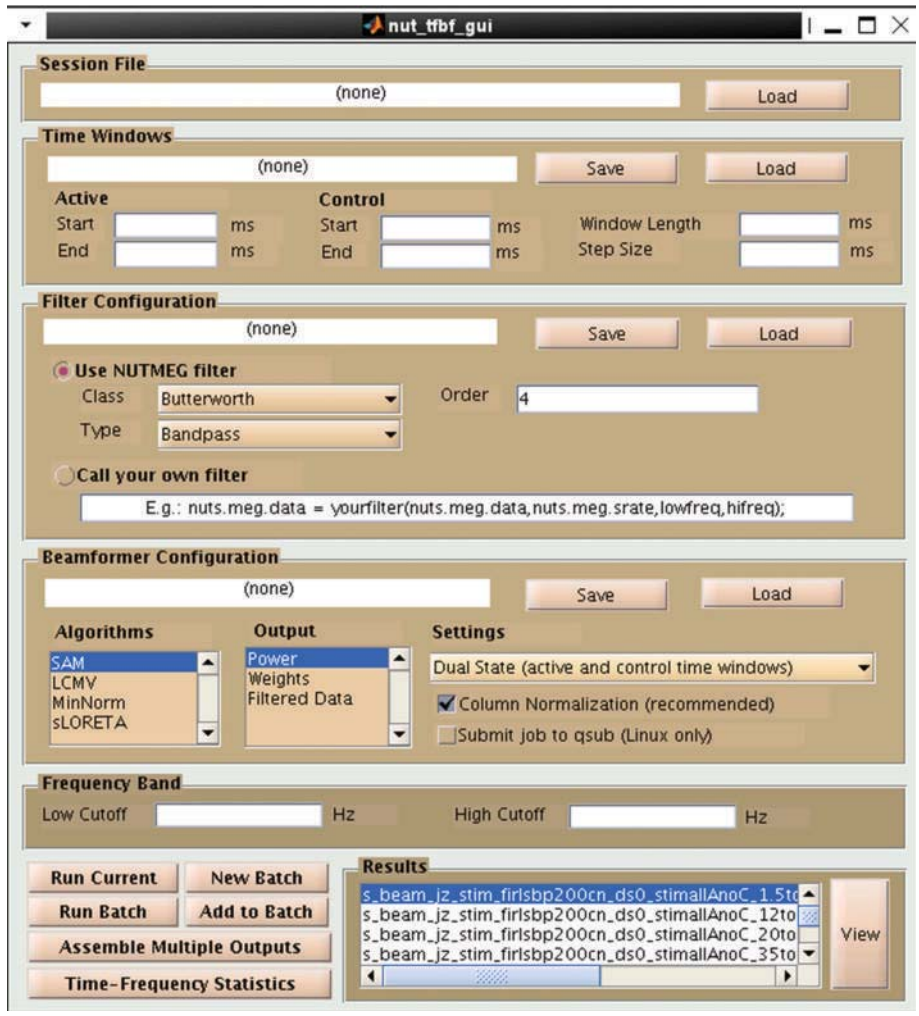


Figure 6: Time-Frequency Beamformer GUI. The pre-specified NUTMEG session including dataset and forward lead field is loaded. Multiple active windows and a single control time window, as well as filtering parameters are selected in the GUI or loaded from prespecified parameter files. Details of the source inversion methods are specified, and the whole setup can then be saved and set to run as a batch process, potentially sending each time-frequency window to a separate computer node. Finally, all the windows are recombined into one file to view the 5-D output.

derived from the separate halves will be high at locations of a true evoked source; this step is repeated for different divisions of the trials, and an average correlation map is obtained to localize the sources. This method has been implemented to work within the NUTMEG work flow and has been successfully applied to auditory evoked data from tone stimulation [29].

3.5. Bayesian Inference Inversions. The following denoising and source localization methods are designed to be used with averaged data. If unaveraged trials are loaded, they will be averaged first prior to input into these methods.

3.5.1. Denoising/Factor Analysis of Sensor Data. To remove background noise from evoked data with a prestimulus baseline, Nagarajan et al. [30] proposed *stimulus evoked factor analysis* (SEFA). The SEFA algorithm uses Bayesian inference to determine which temporal "factors" (like a component in ICA) are stimulus-evoked versus background activity. Using a probabilistic model, hyperparameters over each factor help determine which to keep or to suppress. SEFA can be selected from the drop-down "denoising" menu, and the immediate effects on the cleaned sensor data can be viewed within a subfigure of the Source Analysis: Time-Series GUI.

3.5.2. SAKETINI Inverse Method. In order to estimate source activity using knowledge of event timing and independent from noise and interference (SAKETINI), a probabilistic model [31] was proposed which, for each source voxel, separates the contribution to the sensors from (1) evoked activity at that given voxel, (2) evoked activity at all other voxels, (3) background neural activity present in the prestimulus period, and (4) sensor noise. Using a similar probabilistic model to SEFA but with an additional term for (1), SAKETINI also uses hyperparameters to determine how many factors belong to each category. This method can be

called from the drop-down menu on the Source Analysis: Time-Series GUI (Figure 5) or from batch scripts and can optionally be run on a parallel computing cluster to speed computation time.

3.5.3. NSEFALoc Inverse Method. Using the factors from SEFA as a set of temporal basis functions (TBFs), the neural SEFA localization (NSEFALoc) method [32] determines, for each source voxel, the optimal linear combination of the TBFs with an additive noise term to model voxels at which no evoked activity occurs. NSEFALoc has been shown to be superior to the eigenspace beamformer and minimum-norm methods for evoked activity. Like SAKETINI, it can be called from the GUI or command line and may also optionally be run on a parallel computing cluster.

3.5.4. Champagne Inverse Method. NUTMEG also implements Champagne [33], a tomographic Bayesian inference algorithm that combines SEFA modeling of background noise with sparse Bayesian inference of source activity in all voxels simultaneously using fast, robust update rules with guaranteed convergence under many realistic conditions. Champagne bears some similarities to SAKETINI. Whereas SAKETINI considers each voxel sequentially while statistically modeling contributions to sensors from other voxels, Champagne considers all voxels simultaneously. Champagne has been shown to successfully localize many simultaneous and temporally correlated sources.

3.6. Minimum-Norm Methods. For algorithm performance evaluation as well as comparison with results from the literature, two common minimum-norm methods are also included in NUTMEG. Both sLORETA [34] and dSPM [35] normalize the standard mininum-norm inverse

$$\mathbf{L}^T\left(\mathbf{L}\mathbf{L}^T\right)^{-1} \tag{2}$$

by an estimate of source noise obtained by projecting sensor noise

$$\mathbf{L}^T\left(\mathbf{L}\mathbf{L}^T\right)^{-1}\mathbf{R}_{nn}\left(\mathbf{L}\mathbf{L}^T\right)^{-1}\mathbf{L}. \tag{3}$$

By using a form of the regularized Gram matrix $\mathbf{L}\mathbf{L}^T$ in place of $\mathbf{R}_{yy}$ in (1), the (data-dependent) beamformer can be translated to a (data-independent) weighted minimum-norm method. As the Gram matrix is not full-rank, yet needs to be inverted, performance is highly dependent on choice of regularization. NUTMEG includes two options to regularize the Gram matrix prior to inversion: (1) add the sensor covariance matrix (based on either individual subject data or room noise) weighted by a constant or (2) Tikhonov regularization, that is, add a constant to the diagonal of the Gram matrix, based on the strength of the off-diagonal elements of the inverted matrix.

dSPM traditionally sets $\mathbf{R}_{nn}$ to room noise covariance. If this covariance is taken to be the identity matrix (times a constant), this leads to the regularized Gram matrix in the numerator in place of $\mathbf{R}_{yy}$ and the square of the regularized Gram matrix in the denominator. In contrast, sLORETA sets $\mathbf{R}_{nn}$ to the sensor noise covariance obtained from assuming (in a Bayesian fashion) identity source power and identity sensor noise (times a constant), which is equivalent to a type of regularized Gram matrix.

Note that, while minimum-norm spatial filters are nonadaptive relative to the sensor data, they enforce that all measured activity arises from the defined VOI. A cortically constrained VOI is often used with both methods.

Lastly, NUTMEG implements a recently developed method by Kumihashi and Sekihara [36] called the *Array Gain constraint Minimum-Norm, with Recursively Updated Gram matrix* (AGMN-RUG) method, which estimates the sensor covariance matrix by recursively updating the weighted Gram matrix using the source covariance from the previous estimate. Like the minimum-variance adaptive beamformers, the source estimates are spatially focal, while, like the minimum-norm methods, unhindered by temporally correlated sources or few available time points.

4. Visualization

NUTMEG supports both orthogonal view visualization as well as 3D rendering of cortical surface visualization. NUTMEG utilizes the SPM8 navigator to display functional maps on structural MRIs (Figure 7(a)). This is interactively linked with NUTMEGs time series or time-frequency display (Figure 7(b)). That is, when the user clicks to a different location in the brain, the time series/frequency display updates to show the temporal change at that location. Likewise, the user can click on a different time point or frequency band and the MRI display automatically updates with the 3D functional map corresponding to that new time/frequency point. Additional buttons exist for manipulating the view, for example, displaying different contrast types (simple difference of active and control, % change, etc.), zooming in time, rescaling the colormap, and calling SPM8 functions for projecting functional overlays onto a rendered surface.

In addition to these main tools, data can be exported to analyze format images which can then be further manipulated in CarTool (http://sites.google.com/site/cartoolcommunity/), mri3dX (http://www.cubric.cf.ac.uk/Documentation/mri3dX/), DataViewer3D (https://www.ynic.york.ac.uk/software/dv3d), and MRICro (http://www.cabiatl.com/mricro/), all of which can be used to generate publication-quality surface renderings with superimposed functional maps.

5. Statistics

5.1. Within-Subject Statistics. A nonparametric statistical threshold for time series source reconstructions can be calculated based on the distribution of baseline activity across trials within a single subject [37]. For time-frequency source reconstructions, Wilcoxon Z scores assess the contrast between baseline time-frequency windows versus "active" windows [17].

5.2. Group Statistics. Group statistics can also be performed to assess statistical significance across subjects. The mean and

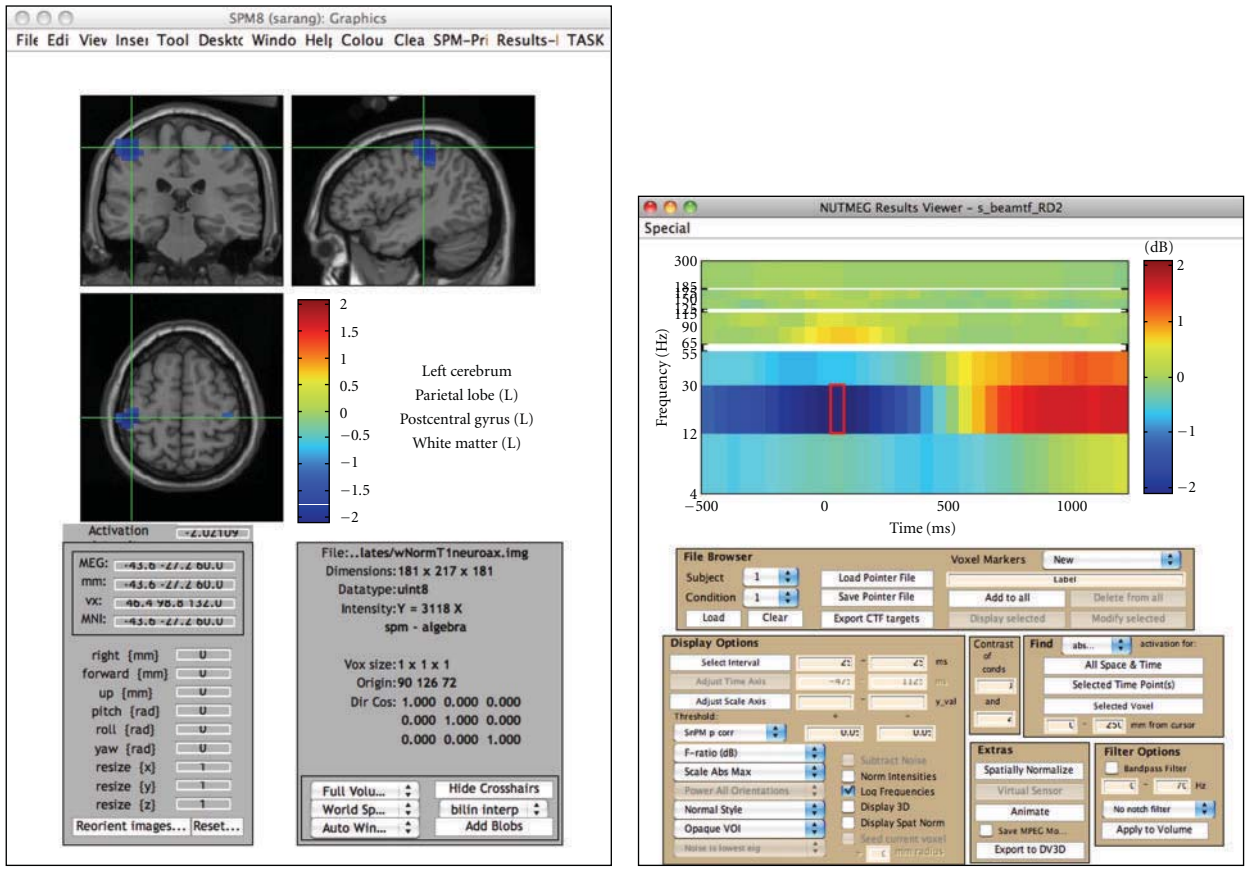


Figure 7: NUTMEGs results viewer, in time-frequency mode. The modified SPM8 viewer shows the functional map corresponding to the selected time-frequency window, marked by a red box in the spectrogram. Conversely, the time-frequency spectrogram corresponds to the voxel indicated by the cross-hairs on the MRI navigator. The MRI display and time-frequency spectrogram are interactively linked to each other; changing a selection in one automatically updates the other. Movies can also be created from animations of the functional images across time.

variance of power across subjects can be computed by first spatially normalizing each subject's source reconstruction and then resampling each subject's result into a common voxel space.

Statistical tests can then be applied to these transformed datasets. For situations in which normal distributions of power change can be expected, or after transformation to a normal distribution, one option is to apply the Student's t-test or ANOVA across multiple conditions.

Alternatively, statistical nonparametric mapping (SnPM) can be applied to data that may not necessarily follow a normal distribution [38]. One of the advantages of SnPM over parametric methods is that it can be applied to a population of as few as 5 subjects, though having more subjects will allow detection of weaker effects. Since variance estimates can be noisy for a relatively low number of subjects, variance maps are smoothed with a 3D Gaussian kernel. From this, a pseudo-t statistic can be obtained at each voxel, time window, and frequency band. Then, a distribution of pseudo-t statistics is created from 2^N permutations of the original N datasets (subjects). Each permutation consists of two steps: (1) inverting the polarity of the power change values for some subjects (with 2^N possible combinations of negations) and (2) finding the current maximum pseudo-t value among all voxels and time windows for each frequency band. Instead of estimating the significance of each nonpermuted pseudo-t value from an assumed normal distribution, it is then calculated from the position within the distribution of these maximum permuted pseudo-t values. The comparison against maximum values effectively corrects for the family-wise error of testing multiple voxels and time windows.

6. Connectivity

The brain is a complex network with abundant functional interactions among local and remote brain areas [39]. The synchronization of oscillations in different brain areas, that is, the so-called *functional connectivity*, is considered as an index of their functional interaction [40, 41]. Techniques based on functional connectivity open an accessible window for a noninvasive assessment of brain function in healthy subjects [42, 43] as well as in patients with brain lesions [14, 44].

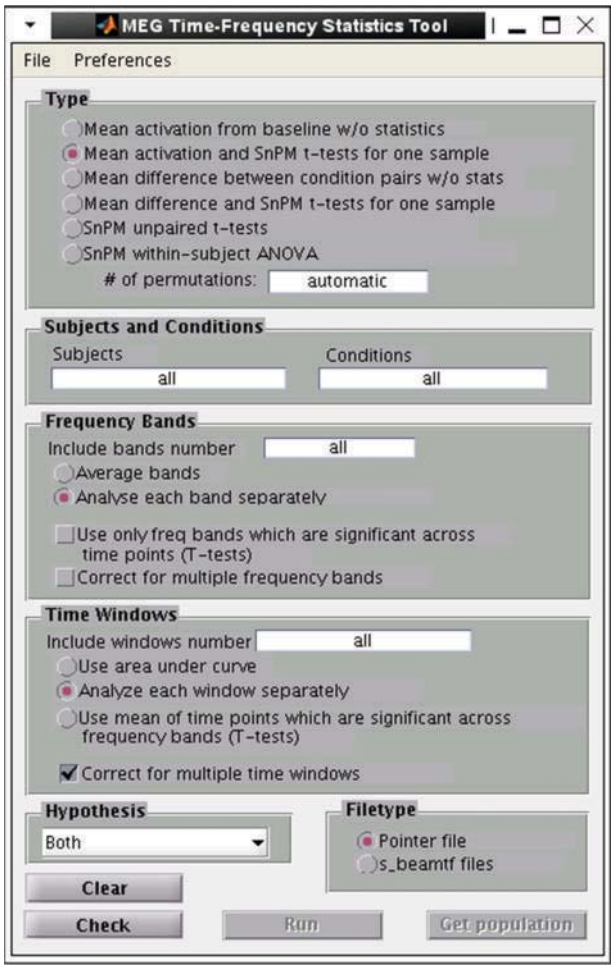


Figure 8: NUTMEGs time-frequency statistics tool, showing various options for calculating statistical significance across subjects for source time-frequency maps.

NUTMEG computes the localization of functional connectivity among brain areas from MEG and EEG recordings by combining source localization algorithms with measures of functional connectivity. The oscillations of neural networks at each brain voxel are estimated by calculating the linear combination of the sensor data matrix with a spatial weighting matrix obtained with inverse solutions.

6.1. FCM Toolbox for Imaginary Coherence. Imaginary coherence, applied to the source time series, is a measure of functional connectivity that is robust to sensor cross-talk and volume conduction [14, 45]. In order to reduce computation times for large datasets or for exploration of numerous connections among brain voxels, the calculations in NUTMEG can be performed in parallel on Linux clusters. The toolbox also offers visualization tools for inspection of the complex functional interactions data as well as a set of statistical tests. Figure 9 shows an example of corticomuscular coherence in a single subject, which is localized to the bilateral motor cortex. Cortico-cortical interaction can also be analyzed.

6.2. Full Coherence. As a standalone command line option or called from a GUI (Figure 10), both the magnitude and imaginary cross-coherence can be computed for an input of voxels' power spectrum (after FFT and windowing of time series) for each trial. The output can be placed into the appropriate NUTMEG data structure to view results overlaid on the MRI.

6.3. Hilbert Envelope Correlation. An alternative metric for MEG/EEG functional connectivity involves computing the correlation of the Hilbert envelope (amplitude) of bandpass filtered time series from source locations [46]. This method may also be called from a GUI (Figure 10) or command-line.

7. Extension to Include Scalp and Intracranial EEG

7.1. Scalp EEG. NUTMEG has been expanded to support beamforming with electroencephalography data via the NUTEEG module. This module allows the import of data recorded from EEG systems, along with electrode coordinates. NUTEEG automatically performs average referencing on EEG data and lead potentials as part of the preprocessing procedure. For situations where an MRI is not available, NUTEEG provides the option of warping a template MRI and corresponding boundary element model to digitized electrode positions, based on the algorithm described by Darvas et al. [47] (see Figure 11).

Forward lead potentials can be calculated either using spherical head models, or via BEM with the previously mentioned toolboxes. If a boundary element model is used, digitized electrode positions can be projected to the scalp surface. Boundary element models can be created from segmented MRI images using a Delaunay triangulation method provided by the ISO2MESH toolbox (http://iso2mesh.sourceforge.net/) (Figure 12), or from BrainSuite Duff surface files via a triangulated sphere wrapping procedure.

NUTEEG allows the user to import cortical surface Duff files from BrainSuite to create a file containing orthogonal dipole orientations for voxels near the cortical surface. These dipole orientations can then be used for implementing cortical constraints, where one assumes that sources are cortical and are oriented tangential to the cortical surface. The imported cortical surface files can also be used to show results in 3D (Figure 13).

After data import and lead field computation/import of EEG, the subsequent steps for source estimation and visualization are straightfoward, as for MEG.

7.2. Intracranial EEG. Invasive electrode implants are sometimes performed in human patients to aid in surgical planning for, for example, intractable epilepsy or brain tumors. Although intracranial EEG is often considered to be the "gold standard" of electrical brain activity, it may also be susceptible to undesired physiological noise sources [48, 49]. Furthermore, intracranial electrodes are not immune to far-field potentials from strong brain sources.

Referencing choice can also complicate interpretation of results. A simple focal source appears as a polarity inversion

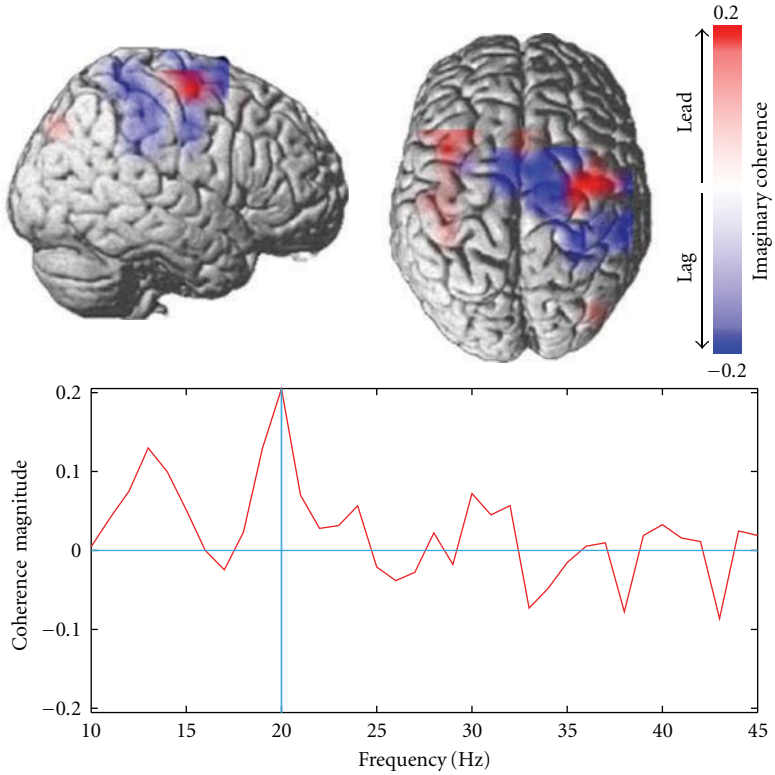


Figure 9: Corticomuscular coherence, showing relationship between motor and somatosensory cortices and left finger EMG. Note that regions that lead and lag the EMG activity can be clearly differentiated.

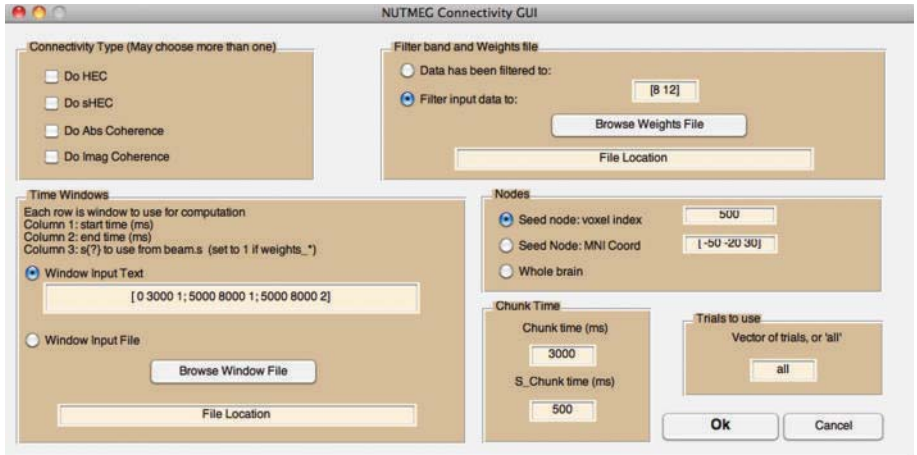


Figure 10: The connectivity GUI allows the user to specify the type of metric, the frequency band of interest, the whole time windows of interest which may be across condition types, the length of time chunk within each time window, and whether to compute over whole brain or seed based. The GUI assumes the inverse weights have already been computed, but the source-level time series or power need not have been saved out previously.

between electrodes in a monopolar scheme but a local peak in a bipolar montage. Furthermore, for complex voltage topographies, the actual source origin may be ambiguous and difficult to deduce from any montage. Finally, traditional voltage topographies are limited by the spatial sampling of the electrode placement.

Source localization techniques from scalp EEG/MEG may provide a solution to these problems. In particular, adaptive spatial filtering methods such as beamforming are particularly well suited [50]. Unlike previous attempts that use minimum-norm-based techniques [51–54], beamformers do not enforce that all source activity arise from the

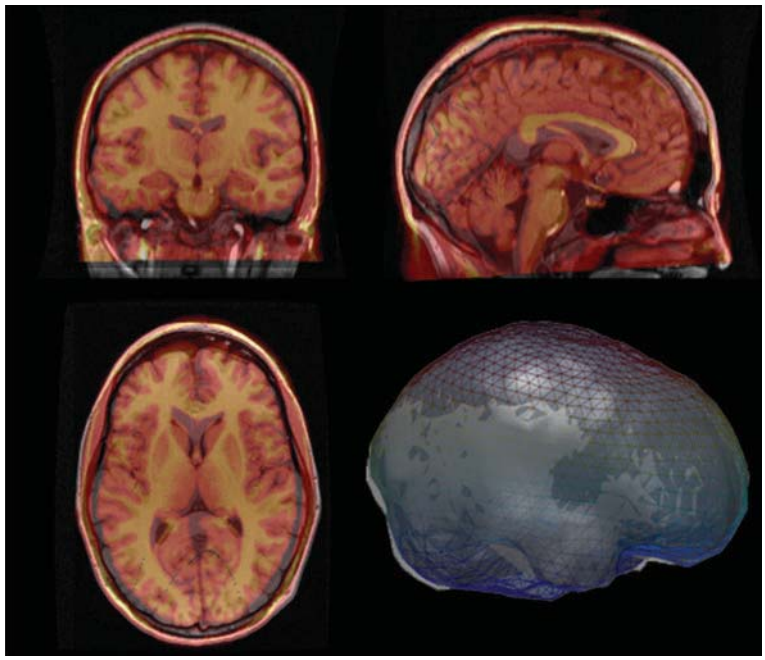
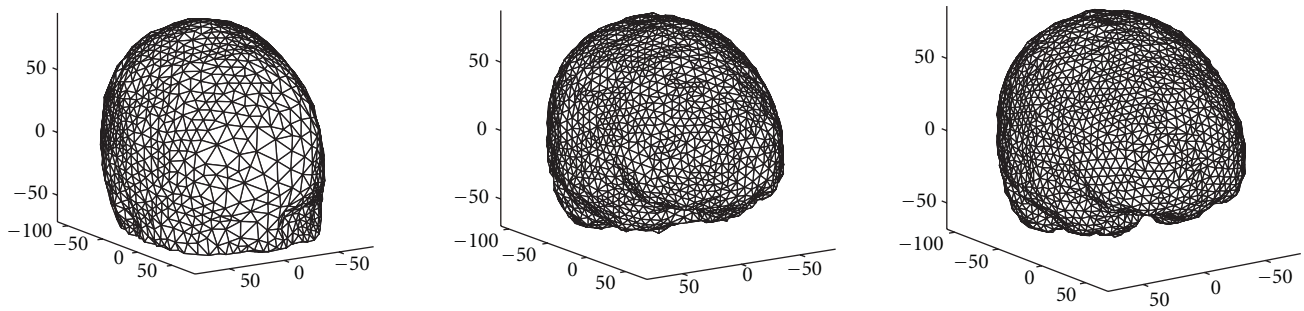


Figure 11: The overlay of the warped template MRI (red/yellow) with the actual MRI (grey) shows a reasonable fit of the scalp surface and neuro-anatomy. A mesh overlay of the warped template brain (blue) and the actual brain (grey) is also shown.



Scalp: 2000 nodes Outer skull: 3000 nodes Inner skull: 4000 nodes

Figure 12: Three-layer boundary element mesh created with Delaunay triangulation. Projected electrode positions are shown as blue dots on the scalp surface.

defined volume of interest. Thus, noise sources such as heart and muscle would be rejected by the spatial filter rather than projected into the brain, and, conversely, brain regions that contribute negligible signal would not distort the localization results. Finally, source localization allows gaps between electrodes to be "filled in" to gain an effectively higher spatial resolution, providing similar benefits to denser electrode coverage.

Therefore, development of intracranial EEG localization and analysis techniques is considered a research priority for NUTMEG. Figure 14 shows preliminary results from a beamformer applied to depth electrode responses evoked by photographic stimuli. Lead fields can be computed within NUTMEG, currently implemented as a simple semi-infinite homogeneous volume conductor; alternatively, a BEM-based lead field can be computed and imported from the OpenMEEG package.

8. NUTMEG in Python

Python is an open-source, general-purpose, object-oriented programming language that is gaining popularity as a tool for scientific computing. As an interpreted language with robust object model support, Python allows a wide variety of programming styles, from line-by-line scripting to abstracted, reusable library code. Its strengths include an emphasis on legibility and ease-of-use, system portability, and straightforward access to system libraries. Additionally, there is a very stable stack of basic computational tools actively developed by the scientific Python community. First among the many

commonly used tools are: NumPy for multidimensional arrays, SciPy for a wealth of computational code, much of it being a Python layer over established, validated libraries such as LAPACK and FFTPACK, and Matplotlib, which provides interactive and scriptable 2D plotting tools that emulate MATLAB plotting. All these features provide a convenient computing environment for the development of modern scientific data processing systems, whose scope may expand over time, and whose core functionality typically demand a design covering a range from optimized algorithms to complex data models for physical phenomena.

NUTMEG-Py is a complementary project that entails a small scale reformulation of NUTMEG components into Python. To date, implementation of visualization and statistical postprocessing have been emphasized, with source reconstruction algorithms remaining in MATLAB.

8.1. From MATLAB Data to Python Objects. The workflow for a NUTMEG-based analysis that incorporates Python tools presents both a design challenge and a technical data translation problem. The latter is a solved problem, thanks to code from SciPy enabling I/O between NumPy arrays and MATLAB data contained in MAT files. The former allows the use of Python's object model.

NUTMEG-Py's core includes very simple data models which, abstractly, have immutable data and metadata, have methods to interrogate or transform the data in some fashion, and finally can read and write itself on disk without loss of precision. The TFBeam is an example of such an object and is the Python analog to the MATLAB "struct" containing a time-frequency reconstruction (NUTMEGs *beam* structure).

The toolbox side of NUTMEG-Py currently includes a nonparametric statistical testing package, based on Nichols and Holmes [55], including cluster level analysis from Hayasaka and Nichols [56]. Both approaches have been adapted to the five-dimensional space of time-frequency MEG imaging. The results are encapsulated in an object oriented manner, as the TimeFreqSnPMResults, which stores the generated null distributions, and has methods available for creating thresholds and maps based on levels of significance.

8.2. Visualization. While the MATLAB/SPM based visualization of results in NUTMEG allows for easy navigation across space, time, and frequency, the interactive viewing is limited to the orthogonal slice projection, which can make widespread global brain activations difficult to visualize. The project to transition NUTMEG into a Python-based toolkit has also spawned a small but powerful visualization effort named Xipy (cross-modality imaging in Python), which lies under the umbrella of the seminal NiPy (Neuroimaging in Python) project (http://nipy.sourceforge.net/). The main ambition of Xipy is to provide a flexible and extensible system for displaying and navigating brain imagery from various data sources (e.g., anatomical MRIs, functional maps, and diffusion tracks) in the same 3D scene (see Figure 15). Xipy is designed to be independent from NUTMEG-Py, and visualization of results from NUTMEG and NUTMEG-Py within Xipy is enabled by a richly featured plugin contained in the NUTMEG-Py package.

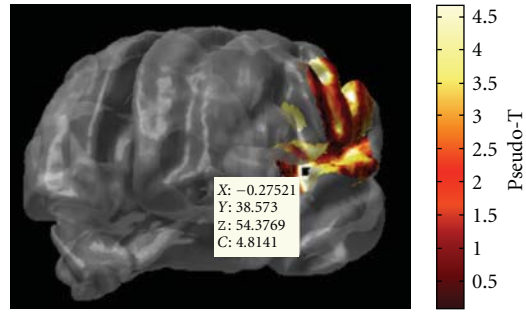


Figure 13: EEG beamformer reconstruction of an evoked response from auditory cortex projected to the cortical surface.

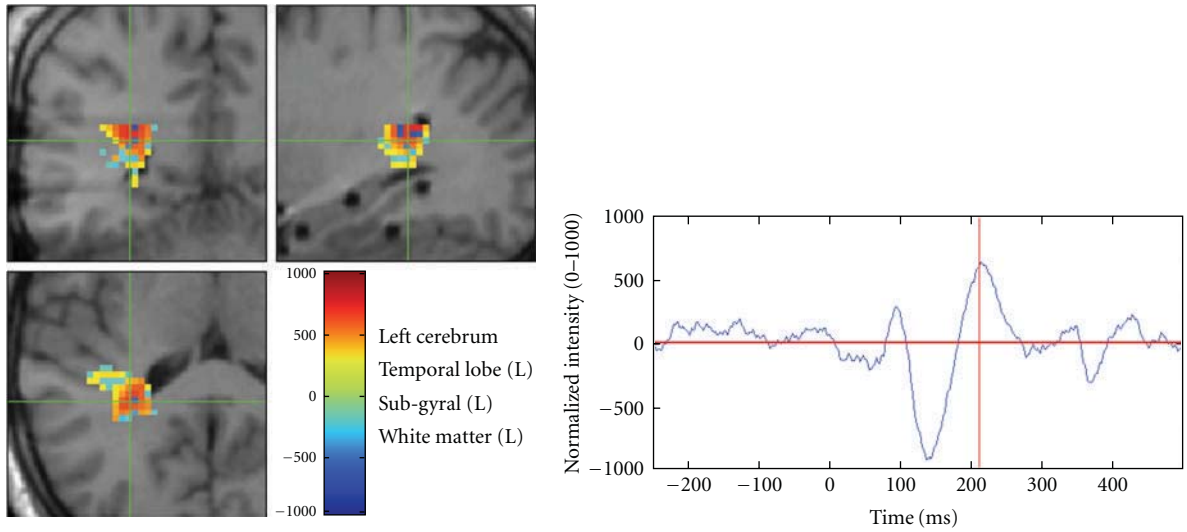
9. NUTMEG's Future Directions

The future of NUTMEG is influenced by both the research priorities of the developers as well as requests from users.

At present, we intend to create more formal links with SPM8, FieldTrip, and Brainstorm. Specifically, as methods developers, we would like to import, view, and directly compare the Multiple Sparse Priors [57] from SPM8 with other source estimation methods included in NUTMEG; further, we would like to enable direct comparison within NUTMEG of Dynamic Causal Modelling (DCM) for M/EEG [58] with other metrics for functional connectivity. The advanced time-frequency analysis and viewing tools for sensor level data within FieldTrip can be useful to NUTMEG users for planning of further analysis in source space. NUTMEG should be able to display source level results computed in FieldTrip. The cluster-based and permutation test statistics for sensor and source space results implemented in FieldTrip would also be of benefit to be more formally linked to the NUTMEG format. Sensor selection via visual inspection is a highly developed tool within Brainstorm, the output of which could be imported to NUTMEG. Brainstorm also contains useful GUIs for dataset, trial-condition selection, and batch processing setup, which could be linked to NUTMEG via a conversion of MATLAB data structures.

As several methods for connectivity analysis have recently become available within NUTMEG and additional methods are planned for inclusion, a means to visually browse the results is needed beyond a simple extension of the current source-space viewer. The eConnectome package (http://econnectome.umn.edu/) already implements the computation and elaborate visualization of connectivity, to which we may link.

The fusion of multiple sensor types (MEG magnetometers and planar gradiometers, scalp EEG, and intracranial EEG) simultaneously recorded for source reconstruction is a compelling need, but is not yet considered directly straightforward or well established; NUTMEG and other open-source software packages would benefit greatly from further developments on this topic.



(a) (b)

Figure 14: Intracranial EEG beamformer reconstruction of a visual evoked response. Depth electrode trajectories are evident on the sagittal MRI view.

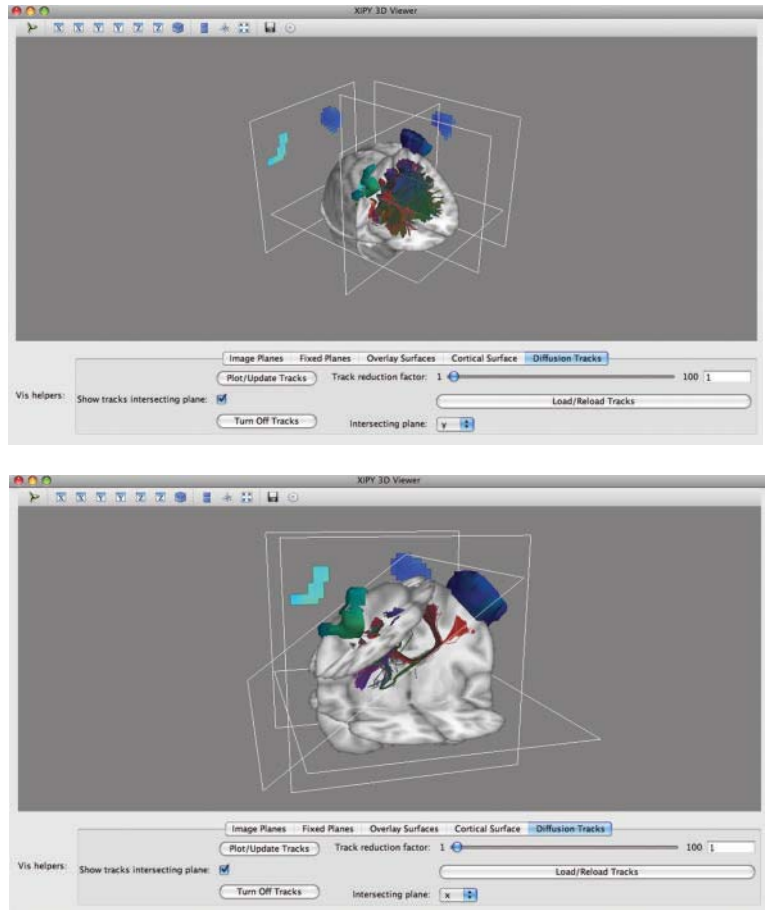


Figure 15: NUTMEG and DTI results in Xipy viewer.

10. Conclusion

NUTMEG provides a full set of MATLAB-based open-source functions with which to compute neural source estimates and additional manipulations thereof, as well as a graphical interface to process and view results. It is linked (to varying degrees) to other open-source packages for processing steps which are better performed by those toolboxes. NUTMEG is flexible to inclusion of new methods at any stage and welcomes new users and developers.

Authors' Contribution

Sarang S. Dalal and Johanna M. Zumer contributed equally to the manuscript.

Acknowledgments

S. S. Dalal was supported by European Commission FP7 Grant PIIF-GA-2008-221097. J. M. Zumer was supported by a Sir Peter Mansfield Fellowship from the University of Nottingham and by the Whitaker International Scholar program. A. G. Guggisberg was supported by Swiss National Science Foundation Grant 320030_129679. This work was funded in part by the following Grants to S. S. Nagarajan: NIH grants R01 DC4855, DC6435, DC10145, NS67962, NIH/NCRR UCSF-CTSI grant UL1 RR024131, and UCSF/REAC. S. S. Dalal thanks Dr. Juan R. Vidal for design of the intracranial EEG experiment. J. M. Zumer thanks Dr. Matthew J. Brookes for helpful discussions and Prof. Peter G. Morris for support. The authors would finally like to thank the long list of other contributors to the NUTMEG codebase, listed in the software under "About NUTMEG."

References

[1] S. S. Dalal, J. M. Zumer, V. Agrawal, K. E. Hild, K. Sekihara, and S. S. Nagarajan, "NUTMEG: a neuromagnetic source reconstruction toolbox," *Neurology & Clinical Neurophysiology*, vol. 2004, p. 52, 2004.

[2] A. C. Evans, D. L. Collins, S. R. Mills, E. D. Brown, R. L. Kelly, and T. M. Peters, "3D statistical neuroanatomical models from 305 MRI volumes," in *Proceedings of IEEE Nuclear Science Symposium & Medical Imaging Conference*, pp. 1813–1817, November 1993.

[3] J. Mazziotta, A. Toga, A. Evans et al., "A probabilistic atlas and reference system for the human brain: International Consortium for Brain Mapping (ICBM)," *Philosophical Transactions of the Royal Society B*, vol. 356, no. 1412, pp. 1293–1322, 2001.

[4] P.-E. Aguera, K. Jerbi, A. Caclin, and O. Bertrand, "ELAN: A software package for analysis and visualization of MEG, EEG, and LFP signals," *Computational Intelligence and Neuroscience*. In press.

[5] S. S. Dalal, K. Sekihara, and S. S. Nagarajan, "Modified beamformers for coherent source region suppression," *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 7, pp. 1357–1363, 2006.

[6] C. L. Dale, A. M. Findlay, R. A. Adcock et al., "Timing is everything: neural response dynamics during syllable processing and its relation to higher-order cognition in schizophrenia and healthy comparison subjects," *International Journal of Psychophysiology*, vol. 75, no. 2, pp. 183–193, 2010.

[7] T. H. Heinks-Maldonado, D. H. Mathalon, J. F. Houde, M. Gray, W. O. Faustman, and J. M. Ford, "Relationship of imprecise corollary discharge in schizophrenia to auditory hallucinations," *Archives of General Psychiatry*, vol. 64, no. 3, pp. 286–296, 2007.

[8] J. M. Zumer, S. S. Nagarajan, L. A. Krubitzer, Z. Zhu, R. S. Turner, and E. A. Disbrow, "MEG in the macaque monkey and human: distinguishing cortical fields in space and time," *Brain Research*, vol. 1345, pp. 110–124, 2010.

[9] S. S. Dalal, A. G. Guggisberg, E. Edwards et al., "Five-dimensional neuroimaging: localization of the time-frequency dynamics of cortical activity," *NeuroImage*, vol. 40, no. 4, pp. 1686–1700, 2008.

[10] J. M. Zumer, M. J. Brookes, C. M. Stevenson, S. T. Francis, and P. G. Morris, "Relating BOLD fMRI and neural oscillations through convolution and optimal linear weighting," *NeuroImage*, vol. 49, no. 2, pp. 1479–1489, 2010.

[11] A. G. Guggisberg, S. S. Dalal, A. M. Findlay, and S. S. Nagarajan, "High-frequency oscillations in distributed neural networks reveal the dynamics of human decision making," *Frontiers in Human Neuroscience*, vol. 1, p. 14, 2008.

[12] V. Van Wassenhove and S. S. Nagarajan, "Auditory cortical plasticity in learning to discriminate modulation rate," *Journal of Neuroscience*, vol. 27, no. 10, pp. 2663–2672, 2007.

[13] L. B. N. Hinkley, S. S. Nagarajan, S. S. Dalal, A. G. Guggisberg, and E. A. Disbrow, "Cortical temporal dynamics of visually guided behavior," *Cerebral Cortex*, vol. 21, no. 3, pp. 519–529, 2011.

[14] A. G. Guggisberg, S. M. Honma, A. M. Findlay et al., "Mapping functional connectivity in patients with brain lesions," *Annals of Neurology*, vol. 63, no. 2, pp. 193–203, 2008.

[15] J. M. Zumer, M. J. Brookes, C. S. Stevenson, P. G. Morris, and S. V. Shinkareva, "Oscillatory power and connectivity changes in a word decision task measured with MEG," Organization for Human Brain Mapping, 2010.

[16] D. D. E. Wong and K. A. Gordon, "Beamformer suppression of cochlear implant artifacts in an electroencephalography dataset," *IEEE Transactions on Biomedical Engineering*, vol. 56, no. 12, pp. 2851–2857, 2009.

[17] S. S. Dalal, S. Baillet, C. Adam et al., "Simultaneous MEG and intracranial EEG recordings during attentive reading," *NeuroImage*, vol. 45, no. 4, pp. 1289–1304, 2009.

[18] J. Sarvas, "Basic mathematical and electromagnetic concepts of the biomagnetic inverse problem," *Physics in Medicine and Biology*, vol. 32, no. 1, pp. 11–22, 1987.

[19] M. X. Huang, J. C. Mosher, and R. M. Leahy, "A sensor-weighted overlapping-sphere head model and exhaustive head model comparison for MEG," *Physics in Medicine and Biology*, vol. 44, no. 2, pp. 423–440, 1999.

[20] L. Spinelli, S. G. Andino, G. Lantz, M. Seeck, and C. M. Michel, "Electromagnetic inverse solutions in anatomically constrained spherical head models," *Brain Topography*, vol. 13, no. 2, pp. 115–125, 2000.

[21] Z. Zhu, E. A. Disbrow, J. M. Zumer, D. J. McGonigle, and S. S. Nagarajan, "Spatiotemporal integration of tactile information in human somatosensory cortex," *BMC Neuroscience*, vol. 8, article 21, 2007.

[22] B. D. Van Veen, W. Van Drongelen, M. Yuchtman, and A. Suzuki, "Localization of brain electrical activity via linearly constrained minimum variance spatial filtering," *IEEE Transactions on Biomedical Engineering*, vol. 44, no. 9, pp. 867–880, 1997.

[23] K. Sekihara, S. S. Nagarajan, D. Poeppel, A. Marantz, and Y. Miyashita, "Reconstructing spatio-temporal activities of neural sources using an MEG vector beamformer technique," *IEEE Transactions on Biomedical Engineering*, vol. 48, no. 7, pp. 760–771, 2001.

[24] D. Cheyne, L. Bakhtazad, and W. Gaetz, "Spatiotemporal mapping of cortical activity accompanying voluntary movements using an event-related beamforming approach," *Human Brain Mapping*, vol. 27, no. 3, pp. 213–229, 2006.

[25] M. J. Brookes, J. M. Zumer, C. M. Stevenson et al., "Investigating spatial specificity and data averaging in MEG," *NeuroImage*, vol. 49, no. 1, pp. 525–538, 2010.

[26] M. Popescu, E. A. Popescu, T. Chan, S. D. Blunt, and J. D. Lewine, "Spatio-temporal reconstruction of bilateral auditory steady-state responses using MEG beamformers," *IEEE Transactions on Biomedical Engineering*, vol. 55, no. 3, pp. 1092–1102, 2008.

[27] M. A. Quraan and D. Cheyne, "Reconstruction of correlated brain activity with adaptive spatial filters in MEG," *NeuroImage*, vol. 49, no. 3, pp. 2387–2400, 2010.

[28] G. Prendergast, S. R. Johnson, M. Hymers, W. Woods, and G. G.R. Green, "Non-parametric statistical thresholding of baseline free MEG beamformer images," *NeuroImage*, vol. 54, no. 2, pp. 906–918, 2011.

[29] M. Serada, J. M. Zumer, M. J. Brookes, and P. Adjamian, "Modulation of AEF amplitude with frequency of tone stimulation," Tech. Rep., University of Nottingham, 2010.

[30] S. S. Nagarajan, H. T. Attias, K. E. Hild, and K. Sekihara, "A probabilistic algorithm for robust interference suppression in bioelectromagnetic sensor data," *Statistics in Medicine*, vol. 26, no. 21, pp. 3886–3910, 2007.

[31] J. M. Zumer, H. T. Attias, K. Sekihara, and S. S. Nagarajan, "A probabilistic algorithm integrating source localization and noise suppression for MEG and EEG data," *NeuroImage*, vol. 37, no. 1, pp. 102–115, 2007.

[32] J. M. Zumer, H. T. Attias, K. Sekihara, and S. S. Nagarajan, "Probabilistic algorithms for MEG/EEG source reconstruction using temporal basis functions learned from data," *NeuroImage*, vol. 41, no. 3, pp. 924–940, 2008.

[33] D. P. Wipf, J. P. Owen, H. T. Attias, K. Sekihara, and S. S. Nagarajan, "Robust Bayesian estimation of the location, orientation, and time course of multiple correlated neural sources using MEG," *NeuroImage*, vol. 49, no. 1, pp. 641–655, 2010.

[34] R. D. Pascual-Marqui, "Standardized low-resolution brain electromagnetic tomography (sLORETA): technical details," *Methods and Findings in Experimental and Clinical Pharmacology*, vol. 24, supplement D, pp. 5–12, 2002.

[35] A. M. Dale, A. K. Liu, B. R. Fischl et al., "Dynamic statistical parametric mapping: combining fMRI and MEG for high-resolution imaging of cortical activity," *Neuron*, vol. 26, no. 1, pp. 55–67, 2000.

[36] I. Kumihashi and K. Sekihara, "Array-gain constraint minimum-norm spatial filter with recursively updated gram matrix for biomagnetic source imaging," *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 6, pp. 1358–1365, 2010.

[37] K. Sekihara, M. Sahani, and S. S. Nagarajan, "A simple nonparametric statistical thresholding for MEG spatial-filter source reconstruction images," *NeuroImage*, vol. 27, no. 2, pp. 368–376, 2005.

[38] K. D. Singh, G. R. Barnes, and A. Hillebrand, "Group imaging of task-related changes in cortical synchronisation using nonparametric permutation testing," *NeuroImage*, vol. 19, no. 4, pp. 1589–1601, 2003.

[39] F. Varela, J. P. Lachaux, E. Rodriguez, and J. Martinerie, "The brainweb: phase synchronization and large-scale integration," *Nature Reviews Neuroscience*, vol. 2, no. 4, pp. 229–239, 2001.

[40] K. J. Friston, "Brain function, nonlinear coupling, and neuronal transients," *Neuroscientist*, vol. 7, no. 5, pp. 406–418, 2001.

[41] P. L. Nunez, R. Srinivasan, A. F. Westdorp et al., "EEG coherency I: statistics, reference electrode, volume conduction, Laplacians, cortical imaging, and interpretation at multiple scales," *Electroencephalography and Clinical Neurophysiology*, vol. 103, no. 5, pp. 499–515, 1997.

[42] J. Gross, J. Kujala, M. Hämäläinen, L. Timmermann, A. Schnitzler, and R. Salmelin, "Dynamic imaging of coherent sources: studying neural interactions in the human brain," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 98, no. 2, pp. 694–699, 2001.

[43] F. De Pasquale, S. Della Penna, A. Z. Snyder et al., "Temporal dynamics of spontaneous MEG activity in brain networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 107, no. 13, pp. 6040–6045, 2010.

[44] J. Martino, S. M. Honma, A. M. Findlay et al., "Resting functional connectivity in patients with brain tumors in eloquent areas," *Annals of Neurology*. In press.

[45] G. Nolte, OU. Bai, L. Wheaton, Z. Mari, S. Vorbach, and M. Hallett, "Identifying true brain interaction from EEG data using the imaginary part of coherency," *Clinical Neurophysiology*, vol. 115, no. 10, pp. 2292–2307, 2004.

[46] M. J. Brookes, J. Hale, J. M. Zumer et al., "Measuring functional connectivity in default mode network using hilbert envelope correlation," in *Biomag Proceedings*, Dubrovnik, Croatia, 2010.

[47] F. Darvas, J. J. Ermer, J. C. Mosher, and R. M. Leahy, "Generic head models for atlas-based EEG source analysis," *Human Brain Mapping*, vol. 27, no. 2, pp. 129–143, 2006.

[48] K. Jerbi, T. Ossandón, C. M. Hamamé et al., "Task-related gamma-band dynamics from an intracerebral perspective: review and implications for surface EEG and MEG," *Human Brain Mapping*, vol. 30, no. 6, pp. 1758–1771, 2009.

[49] T. Ball, M. Kern, I. Mutschler, AD. Aertsen, and A. Schulze-Bonhage, "Signal quality of simultaneously recorded invasive and non-invasive EEG," *NeuroImage*, vol. 46, no. 3, pp. 708–716, 2009.

[50] N. Chang, R. Gulrajani, and J. Gotman, "Dipole localization using simulated intracerebral EEG," *Clinical Neurophysiology*, vol. 116, no. 11, pp. 2707–2716, 2005.

[51] B. Yvert, C. Fischer, O. Bertrand, and J. Pernier, "Localization of human supratemporal auditory areas from intracerebral auditory evoked potentials using distributed source models," *NeuroImage*, vol. 28, no. 1, pp. 140–153, 2005.

[52] M. Fuchs, M. Wagner, and J. Kastner, "Development of volume conductor and source models to localize epileptic foci," *Journal of Clinical Neurophysiology*, vol. 24, no. 2, pp. 101–119, 2007.

[53] O. Korzyukov, M. E. Pflieger, M. Wagner et al., "Generators of the intracranial P50 response in auditory sensory gating," *NeuroImage*, vol. 35, no. 2, pp. 814–826, 2007.

[54] Y. Zhang, W. van Drongelen, M. Kohrman, and B. He, "Three-dimensional brain current source reconstruction from intracranial ECoG recordings," *NeuroImage*, vol. 42, no. 2, pp. 683–695, 2008.

[55] T. E. Nichols and A. P. Holmes, "Nonparametric permutation tests for functional neuroimaging: a primer with examples," *Human Brain Mapping*, vol. 15, no. 1, pp. 1–25, 2002.

[56] S. Hayasaka and T. E. Nichols, "Combining voxel intensity and cluster extent with permutation test framework," *NeuroImage*, vol. 23, no. 1, pp. 54–63, 2004.

[57] K. Friston, L. Harrison, J. Daunizeau et al., "Multiple sparse priors for the M/EEG inverse problem," *NeuroImage*, vol. 39, no. 3, pp. 1104–1120, 2008.

[58] J. Daunizeau, S. J. Kiebel, and K. J. Friston, "Dynamic causal modelling of distributed electromagnetic responses," *NeuroImage*, vol. 47, no. 2, pp. 590–601, 2009.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 852961, 32 pages
doi:10.1155/2011/852961

Research Article

EEG and MEG Data Analysis in SPM8

Vladimir Litvak,[1] Jérémie Mattout,[2] Stefan Kiebel,[3] Christophe Phillips,[4] Richard Henson,[5] James Kilner,[1] Gareth Barnes,[1] Robert Oostenveld,[6] Jean Daunizeau,[1] Guillaume Flandin,[1] Will Penny,[1] and Karl Friston[1]

[1] *The Wellcome Trust Centre for Neuroimaging, UCL Institute of Neurology, Queen Square, London WC1N 3BG, UK*
[2] *INSERM U1028, CNRS UMR5292, Lyon Neuroscience Research Centre, Brain Dynamics and Cognition Team, Lyon, F-69500, France*
[3] *Max Planck Institute for Human Cognitive and Brain Sciences, 04303 Leipzig, Germany*
[4] *Cyclotron Research Centre, University of Liège, 4000 Liège, Belgium*
[5] *MRC Cognition and Brain Sciences Unit, Cambridge CB2 7EF, UK*
[6] *Donders Institute for Brain, Cognition, and Behaviour, Radboud University Nijmegen, 6500 HB Nijmegen, The Netherlands*

Correspondence should be addressed to Vladimir Litvak, v.litvak@ion.ucl.ac.uk

Received 24 September 2010; Accepted 7 December 2010

Academic Editor: Sylvain Baillet

SPM is a free and open source software written in MATLAB (The MathWorks, Inc.). In addition to standard M/EEG preprocessing, we presently offer three main analysis tools: (i) statistical analysis of scalp-maps, time-frequency images, and volumetric 3D source reconstruction images based on the general linear model, with correction for multiple comparisons using random field theory; (ii) Bayesian M/EEG source reconstruction, including support for group studies, simultaneous EEG and MEG, and fMRI priors; (iii) dynamic causal modelling (DCM), an approach combining neural modelling with data analysis for which there are several variants dealing with evoked responses, steady state responses (power spectra and cross-spectra), induced responses, and phase coupling. SPM8 is integrated with the FieldTrip toolbox , making it possible for users to combine a variety of standard analysis methods with new schemes implemented in SPM and build custom analysis tools using powerful graphical user interface (GUI) and batching tools.

1. Introduction

Statistical parametric mapping (SPM) is a free and open source academic software distributed under GNU General Public License. The aim of SPM is to communicate and disseminate methods for neuroimaging data analysis to the scientific community that have been developed by the SPM coauthors associated with the Wellcome Trust Centre for Neuroimaging, UCL Institute of Neurology.

The origins of SPM software go back to 1990, when SPM was first formulated for the statistical analysis of positron emission tomography (PET) data [1, 2]. The software incorporated several important theoretical advances, such as the use of general linear model (GLM) to describe, in a generic way, a variety of experimental designs [3] and random field theory (RFT) to solve the problem of multiple comparisons arising from the application of mass univariate tests to images with multiple voxels [4]. As functional magnetic resonance imaging (fMRI) gained popularity later in the decade, SPM was further developed to support this new imaging modality, introducing the notion of a hemodynamic response function and associated convolution models for serially correlated time series. This formulation became an established standard in the field and most other free and commercial packages for fMRI analysis implement variants of it. In parallel, increasingly more sophisticated tools for registration, spatial normalization, and segmentation of functional and structural images were developed [5]. In addition to finessing fMRI and PET analyses, these methods made it possible to apply SPM to structural MRIs [6], which became the field of voxel-based morphometry (VBM).

The first decade of the 21st century brought about two further key theoretical developments for SPM: increasing

use of Bayesian methods (e.g., posterior probability mapping [7]) and a focus on methods for studying functional integration rather than specialization. Dynamic causal modelling (DCM [8]) was introduced as a generic method for studying functional integration in neural systems. This approach uses Bayesian methods for fitting dynamic models (formulated as systems of differential equations) to functional imaging data, making inferences about model parameters and performing model comparison. Bayesian model comparison uses an approximation to the model evidence (probability of the data given the model). Model evidence quantifies the properties of a good model; that is, that it explains the data as accurately as possible and, at the same time, has minimal complexity [9–11]. Further development and refinement of DCM and related methods are likely to remain the focus of research in the future.

In the second half of the decade, the research focus of the SPM group shifted towards the analysis of MEG and EEG (M/EEG). This resulted in three main developments. First, the "classical" SPM approach was extended to the analysis of M/EEG scalp maps [12–14] and time-frequency images [15]. Second, a new approach to electromagnetic source reconstruction was introduced based on Bayesian inversion of hierarchical Gaussian process models [16–18]. The Bayesian perspective was also applied to the problem of equivalent current dipole modelling [19]. Third, DCM was extended to M/EEG data and several variants of the approach were validated, focusing on evoked responses [20], induced responses [21], steady state responses [22], and phase coupling [23]. In order to make it possible for our colleagues to apply these methods easily to their data, infrastructure for conversion and pre-processing of M/EEG data from a wide range of recording systems were incorporated in the SPM software, with notable contribution from the developers of the FieldTrip software (http://www.ru.nl/donders/fieldtrip, see Oostenveld et al. in this issue). SPM for M/EEG is constructed to support the high-level functionality developed by our group and is not intended as a generic repository of useful methods. This distinguishes SPM from some other toolboxes (e.g., FieldTrip).

The present paper focuses on the implementation of these tools in the most recent SPM version, SPM8. We will not rehearse all the technical details of the methods, for which the reader is referred to the relevant papers. Moreover, we will also avoid focusing on specific interface details, as these often change with intensive SPM development. The details we do mention are correct for SPM8 version 4010, released on July 21, 2010. Our aim is to provide an overview of SPM functionality and the data analysis pathways that the software presently supports. This overview is quite long and inclusive. This reflects the fact that the software covers three distinct domains, source reconstruction, statistical parametric mapping (topological inference over various spaces) and dynamic causal modeling. Each entails a set of assumptions and procedures, some of which are fairly basic and common to most analyses of electromagnetic data and some of which are unique to the applications we consider. We have elected to cover all the basic issues for completeness and to relate them to the specific issues within each domain. However, the readers who are familiar with the basic parts could easily skip these sections. The paper is organized as follows. After presenting a brief overview of the SPM8 user interface, we focus on each of the three core parts of SPM for M/EEG presented above: (i) statistical analysis of images, (ii) Bayesian source reconstruction, and (iii) DCM for M/EEG. The Appendix describes the M/EEG pre-processing infrastructure in SPM8 and explains how to get from raw M/EEG data to the format suitable for analysis with one of the core SPM methods.

2. SPM8 Interface and Overview

The SPM software consists of a library of MATLAB M-files and a small number of C-files, using the MATLAB MEX gateway for the most computer-intensive operations. Its installation simply consists of unpacking a ZIP archive on the user computer and adding the root SPM directory to the MATLAB path. More details on the installation (especially compilation of the MEX files if needed) can be found on the SPM wiki on Wikibooks (http://en.wikibooks.org/wiki/SPM). SPM requires a prior installation of MATLAB, a commercial high-end numerical software platform developed by the MathWorks, Inc. (Natick, USA). More specifically, SPM requires version R14SP3 (released in 2005) or any more recent version (up to the latest R2010b). It runs on any platform supported by MATLAB, that is, Microsoft Windows, Macintosh OS, and Linux, 32- and 64-bit. A standalone version of SPM8, compiled using the MATLAB compiler, is available upon request—it allows using most of the SPM functionalities without requiring the availability of a MATLAB licence.

SPM for M/EEG can be invoked by typing **spm eeg** on the MATLAB command line and pressing Enter. After a brief initialization, the SPM GUI will appear. It consists of three windows (see Figure 1). The menu window on the top left (Figure 1(a)) contains buttons and other GUI elements used to access different SPM functions. This window's contents only change if the user switches modalities (fMRI/PET/MEEG). The interactive window (bottom left, Figure 1(b)) is used by SPM functions for creating dynamic GUI elements, when necessary (for instance, to present the user with a choice or ask for input). The graphics window on the right (Figure 1(c)) is where SPM presents intermediate and final results of its analyses. It is also used by the SPM M/EEG reviewing tool. Additional graphics windows are created when necessary.

There are three ways to access SPM M/EEG functionality. The first is to use the GUI. Since SPM8 is a GUI-based application, all the standard pre-processing and analysis procedures can be accessed this way with no need for programming. We recommend that beginners use the GUI first, because this will prompt SPM to ask for all relevant information needed to process the data. The second way is to use the matlabbatch tool (Figure 1(d)). Matlabbatch (http://sourceforge.net/projects/matlabbatch/) is a standalone batch system for MATLAB developed by Volkmar Glauche, based on the job manager, originally

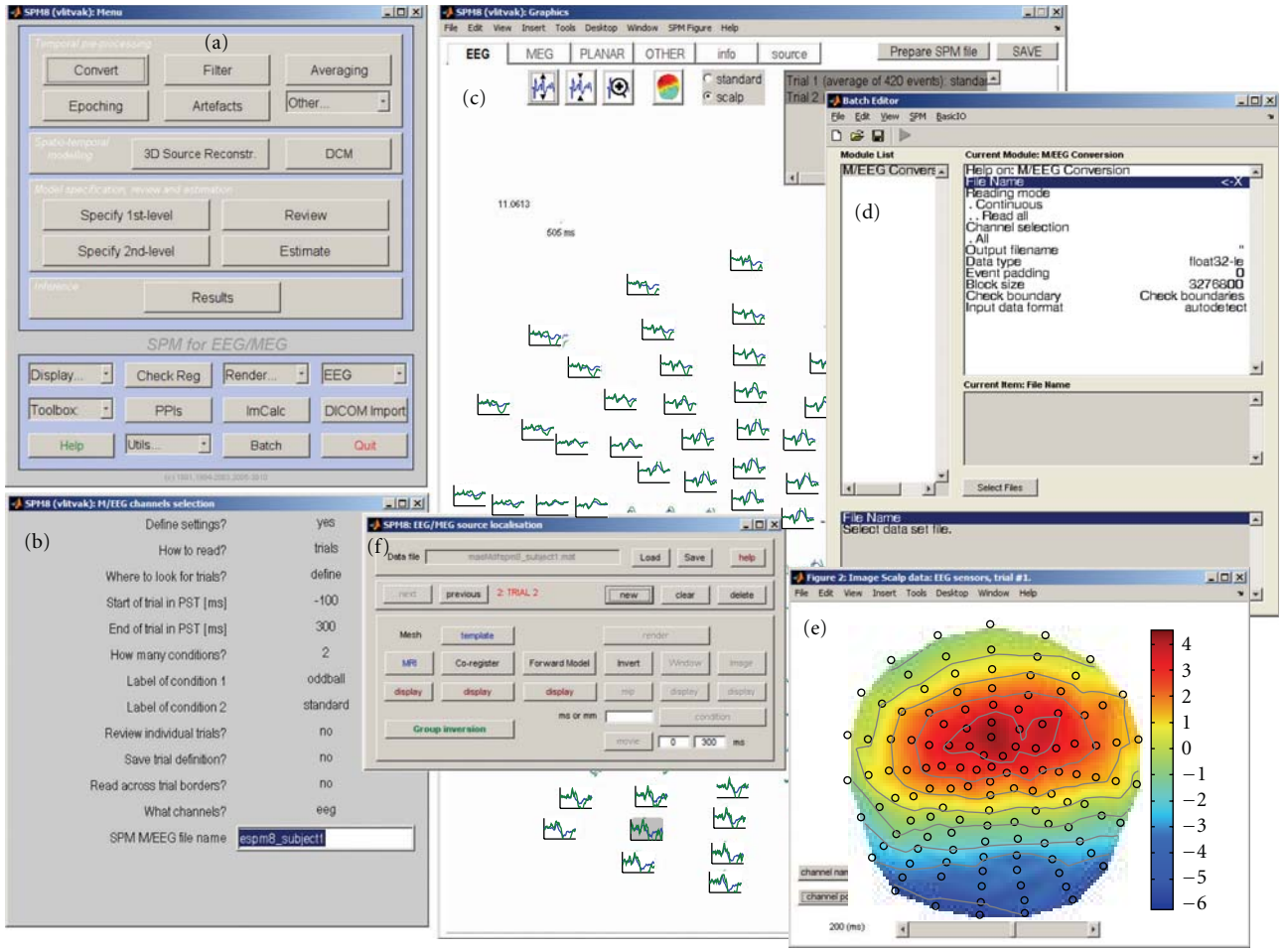


Figure 1: SPM8 for M/EEG graphical user interface tools; see also Figure 11. (a) Menu window, (b) interactive window showing a series of inputs required for conversion of an EEG dataset, (c) graphics window with the SPM8 for M/EEG reviewing tool. Evoked responses recorded in a mismatch negativity experiment are displayed in a topographical plot, (d) MATLAB batch tool with the configuration interface for data conversion, (e) Scalp map of potential distribution for mismatch negativity data that was created from the reviewing tool, and (f) 3D source reconstruction interface.

developed for SPM5. Matlabbatch basically allows "programming without programming". Processing pipelines can be built and configured using a specialized batch GUI and then applied to multiple datasets in noninteractive mode. The batch system is designed for repetitive analyses of data, once the user knows what should be done, and in which order. Matlabbatch can be accessed by pressing the "Batch" button in the SPM menu window. This will open the batch tool window. SPM functionality can be accessed via the "SPM" menu in this window. Finally, users familiar with MATLAB programming can call SPM functions directly from their scripts without using the GUI. We will refer to this way of using SPM as "scripting" as opposed to "batching"; that is, using the batch tool. The use of GUI, batching, and scripting are not always clearly separated, as for some functions the batch tool is the only available GUI. Also, batch pipelines can be created, modified, and run via scripts. In fact, creating a template batch and then invoking it from a script with specific inputs is the most convenient way to prescribe some of the more complicated analyses in SPM. Thus, SPM scripts can combine the user's own code with invoking SPM functions directly or via batch pipelines. The facilities used by SPM programmers to create dynamic GUIs and batch tools are also available to users for their own custom tools.

All the analysis procedures in SPM are optimized to reduce computation time: typically an analysis of a single dataset (e.g., source reconstruction or DCM) can be completed in minutes (or in the worst case, tens of minutes) on a standard desktop computer. SPM does not require any special computer infrastructure or parallel computations, although one of our development directions is to introduce parallelization to finesse analysis of multiple subjects and fitting multiple alternative models to the data.

In what follows, we consider the three main domains in which SPM functionality is used. We start with analyses of M/EEG data in sensor space and then proceed to source space analyses in the subsequent sections. Typically, sensor-level analyses are used to identify peristimulus time or frequency windows, which are the focus of subsequent analyses in source space. These sensor space analyses use, effectively,

standard SPM procedures (topological inference) applied to a variety of electromagnetic data features that are organised into images.

3. Sensor-Level Analysis and Topological Inference

EEG and MEG typically produce a time-varying modulation of signal amplitude or frequency-specific power in some peristimulus time period, at each electrode or sensor. Often, researchers are interested in whether condition-specific effects (observed at particular sensors and peristimulus times) are statistically significant. However, this inference must correct for the number of statistical tests performed. One way to do so is to control the family-wise error rate (FWER), the probability of making a false positive over the whole search space [24]. For independent observations, the FWER scales with the number of observations. A simple method for controlling FWER is the Bonferroni correction. However, this procedure is rarely adopted in neuroimaging because it assumes that neighbouring observations are independent. When there is a high degree of correlation among neighbouring samples (e.g., when data features are smooth), this correction is far too conservative.

Although the multiple comparisons problem has always existed for M/EEG analyses (due to the number of time bins in the peristimulus time window), the need for a correction method has become more acute with the advent of high-density EEG caps and MEG sensor arrays that increase the number of observations across the scalp. In many analyses, the multiple comparisons problem is circumvented by restricting the search space prior to inference, so that there is only one test per repeated measure. This is usually accomplished by averaging the data over prespecified sensors and time bins of interest. This produces one summary statistic per subject per condition. In many instances, this is a powerful and valid way to sidestep the multiple comparisons problem; however, it requires the space of interest to be specified *a priori*. A principled specification of this space could use orthogonal or independent data features. For example, if one were interested in the attentional modulation of the N170 (a typical event-related wave recorded 170 ms after face presentation), one could first define the electrodes and time bins that expressed an N170 (compared to baseline) and then test for the effects of attention on their average. Note that this approach assumes that condition-specific effects occur at the same sensors and time, and is only valid when selection is not biased [25]. In situations where the location of evoked or induced responses is not known *a priori* or cannot be localized independently, one can use topological inference to search over some space for significant responses; this is the approach implemented in SPM. It is based on the random field theory (RFT [4]). RFT provides a way of adjusting the P-values that takes into account the fact that neighbouring sensors are not independent, by virtue of continuity in the original data. Provided the data are smooth, the RFT adjustment is less severe (i.e., is more sensitive) than a Bonferroni correction for the number of sensors. The theoretical basis of topological inference for M/EEG has been recently reviewed by Kilner and Friston [14]. Here, we rehearse some of the points from this review and provide more details about the SPM implementation of the method.

Statistical analyses of M/EEG data in SPM use the same mechanisms as all other data types (PET, fMRI, and structural MRI in VBM). This simply requires transforming data from SPM M/EEG format to image files (NIfTI format, http://nifti.nimh.nih.gov/nifti-1/). Once the data are in this image format, statistical analyses for M/EEG are procedurally identical to between-subject analyses of PET or VBM data (e.g., second level analyses in fMRI [26]). These analyses assume one summary statistic image per subject per condition (or level of an experimental factor). Here, a summary statistic image is just a technical term for the data feature summarising treatment effects that one wants to make an inference about. More formally, when this summary statistic is itself a maximum likelihood estimate based on within-subject data, the analysis is called a summary-statistic procedure for random effect models. In the present context, we will see that the summary statistic can comprise many different data features.

3.1. Creating Summary Statistics: Conversion to Images. This function takes SPM M/EEG sensor data as input and generates an image for each trial (for trials that were not rejected). This analysis can be applied to EEG and MEG (separately). In the case of MEG systems with planar gradiometers, images can be generated from root-mean-square values combining the two planar gradiometers at each location. In an averaged dataset, this will produce a single image per condition and enable statistical comparisons across subjects. In an epoched dataset, there will be an image per trial and multiple images per condition. It is, therefore, possible to perform within-subject statistical tests and then also take further summary statistic images (usually contrasts of parameter estimates from the within-subject models) from each subject to second level analyses between subjects.

3.1.1. Images over Time. Data in the time domain are converted into an image by generating a scalp map for each time frame and stacking scalp maps over peristimulus time (see Figure 2). Scalp maps are generated using the 2D sensor layout specified in the dataset (see Appendix B.3) and linear interpolation between sensors. The user is asked to specify the output dimensions of the interpolated scalp map. Typically, we suggest 64 pixels in each spatial direction. There is also an option to either interpolate or remove bad channels from the images. Interpolation is the preferred option when there is a sufficient number of good channels around each bad channel. If bad channels are removed, there will be "holes" in the resulting images and these holes will be propagated throughout the statistical analysis. A directory is created with the same name as the input dataset. In this directory there will be a subdirectory for each trial type. These directories will contain 3D image files, where the dimensions are space (x, y) and time (z). In the case of

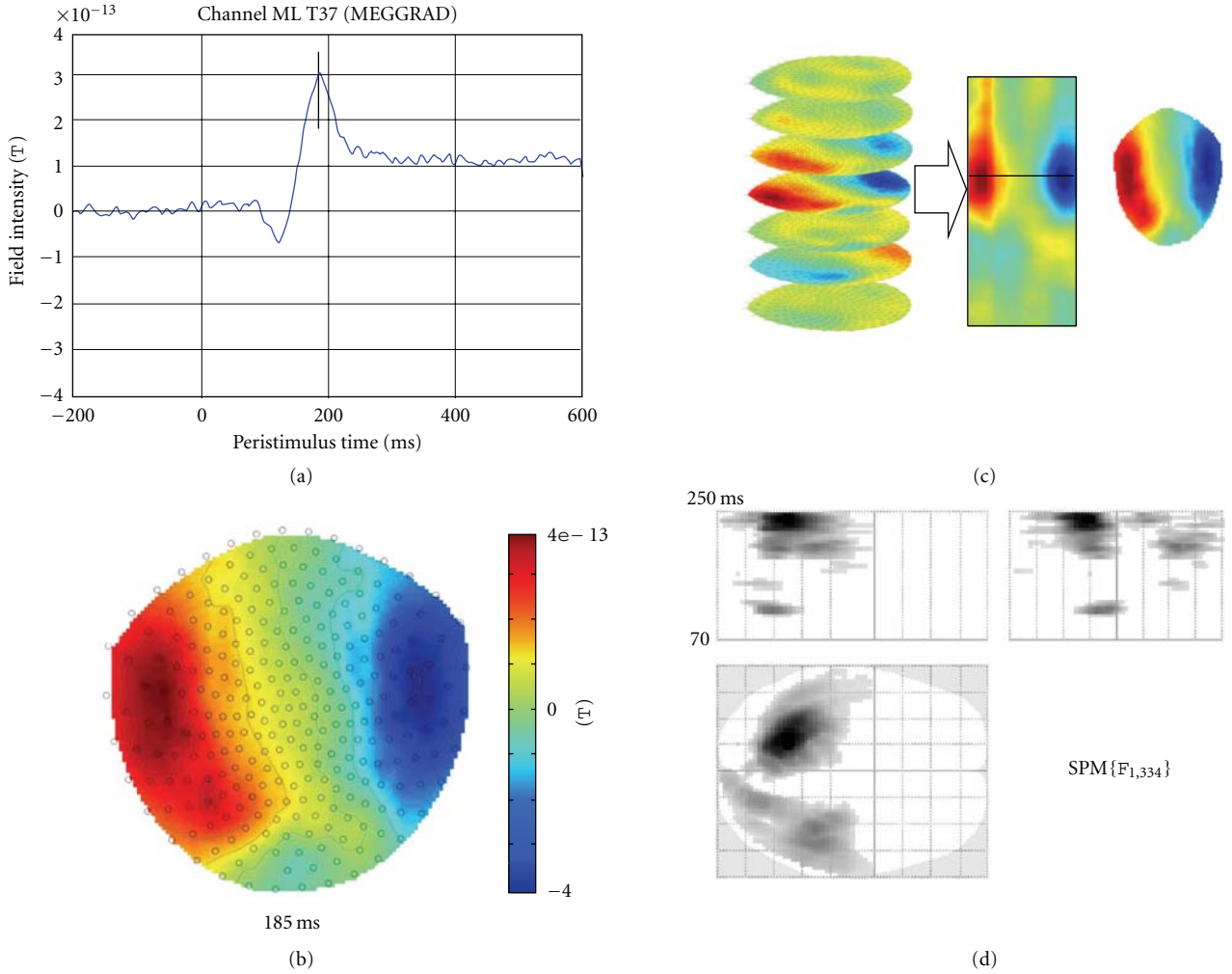


FIGURE 2: Construction of (space × space × time) summary statistic image and the ensuing SPM inference. The data are MEG responses to presentation of images of faces and scrambled faces. (a) Average ERF for a single subject recorded at a left temporal sensor in response to face presentation. The vertical line indicates the maximum positive value of this ERF. (b) Sensor-space map interpolated across all sensors at 185 ms after the stimulus, indicated by the line in (a). (c) Construction of a 3D (space × space × time) data volume from sensor-space maps, such as shown in (b). (d) Results of F test for difference between responses to faces and scrambled faces. Overall, single trials (168 for each condition) were converted to images as shown in (c). A two-sample t-test was performed and the results were assessed with an F-contrast to test for differences of either polarity. The results were thresholded at $P = .05$ with FWER correction based on random field theory. The red arrow indicates the peak value of the F-statistic (at 245 ms).

averaged data (e.g., an event-related potential-ERP), a single image is placed in each directory. In the case of epoched data, there will be an image for each trial.

3.1.2. Averaging over Time. If the time window of interest is known in advance (e.g., in the case of a well-characterized ERP or event-related field (ERF) peak) one can average over this time window to create a 2D image with just the spatial dimensions.

3.1.3. Time-Frequency Data. Although, in principle, topological inference can be done for any number of dimensions, the present implementation in SPM8 is limited to 3 dimensions or less. Thus, when time-frequency features are exported to summary statistic images, it is necessary to reduce the data dimensionality from 4D (space × space × time × frequency) to either a 3D image (space × space × time) or a 2D time-frequency image (time × frequency). This is achieved by averaging either over channels (space × space) or frequencies. Averaging over channels (or as a common special case, selecting one channel) furnishes 2D time-frequency images (Figure 3).

When averaging over frequencies, one needs to specify the frequency range of interest. The power is then averaged over the specified frequency band to produce channel waveforms. These waveforms are saved in a new time-domain M/EEG dataset. This dataset can be reviewed and further processed in the same way as ordinary time domain datasets (source reconstruction or DCM would not be appropriate because the data features are power or energy [21]). Once this

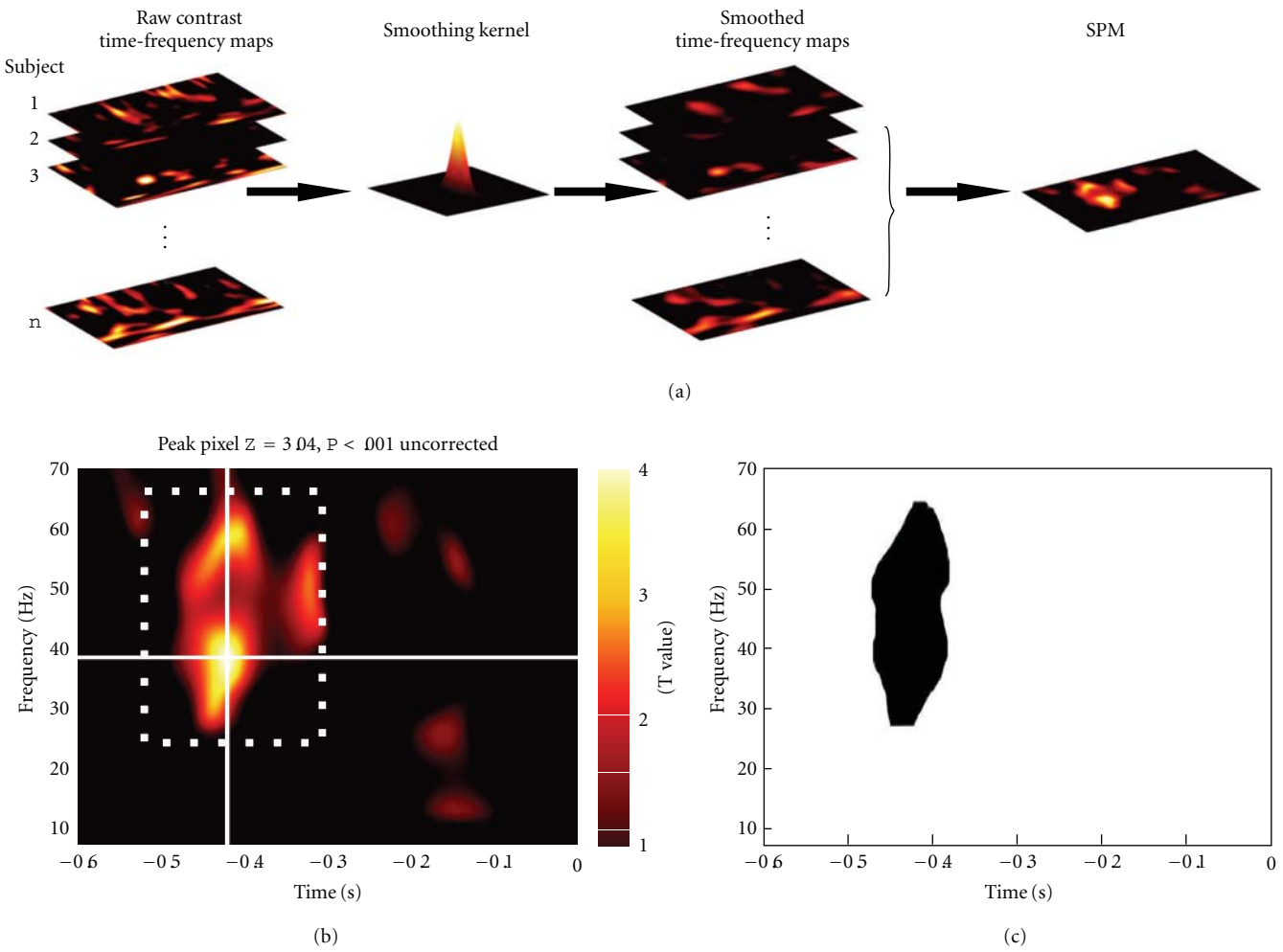


Figure 3: SPM analysis of time × frequency images. (a) Time-frequency images were calculated for each subject and smoothed by convolution with a Gaussian kernel. (b) A t-statistic image was calculated from the smoothed time-frequency images and thresholded at $P < .01$ (uncorrected). The location of the peak bin is shown. The white dotted box indicates our illustrative *a priori* window of interest. (c) Statistical test restricted to the window of interest shown in (b) revealed a significant cluster ($P = .005$, cluster-level FWER correction). This figure was adapted with permission from [15].

dataset is generated, it is automatically exported to images in the same way as data in the time domain (see above).

3.1.4. Smoothing. The images generated from M/EEG data are generally smoothed prior to second level (i.e., group level) analysis by multidimensional convolution with a Gaussian kernel (standard image smoothing available in SPM). Smoothing is necessary to accommodate spatial/temporal variability over subjects and ensure the images conform to the assumptions of the topological inference approach. The dimensions of the smoothing kernel are specified in the units of the original data: [mm × mm × ms] for space-time, [Hz × ms] for time-frequency images. The guiding principle for deciding how much to smooth is based on the matched filter theorem, which says that the smoothing kernel should match the scale of data features one expects. Therefore, the spatial extent of the smoothing kernel should be more or less similar to the extent of the dipolar patterns expected in the data (probably of the order of a few cm). In practice, one can try smoothing the images with different kernels, according to the principle above; this is a form of scale space search or feature selection. Smoothing in time is not always necessary, as temporal filtering has the same effect. Once the images have been smoothed, one can proceed to the second level analysis.

Figure 2 is a schematic illustrating the construction of (space × space × time) summary-statistic image and the ensuing SPM testing for an effect of faces versus scrambled faces stimuli over subjects. This example highlights the role of topological inference (based on random field theory) to identify significant sensor-time regions that contain a significant condition-specific response. Figure 3 illustrates a typical analysis in time × frequency space using a single channel. These analyses can then be reported directly or used to finesse the subsequent characterisation of the appropriate peristimulus time window and frequency bands in source space.

4. Source Analysis

This section focuses on the imaging (or distributed) methods for EEG/MEG source reconstruction in SPM. This approach results in a spatial projection of sensor data into (3D) brain space and considers brain activity as comprising a very large number of dipolar sources spread over the cortical sheet, with fixed locations and orientations. This renders the observation model linear, the unknown variables being the source amplitudes. Given epoched and preprocessed data, the evoked and/or induced activity for each dipolar source can be estimated, for either a short time segment or a wider peristimulus time window. The reconstructed activity is in 3D voxel space and can then be analyzed using mass univariate analysis in SPM, using appropriate summary statistic images over time and/or frequency.

In contrast to PET/fMRI image reconstruction, M/EEG source reconstruction is a nontrivial operation. Often compared to estimating a body shape from its shadow, inferring brain activity from scalp data is mathematically ill-posed and requires prior information such as anatomical, functional, or mathematical constraints to isolate a unique and highly probable solution [27]. Distributed linear models have been around for more than a decade now [28], and the recommended pipeline in SPM for an imaging solution is very similar to common approaches in the field [29, 30]. However, at least three aspects are original and should be emphasized here.

(i) Based on an empirical Bayesian formalism, the inversion is meant to be generic, in the sense that it can incorporate and estimate the relevance of multiple constraints of a varied nature (i.e., it can reproduce a variety of standard constraints of the sort associated with minimum norm [29], LORETA [30], and other well-known solutions to the inverse problem). The data-driven relevance of different constraints (priors) is established through Bayesian model inversion, and different sets of constraints can be evaluated using Bayesian model comparison [16–18, 31, 32].

(ii) Subject-specific anatomy is incorporated in the generative model of the data, in a fashion that eschews individual cortical surface extraction. The individual cortical mesh is obtained automatically from a canonical mesh in MNI space, providing a simple and efficient way of reporting results in stereotactic coordinates [33].

(iii) SPM uses a Gaussian process model [34, 35] for source reconstruction based on the sample channel $\times$ channel covariance of the data over time. Crucially, this means it does not reconstruct one time bin at a time but uses the variance over time to furnish a full spatiotemporal inversion for each time series. This finesses any problems with specifying baselines, because only the variance (change from prestimulus baseline) contributes to the sample covariance and, therefore, the solution. In short, SPM reconstructs changes in source activity (not activity *per se*). This becomes important when specifying the time window for inversion (see below).

The M/EEG imaging pipeline is divided into four consecutive steps, which characterize any inverse procedure with an additional step of summarizing the results. In this section, we go through each of the steps that comprise a full inverse analysis.

(i) Source space modelling.

(ii) Data coregistration.

(iii) Forward computation.

(iv) Inverse reconstruction.

(v) Summarizing the reconstructed response as an image.

Whereas the first three steps specify the forward or generative model, the inverse reconstruction step is concerned with Bayesian inversion of that model and is the only step that requires the EEG/MEG data.

4.1. Getting Started. Everything described below is accessible from the SPM user interface by pressing the "3D Source Reconstruction" button. A new window will appear with a GUI that guides the user through the necessary steps to obtain an imaging reconstruction of their data (see Figure 1(f)). At each step, the buttons not yet relevant for this step will be disabled. At the beginning, only two buttons are enabled: "Load", which is used to load a preprocessed SPM M/EEG dataset and the "Group inversion" button that will be described below. One can load a dataset that is either epoched with single trials for different conditions, averaged with one ERP/ERF per condition, or grand averaged. An important precondition for loading a dataset is that it should contain sensors and fiducials (see "Section 4.3."). This will be checked when loading a file and loading will fail if there is a problem. The user should make sure that for each modality in the dataset as indicated by channel types (either EEG or MEG), there is a sensor description. For instance, to load MEG data with some EEG channels that are not actually used for source reconstruction, the type of these channels should be changed to "LFP" (local field potential) or "Other" before trying to load the dataset. Unlike "Other" channels, "LFP" channels are filtered and are available for artefact detection. MEG data converted by SPM from their raw formats will usually contain valid sensor and fiducial descriptions. In the case of EEG, for some supported channel setups (such as extended 10–20 or Biosemi), SPM will provide default channel locations and fiducials that can be used for source reconstruction. Sensor and fiducial descriptions can be modified using the "Prepare" interface (see Appendix B.3).

When a dataset is loaded, the user is asked to give a name to the reconstruction. In SPM, it is possible to perform multiple reconstructions of the same dataset with different parameters. The results of these reconstructions will be stored with the dataset after pressing the "Save" button. They can be loaded and reviewed using the "3D

Source Reconstruction" GUI and also with the SPM M/EEG reviewing tool. From the command line, one can access source reconstruction results via the D.inv field of the @meeg object. This field (if present) is a cell array of structures. Each cell contains the results of a different reconstruction. One can navigate among these cells in the GUI, using the buttons in the second row. One can also create, delete, and clear analysis cells. The label provided at the beginning will be attached to the cell for the user to identify it.

4.2. Source Space Modelling. After entering the label, the "Template" and "MRI" buttons will be enabled. The "MRI" button creates individual head meshes describing the boundaries of different head compartments based on the subject's structural scan. SPM will ask for the subject's structural image. It might take some time to prepare the model, as the image needs to be segmented as part of computing the nonlinear transformation from individual structural spaces to the template space [5]. The individual meshes are generated by applying the inverse of the spatial deformation field, which maps the individual structural image to the MNI template, to canonical meshes derived from this template [33], Figure 4(b). This method is more robust than deriving the meshes from the structural image directly and can work even when the quality of the individual structural images is low.

In the absence of an individual structural scan, combining the template head model with the individual head shape also results in a fairly precise head model. The "Template" button uses SPM's template head model based on the MNI brain. The corresponding structural image can be found under canonical/single_subj_T1.nii in the SPM directory. When using the template, different things are done depending on whether the data are EEG or MEG. For EEG, the electrode positions will be transformed to match the template head. So even if the subject's head is quite different from the template, one should be able to obtain reasonable results. For MEG, the template head will be transformed to match the fiducials and head shape that come with the MEG data. In this case, having a head shape measurement can be helpful in providing SPM with more data to scale the head correctly.

Irrespective of whether the "MRI" or "Template" button is used, the cortical mesh describing the locations of possible sources of the EEG and MEG signal is obtained from a template mesh (Figure 4(a)). In the case of EEG, the mesh is used as is, and in the case of MEG it is transformed with the head model. Three cortical mesh sizes are available: "coarse", "normal", and "fine" (5124, 8196, and 20484 vertices, resp.). We advise to work with the "normal" mesh. "Coarse" is useful for less powerful computers and "fine" will only work on 64-bit systems with enough main memory. The inner-skull, outer-skull, and scalp canonical surfaces each comprise 2562 vertices, irrespective of the cortical mesh size.

For the purposes of forward computation, the orientations of the sources are assumed to be normal to the cortical mesh. This might seem as a hard constraint at first glance, especially for the "Template" option, where the details of the mesh do not match the individual cortical anatomy. However, in our experience, when a detailed enough mesh is used, the vertices in any local cortical patch vary sufficiently in their orientation to account for any activity that could come from the corresponding brain area; provided the mesh is sufficiently dense. All the (three) mesh resolutions offered by SPM provide sufficient degrees of freedom in this context. When comparing meshes with free and fixed orientation, Henson et al. [36] found the latter to be superior for SPM's default source reconstruction method.

4.3. Data Coregistration. For SPM to provide a meaningful interpretation of the results of source reconstruction, it should map the coordinate system in which sensor positions are originally represented to the coordinate system of a structural MRI (MNI coordinates).

There are two possible ways of coregistering M/EEG data to the structural MRI space.

(i) A landmark-based coregistration (using fiducials only). The rigid-body transformation matrices (rotation and translation) are computed such that they match each fiducial in the M/EEG space to the corresponding one in MRI space. The same transformation is then applied to the sensor positions.

(ii) Surface matching (between some head shape in M/EEG space and some MRI-derived scalp tessellation).

For EEG, the sensor locations can be used instead of the head shape. For MEG, the head shape is first coregistered with MRI space; the inverse transformation is then applied to the head model and the mesh. Surface matching is performed using an iterative closest point algorithm (ICP). The ICP algorithm [37] is an iterative alignment algorithm that works in three phases.

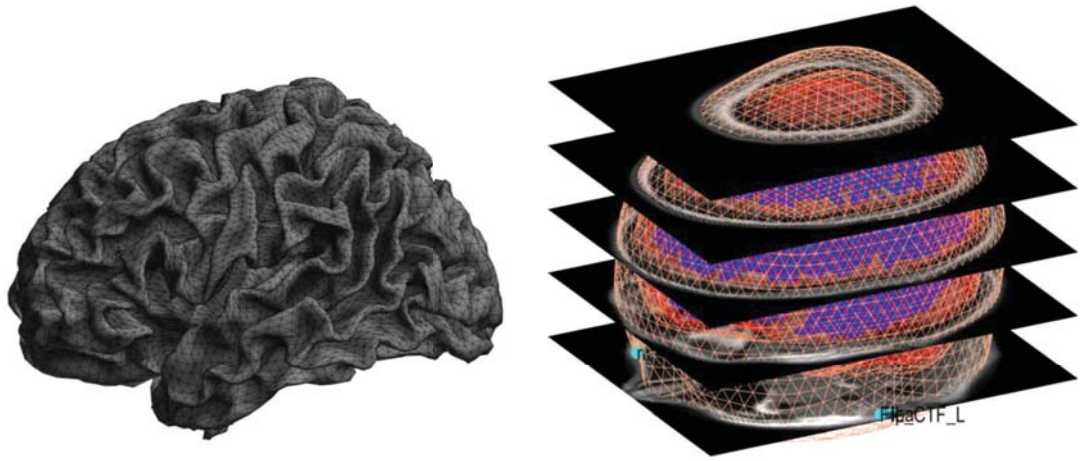
(i) Establish correspondence between pairs of features in the two structures that are to be aligned, based on proximity.

(ii) Estimate the rigid transformation that best maps the first member of the pair onto the second.

(iii) Apply that transformation to all features in the first structure. These three steps are then reapplied until convergence. Although simple, the algorithm works quite effectively when given a good initial estimate.

In practice, after pressing the "Coregister" button one needs to specify the points in the MRI that correspond to the M/EEG fiducials. If more than three fiducials are available (which may happen for EEG as, in principle, any electrode can be used as a fiducial), the user is asked at the first step to select the fiducials to use. It is possible to select more than three, but not less. Then for each M/EEG fiducial selected, the user is asked to specify the corresponding position in the MRI in one of three ways.

(i) "Select"—locations of some points such as the commonly used nasion and preauricular points and



(a) (b)

Figure 4: Template meshes used for distributed source imaging. (a) "Normal" cortical template mesh (8196 vertices), left view. The triangular grid shows the representation of the cortical surface used by SPM. (b) All template meshes (cortex, inner skull, outer skull, and scalp) superimposed on the template MRI. Default fiducial locations associated with the template anatomy are displayed in light blue.

also CTF-recommended fiducials for MEG are hard-coded in SPM. If an M/EEG fiducial corresponds to one of these points, the user can select this option and then select the correct point from a list.

(ii) "Type"—here it is possible to enter the MNI coordinates for the fiducial (1×3 vector). If the fiducial is not in the SPM hard-coded list, it is advised to carefully find the correct point on either the template image or on the subject's own image registered to the template. This can be done by opening the image using SPM's image display functionality. One can then record the MNI coordinates and use them in subsequent coregistration, using the "type" option.

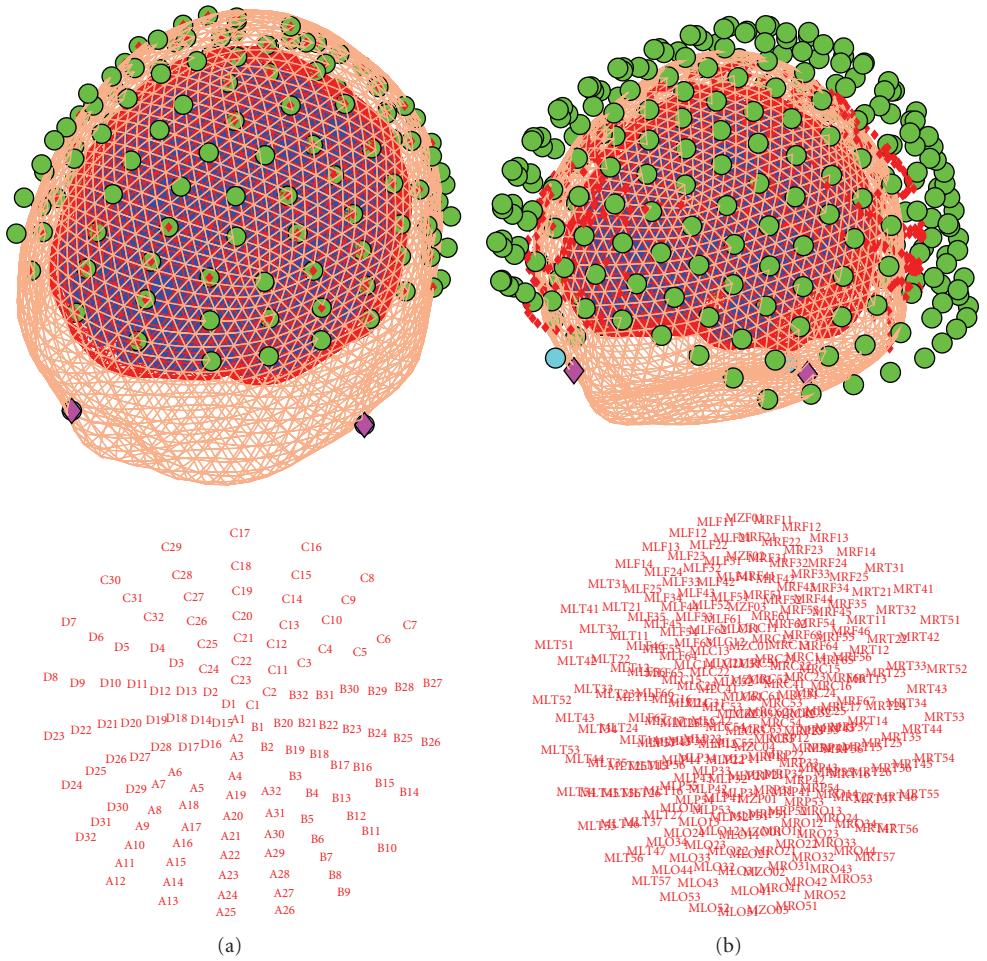
(iii) "Click"—the user is presented with a structural image and can click on the correct point. This option is good for "quick and dirty" coregistration or to try out different options.

After specifying the fiducials, the user is asked whether to use the head shape points if they are available. For EEG this is advised. For MEG, the head model is based on the subject's MRI, and precise information about the fiducials is available (e.g., from a MRI with fiducials marked by vitamin E capsules); using the head shape might actually do more harm than good.

The results of the coregistration are presented in SPM's graphics window (see Figure 5). It is important to examine the results carefully before proceeding. The top panel shows the scalp, the inner skull, and the cortical mesh, with the sensors and the fiducials. For EEG one should make sure that the sensors are on the scalp surface. For MEG one should check that the head position, in relation to the sensors, makes sense and the head does not, for instance, protrude outside the sensor array. In the bottom panel, the sensor labels are shown in topographical array. One should check that the top labels correspond to anterior sensors, bottom to posterior, left to left, and right to right and also that the labels are where expected topographically (e.g., that there is no shift when matching positions to channels).

4.4. Forward Computation. This refers to computing, for each dipole on the cortical mesh, the effect it would have on the sensors. The result is an $N \times M$ matrix where N is the number of sensors and M is the number of mesh vertices (chosen from several options at a previous step). This matrix can be quite large and is therefore stored in a separate MAT-file (whose name starts with "SPMgainmatrix"). This file is written to the same directory as the dataset. Each column in this matrix is a so-called "lead field", corresponding to one mesh vertex. The lead fields are computed using the "forward" toolbox, which SPM shares with FieldTrip (see Oostenveld et al., this issue). This computation is based on Maxwell's equations and makes assumptions about the physical properties of the head. There are different ways to specify these assumptions which are known as "forward models".

The "forward" toolbox supports different forward models. After pressing the "Forward Model" button (which should be enabled after successful coregistration), the user has a choice of several head models, depending on the modality of the data. In SPM8, we recommend using a "single shell" model [38] for MEG and "EEG BEM" (Boundary Elements Model [39–43]) for EEG. One can also try other options and compare them using their model evidence ([36], see below). The first time the EEG BEM option is used with a new structural image (and also the first time the "Template" option is used) a lengthy computation will take place that prepares the BEM model based on the head meshes. The BEM will then be saved in a large MAT-file with ending "_EEG_BEM.mat" in the same directory as the structural image (this is the "canonical" subdirectory of SPM for the template). When the head model is ready, it will be displayed in the graphics window, with the cortical mesh and



(a) (b)

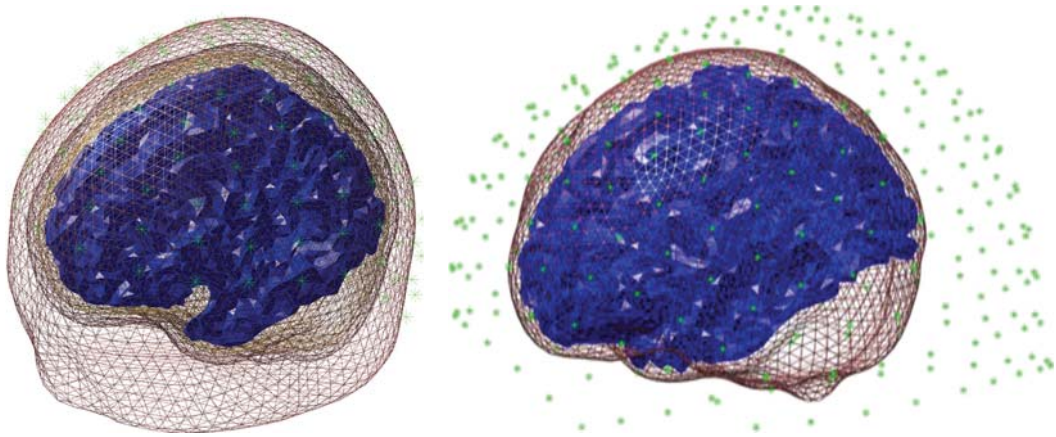
Figure 5: Examples of coregistration display (appears after the co-registration step has been completed). Top row shows the 3D outcome of the co-registration, while the bottom row shows the sensor arrangement in 2D with corresponding labels. (a) EEG data (128 Biosemi system) from the multimodal face perception experiment available from the SPM website. EEG sensor locations have been adjusted to fit the scalp surface. (b) MEG data (275-channel CTF system) from the same experiment.

sensor locations, for verification (Figure 6). The actual lead field matrix is computed at the beginning of the next step and saved. This is a time-consuming step, particularly for high-resolution meshes. The lead field file will be used for all subsequent inversions, if the coregistration and the forward model are not changed.

4.5. Inverse Reconstruction. The inverse reconstruction is invoked by pressing the "Invert" button. The first choice one gets is between "Imaging", "VB-ECD", and "DCM". For reconstruction based on an empirical Bayesian approach (to localize evoked responses, evoked power, or induced power) one should press the "Imaging" button. The other options are explained in greater detail below. When there are several conditions (trial types) in the dataset, then the next choice is whether to invert all the conditions together or to choose a subset. If one is planning a statistical comparison between a set of conditions, one should invert all of them together. After selecting the conditions one gets a choice between "Standard" and "Custom" inversion. For "Standard" inversion, SPM will start the computation with default settings. These correspond to the multiple sparse priors (MSP) algorithm [17], which is then applied to the whole time series.

To fine tune the parameters of the inversion, the "Custom" option can be chosen. There will then be a possibility to choose among several types of inversion, differing in terms of their hyperpriors (priors on priors or constraints): IID—equivalent to classical minimum norm [29], COH—smoothness prior similar to methods such as LORETA [30], or multiple sparse priors (MSP) [17]. The latter gives the most plausible results and has been shown to have greater model evidence in relation to other priors [36].

One can then choose the time window that will be used for inversion. Based on our experience, we recommend the time window be limited to periods in which the activity of interest is expressed. The reason is that if irrelevant high-amplitude activity is included, source reconstruction will focus on reducing the error for reconstructing this activity and might suppress the responses of interest. There is also



(a) (b)

Figure 6: Examples of forward model display (appears after the forward modelling step has been completed). The figure includes the cortical mesh, the sensor locations and the other layers used to compute the lead-field matrix. (a) EEG data (128 Biosemi system) from the multimodal face perception experiment available from the SPM website. This figure shows the head model that was used to compute the BEM forward solution for these data. (b) MEG data (275-channel CTF system) from the same experiment. This figure shows the head model that was used to compute the realistic single shell solution for these data.

an option to apply a Hanning taper to the time series to down-weight possible baseline noise at the beginning and end of the trial. The next option is to prefilter the data. This is mainly for focusing on certain temporal scales during reconstruction (e.g., alpha band for ERPs or gamma for faster induced responses or sensory-evoked responses). The next option allows for extra source priors. This makes it possible to integrate prior knowledge from the literature or from fMRI/PET/DTI into the inversion [44]. Here, one can just provide a thresholded statistical image and SPM will generate the priors based on its suprathreshold clusters. Custom priors are not a "hard" way to restrict the solution. They will only be used when leading to a solution with higher model evidence. A "hard" restriction of the inverse solution is provided by the next option.

Here, one can restrict solutions to particular brain areas by loading (or specifying) a MAT-file with a $K \times 3$ matrix, containing MNI coordinates of the areas of interest. This option may seem strange initially; as it may seem to overly bias the source reconstruction. However, in the Bayesian inversion framework, it is possible to compare different inversions of the same data using Bayesian model comparison. By limiting the solutions to particular brain areas, one can greatly simplify the model, and if this simplification appropriately captures the sources generating the response, then the restricted model will have higher model evidence than the unrestricted one. If, however, the restricted sources cannot account for the data, the restriction will result in a worse model fit and the unrestricted model might be better (note that for model comparison to be valid, all settings that affect the data, like the time window and filtering, should be identical).

SPM8 imaging source reconstruction also supports multimodal datasets. These are datasets that have both EEG and MEG data from a simultaneous recording. Datasets from the Elekta/Neuromag Vectorview MEG system, which has two kinds of MEG sensors, are also treated as multimodal. If the dataset is multimodal, a dialogue box will appear asking one to select the modalities for source reconstruction from a list. When selecting more than one modality, multimodal fusion will be performed. This option uses a heuristic to rescale the data from different modalities so that they can be fused [45].

Once the inversion is complete, the time course of the source with maximal activity is presented in the top panel of the graphics window (see Figure 7). The bottom panel shows the maximum intensity projection (MIP) at the time of the maximum activation. The log-evidence, which can be used for model comparison as explained above, is also shown. Note that not all of the output of the inversion is displayed. The full output consists of time courses for all the sources and conditions for the entire time window. It is possible to view more of these results using the controls in the bottom right corner of the 3D GUI. These allow one to focus on a particular time, brain area, and condition. One can also display a movie of the evolution of source activity.

4.6. Summarizing the Reconstructed Response as an Image. SPM allows one to create summary statistic images in terms of contrasts (mixtures of parameter or activity estimates) over time and frequency. These are in the form of 3D NIfTI images, so that one can proceed to GLM-based statistical analysis in the usual way (at the between-subject level). This entails summarizing the trial- and subject-specific responses with a single 3D image in source space and involves specifying a time-frequency window for each contrast image. This is a flexible and generic way of specifying the data features one wants to make an inference about (e.g., gamma activity around 300 ms or average response between 80 and 120 ms). The contrast is specified by pressing the "Window"

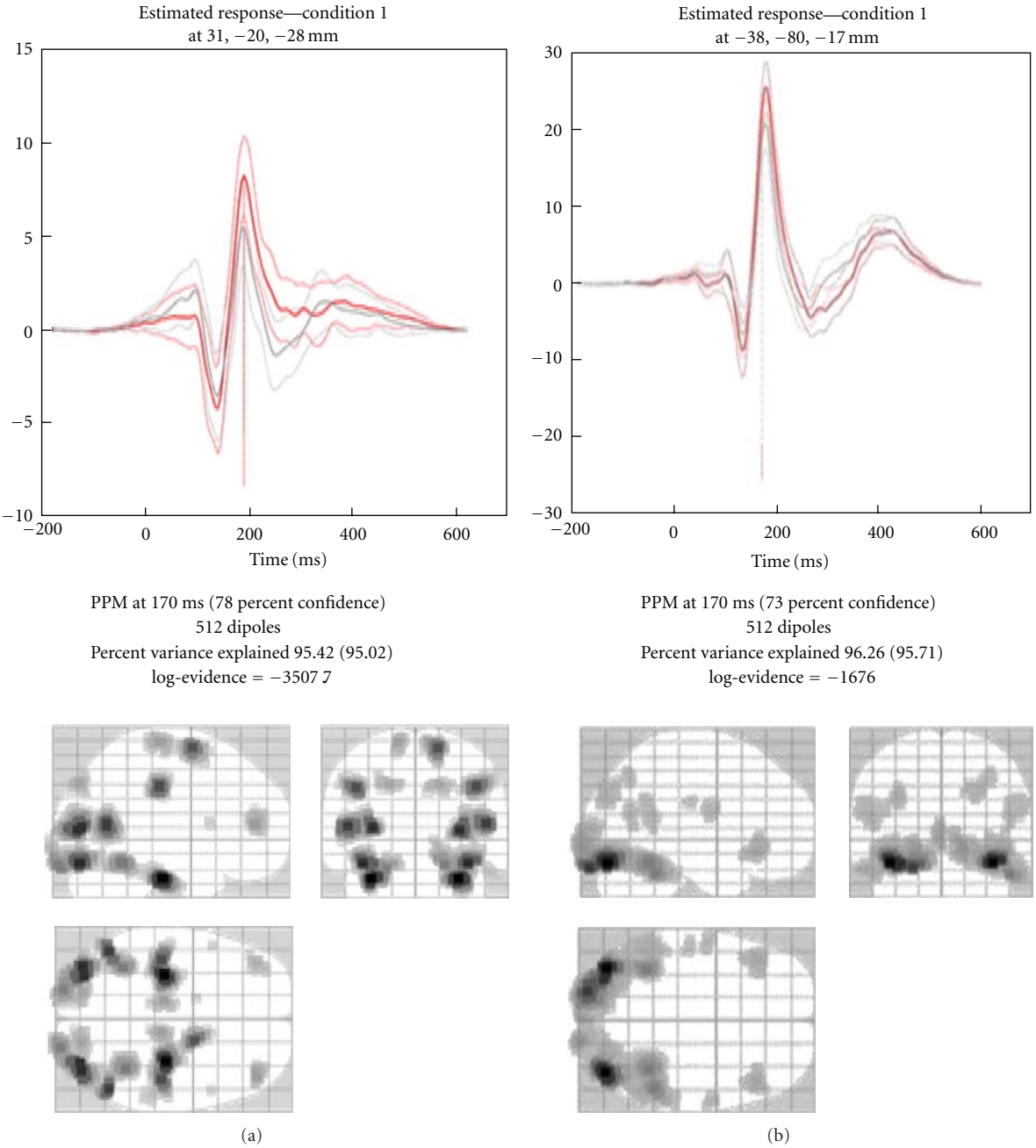


Figure 7: Display of the estimated distributed solution for evoked responses. The top panel shows the time course of the source having maximal activity while the bottom panel shows the maximum intensity projections (MIP) at the time of maximum activation. (a) EEG data from the multimodal face perception experiment available from the SPM website. (b) MEG data from the same experiment. Time course in red is for the face stimuli while the light grey is for scrambled faces. The lighter red and gray lines indicate 90% confidence intervals.

button. The user will then be asked about the time window of interest (in ms, peristimulus time). It is possible to specify one or more time segments (separated by a semicolon). To specify a single time point the same value can be repeated twice. The next prompt pertains to the frequency band. To average the source time course one can simply leave this at the default of zero. In this case, the window will be weighted by a Gaussian function. In the case of a single time point, this will be a Gaussian with 8 ms full width half maximum (FWHM). If one specifies a particular frequency or a frequency band, then a series of Morlet wavelet projectors will be generated, summarizing the energy in the time window and frequency band of interest.

There is a difference between specifying a frequency band of interest as zero, as opposed to specifying a wide band that covers the whole frequency range of the data. In the former case, the time course of each dipole is averaged over time, weighted by a Gaussian. Therefore, if within the selected time

window this time course changes polarity, the activity can average out and even a strong response can produce a value of zero. In the latter case, the power is integrated over the whole spectrum ignoring phase, and this would be equivalent to computing the sum of squared amplitudes in the time domain.

Finally, if the data file is epoched rather than averaged, there is a choice between "evoked", "induced", and "trials". The projectors generated at the previous step can either be applied to each trial and the results averaged (induced) or applied to the averaged trials (evoked). Thus, it is possible to localize induced activity that has no phase locking to the stimulus. It is also possible to focus on frequency content of the ERP using the "evoked" option. Clearly the results will not be the same. The projectors specified (bottom panel of Figure 8) and the resulting MIP (top panel) will be displayed when the operation is completed. The "trials" option makes it possible to export an image per trial, which is useful for performing parametric within-subject analyses (e.g., looking for the correlates of reaction times).

The values of the exported images are normalized to reduce between-subject variance. Therefore, for best results one should export images for all the time windows and conditions that will be included in the same statistical analysis together. Note that the images exported from the source reconstruction are a little peculiar because of smoothing from a 2D cortical sheet into 3D volume (Figure 9). SPM's statistical machinery has been optimized to deal with these peculiarities and ensure sensible results.

In what follows, we consider some auxiliary functions associated with source reconstruction.

4.6.1. Rendering Interface. By pressing the "Render" button one can open a new GUI window, which displays a rendering of the inversion results on the brain surface. One can rotate the brain, focus on different time points, run a movie, and compare the predicted and observed scalp topographies and time series. A useful option is the "virtual electrode", which allows one to extract the time course from any point on the mesh and form the MIP at the time of maximum activation at this point. An additional tool for reviewing the results is available in the SPM M/EEG reviewing tool.

4.6.2. Group Inversion. A problem encountered with MSP inversion is that it sometimes produces solutions that are so focal in each subject that the spatial overlap between the activated areas across subjects is not sufficient to yield a significant result at the between-subject level. This could be finessed by smoothing, but smoothing compromises the spatial resolution and thus subverts the main advantage of using an inversion method that can produce focal solutions. The more principled solution is to tell the model that the same distributed brain system has been engaged in all subjects or sessions (by design). This is simple to do using a hierarchical extension of the MSP method [46] that effectively ensures the activated sources are the same in all subjects (only the time course of activation is allowed to vary over subjects). We showed that this modification makes it possible to obtain significance levels close to those of nonfocal methods such as minimum norm, while preserving accurate spatial localization. Group inversion can yield much better results than individual inversions because it introduces an additional constraint for the ill-posed inverse problem, namely, that the responses in all subjects should be explained by the same set of sources. It is the inversion method of choice, when analyzing an entire study with subsequent topological inference on the contrast images.

Operationally, group inversion involves computing prior spatial covariances in source space that are common to all subjects. This rests on realigning the sensor-level data to pool sample covariances over subjects. In principle, this is straightforward because the linear mappings from each subject's sensors to a canonical set of cortical sources imply there is a unique linear mapping from one subject's montage to another; in other words, we can compute what we would have seen if one subject had been studied with the montage of another subject. However, in practice, the requisite realignment is a little more difficult because the "average" montage must converse information from all subjects. Imagine that two subjects have been studied with a single electrode and that the lead fields of these two electrodes are orthogonal. This means that realigning the sensor from one subject with the other would lose all the information from the subject being realigned. What we seek is an average sensor that captures the signals from both subjects in a balanced way. This can be achieved by iteratively solving a set of linear equations under the constraint that the mutual information between the average (realigned) sensor data and each subject's data is maximized. SPM8 uses a recursive (generalised) least squares scheme to do this.

Group inversion can be started by pressing the "Group inversion" button right after opening the 3D source reconstruction GUI. The user is asked to specify a list of M/EEG datasets to invert together. Then one is asked to coregister each of the files and specify all the inversion parameters in advance. It is also possible to specify contrasts in advance. Then the inversion will proceed by computing the inverse solution for all the files and will write out the output images. The results for each subject are saved in the header of the corresponding input file. It is possible to load this file into the 3D GUI, after inversion, and explore the results as described above.

4.6.3. Batching Source Reconstruction. One can also run imaging source reconstructions using the matlabbatch tool. It can be accessed by pressing the "Batch" button in the main SPM window and then going to "M/EEG source reconstruction" under "SPM" and "M/EEG". There are three separate tools here for building head models, computing the inverse solution and creating contrast images. This makes it possible to generate images for several different contrasts from the same inversion. All the three tools support multiple datasets as inputs. Group inversion is used automatically for multiple datasets.

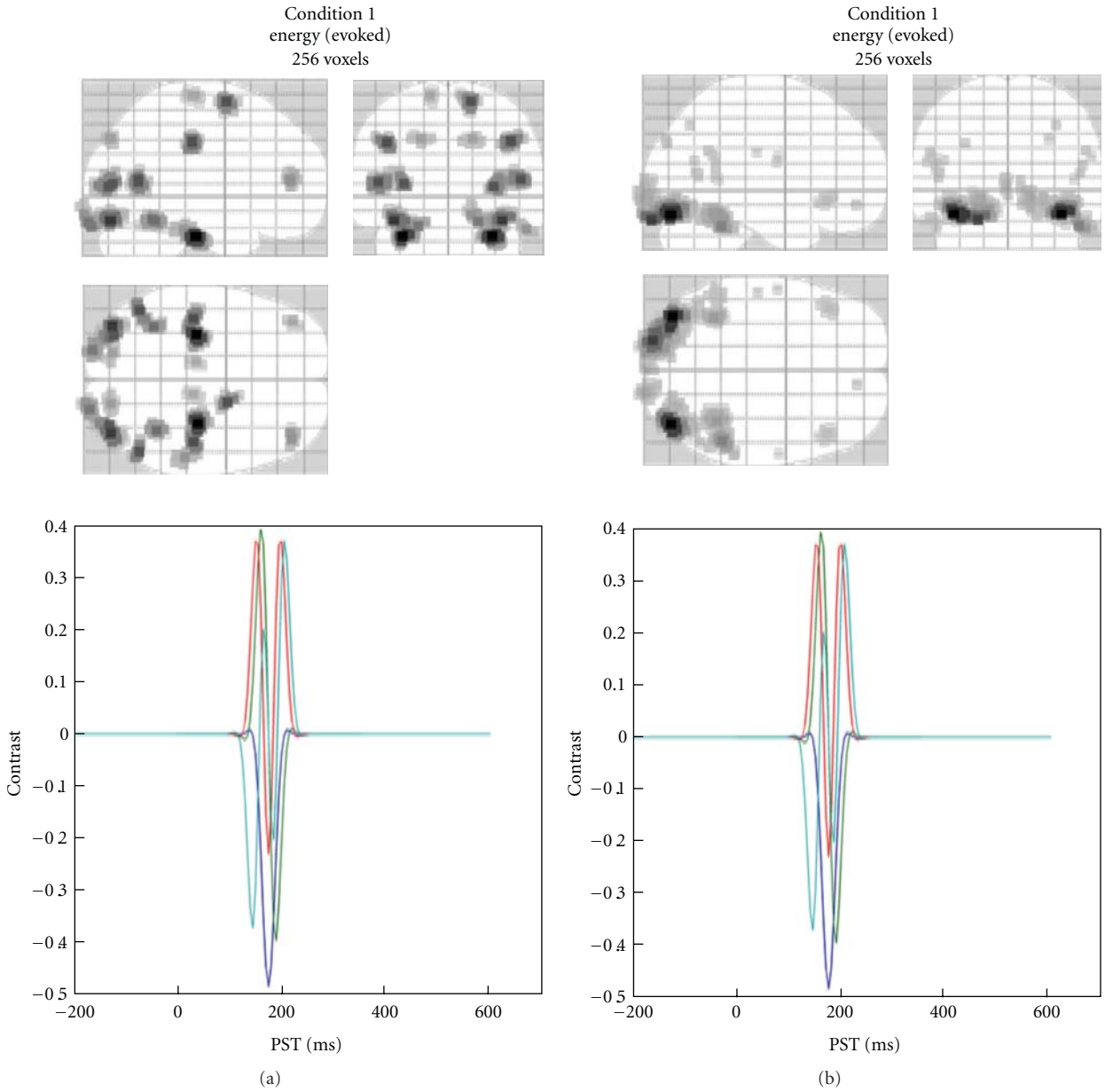


Figure 8: Display of the estimated distributed solution for evoked power in a specific frequency band. The same can be obtained for induced power and for each trial. The top panel shows the maximum intensity projections (MIP) while the bottom panel shows the applied time-frequency contrast. (a) EEG data from the multimodal face perception experiment available from the SPM website. (b) MEG data from the same experiment. Evoked power was computed between 150 and 200 ms, for frequencies between 1 and 20 Hz.

This completes our review of distributed source reconstruction. Before turning to the final section, we consider briefly the alternative sort of source space model, which is much simpler but, unlike the cortical mesh described in this section, leads to a nonlinear forward model parametrisation.

5. Localization of Equivalent Current Dipoles

This section describes source reconstruction based on Variational bayesian equivalent current dipoles (VB-ECDs) [19]. 3D imaging (or distributed) reconstruction methods consider all possible source locations simultaneously, allowing for large and distributed clusters of activity. This is to be contrasted with "equivalent current dipole" (ECD) approaches, which rely on two hypotheses.

(i) Only a few (say less than ∼5) sources are active simultaneously, and that

(ii) those sources are focal.

This leads to the ECD forward model, where the observed scalp potential is explained by a handful of discrete current sources; that is, dipoles, located inside the brain volume.

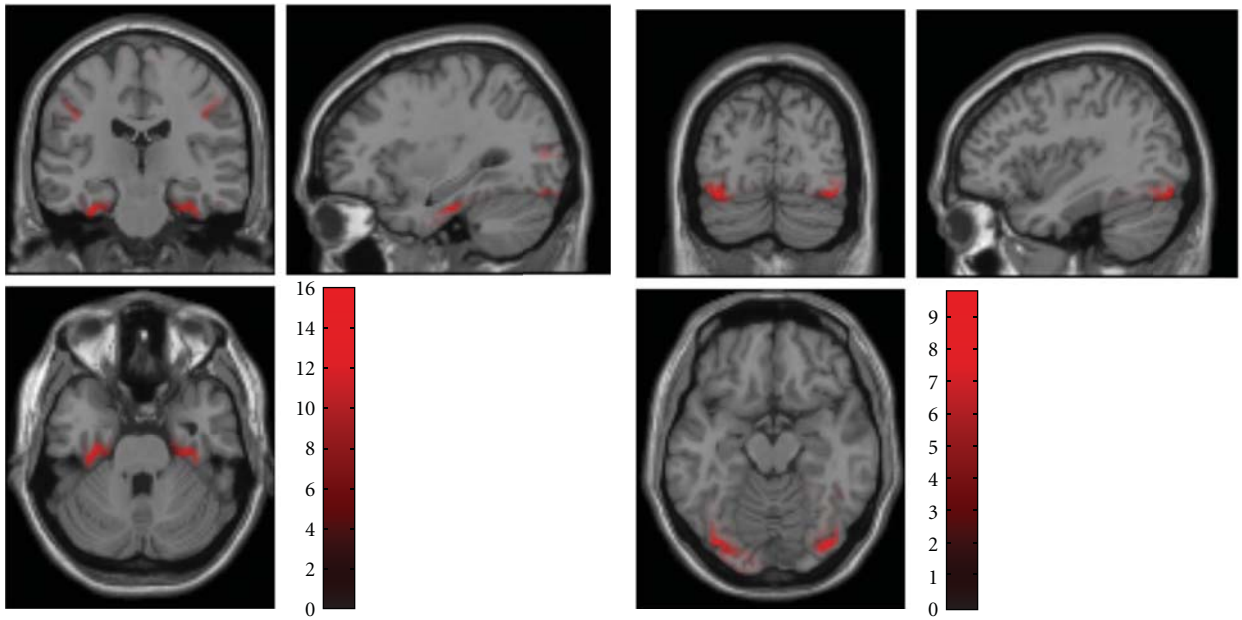


Figure 9: Axial, sagittal, and coronal views of the contrast image shown in Figure 8, projected into MNI voxel space and superimposed on the template structural MRI image. (a) EEG data from the multimodal face perception experiment available from the SPM website. (b) MEG data from the same experiment. The intensity was normalised to the mean over voxels to reduce intersubject variance.

In contrast to imaging reconstruction, the number of ECDs considered in the model, that is, the number of active locations, has to be defined *a priori*. This is a crucial step, as the number of sources considered defines the ECD model. This choice should be based on empirical knowledge about the brain activity observed or any other source of information (e.g., by looking at the scalp potential distribution). Note that the number of ECDs can be optimised *post hoc* using model comparison (see below). In general, each dipole is described by six parameters: three for its location, two for its orientation, and one for its amplitude. To keep the inverse problem overdetermined, the number of ECDs therefore must not exceed the number of channels divided by 6, and preferably should be well below this threshold. Once the number of ECDs is fixed, a nonlinear Variational Bayesian scheme is used to optimise the dipole parameters (six times the number of dipoles) given the observed potentials.

Classical ECD approaches use a simple best fitting optimisation using "least square error" criteria. This leads to relatively simple algorithms but presents a few drawbacks.

(i) Constraints on the dipoles are difficult to include in the framework.

(ii) Noise cannot be properly taken into account, as its variance should be estimated alongside the dipole parameters.

(iii) It is difficult to define confidence intervals on the estimated parameters, which could lead to overconfidence in the results.

(iv) Models with different numbers of ECDs cannot be compared, except through their goodness-of-fit, which can be misleading. As adding dipoles to a model will necessarily improve the overall goodness of fit, one could erroneously be tempted to use as many ECDs as possible and to perfectly fit the observed signal.

However, using Bayesian techniques, it is possible to circumvent all of the above limitations of classical approaches. Briefly, a probabilistic generative model is built, providing a likelihood model for the data. This assumes an independent and identically distributed normal distribution for the errors, but other distributions could be specified. The model is completed by priors on the various parameters, leading to a Bayesian forward model, which allows the inclusion of user-specified prior constraints.

An iterative Variational Bayesian scheme is then employed to estimate the posterior distribution of the parameters (in fact the same scheme used for distributed solutions). The confidence interval on the estimated parameters is therefore directly available through the posterior variance of the parameters. Crucially, in a Bayesian context, different models can be compared using their evidence. This model comparison is superior to classical goodness-of-fit measures, because it takes into account the complexity of the models (e.g., the number of dipoles) and, implicitly, uncertainty about the model parameters. VB-ECD can therefore provide an objective and accurate answer to the question: would this dataset be better modelled by two or three ECDs? We now describe the procedure for using the VB-ECD approach in SPM8.

The engine calculating the projection (lead field) of the dipolar sources to the scalp electrodes comes from FieldTrip

and is the same for the 3D imaging or DCM. The head model should thus be prepared the same way, as described in the previous section. For the same data set, differences between the VB-ECD and imaging reconstructions are, therefore, only due to the reconstruction chosen.

5.1. VB-ECD Reconstruction. After loading and preparing the head model, one should select the VB-ECD option after pressing the "Invert" button in the "3D source reconstruction" window. The user is then invited to fill in information about the ECD model and click on buttons in the following order.

(i) Indicate the time bin or window for the reconstruction. Note that the data will be averaged over the selected time window. VB-ECD will thus always be calculated for a single scalp topography.

(ii) Enter the trial type(s) to be reconstructed. Each trial type will be reconstructed separately.

(iii) Add a single (i.e., individual) dipole or a pair of symmetric dipoles to the model.

(iv) Select "Informative" or "Noninformative" location priors. "Non-informative" invokes flat priors over the brain volume. With "Informative", one can enter the *a priori* location of the source (for a symmetric pair of dipoles, only one set of dipole coordinates is required).

(v) At this point, it is possible to go back and add more dipole(s) to the model, or stop adding dipoles.

(vi) Specify the number of iterations. These are repetitions of the fitting procedure with different initial conditions. Since there are multiple local maxima in the objective function, multiple iterations are necessary to ensure good results, especially when non-informative location priors are chosen.

The routine then proceeds with the VB optimization scheme to estimate the model parameters. There is a graphical display of intermediate results. When the best solution is selected, the model evidence will be shown at the top of the SPM graphics window (see Figure 10(a)). This can be used to compare solutions with different priors or number of ECDs. Results of the inversion are saved to the data structure and displayed in the graphics window.

5.1.1. Result Display. The VB-ECD results can be displayed again by pressing the "dip" button, located under the "Invert" button that will be enabled after computing VB-ECD solution. In the upper part, the three main figures display orthogonal views of the brain with the dipole location and orientation superimposed (see Figure 10(b)). The location confidence interval is reported by the dotted ellipse around the dipole location (Figure 10(c)). The lower left table displays the current dipole location, orientation (Cartesian or polar coordinates), and amplitude in various formats. The lower right table allows for the selection of trial types and dipoles. Display of multiple trial types and multiple dipoles is also possible. The display will centre itself on the average location of the dipoles.

This completes our discussion of source reconstruction. The previous sections introduced distributed and ECD solutions based on forward models mapping from sources to sensors. These models are not constrained to produce physiologically plausible neuronal activity estimates and ignore the neuronal coupling among different dipolar sources in generating observed sensor signals. In the final section, we turn to dynamic causal modelling (DCM), which effectively puts a neuronal model underneath the electromagnetic forward models considered above. Usually, source reconstruction (imaging or ECD) is used to answer questions about the functional anatomy of evoked or induced responses, in terms of where sources have been engaged. This information is generally used to specify the location priors of sources in DCM.

6. Dynamic Causal Modelling for M/EEG

Dynamic causal modelling (DCM) is based on an idea initially developed for fMRI data [8]. Briefly, measured data are explained by a network model consisting of a few sources, which are dynamically coupled (cf. spatiotemporal dipole modelling introduced by Scherg and colleagues [47, 48]). This network model is inverted using the same Variational Bayesian scheme used for source reconstruction. Model inversion furnishes the model evidence (used to search model spaces or hypotheses) and the posterior density on model parameters (used to make inferences about connections between sources or their condition-specific modulation), under the model selected. David et al. [20] extended the DCM idea to modelling ERPs. At its heart DCM for ERP (DCM-ERP) is a source reconstruction technique, and for the spatial domain we use exactly the same forward model as the approaches in previous sections. However, what makes DCM unique is that it combines the spatial forward model with a neurobiologically informed temporal forward model, describing the connectivity among sources. This crucial ingredient not only makes the source reconstruction more robust, by implicitly constraining the spatial parameters, but also allows inference about connectivity architectures.

For M/EEG data, DCM can be a powerful technique for inferring (neuronal) parameters not observable with M/EEG directly. Specifically, one is not limited to questions about source strength, as estimated using a source reconstruction approach, but can test hypotheses about connections between sources in a network. As M/EEG data are highly resolved in time, as compared to fMRI, precise inferences about neurobiologically meaningful parameters (e.g., synaptic time constants) are possible. These relate more directly to the causes of the underlying neuronal dynamics. In the recent years, several variants of DCM for M/EEG have been developed. DCM for steady state responses (DCM-SSR) [22, 49, 50] uses the same neural models as DCM-ERP to generate predictions for power spectra and cross-spectra measured under steady state assumptions. There are also (phenomenological) DCMs that model specific data features, without an explicit neural model. DCM for

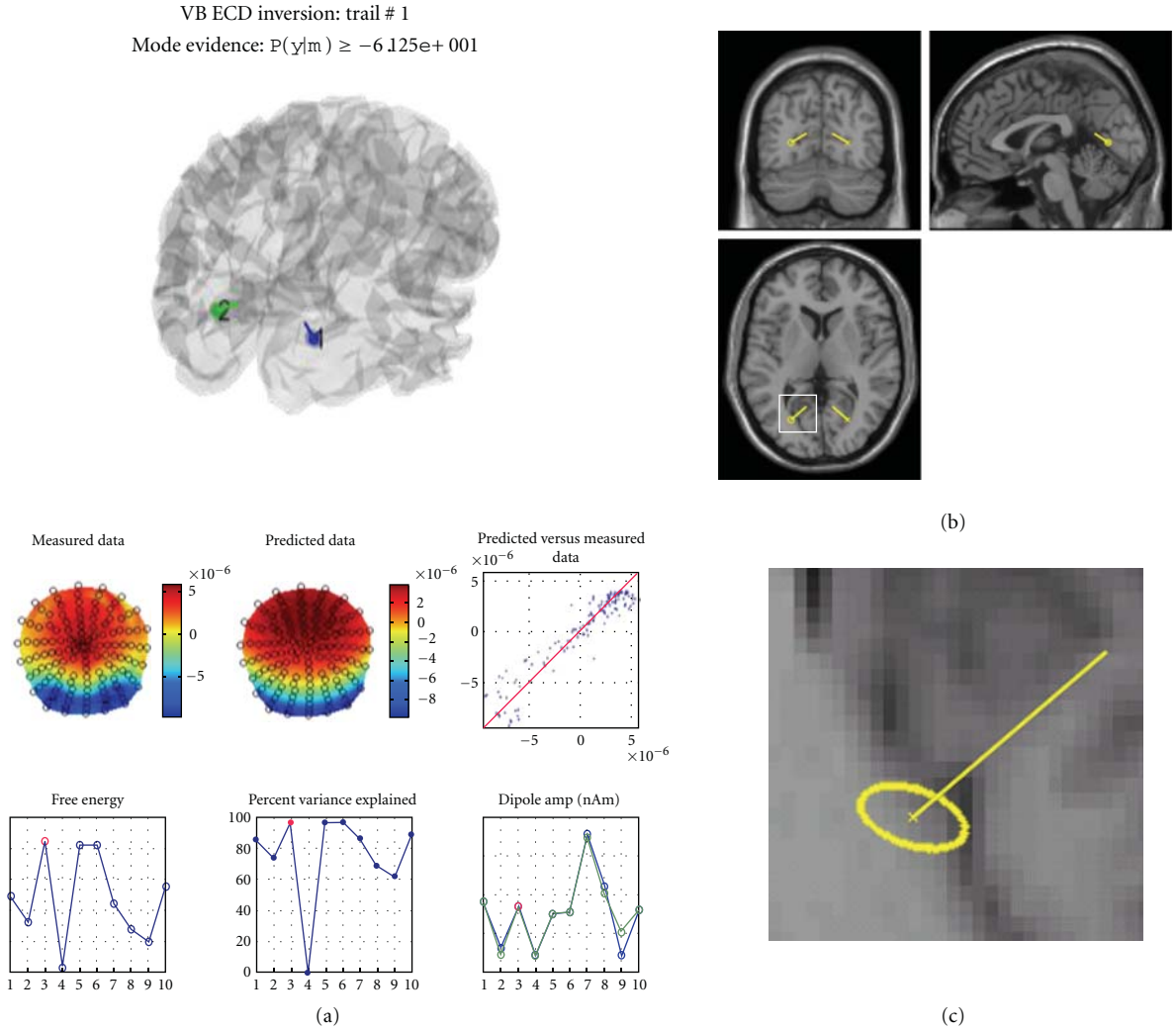


Figure 10: VB-ECD solution illustrated here on EEG data from the multimodal face perception experiment available from the SPM website. A symmetric dipole pair was fitted to the topography of the difference between faces and scrambled faces averaged between 170 and 180 ms. (a) The upper part shows the dipole location through the transparent cortical mesh; the middle part shows the correspondence between observed and predicted scalp data in two ways (topographies and dot plot); the bottom part shows the free energy, the explained variance, and the estimated dipole amplitude (from left to right) obtained from each of the ten repetitions of the procedure with different initial locations. Results correspond to the one with highest free energy (red point). (b) Orthogonal views of the brain with dipole locations obtained from the solution with highest free energy. (c) Enlarged fragment of the axial image shown by white square in (b). The ellipse shows the 95% confidence volume for dipole location.

induced responses (DCM-IR) [21] models event-related power dynamics (time-frequency features). DCM for phase coupling (DCM-PHA) [23] models event-related changes in phase relations between brain sources: DCM-PHA can be applied to one frequency band at a time. Presently, all M/EEG DCMs share the same interface, as many of the variables that need to be specified are the same for all four approaches. Therefore, we will focus on DCM for evoked responses and then point out where the differences to the other DCMs lie.

In this section, we only provide a procedural guide for the practical use of DCM for M/EEG. For the scientific background, the algorithms used or how one would typically use DCM in applications, we recommend the following. A general overview of M/EEG DCMs can be found in [51]. The two key technical contributions for DCM-ERP can be found in [20, 52]. Tests of interesting hypotheses about neuronal dynamics are described in [53, 54]. Other examples of applications demonstrating the kind of hypotheses testable with DCM can be found in [55, 56]. Another good source of background information is the recent SPM book [57] where parts 6 and 7 cover not only DCM for M/EEG but contextualise DCM with related research from our group. DCM-IR is covered in [21, 58], DCM-SSR in [22, 49, 50], and DCM-PHA in [23].

6.1. Overview. The goal of DCM is to explain measured data (such as evoked responses) as the output of an interacting network consisting of several areas, some of which

receive input (i.e., the stimulus). The differences between evoked responses, measured under different conditions, are modelled as a modulation of specified DCM parameters; for example, cortico-cortical connections [20]. The implicit model of evoked responses makes hypotheses about connectivity directly testable. For example, one can ask whether the difference between two evoked responses can be explained by top-down modulation of early areas [55]. Importantly, because model inversion is implemented using a Bayesian approach, one can compare Bayesian model evidences. These can be used to compare alternative, equally plausible, models and select the best [9–11].

DCM for evoked responses takes the spatial forward model into account. This makes DCM-ERP a spatiotemporal model of the full data set (over channels and peristimulus time). Alternatively, one can describe DCM as a spatiotemporal source reconstruction algorithm, which uses additional temporal constraints given by neural mass dynamics and long-range effective connectivity. This is achieved by parameterising the lead field, that is, the spatial projection of source activity to the sensors. In the current version, this can be done using two different approaches. The first assumes that the lead field of each source is modelled by a single equivalent current dipole (ECD) [52]. The second models each source as a "patch" of dipoles on the grey matter sheet [59]. This spatial model is complemented by a model of the temporal dynamics of each source. Importantly, these dynamics not only describe how the intrinsic source dynamics evolve over time, but also how a source reacts to external input, from subcortical areas (stimulus) or from other cortical sources.

The GUI allows one to enter all the information necessary for specifying a spatiotemporal DCM for a given data set. To fit multiple models, we recommend using a batch script. An example of such a script can be found in the man/example_scripts folder of the distribution.

6.2. Getting Started. The button for calling the DCM GUI is found in the menu window of SPM. When pressing the button, the GUI pops up (Figure 11). The GUI is partitioned into five parts, going from the top to the bottom. The first part deals with loading and saving existing DCMs, and selecting the type of model. The second part is about selecting data, the third is for specification of the spatial forward model, and the fourth is for specifying neuronal connections. The last row of buttons calls the DCM inversion and results display.

Data selection and model specification must be performed in a fixed order (data selection > spatial model > connectivity model). This order is necessary because there are dependencies among the three parts that would otherwise be hard to resolve. At any time, it is possible to switch back and forth from one part to the next. Also, within each part, information can be specified in any order.

6.3. Load, Save, and Select Model Type. The buttons at the top part of the GUI allow one to load an existing DCM or save the current one. In general, saving is possible during model specification at any time. There are two drop-down

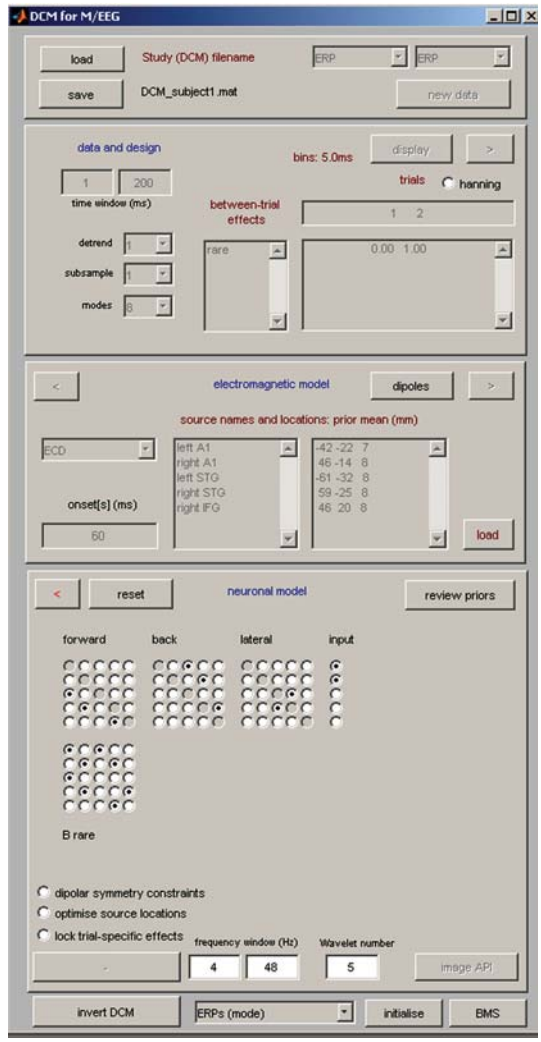


Figure 11: DCM for M/EEG graphical user interface. The configuration shown corresponds to the example DCM for mismatch negativity experiment available from the SPM website; see text for additional details.

boxes in this part of the DCM-GUI. The one on the left is for switching between different DCM variants. The default is ERP which is the DCM for evoked responses described here. Currently, there are three additional options: IND, SSR, and PHA as mentioned above. The menu on the right-hand side is for choosing the neuronal model. Currently, there are four model types. The first is ERP which is the standard model described in most of the application papers; for example [55]. The second is SEP which uses a variant of this model; however, priors on the neuronal dynamics make them faster to model early evoked responses [60]. The third is NMM which is a nonlinear (conductance-based) neural mass model [61]. The fourth is a mean field model MFM which is also nonlinear and is based on a second-order approximation to population dynamics [62]. Finally, data can be loaded using the "new data" button. The data can be either averaged or epoched EEG or MEG. For DCM-ERP, epoched data will be averaged to produce evoked potentials or fields.

6.3.1. Data and Design. This part deals with selecting and refining the data and modelling between trial effects. On the right-hand side of the DCM GUI, there are three text boxes that specify between-trial effects. These are the effects that are mediated by changing connection strengths. The top box should contain the indices of conditions to include in the model. For example, to model the second and third evoked response contained within a dataset, 2 and 3 should be specified. The indices correspond to the order, which can be specified by the user (see Appendix D.7). If the two evoked responses, for some reason, are in different files, these files need to be merged prior to DCM. Below the condition selection box, there is a box for specification of effects. This is used to define different options for modelling the experimental effects (i.e., the differences between conditions). For example, if trial 1 is the standard and trial 2 is the deviant response in an oddball paradigm, one can use the standard as the baseline and model the differences by modulations of the connections that are necessary to fit the deviant. To do this the effect should be specified as [0 1]. Alternatively, if the effect is specified as [−1 1], then the baseline will be the average of the two conditions and the same factor will be subtracted from the baseline connections to model the standard and added to the connections to model the deviant. The latter option is perhaps not optimal for an oddball paradigm but might be suitable for other paradigms where there is no clear "baseline condition". When modelling three or more evoked responses, one can model modulations of connection strength over multiple conditions as two effects relative to the first evoked response. However, one can also choose to couple the connection strength over conditions by imposing a relationship on how this connection changes. For example a single linear effect, over three trials or conditions, can be specified as [−1 0 1]. This can be useful when one wants to add constraints on how connections (or other DCM parameters) change. A compelling example of this can be found in [63]. For each experimental effect specified, one later selects the connections in the model that it affects (see below).

The leftmost textbox can be used to define names for the experimental effects (e.g., "oddball"). Further to the left there are several more controls whose purpose is refining the data before modelling. Under "time window (ms)" one has to enter the peristimulus times to model, for example, 1 to 200 ms. One can also choose whether to model the mean or drifts of the data at the sensor level. Under "detrend" one can select the number of discrete cosine transform terms to use to model low frequency drifts (selecting 1 means that just the mean will be removed). In the "subsample" option, one may choose to downsample the data before computing the ERP. This subsampling is not proper down sampling but decimation, so it is not advised to use it routinely. If necessary, it is preferable to down-sample the data during pre-processing. In DCM, we use a projection of the data to a subspace (mixtures of channels) to reduce the amount of data and suppress noise. This spatial projection is described in [54]. One can select the number of modes: the default is 8. One can also choose to window the data, in peristimulus time, with a Hanning window (radio button). This will reduce the influence of the beginning and end of the time series, which might be noisy or not captured by the idealised responses used to predict observed data.

Once satisfied with data selection, the projection and the detrending terms, the user can click on the ">" (forward) button to go to the next stage, electromagnetic model. From this, the red "<" button can be used, if necessary, to get back to the data and design specification.

6.3.2. Electromagnetic Model. Presently, there are three options for how to model evoked responses spatially. The first is to use a single equivalent current dipole (ECD) for each source, the second is to use a patch on the cortical surface (IMG), and the third (LFP) is to not use a spatial model at all (and assume that each channel samples a source with unknown gain). In all three cases, it is necessary to enter the source names (one name in one row). For ECD and IMG, the prior source locations (in mm in MNI coordinates) must be specified. Note that by default DCM uses uninformative priors on dipole orientations, but tight priors on locations. This is because M/EEG data contains limited information about location but supports precise estimates of orientation [64, 65]. This means each dipole stays in its designated area and retains its meaning in terms of anatomical designation. The prior location for each dipole can be found either by using available anatomical knowledge or by relying on source reconstructions (see the previous sections). Also note that the prior location does not need to be overly exact, because the spatial resolution of M/EEG, depending on location, can be on a scale of several centimetres [65, 66]. It is also possible to load the prior locations from a file ("load"). The locations can be visualized by pressing "dipoles".

An "onset" parameter determines when the stimulus, presented at 0 ms peristimulus time, is assumed to activate the cortical area to which it is connected. In DCM, we usually do not model small early responses, but start modelling at the first large deflection. Because the propagation of the stimulus impulse through the input nodes causes a delay, we find that the default value of 60 ms onset time is a good value for many evoked responses where the first large deflection (population response) is seen around 100 ms. However, this value is a prior; that is, the inversion routine can optimise it. The prior mean should be chosen according to the specific responses of interest. This is because the time until the first large deflection is dependent on the paradigm or the modality; for example, audition or vision, cortical or subcortical, and so forth. Changing the onset prior might have an effect on how the data are fitted. This is because the onset time has strongly nonlinear effects (a delay) on the predicted responses, which might induce local minima in the solution space, for different prior values. It is also possible to type several numbers in this box (identical or not). Each value invokes its own separate input, whose timing will be optimised separately. These inputs can be connected to the same or different sources of the model. This can be useful, for instance, for modelling a paradigm with combined auditory and visual stimulation.

To proceed to the next model specification stage, hit the ">" (forward) button and proceed to the "Neuronal model". If this is the first estimation and source reconstruction has

not been previously done with the same dataset, DCM will build a spatial forward model. The steps here are the same as described in the "3D source reconstruction" section above.

6.3.3. Neuronal Model. This section is the most critical for model specification. In DCM, one usually specifies a series of models (model space) for model comparison. For the model comparison to be valid, the models should only differ in their connectivity; that is, in the parameters specified in this part. DCM-ERP (as well as DCM-SSR) makes it possible to compare models with different sources. However, because of the data projection used (see above and [54]), this should be done by specifying all the sources in "Electromagnetic model" and then leaving the sources not participating in a particular neuronal model unconnected.

In this part of the GUI, there are five (or more) matrices, which are specified by radio button switches. The first three are the connection strength (A) matrices. For ERP and SEP models, there are three types of connections: forward, backward, and lateral. For NMM and MFM models the corresponding types are excitatory, inhibitory, and mixed (excitatory and inhibitory) connections, respectively. These matrices encode connections from source regions to target regions. For example, switching on the element **(2, 1)** (i.e., second row, first column) in the intrinsic forward connectivity matrix means that a forward connection from area 1 to 2 is enabled (can take nonzero values). This is basically an adjacency matrix for those familiar with graph theory. Some people find the meaning of each element slightly counterintuitive, because the column index corresponds to the source region, and the row index to the target region. This convention is motivated by the direct correspondence between the matrices in the GUI and connectivity matrices in DCM equations, and is probably intuitive to anyone familiar with matrix algebra.

The one or more inputs (onsets) specified previously can go to any source or to multiple sources. Receiving sources can be specified by selecting indices in the input (C) matrix. The number of columns in this matrix corresponds to the number of inputs specified previously. For a single input, C is a column vector. The bottom set of matrices (B-matrices) specify gain modulations of connection strengths as set in the A-matrices. These modulations are specified by the experimental effects described above. For example, for two evoked responses and experimental effect specified as [0 1], DCM explains the first response by using the A-matrix only. The 2nd response is modelled by modulating the connections specified by the B-matrix. The number of B-matrices is the same as the number of experimental effects. Since it is assumed that a connection between any two sources is of one type (forward, backward, or lateral), only one B-matrix per effect is necessary. The diagonal entries in the B-matrices allow modulation of the intrinsic (within source) connections. As described in [53], this makes it possible to model local changes in the excitability of a cortical area.

The "Review priors" button, also located in this part of the GUI, is for power users and opens another window making it possible to directly specify and refine priors on the neuronal model parameters and look at how they affect the model's dynamical responses.

Several additional radio buttons located below the connectivity matrices are for toggling options specific to DCM-ERP.

(i) The "Dipolar symmetry constraints" option is useful for modelling bilateral symmetric sources (e.g., auditory cortices).

(ii) "Optimise source locations" only works in combination with the "ECD" option and allows DCM more freedom with moving the dipoles as part of the optimisation process.

(iii) "Lock trial-specific effects" ensures that all the changes in connectivity are the same. This is useful when there is a specific hypothesis that some experimental factor increases (or decreases) all connection strengths.

6.3.4. Estimation. After model specification, the "Estimate" button can be pressed to invert the model. DCM then estimates the model parameters, which can take some time (typically from several minutes to an hour, depending on model complexity). One can follow the optimisation by observing the iterative model fit in a graphics window. In the MATLAB command window, the code will display the predicted and actual change in free energy (a bound approximation to the model's log-evidence that is being optimised) following each iteration. At convergence, DCM saves the results in a DCM file, by default named "DCM_ERP.mat". The name can be changed by pressing "save" at the top of the GUI and saving to a different name.

6.3.5. Results. After estimation is finished, the results can be assessed by choosing from the pull-down menu at the bottom (middle). With "ERPs (mode)" one can plot, for each mode, the data for both evoked responses and the model fit (see Figure 12(a)). When selecting "ERPs (sources)", the (posterior expectations of) dynamics in each source are plotted (see Figure 12(b)). The activities of the pyramidal cells (which are the reconstructed source activities) are plotted in solid lines, and the activities of the two other populations (inhibitory and excitatory interneurons) are plotted as dotted lines. The option "coupling (A)" will display a summary of the posterior distributions over the connections in the A-matrix. In the upper row (Figure 12(c)), the posterior means for all intrinsic connections are shown. As above, element **(i, j)** corresponds to a connection from area j to i. In the lower row (Figure 12(d)), for each connection, one can find the probability that its posterior mean is different from the prior mean, taking into account the posterior variance. With the option "coupling (B)" one can access the posterior means for the gain modulations of the connections. With "coupling (C)" one can see a summary of the posterior distribution for the strength of the input into the input-receiving source(s). On the left-hand side, DCM plots the posterior means for each area. On the right-hand side, the corresponding probabilities are provided. See Figure 12 for examples of

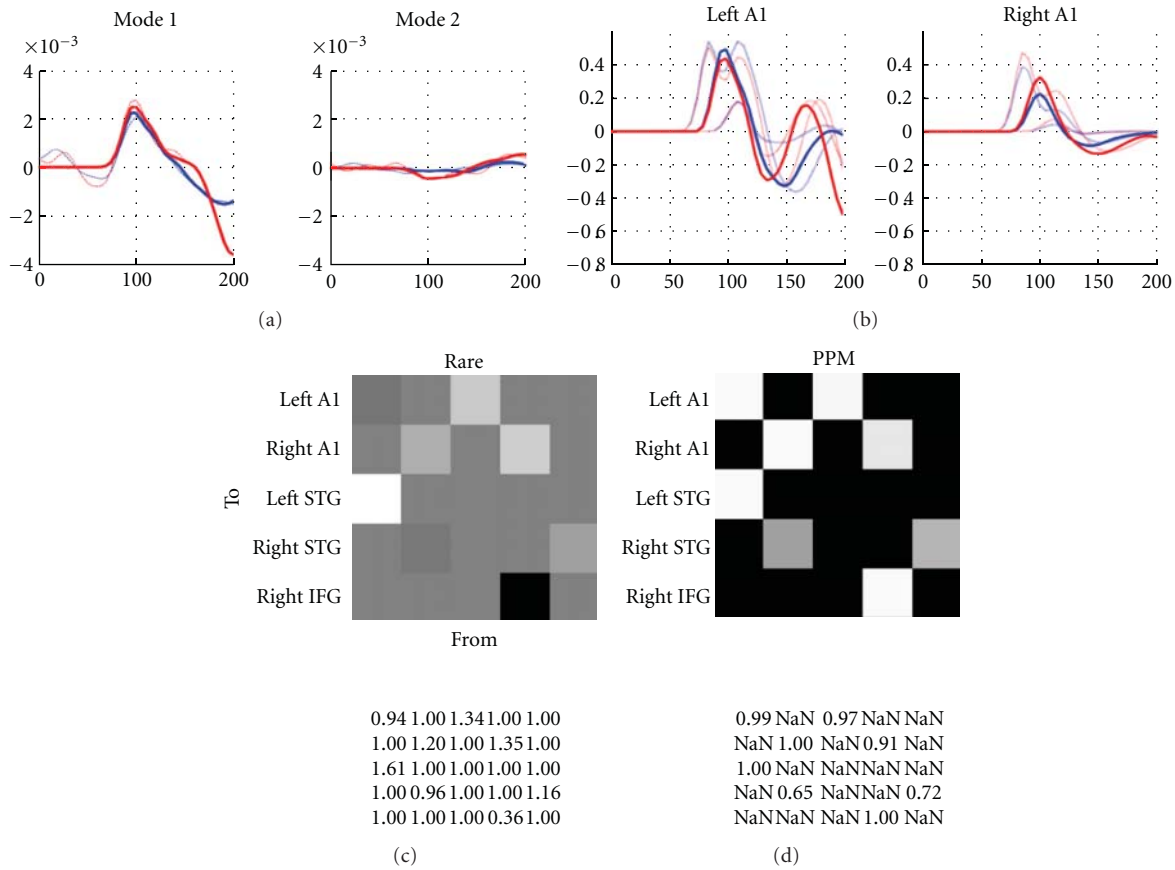


Figure 12: Examples from results display of DCM-ERP. The EEG data was taken from the mismatch negativity experiment available from the SPM website (a) "ERPs (mode)" display for the first two (out of eight) modes. The thin lines show the data and the thick lines the model prediction. Response to frequently presented tone is shown in blue and response to rare tone—in red. (b) "ERPs (sources)" display showing the predicted dynamics of two of the sources in the model (left and right auditory cortices). Each source consists of 3 neuronal populations. Response to frequently presented tone is shown in blue and response to rare tone—in red. (c) "Coupling (B)" display showing the posterior means for the gain modulations of the connections. (d) Posterior probability map from the same display showing for each modulated connection the probability that the effect of experimental conditions (frequent versus rare) on the connection was different from zero.

these summaries based on the posterior (conditional) density over the model's hidden states and parameters. There are several other quantities that can be examined (see the SPM manual for further details).

There are two additional buttons to the right of the "Results" menu. The "Initialise" button makes it possible to assign parameter values as initial starting points for the inversion. These values are taken from another already estimated DCM, which the user can select. The "BMS" button opens the SPM batch tool for model selection. This tool allows one to perform Bayesian model comparison and Bayesian model averaging [9–11], which is usually the prelude to examining the parameter estimates of the selected model or Bayesian model average.

This concludes our description of the specification and inversion of a DCM-ERP. We will now consider briefly the other DCMs and the interface features specific to these variants.

6.4. DCM for Steady State Responses. DCM-SSR is accessed by selecting "SSR" in the top-left drop-down menu. The second drop-down menu in the right of the top panel specifies (as with all DCMs) whether the analysis should be performed using a model that is linear in the states (ERP) or a conductance-based model (NMM) that is nonlinear in the states. The data selection and specification of between-trial effects are the same as for the case of ERPs, described above. The same is true for the electromagnetic model. DCM-SSR is commonly used with intracranial data, particularly from animal models and, therefore, the LFP option is especially relevant for this form of DCM. In the "Neuronal Model" the main difference from DCM-ERP is in the inputs. In DCM-SSR, inputs are not discrete events in time but endogenous noise sources. Therefore, the "onset" parameter is not relevant. The C matrix makes it possible to specify how the model sources are driven by noise. Usually, in this context, C is an identity matrix, prescribing endogenous fluctuations (noise) in all sources.

6.4.1. Cross-Spectral Densities. In addition to variables discussed above, with DCM-SSR, it is necessary to select the frequencies that will be modelled. These could be

part of a broad frequency range; for example, like the default 4–48 Hz, or one could enter a narrow band; for example, 8 to 12 Hz, which would model the alpha band. This specification is implemented via "frequency window (Hz)" boxes close to the bottom of the DCM window. After pressing the "invert DCM" button, the cross spectral densities are computed automatically (using the "spectral" toolbox in SPM [67]). The data features used for model inversion (i.e., those features generated or predicted by the model) include the auto-spectra and cross-spectra between channels (or modes). These data features are evaluated using a multivariate autoregressive model, which can accurately measure periodicities in the time-domain data. The resulting spectra are then presented as an upper-triangular, channel $\times$ channel matrix (or mode $\times$ mode), with autospectra on the main diagonal and cross-spectra in the off-diagonal terms.

6.4.2. Output and Results. The "Results" menu provides several estimates. By examining the "spectral data", one can see observed spectra in the matrix format described above. Selecting "Cross-spectral density" gives both observed and predicted responses. To examine the connectivity estimates one should select the "coupling (A)" results option, or for the modulatory parameters, the "coupling (B)" option. Also one can examine the input strength at each source by selecting the "coupling (C)" option, as in DCM-ERP. To examine the spectral input to these sources the "Input" option should be selected; this is a mixture of white and pink noise.

6.5. DCM for Induced Responses. DCM-IR models coupling within and between frequencies that are associated with linear and nonlinear mechanisms, respectively. DCM-IR is accessed by selecting "IND" in the top-left drop-down menu. Since this is a phenomenological DCM, the neural model menu is not relevant (i.e., it uses a simple bilinear approximation).

6.5.1. Data Features. The data features (time-frequency response) modelled are formed from single-trial, epoched data. DCM-IR models the entire spectra, including both the evoked (phase locked to the stimulus) and induced (non-phase-locked) components. The "modes" variable has a different meaning in DCM-IR than in DCM-ERP. Here, these are not spatial modes but frequency modes that are taken from singular value decomposition of the time-frequency data concatenated across sources. The more modes selected, the more details of the time-frequency response will be modelled. However, when there are too many modes, DCM inversion is slow, and the higher order modes usually capture noise rather than physiologically meaningful dynamics. Usually the first 3-4 modes capture most of the interesting features so the default value of 8 should be more than sufficient in most cases.

6.5.2. Electromagnetic Model. Unlike the DCMs above, DCM-IR does not model the data features in sensor space. The reason for this is that DCM-IR only models power and discards phase information. This makes it impossible to predict the sensor data given modelled source dynamics. Consequently, DCM-IR first inverts the electromagnetic model by using the pseudoinverse of the lead field matrix and then computes the source power, which is subsequently modelled. The IMG option is not relevant for this two-step procedure and, therefore, only the ECD and LFP options are available. When using the ECD option, the location parameters of the spatial model are not optimized. This means that DCM-IR will project the data into source space using the spatial locations specified by the user. We are currently considering more spatially specific methods for extracting source waveforms (e.g., beamforming). These methods will be implemented in the future.

6.5.3. Neuronal Model. In DCM for induced responses, the A-matrices encode the strength of linear and nonlinear coupling between sources. The leftmost matrix in the first row specifies the linear connections. These are the connections where frequency energy in one source affects the dynamics of the same frequency in another source. Note that all connections in the model should be at least linear, so if a connection is present, the corresponding button in this matrix should be on. Also, the buttons on the leading diagonal of the matrix are always on because each node in the model has a linear intrinsic connection with negative sign. This ensures that induced activity has a tendency to dissipate. To the right of the linear connectivity matrix there is a nonlinear connectivity matrix. The idea here is the same. Note that the corresponding linear connection should be enabled as well. When a connection is nonlinear, a frequency in the source node can affect all the frequencies in the target node. Intrinsic connections can be made nonlinear as well so as to explain nonlinearities among putative subpopulations within each source.

The use of the input matrix and the onset parameters are similar to DCM-ERP. The B-matrices are also used as described above. It does not matter whether a connection is linear or nonlinear when specifying modulation by experimental effects. Hence, there is only one modulation matrix per experimental effect. Self-connections can be modified by experimental effects, thus the diagonal entries of the B-matrices can be toggled.

6.5.4. Wavelet Transform. This button, located below the connectivity matrices allows one to transfer data into the time-frequency domain using a Morlet wavelet transform. One must also specify the frequency window that defines the desired frequency band and the number of cycles in the wavelet, which specifies the temporal-frequency resolution (see Appendix C.7). For the latter, we recommend values greater than 5 to obtain a stable estimation.

6.5.5. Results. The "Frequency modes" option will display the frequency modes, identified using singular value decomposition of spectral dynamics in source space (over time and sources). "Time modes" will display the observed time courses of the frequency modes (dashed lines) and the model predictions (solid lines). Here, one can also see whether

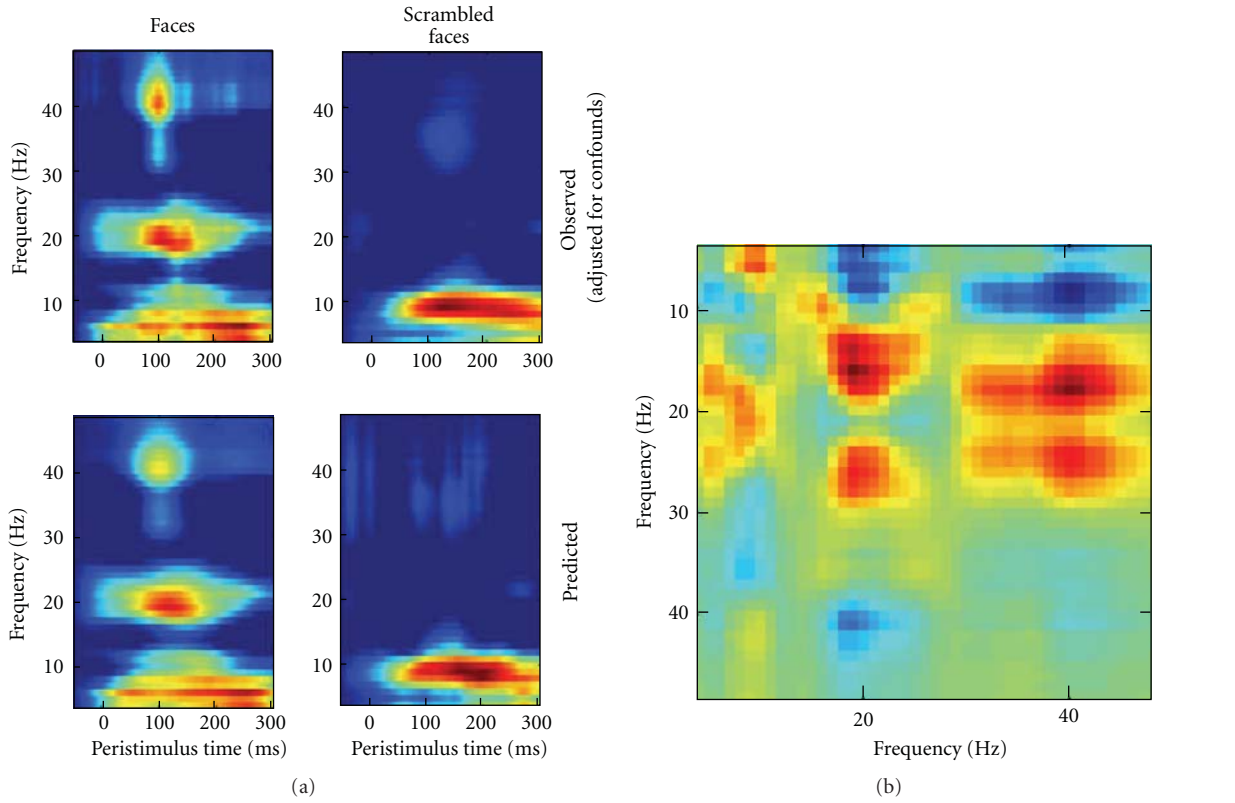


Figure 13: Examples from results display of DCM-IR. The MEG data was taken from the multimodal face perception experiment available from the SPM website. (a) Part of "Time-Frequency" display showing the observed and predicted time-frequency power for one of the model sources (right fusiform face area, rFFA). (b) Part of "Coupling (B-Hz)" display showing an example of a cross-frequency coupling matrix, in this case for connection between rFFA and right occipital source.

the activity picked up by the minor modes is noise, which is helpful for optimizing the number of modes. "Time-Frequency" will display the observed time-frequency power data for all prespecified sources (upper panel) and the fitted data (lower panel); see Figure 13(a) for an example. "Coupling (A-Hz)" will display the coupling matrices representing the coupling strength from source to target frequencies. These matrices are obtained by multiplying the between-mode coupling estimates with the frequency profiles of the modes [21]. "Coupling (B-Hz)" is similar to the above and reports modification of coupling by experimental effects (see Figure 13(b)). "Coupling (A-modes)" will display the coupling matrices between modes and the posterior probabilities that the coefficients are different from zero. This representation is useful for diagnostics when the inversion fails but the physiological interpretation is less straightforward. See the SPM manual for a more complete description and other options for reviewing the conditional estimates from DCM-IR.

A "Save as img" option allows one to save the cross-frequency coupling matrices as images. When analyzing a group of subjects one can use these images as summary statistics in SPM to find common features in coupling and coupling changes across subjects. The image names will include identifiers like "A12" or "B31" which relate to the source connection matrices; either the basic (A) or experimental effects (B).

6.6. DCM for Phase Coupling. DCM-PHA is based on a weakly coupled oscillator model of neuronal interactions. This approach is used to describe dynamic phase changes in a network of oscillators; see Figure 14 for examples of this kind of dynamics. The influence that the phase of one oscillator has on the rate of change of phase of another is characterised in terms of a phase interaction function (PIF) as described in [23]. SPM supports PIFs specified using arbitrary order Fourier series. However, to simplify the interface, one is restricted to simple sinusoidal PIFs when using the GUI.

6.6.1. Data Features. The data features (instantaneous phase) are computed from multiple trial, epoched data. Multiple trials are required so that the full state space of phase differences can be explored. This occurs because each trial is likely to contain different initial relative phase offsets. Information about different trial types is entered as it is with DCM-ERP. DCM for phase coupling is intended to model dynamic transitions toward synchronization states. As these transitions are short, it is advisable to model short time windows; the higher the frequency of the oscillations one is interested in, the shorter this time window should be.

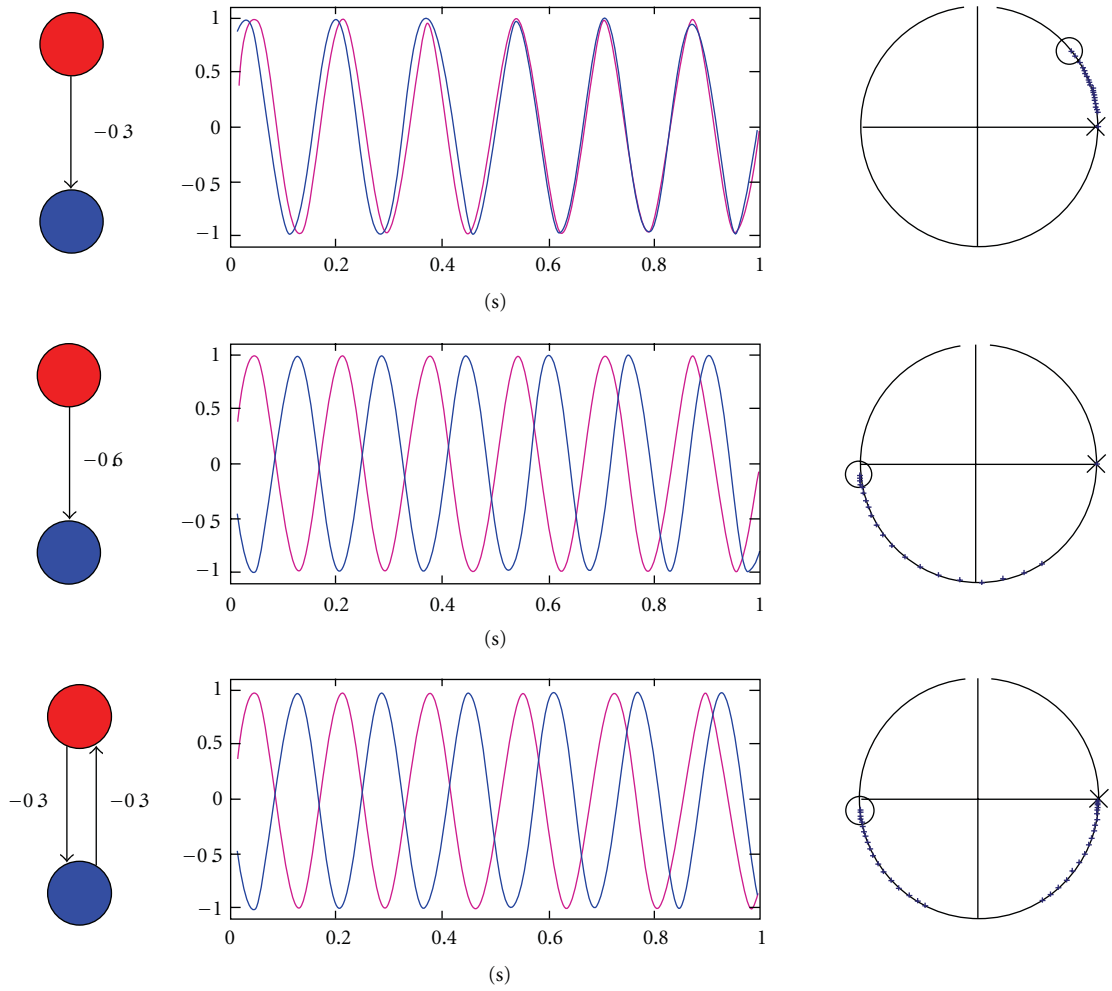


Figure 14: Examples for interactions between coupled oscillators that can be modelled with DCM-PHA. The left column shows the network structure used to generate the data in each row. The middle column shows the corresponding bivariate time series for two oscillators. The right column shows the corresponding phase diagrams on the unit circle with initial phases marked as a red cross for the first oscillator, and as a blue circle for the second. Subsequent phase evolutions are shown using dots. This figure was taken with permission from [23].

DCM for phase coupling will probably run into memory problems when using long time windows or large numbers of trials. Therefore, instead of a "modes" option (not relevant for DCM-PHA) there is a "subtrials" option which allows a subsampling of trials (i.e., using every second, every third, etc.).

6.6.2. Electromagnetic and Neuronal Model. As with DCM-IR, DCM-PHA projects the data onto source space, using the spatial locations, provided by the user, and does not optimize the spatial model. ECD or LFP options are available. There is only one A-matrix (called "endog" meaning endogenous). If using the GUI, the phase interaction functions are given by $a_{ij} \sin(\phi_i - \phi_j)$, where a_{ij} are the connection weights that appear in the A-matrix and ϕ_i and ϕ_j are the phases in sources i and j. DCM for phase coupling can also be run from a MATLAB script. This provides greater flexibility, in that the phase interaction functions can be approximated using arbitrary order Fourier series. There is an example in the man/example_scripts subdirectory of the SPM distribution.

6.6.3. Hilbert Transform and Results. Pressing the "Hilbert transform" button does two things. First, source data are bandpass filtered into the specified range. Second, a Hilbert transform is applied, from which time series of phase variables are obtained. The "Results" drop-down menu allows examining hidden states and parameter estimates; "Sin(Data)—Region I" plots the sine of the phase variable and the corresponding model fit for the i-th region. "Coupling (A)"and "Coupling (B)" will display the intrinsic and modulatory coupling matrices, respectively. The elements of A specify how quickly one source changes its phase to align with another. The corresponding entry in B shows how these values are changed by experimental manipulation.

7. Conclusion

In summary, we have reviewed the three main sorts of data analysis supported by the SPM software (and adopted in our scientific studies). These comprise analyses of sensor-level

data to identify significant treatment effects in evoked or induced responses. The idea here is to use standard SPM (topological inference) to find significant regions in time × frequency or time × channels search spaces, whose *P*-values are properly adjusted for multiple comparisons and profound correlations over the implicit search spaces. These analyses can be the endpoint or a device to identify peristimulus time or frequency windows for subsequent source reconstruction or DCM. The second analysis domain we have looked at is source reconstruction, using either distributed or ECD forward models. These analyses allow one to identify where in the brain various treatment effects are expressed. Usually, distributed solutions are used simply to create summary statistic (time-frequency contrast) images for topological inference. Again, the ensuing SPM may be the final analysis or used to specify the number and location of sources that form the basis of DCM analysis. DCM and its variants have been reviewed as enabling hypothesis testing (model comparison) about the functional architectures subtending observed responses. There is an increasing portfolio of models available and DCM for electromagnetic signals is likely to be a focus of development for many years to come.

Appendices

These appendices deal with the computer science and pragmatic aspects of handling data in SPM8. We include here substantial Appendices B and C on data formats and preprocessing, which are important parts of the analysis stream and share many protocols (and code) with other software platforms.

A. SPM8 and FieldTrip

SPM developers have a formal collaboration with the developers of the FieldTrip toolbox (see Oostenveld et al., this issue) on many analysis themes. For example, SPM and FieldTrip share routines for converting data to MATLAB and some of the basic pre-processing and forward modelling for M/EEG source reconstruction. The SPM8 distribution contains a version of FieldTrip so that FieldTrip and SPM functions can be combined easily in custom scripts. For using FieldTrip functions within SPM, it is not necessary to install the full FieldTrip version. SPM and FieldTrip complement each other; SPM is geared toward very specific analysis tools whereas FieldTrip is a more generic repository of different methods that can be assembled in flexible ways to perform a variety of analyses. This flexibility of FieldTrip, however, comes at the expense of accessibility to a nonexpert user. FieldTrip does not have a GUI and its functions are accessed through custom scripts. By combining SPM8 and FieldTrip, the flexibility of FieldTrip can be complemented by SPM's GUI tools and batching system. Within this framework, power users can easily and rapidly develop specialized analysis tools with GUIs that can then also be used by nonproficient MATLAB users.

B. Data Formats and Handling

In this appendix, our goal is to provide a comprehensive description of how the software can be used to preprocess M/EEG data, up to the point where one would use source reconstruction, DCM or statistical analysis of M/EEG channel data, as described in the main text.

Pre-processing functions can be called via GUI, the batch system or scripts. For scripting and command-line specification, we adhere to the principle of providing only one input argument to each function. This input argument is usually a MATLAB structure (struct) that contains all input arguments as fields. This approach has the advantage that the input does not need to follow a specific input argument order. If an obligatory input argument is missing, the function will invoke the GUI and ask the user for the missing argument. When using the GUI, a function is called without any input argument; that is, SPM will ask for the requisite input arguments.

B.1. SPM M/EEG Data Format. The first step of any analysis is the conversion of data from its native machine-dependent format to a MATLAB-based, common SPM format. A dataset in this format consists of two files: DAT-file which is a binary file containing just the data and MAT-file containing a structure with all the additional information related to the dataset.

There are two important programming devices that SPM uses when working with its M/EEG dataset: using object-oriented programming to access the header information and using memory mapping to access the data. The header data is stored in the MAT-file as a struct but when loaded by an SPM function into MATLAB memory, it is converted to an @meeg object. The internal structure of the header is thereby hidden from the user and is only accessed via special functions called "methods". This has several advantages. First, using an object enforces internal consistency checks when the object is created or modified. If, for any reason, the header data becomes inconsistent, SPM will report this as soon the user tries to load these data. SPM will also report where the check flagged an inconsistency. If there is enough information available to fix the inconsistency, it is fixed on the fly. Second, using methods simplifies internal book-keeping, which makes it much easier to program functions operating on the M/EEG object. There is no need to check in higher level functions whether a particular header field is present and valid, as the object takes care of that. Third, using objects lends more flexibility to SPM developers because some details of the format can be changed without the need to update higher level SPM functions. It is therefore recommended that users writing scripts familiarise themselves briefly with the object functionality and work with the object rather than with the header struct directly.

The second important programming device is the use of memory mapping to access the data stored in the DAT-file. Memory mapping treats the data stored on the hard drive as if they were in memory, without the need to actually load all of the data at the same time. The technical details are handled by the operating system, but the important

thing is that SPM can work with large files in a memory-efficient fashion. This means that SPM can handle large data sets without running into memory errors. The price can be a slight slowing in performance for some applications. Users writing custom code that needs to access the data are advised to do so in chunks (i.e., write a loop loading the data trial-by-trial or channel-by-channel in a continuous file) to take full advantage of memory mapping. Due to the way memory mapping is presently implemented, SPM only supports fixed-length trials (unlike for instance FieldTrip).

B.2. Conversion of Data. There are two ways to convert data from another format and create an SPM M/EEG dataset. The first is geared towards EEG and MEG datasets stored in their native formats or in formats of other analysis packages (e.g., EEGLAB). This conversion facility is based on "fileio" module (see http://fieldtrip.fcdonders.nl/development/fileio), which is shared between SPM8, FieldTrip and EEGLAB toolboxes and jointly developed by the users of these toolboxes. At the moment, most common EEG and MEG data formats are supported. For some formats, it might be necessary to install additional MATLAB toolboxes (there is an error message if these toolboxes are missing). If the data format is not recognized by "fileio", it will attempt to read the file using the Biosig toolbox (http://biosig.sourceforge.net/), if available. This can work for some EEG systems, particularly clinical ones. One of the consequences of using Biosig as a fallback option is that error messages from Biosig or mentioning Biosig can appear if the data format is not supported. If necessary the "fileio" toolbox can be extended easily to support additional formats.

The majority of formats supported by "fileio" are also automatically recognized. Therefore, format-specific details are hidden in the library and SPM can deal with conversion in a generic fashion. After selecting a file for conversion, the user can let SPM and fileio convert it automatically and read all the data in the file. There is also a facility to configure parameters for conversion. This is useful for instance when only a part of a large data file needs to be read or only a subset of channels are of interest.

In many cases, it is necessary to work with data that is not stored in any standard format; for instance, data stored as a MATLAB variable or an ASCII file. There are two options for importing these data, depending on whether the user can write a MATLAB script or wants to use the GUI. For users who would like to rely on the GUI, we suggest they assemble their dataset in EEGLAB (Delorme et al., this issue) and then save it and convert to SPM as with any other format. The developers of EEGLAB have invested a lot of effort into making it possible to build a dataset from scratch, without using the command line or scripts.

For those who prefer to work with MATLAB scripts, the most straightforward way to convert custom data is to create a simple FieldTrip raw data structure and then use SPM's spm_eeg_ft2spm.m function to convert this structure to SPM. Missing information can then be supplemented using @meeg methods and SPM functions.

FieldTrip raw struct must contain the following fields.

(i) .fsample—sampling rate (Hz).

(ii) .trial—cell array of trials containing matrices with identical dimensions (channels × time).

(iii) .time—cell array of time vectors (in sec)—one cell per trial, containing a time vector the same length as the second dimension of the data. For SPM, the time vectors must be identical.

(iv) .label—cell array of strings; a list of channel labels. Same length as the first dimension of the data.

B.3. Data Reviewing and Augmentation. Following conversion and throughout pre-processing, the data can be reviewed using the SPM8 reviewing tool. This tool allows the user to visualise data and source reconstructions and review or modify much of the header information on the dataset (e.g., channel labels and types). When reviewing continuous data, it is possible to add events manually (e.g., mark epileptic spikes). It is also possible to mark trials and channels as bad.

fMRI artefact rejection and sleep scoring toolbox (FASST, http://www.montefiore.ulg.ac.be/~phillips/FASST.html) can be used as an alternative to the SPM8 reviewing tool to speed up reviewing of long continuous multichannel datasets (see Leclercq et al., in this issue).

SPM tries to do its best to extract information automatically from the various data formats. In some cases it can also supplement the converted dataset with information not directly present in the raw data. For instance, SPM can recognise common EEG setups (extended 10–20, Biosemi, EGI) based on channel labels and assigns "EEG" channel type and default electrode locations for these cases. However, there are data types that are either not yet supported in this way or do not contain sufficient information for SPM to make automatic choices. Furthermore, channel labels do not always correctly describe the actual electrode locations in an experiment. In these cases, further information needs to be supplied by the user. The SPM "Prepare" tool, accessible from the reviewing tool, makes it possible to augment and modify the data in several important ways. First, it is possible to review and modify channel types. Channel types are important in SPM because they determine how SPM interprets the information in the channels and the dataset as a whole. For instance, if EEG or MEG channels are present in the dataset, SPM will expect sensor locations to be defined when this dataset is loaded for 3D source reconstruction or DCM analysis. Second, using the "Prepare" interface one can load individual EEG electrode locations and individual head shapes used for MEG coregistration. Sensor positions and fiducials for MEG are extracted from the raw data automatically and are already present after conversion. However, sometimes head shape measurement outside the scanner is used to relate the locations of MEG head position indicator coils to anatomical fiducials that can be marked on an MRI. SPM handles this by allowing the user to load the head shape via "Prepare". Third, an important function of the "Prepare" interface is to create

2D channel layouts used for creating topographical scalp plots and converting M/EEG data to images for statistical analysis. These layouts can be created automatically by projecting 3D sensor locations to 2D, but they can also be built manually by the user or loaded from a fixed template. When possible, SPM creates 2D layouts automatically at conversion. Without meaningful 2D coordinates, all SPM functions will work correctly but the channels will be laid out in a topographically meaningless rectangular pattern.

C. Data Processing

In this appendix, we will describe the high-level SPM functions which are used for pre-processing of converted M/EEG data (e.g., epoching, filtering, averaging). The majority of these functions have functionality similar to that implemented in commercial and other open source M/EEG analysis packages. Here, our aim was to make SPM self-contained so that the users will not usually need other software to prepare their data for SPM analyses proper. Where possible, SPM shares low-level code with FieldTrip. The general syntax is the same for all functions. If called from the command line, and if no input arguments are specified, the function will behave exactly as if called from the GUI. However, on the command line (or from a script) it is possible to supply input arguments. When all required input arguments are specified, the function will run without user interaction. In this way, one can write a noninteractive script. Input arguments are provided in a struct called S, whose fields contain the arguments. One of the fields (usually S.D) specifies the input dataset and can be either an @meeg object or file name. The majority of SPM pre-processing functions create a new output dataset, rather than modify the input dataset. This is somewhat wasteful in terms of disk space, but makes it possible to backtrack and try different options for every analysis step without the need to recompute. The filenames of the output MAT- and DAT-files are generated by prepending a single letter to the input file name. In the example of epoching, this would be an "e". The idea is that after calling a sequence of functions on a file, the file name encodes the pre-processing steps that were called to produce this file. Note that another way of calling SPM functions and specifying all input parameters is to use the batch interface.

The order of the pre-processing steps is not fixed and depends on different considerations. For instance, it is important to set the channel types correctly prior to filtering because channels with undefined ("Other") type are not filtered. The logic here is to preserve information in channels for which filtering is not relevant (e.g., trigger channels).

C.1. Epoching. Epoching cuts out little chunks of continuous data and saves them as "single trials". In M/EEG research, this is a standard data selection procedure to remove long gaps between trials. An epoch starts at some user-specified pre-stimulus time and ends at some poststimulus time; for example, from −100 to 400 milliseconds in peristimulus time. By default, the epoched data will also be baseline corrected; that is, the mean of the pre-stimulus time is subtracted from the whole trial.

The epoching function can implement two ways of specifying trials to epoch. The first is to specify trials using labelled events stored in the file. The user should define the prestimulus and poststimulus interval, and specify the events (triggers) around which the epochs will be "cut". SPM identifies events by their "event type" and "event value". These are strings or numbers, which the software used by the EEG or MEG vendor produces when generating the measurement file. These can sometimes look strange but are usually the same for a given system, so it is only necessary for the user to find out once which triggers are relevant for the experiment. It is also necessary to define a "condition label" for each trial type, which can be any string. This is the label that SPM will use to indicate the trial type of a trial at later processing stages. It is possible to use several types of triggers for defining trials with the same label.

For most users, the epoching described above is the most convenient but there are situations when a single trigger is not sufficient to specify the trial type. This happens, for instance, when the epoching is done around the stimulus but the trial type also takes into account the response that comes later. This situation can be dealt with by epoching around the stimulus trigger and defining a single trial type but relabeling the trials at a later stage. Relabeling can be done using @meeg object methods or the reviewing tool. In even more complicated cases, or when there is no event information available in the raw data, it is up to the user to write a custom script for trial definition and specify explicitly where each trial is located in the measured time series.

This script should generate an $N\times 2$ so-called "trl" matrix, where each row contains the start and end of a trial (in samples). Optionally, there can be a third column containing the offset of the trigger, with respect to the trial. An offset of zero (the default) means that the first sample of the trial corresponds to the trigger. A positive offset indicates that the first sample is later than the trigger and a negative offset indicates that the trial begins before the trigger. In SPM, the offset should be the same for all trials. The need to specify a whole column is for interoperability with FieldTrip, where trials can have different time axes. In addition, condition labels need to be specified either as a single string or a cell array of strings with a label per trial.

C.2. Filtering. Continuous or epoched data can be filtered, over time, with a lowpass, highpass, bandstop, or bandpass-filter.

C.3. Downsampling. The data can be downsampled to any sampling rate.

C.4. Baseline Correction. This function subtracts the baseline from channel data. The baseline period needs to be specified in ms; for example, [−100 0].

C.5. Artefact Detection and Rejection. Some trials not only contain (neuronal) signals of interest, but also large signals

from other sources, like eye movements or muscular activity. These signal components are referred to as artefacts. There are many kinds of artefacts and methods for detecting them. The artefact detection function in SPM is, therefore, extendable and can automatically detect and use plugin functions that implement particular detection schemes. Simple algorithms presently implemented include thresholding of the data, thresholding of the difference between adjacent samples (to detect jumps), thresholding peak-to-peak amplitude, and detection of flat segments. Channels containing artefacts in a large proportion of trials are automatically marked as bad. Note that the function only indicates which trials are artefactual or clean and subsequent processing steps (e.g., averaging) will take this information into account. However, no data are actually removed from the DAT-file.

C.6. Montage Application. The montage function in SPM basically multiplies the channel data by a matrix. This can be used to implement various processing steps. One common example is EEG re-referencing: Presently, for source analysis and DCM, EEG data should be re-referenced to the channel average, to meet the assumptions of the forward model used. For sensor level analysis, it is sometimes useful to use a reference that emphasizes the effects of interest. Other common uses are deriving bipolar channels, when data are recorded with common reference (e.g., EMG, EOG), and adding or removing channels.

The montage function can also be used for denoising and artefact reduction. Removing artefactual components, using projections derived from independent component analysis (ICA), signal space projection (SSP), or the application of synthetic gradients implemented in the CTF MEG system, are all particular cases of montage. An important point here is that the application of the montage function often changes the dimensionality of the data or the relationship between sensors and channels. This should be taken into account when computing the forward model for source analysis. Presently, this is only done for MEG, where the montage function automatically updates the sensor representation to ensure consistency with the data. Similar mechanisms for EEG will be implemented in the future but for now, removing too many spatial components from the data should be avoided, especially with a small number of channels.

C.7. Time-Frequency Analysis. Time-frequency analysis makes it possible to study oscillatory neural activity that appears consistently at particular times, relative to the event of interest; even if this activity is not phase locked to the event and therefore averages out in conventional analyses of evoked responses. This usually involves transforming the data to the frequency domain in short (possibly overlapping) time windows. Combining the spectra for all time windows yields 2D images that can be analysed using SPM statistical machinery (topological inference). All methods of time-frequency analysis are inherently limited by the uncertainty principle; stating that the resolution in time is inversely related to frequency resolution. Therefore, in order to estimate the frequency of particular response more precisely, one should examine longer data segments and sacrifice time resolution. The opposite holds when trying to localize a signal precisely in time. Different methods of time-frequency analysis handle this resolution trade-off slightly differently and are, therefore, optimal for certain kinds of signals and suboptimal for others. For example, in the 300 ms following an event (which is more or less the reaction time in simple tasks) there are only three cycles at alpha frequency (10 Hz) and 6 cycles at beta frequency (20 Hz). Thus, analysis of event-related activity at these frequencies requires a method that can reliably estimate power and phase of an oscillation, based on a small number of cycles. Wavelet analysis using Morlet wavelets compares the signal of interest with short segments of an oscillation multiplied by a Gaussian. This results in very high time resolution and makes wavelet analysis the method of choice for low frequencies. However, when we look at higher frequencies, especially in high gamma range (above 50 Hz), the number of cycles is no longer an issue. In 300 ms, there are 24 cycles of an 80 Hz oscillation. The main problem is now the fact that high gamma activity may be highly variable in timing and precise frequency. Therefore, the high time and frequency resolution of wavelets is rather a disadvantage in this case and results in patchy time-frequency plots with characteristic "fingers". Multitaper spectral analysis is the optimal way to "smooth" the spectral estimates in both time and frequency, and thereby gain more power for detecting high gamma activity. This method is based on premultiplying the data with a series of tapers optimised for producing uncorrelated estimates of the spectrum in the given frequency band. This makes it possible to sacrifice some of the frequency resolution in a well-controlled manner to gain a higher signal-to-noise ratio, by effectively multiplying the number of trials by the number of tapers used.

The above example shows why no single method of estimating power over time and frequency is optimal for all applications and even within a method parameters need to be optimized to get the best spectral estimate for a particular dataset. SPM makes it possible to perform such optimization by offering a flexible and extendible interface based on the matlabbatch GUI. Different spectral estimation methods are implemented as automatically detectable plug-ins. Algorithms presently implemented include continuous Morlet wavelet transform, Hilbert transform, and tapered fast Fourier transform (FFT), including multitaper spectral estimation. The result is written to one or two result files, one containing the instantaneous power and the other (optional) containing phase estimates (phase estimation is not possible for some algorithms). One can select the channels and frequencies for which power and phase should be estimated. In the future, additional algorithms will be added and we are planning to share much of the low-level code with FieldTrip.

C.8. Averaging. Averaging of single trial data is the crucial step to obtain the evoked response. When averaging single trial data, single trials are averaged within trial type. Power and phase data of single trials can also be averaged by using the SPM averaging function.

A special feature implemented in SPM is robust averaging. This is a rather simple case of robust general linear modelling [68]. The idea is that for each channel and time bin (or time-frequency pixel), the distribution of values over trials is used to downweight outliers, when computing the average. This suppresses artefacts restricted to narrow time and frequency ranges, without rejecting whole trials. Moreover, a clean average can be computed with no clean trials; given that the artefacts do not consistently overlap and only corrupt (different) parts of trials.

The robust averaging algorithm estimates weights, lying between 0 and 1, that indicate how artefactual a particular sample in a trial is. Later, when averaging to produce evoked responses, each sample is weighted by this number. If the weight of a sample is close to zero, it has little influence on the average. The sensitivity of the algorithm to outliers is controlled by the "offset for the weighting function" parameter. This value (3 by default) defines the weighting function used for averaging the data. This will preserve roughly 95% of data points drawn randomly from a Gaussian distribution. Another choice the user should make is whether to compute the weights by condition (as opposed to all the trials). If one condition has fewer trials than the others, it is generally safer to estimate the weights separately for each condition; otherwise, evoked responses in the rarer condition will be downweighted and become more similar to the more common condition(s). The weights can be saved as a separate dataset, which is useful for finding out what parts of the data were down-weighted and adjusting the parameters if necessary. Robust averaging can be applied to either time or time-frequency data. In the case of time data, if a low-pass filter was applied before averaging, one should apply it again (after averaging) because the differential weighting of adjacent points may introduce high frequencies.

D. Additional Utility Functions

D.1. Grand Mean. The grand mean is usually understood as the average of evoked responses over subjects. The grand mean function in SPM is typically used to compute this, but can also be used to average over multiple EEG files; for example, multiple sessions of a single subject. There is an option to weight by the number of trials in each file (suitable for averaging across sessions within a subject) or do unweighted averaging (suitable for averaging across subjects).

D.2. Merge. Merging several M/EEG files can be useful for concatenating multiple sessions of a single subject. Another use is to merge files and then use the reviewing tool to display data from different files in the same graph. The function has a mechanism for recoding condition labels so that, for instance, all trials coming from the same original file will have an identifier added to their condition label in the merged file.

D.3. Multimodal Fusion. SPM supports datasets containing simultaneously recorded MEG and EEG. For imaging source reconstruction, it is possible to use both modalities to inform the source solution. Usually, combined MEG/EEG data is contained within the same raw dataset and can be preprocessed together from the beginning. If this is not the case, the fusion functionality makes it possible to combine two datasets with different channels into a single dataset; given that the sets of channels do not overlap and the datasets are identical in the other dimensions (i.e., have the same sampling rate and time axis, the same number of trials, and the same condition labels in the same order). This function can be used to create a multimodal dataset from separately recorded MEG and EEG, for experiments with highly reproducible ERP/ERFs.

D.4. Rescaling and Baseline Correction of Time-Frequency. Usually, raw event-related power is not the most informative thing to look at (although contrasts of raw power between conditions can be informative). To see the event-related effects better, the power can be either transformed or baseline-corrected separately for each frequency. There are several different ways to do this. "LogR" (log-ratio) method first computes the log of power and then baseline-corrects and scales the result to produce values in dB. "Diff" just does simple baseline subtraction. "Rel" expresses the power in percentage of the baseline units. Finally, the "Log" and "Sqrt" options just compute the respective functions without baseline correction. If necessary, the baseline period needs to be specified. The baseline can also be taken from a different dataset. This allows a baseline condition rather than baseline period.

D.5. Contrast of Trials. As an extension to the averaging functionality, SPM can also be used to compute linear combinations of single trials or evoked responses. A simple example is computing the difference between two evoked responses with a contrast weight vector [−1 1]. Another example is pooling across conditions when several trial types are to be treated as one. In this case there is an option to weight the contrast coefficients by the number of replications in each trial type. In principle, any contrast that can be formulated in the GLM framework can be applied to the data. This can be useful for a "localisation of contrast" approach to source reconstruction.

D.6. Copy. This function makes it possible to make a copy of a dataset. Note that one cannot just copy and rename the files in the usual way because the name of the data file is stored in the header file and this should be updated. The user will be asked to specify the new dataset name.

D.7. Sort Conditions. In many cases in SPM, the order of the conditions in the file is important (e.g., in 3D source reconstruction and in DCM). This function makes it possible to change the specification of the order (without actually changing the data file). Subsequently, every time the order of the conditions is called, the order thereby specified will be used. For instance, if one sorts conditions in an epoched file and then averages it, the conditions in the average file

will be ordered as specified. If trials were originally defined by selecting events from a list, then the order in which the selection was made will be preserved.

D.8. Remove Bad Trials. This function physically removes trials marked as bad from a dataset. This can be useful, for instance, before time-frequency computation, as processing bad trials entails an unnecessary overhead. Also, when it is necessary to remove trials from a dataset (e.g., to remove unused conditions), these trials can be first marked as bad and then removed using this function.

D.9. Crop. This function makes it possible to trim the time axis and/or frequency axis (if available). This is useful when computing time-frequency decompositions and padding the trials with extra data to prevent edge effects. The padding can then be removed using the crop functionality.

E. Script Generation

Data pre-processing in SPM can be automated with MATLAB scripts. Furthermore, these scripts can be generated automatically. To do this, one dataset needs to be analyzed first, using the GUI or batch system. Whenever a pre-processing function is called, all the input arguments, once they have been assembled by the GUI, are stored in the dataset's "history". This history can then be accessed via the reviewing tool to not only see which functions have operated on a dataset, but also to generate a script that reproduces the same operations. The big difference is that, this time, no more GUI interactions are necessary because the script already has all the input arguments, which were provided during the first run. Since the script is automatically generated, it might not be as concise and elegant as a script written by a programmer. For instance, automatically generated scripts sometimes include specification of large montage matrices or long lists of channel labels. But with a basic understanding of MATLAB programming, such scripts can be shortened and simplified.

Automatically generated scripts can be used not only to repeat an analysis, but can also be treated as a template for other analyses. For instance, file names appearing in the script can be changed to preprocess a different subject or (with some adjustments) automatically generated code can be placed in a loop to cycle over a large number of subjects. Modifying automatically generated scripts is an easy way to get started with programming in SPM for M/EEG.

F. Example Datasets

We provide several example datasets on which SPM methods can be tested. The full list of datasets with links can be found under http://www.fil.ion.ucl.ac.uk/spm/data/. Detailed instructions for analysing these datasets are available in the SPM manual (http://www.fil.ion.ucl.ac.uk/spm/doc/spm8_manual.pdf).

G. Toolboxes and Contributing to SPM

We try to maintain SPM for M/EEG as generic, well-structured, and fully documented code. This is important to ensure stability and facilitate future development. However, this can make it more difficult for users to contribute their own tools to SPM. Therefore, SPM includes a protocol for the users to add their own code as toolboxes. These toolboxes are detected automatically and added to the "Toolbox" menu in the main SPM window. The complete list of toolboxes can be found at http://www.fil.ion.ucl.ac.uk/spm/ext/.

Presently, there are two toolboxes for SPM for M/EEG included in the SPM distribution.

G.1. MEEGtools Toolbox. This toolbox includes some useful functions contributed by SPM developers and power users. Many of these functions combine SPM and FieldTrip functionality. Other functions solve system-specific problems that cannot be handled by the main SPM code. For example, there is a set of functions for topography-based artefact correction that have been found to work well for eye blinks [69] and transcranial magnetic stimulation artefact in the EEG [70].

G.2. Beamforming Toolbox. Functions in this toolbox make it possible to perform source reconstruction using beamforming methods in the time [71] and frequency [72] domains and extract source activity using beamformer spatial filters. They make use of SPM-generated forward models (see "Source reconstruction") and (where relevant) generate images that can be entered into the SPM statistics pipeline. Some of these functions are based on FieldTrip code and others are being developed by one of the authors (G. Barnes). In the future, this code is likely to be reorganised with some of the functionality transferred to SPM 3D source reconstruction interface and some integrated into FieldTrip.

We welcome code contributions from SPM users. Single general-use functions can be contributed to MEEGtools. Users who develop useful sets of functions may consider maintaining and distributing them as an external SPM toolbox. Active external developers, who are sufficiently familiar with the code, can be granted write access to our version control system and contribute to development and improvement of the code.

Acknowledgments

This work was funded by the Wellcome Trust. The authors would like to thank all their colleagues and SPM coauthors who have contributed directly and indirectly to the software described in this paper.

References

[1] K. J. Friston, C. D. Frith, P. F. Liddle, and R. S. J. Frackowiak, "Comparing functional (PET) images: the assessment of significant change," *Journal of Cerebral Blood Flow and Metabolism*, vol. 11, no. 4, pp. 690–699, 1991.

[2] K. J. Friston, C. D. Frith, P. F. Liddle, R. J. Dolan, A. A. Lammertsma, and R. S. J. Frackowiak, "The relationship

between global and local changes in PET scans," *Journal of Cerebral Blood Flow and Metabolism*, vol. 10, no. 4, pp. 458–466, 1990.

[3] K. J. Friston, A. P. Holmes, K. J. Worsley, J. P. Poline, C. D. Frith, and R. S. J. Frackowiak, "Statistical parametric maps in functional imaging: a general linear approach," *Human Brain Mapping*, vol. 2, no. 4, pp. 189–210, 1994.

[4] K. J. Worsley and K. J. Friston, "Analysis of fMRI time-series revisited—again," *NeuroImage*, vol. 2, no. 3, pp. 173–181, 1995.

[5] J. Ashburner and K. J. Friston, "Unified segmentation," *NeuroImage*, vol. 26, no. 3, pp. 839–851, 2005.

[6] J. Ashburner and K. J. Friston, "Voxel-based morphometry—the methods," *NeuroImage*, vol. 11, no. 6 I, pp. 805–821, 2000.

[7] K. J. Friston and W. Penny, "Posterior probability maps and SPMs," *NeuroImage*, vol. 19, no. 3, pp. 1240–1249, 2003.

[8] K. J. Friston, L. Harrison, and W. Penny, "Dynamic causal modelling," *NeuroImage*, vol. 19, no. 4, pp. 1273–1302, 2003.

[9] W. D. Penny, K. E. Stephan, A. Mechelli, and K. J. Friston, "Comparing dynamic causal models," *NeuroImage*, vol. 22, no. 3, pp. 1157–1172, 2004.

[10] W. D. Penny, K. E. Stephan, J. Daunizeau et al., "Comparing families of dynamic causal models," *PLoS Computational Biology*, vol. 6, no. 3, Article ID e1000709, 2010.

[11] K. E. Stephan, W. D. Penny, J. Daunizeau, R. J. Moran, and K. J. Friston, "Bayesian model selection for group studies," *NeuroImage*, vol. 46, no. 4, pp. 1004–1017, 2009.

[12] S. J. Kiebel and K. J. Friston, "Statistical parametric mapping for event-related potentials (II): a hierarchical temporal model," *NeuroImage*, vol. 22, no. 2, pp. 503–520, 2004.

[13] S. J. Kiebel and K. J. Friston, "Statistical parametric mapping for event-related potentials: I. Generic considerations," *NeuroImage*, vol. 22, no. 2, pp. 492–502, 2004.

[14] J. Kilner and K. Friston, "Topological inference for EEG and MEG," *Annals of Applied Statistics*, vol. 4, no. 3, pp. 1272–1290, 2010.

[15] J. M. Kilner, S. J. Kiebel, and K. J. Friston, "Applications of random field theory to electrophysiology," *Neuroscience Letters*, vol. 374, no. 3, pp. 174–178, 2005.

[16] C. Phillips, J. Mattout, M. D. Rugg, P. Maquet, and K. J. Friston, "An empirical Bayesian solution to the source reconstruction problem in EEG," *NeuroImage*, vol. 24, no. 4, pp. 997–1011, 2005.

[17] K. Friston, L. Harrison, J. Daunizeau et al., "Multiple sparse priors for the M/EEG inverse problem," *NeuroImage*, vol. 39, no. 3, pp. 1104–1120, 2008.

[18] J. Mattout, C. Phillips, W. D. Penny, M. D. Rugg, and K. J. Friston, "MEG source localization under multiple constraints: an extended Bayesian framework," *NeuroImage*, vol. 30, no. 3, pp. 753–767, 2006.

[19] S. J. Kiebel, J. Daunizeau, C. Phillips, and K. J. Friston, "Variational Bayesian inversion of the equivalent current dipole model in EEG/MEG," *NeuroImage*, vol. 39, no. 2, pp. 728–741, 2008.

[20] O. David, S. J. Kiebel, L. M. Harrison, J. Mattout, J. M. Kilner, and K. J. Friston, "Dynamic causal modeling of evoked responses in EEG and MEG," *NeuroImage*, vol. 30, no. 4, pp. 1255–1272, 2006.

[21] C. C. Chen, S. J. Kiebel, and K. J. Friston, "Dynamic causal modelling of induced responses," *NeuroImage*, vol. 41, no. 4, pp. 1293–1312, 2008.

[22] R. J. Moran, K. E. Stephan, T. Seidenbecher, H. C. Pape, R. J. Dolan, and K. J. Friston, "Dynamic causal models of steady-state responses," *NeuroImage*, vol. 44, no. 3, pp. 796–811, 2009.

[23] W. D. Penny, V. Litvak, L. Fuentemilla, E. Duzel, and K. Friston, "Dynamic causal models for phase coupling," *Journal of Neuroscience Methods*, vol. 183, no. 1, pp. 19–30, 2009.

[24] M. Brett, W. Penny, and S. Kiebel, "Parametric procedures," in *Statistical Parametric Mapping*, K. Friston, J. Ashburner, S. Kiebel, T. Nichols, and W. Penny, Eds., pp. 223–231, Elsevier, Amsterdam, The Netherlands, 2007.

[25] N. Kriegeskorte, W. K. Simmons, P. S. Bellgowan, and C. I. Baker, "Circular analysis in systems neuroscience: the dangers of double dipping," *Nature Neuroscience*, vol. 12, no. 5, pp. 535–540, 2009.

[26] W. Penny and R. Henson, "Hierarchical models," in *Statistical Parametric Mapping*, K. Friston, J. Ashburner, S. Kiebel, T. Nichols, and W. Penny, Eds., pp. 149–155, Elsevier, Amsterdam, The Netherlands, 2007.

[27] S. Baillet, J. J. Rira, G. Main, J. F. Magin, J. Aubert, and L. Ganero, "Evaluation of inverse methods and head models for EEG source localization using a human skull phantom," *Physics in Medicine and Biology*, vol. 46, no. 1, pp. 77–96, 2001.

[28] A. M. Dale and M. I. Sereno, "Improved localization of cortical activity by combining EEG and MEG with MRI cortical surface reconstruction: a linear approach," *Journal of Cognitive Neuroscience*, vol. 5, no. 2, pp. 162–176, 1993.

[29] M. S. Hämäläinen and R. J. Ilmoniemi, "Interpreting magnetic fields of the brain: minimum norm estimates," *Medical and Biological Engineering & Computing*, vol. 32, no. 1, pp. 35–42, 1994.

[30] R. D. Pascual-Marqui, C. M. Michel, and D. Lehmann, "Low resolution electromagnetic tomography: a new method for localizing electrical activity in the brain," *International Journal of Psychophysiology*, vol. 18, no. 1, pp. 49–65, 1994.

[31] K. Friston, R. Henson, C. Phillips, and J. Mattout, "Bayesian estimation of evoked and induced responses," *Human Brain Mapping*, vol. 27, no. 9, pp. 722–735, 2006.

[32] K. J. Friston, W. Penny, C. Phillips, S. Kiebel, G. Hinton, and J. Ashburner, "Classical and Bayesian inference in neuroimaging: theory," *NeuroImage*, vol. 16, no. 2, pp. 465–483, 2002.

[33] J. Mattout, R. N. Henson, and K. J. Friston, "Canonical source reconstruction for MEG," *Computational Intelligence and Neuroscience*, vol. 2007, Article ID 67613, 10 pages, 2007.

[34] H. C. Kim and Z. Ghahramani, "Bayesian Gaussian process classification with the EM-EP algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, pp. 1948–1959, 2006.

[35] B. Ripley, "Flexible non-linear approaches to classification," in *From Statistics to Neural Networks*, V. Cherkassy, J. Friedman, and H. Wechsler, Eds., pp. 105–126, Springer, New York, NY, USA, 1994.

[36] R. N. Henson, J. Mattout, C. Phillips, and K. J. Friston, "Selecting forward models for MEG source-reconstruction using model-evidence," *NeuroImage*, vol. 46, no. 1, pp. 168–176, 2009.

[37] P. J. Besl and N. D. McKay, "A method for registration of 3-D shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[38] G. Nolte, "The magnetic lead field theorem in the quasi-static approximation and its use for magnetoenchephalography forward calculation in realistic volume conductors," *Physics in Medicine and Biology*, vol. 48, no. 22, pp. 3637–3652, 2003.

[39] C. Phillips, J. Mattout, and K. J. Friston, "Forward models for EEG," in *Statistical Parametric Mapping*, pp. 352–366, Elsevier, Amsterdam, The Netherlands, 2007.

[40] D. Geselowitz, "On bioelectric potentials in an inhomogeneous volume conductor," *Biophysical Journal*, vol. 7, pp. 1–11, 1967.

[41] M. S. Hämäläinen and J. Sarvas, "Realistic conductivity geometry model of the human head for interpretation of neuromagnetic data," *IEEE Transactions on Biomedical Engineering*, vol. 36, no. 2, pp. 165–171, 1989.

[42] J. C. de Munck, "A linear discretization of the volume conductor boundary integral equation using analytically integrated elements," *IEEE Transactions on Biomedical Engineering*, vol. 39, no. 9, pp. 986–990, 1992.

[43] J. C. Mosher, R. M. Leahy, and P. S. Lewis, "EEG and MEG: forward solutions for inverse methods," *IEEE Transactions on Biomedical Engineering*, vol. 46, no. 3, pp. 245–259, 1999.

[44] R. N. Henson, G. Flandin, K. J. Friston, and J. Mattout, "A Parametric empirical bayesian framework for fMRI-constrained MEG/EEG source reconstruction," *Human Brain Mapping*, vol. 31, no. 10, pp. 1512–1531, 2010.

[45] R. N. Henson, E. Mouchlianitis, and K. J. Friston, "MEG and EEG data fusion: simultaneous localisation of face-evoked responses," *NeuroImage*, vol. 47, no. 2, pp. 581–589, 2009.

[46] V. Litvak and K. Friston, "Electromagnetic source reconstruction for group studies," *NeuroImage*, vol. 42, no. 4, pp. 1490–1498, 2008.

[47] M. Scherg and D. Von Cramon, "Evoked dipole source potentials of the human auditory cortex," *Electroencephalography and Clinical Neurophysiology*, vol. 65, no. 5, pp. 344–360, 1986.

[48] M. Scherg and P. Berg, "New concepts of brain source imaging and localization," *Electroencephalography and Clinical Neurophysiology. Supplement*, vol. 46, pp. 127–137, 1996.

[49] R. J. Moran, S. J. Kiebel, K. E. Stephan, R. B. Reilly, J. Daunizeau, and K. J. Friston, "A neural mass model of spectral responses in electrophysiology," *NeuroImage*, vol. 37, no. 3, pp. 706–720, 2007.

[50] R. J. Moran, K. E. Stephan, S. J. Kiebel et al., "Bayesian estimation of synaptic physiology from the spectral responses of neural masses," *NeuroImage*, vol. 42, no. 1, pp. 272–284, 2008.

[51] S. J. Kiebel, M. I. Garrido, R. Moran, C. C. Chen, and K. J. Friston, "Dynamic causal modeling for EEG and MEG," *Human Brain Mapping*, vol. 30, no. 6, pp. 1866–1876, 2009.

[52] S. J. Kiebel, O. David, and K. J. Friston, "Dynamic causal modelling of evoked responses in EEG/MEG with lead field parameterization," *NeuroImage*, vol. 30, no. 4, pp. 1273–1284, 2006.

[53] S. J. Kiebel, M. I. Garrido, and K. J. Friston, "Dynamic causal modelling of evoked responses: the role of intrinsic connections," *NeuroImage*, vol. 36, no. 2, pp. 332–345, 2007.

[54] M. Fastenrath, K. J. Friston, and S. J. Kiebel, "Dynamical causal modelling for M/EEG: spatial and temporal symmetry constraints," *NeuroImage*, vol. 44, no. 1, pp. 154–163, 2009.

[55] M. I. Garrido, J. M. Kilner, S. J. Kiebel, and K. J. Friston, "Evoked brain responses are generated by feedback loops," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 104, no. 52, pp. 20961–20966, 2007.

[56] M. I. Garrido, J. M. Kilner, S. J. Kiebel, K. E. Stephan, and K. J. Friston, "Dynamic causal modelling of evoked potentials: a reproducibility study," *NeuroImage*, vol. 36, no. 3, pp. 571–580, 2007.

[57] W. D. Penny, K. J. Friston, J. T. Ashburner, S. J. Kiebel, and T. E. Nichols, Eds., *Statistical Parametric Mapping: The Analysis of Functional Brain Images*, Elsevier, Amsterdam, The Netherlands, 2007.

[58] C. C. Chen, R. N. Henson, K. E. Stephan, J. M. Kilner, and K. J. Friston, "Forward and backward connections in the brain: a DCM study of functional asymmetries," *NeuroImage*, vol. 45, no. 2, pp. 453–462, 2009.

[59] J. Daunizeau, S. J. Kiebel, and K. J. Friston, "Dynamic causal modelling of distributed electromagnetic responses," *NeuroImage*, vol. 47, no. 2, pp. 590–601, 2009.

[60] A. C. Marreiros, J. Daunizeau, S. J. Kiebel, and K. J. Friston, "Population dynamics: variance and the sigmoid activation function," *NeuroImage*, vol. 42, no. 1, pp. 147–157, 2008.

[61] A. C. Marreiros, S. J. Kiebel, and K. J. Friston, "A dynamic causal model study of neuronal population dynamics," *NeuroImage*, vol. 51, no. 1, pp. 91–101, 2010.

[62] A. C. Marreiros, S. J. Kiebel, J. Daunizeau, L. M. Harrison, and K. J. Friston, "Population dynamics under the Laplace assumption," *NeuroImage*, vol. 44, no. 3, pp. 701–714, 2009.

[63] M. I. Garrido, K. J. Friston, S. J. Kiebel, K. E. Stephan, T. Baldeweg, and J. M. Kilner, "The functional anatomy of the MMN: a DCM study of the roving paradigm," *NeuroImage*, vol. 42, no. 2, pp. 936–944, 2008.

[64] M. Scherg and T. W. Picton, "Separation and identification of event-related potential components by brain electric source analysis," *Electroencephalography and Clinical Neurophysiology. Supplement*, vol. 42, pp. 24–37, 1991.

[65] B. Lütkenhöner, "Magnetoencephalography and its Achilles' heel," *Journal of Physiology Paris*, vol. 97, no. 4-6, pp. 641–658, 2003.

[66] M. Scherg, N. Ille, H. Bornfleth, and P. Berg, "Advanced tools for digital EEG review: virtual source montages, whole-head mapping, correlation, and phase analysis," *Journal of Clinical Neurophysiology*, vol. 19, no. 2, pp. 91–112, 2002.

[67] W. D. Penny and S. J. Roberts, "Bayesian multivariate autoregressive models with structured priors," *IEE Proceedings: Vision, Image and Signal Processing*, vol. 149, no. 1, pp. 33–41, 2002.

[68] T. D. Wager, M. C. Keller, S. C. Lacey, and J. Jonides, "Increased sensitivity in neuroimaging analyses using robust regression," *NeuroImage*, vol. 26, no. 1, pp. 99–113, 2005.

[69] P. Berg and B. M. Scherg, "A multiple source approach to the correction of eye artifacts," *Electroencephalography and Clinical Neurophysiology*, vol. 90, no. 3, pp. 229–241, 1994.

[70] V. Litvak, S. Komssi, M. Scherg et al., "Artifact correction and source analysis of early electroencephalographic responses evoked by transcranial magnetic stimulation over primary motor cortex," *NeuroImage*, vol. 37, no. 1, pp. 56–70, 2007.

[71] B. D. van Veen, W. van Drongelen, M. Yuchtman, and A. Suzuki, "Localization of brain electrical activity via linearly constrained minimum variance spatial filtering," *IEEE Transactions on Biomedical Engineering*, vol. 44, no. 9, pp. 867–880, 1997.

[72] J. Gross, J. Kujala, M. Hämäläinen, L. Timmermann, A. Schnitzler, and R. Salmelin, "Dynamic imaging of coherent sources: studying neural interactions in the human brain," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 98, no. 2, pp. 694–699, 2001.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 129365, 9 pages
doi:10.1155/2011/129365

Research Article

EEGIFT: Group Independent Component Analysis for Event-Related EEG Data

Tom Eichele,[1] Srinivas Rachakonda,[2] Brage Brakedal,[1] Rune Eikeland,[1] and Vince D. Calhoun[2,3]

[1] *Department of Biological and Medical Psychology, University of Bergen, Jonas Lies Vei 91, 5011 Bergen, Norway*
[2] *Mind Research Network, 1101 Yale Boulevard, N.E, Albuquerque, NM 87131, New Mexico, USA*
[3] *Department of Electrical and Computer Engineering, University of New Mexico, Albuquerque, USA*

Correspondence should be addressed to Tom Eichele, tom.eichele@gmail.com

Received 1 October 2010; Revised 4 March 2011; Accepted 9 April 2011

Academic Editor: Sylvain Baillet

Independent component analysis (ICA) is a powerful method for source separation and has been used for decomposition of EEG, MRI, and concurrent EEG-fMRI data. ICA is not naturally suited to draw group inferences since it is a non-trivial problem to identify and order components across individuals. One solution to this problem is to create aggregate data containing observations from all subjects, estimate a single set of components and then back-reconstruct this in the individual data. Here, we describe such a group-level temporal ICA model for event related EEG. When used for EEG time series analysis, the accuracy of component detection and back-reconstruction with a group model is dependent on the degree of intra- and interindividual time and phase-locking of event related EEG processes. We illustrate this dependency in a group analysis of hybrid data consisting of three simulated event-related sources with varying degrees of latency jitter and variable topographies. Reconstruction accuracy was tested for temporal jitter 1, 2 and 3 times the FWHM of the sources for a number of algorithms. The results indicate that group ICA is adequate for decomposition of single trials with physiological jitter, and reconstructs event related sources with high accuracy.

1. Introduction

Event related brain responses to simple cognitive tasks are composed of multiple dynamic, temporally and regionally overlapping, functionally separable sub-processes which add to existing oscillatory background activity [1–5]. In other words, event-related processes are spatially and temporally mixed across the brain, and the scalp EEG samples a volume-conducted, spatially degraded version of the responses, where the potential at any location and latency can be consider-ed a mixture of multiple independent sources that stem from large-scale synchronous field potentials [6, 7]. The exploration of the trial-to-trial variability in these responses provides important clues about the dynamics and adaptability of cognitive processes [2, 6, 8–10]

One powerful and increasingly popular method that allows for decomposition of EEG data and assessment of single trial variability is blind source separation with independent component analysis (ICA). ICA algorithms solve a two-dimensional linear mixing problem of spatially, and/or temporally independent sources [11, 12]. ICA models spatio-temporal data as a linear combination of maps and timecourses while attempting to maximize the statistical independence between either the maps (spatial ICA, sICA) or the time courses (temporal ICA, tICA). The method has general applicability to Gaussian mixtures, regarding psychophysiological and neuro imaging data it has been successfully used with averaged ERPs [13], single trial EEG [6, 7, 14], structural and functional MRI [15], and recently also in EEG-fMRI integration [16–22]. Tools for data analysis with ICA are implemented for example in the academic freeware toolboxes EEGLAB [23], ICALAB [24] and GIFT [15, 25], both running in Matlab, as well as the stand-alone package FSL-MELODIC [26].

The basic ICA model applies to single subject data, thus one inherent limitation to the use of ICA in typical multi-subject/session EEG studies is that this method is not naturally suited to generalize results from a group of subjects.

This is because ICAs from separate runs or participants will generate different sets of components with different order and scaling that need to be matched across datasets to allow group inferences. This is in contradistinction to the straightforward way of making group inferences from, for example, ERP component averages from selected channels and latencies in the general linear model [27, 28]. Therefore, a method combining individual components is desired where group inferences are straightforward. There are two strategies to allow for matching of independent components across individuals: one is to combine individual ICs across subjects with clustering techniques [7, 29–31]. Clustering usually involves selection of suitable algorithms and features of interest, that is, topography, timecourse, spectrum and so forth by which between-subject correspondences of components are identified. This requires additional assumptions about the data and expert user input, and differences in algorithm and feature selection, as well as user bias can then create equivocal results. Alternatively, a more parsimonious approach is to create aggregate data containing observations from all subjects, directly estimate components that are consistently expressed in the population in a single set of ICs and then back-reconstruct estimated components to the individual data. This approach has so far predominantly been used for spatial ICA of fMRI [25, 26, 32]. We have recently adopted a group ICA method for parallel and joint decomposition of concurrent EEG-fMRI recordings [21, 22, 33]. Here, we present a group-level temporal ICA model based on the rationale proposed by Calhoun et al. [25] for single-trial analysis of event related EEG timeseries that we also implement in the toolbox EEGIFT, which is available from the Mind Research Networks' medical image analysis lab webpages at http://icatb.sourceforge.net/gift/eegift_startup.php along with documentation and tutorial datasets. EEGIFT runs in Matlab (The Mathworks, Natick, MA), and employs preprocessed data from EEGLAB [23], a popular free toolbox for EEG processing which can be downloaded from http://sccn.ucsd.edu/eeglab/.

In order to make the component estimation computationally feasible, we employ a data reduction using principal component analysis (PCA). The number of components estimated from the data can be based on minimum description length [34] principle (MDL) or on other estimates of dimensionality. Due to aggregation and data reduction with PCA preceding component estimation, group ICA of EEG time-domain data is preferentially suited to the detection of components that contribute to event-related potentials. Processes that are not time/phase-locked within and across subjects, such as background rhythms cannot be satisfyingly reconstructed, in these cases the transformation of the data into the frequency domain prior to ICA decomposition is useful [35]. For time domain data it follows that the accuracy of component detection and back-reconstruction with a group model is dependent on the degree of intra- and inter-individual time and phase-locking of event related EEG processes. Similar to findings in early studies of PCA decomposition of ERP averages [36, 37], excess latency jitter results in splitting of a single source into two (or more) independent components representing the source and its approximate time derivative [21]. Here, we illustrate this dependency in a group analysis of 20 hybrid datasets consisting of three simulated event-related sources with varying degrees of latency jitter, mixed with real EEG data from 20 subjects that participated in a passive listening experiment. It is not trivial to recover reliable estimates about ERP latency jitter from real EEG data, we assume from a literature search and unpublished observations in our lab that single trial peak latencies of larger components such as the auditory N1 and P3 vary approximately 20–40 ms^2, roughly corresponding to the full width at half maximum (FWHM) of these components (for comparison, see, e.g., [8]). We tested the reconstruction accuracy (RA), expressed as the variance of the source accounted for by the reconstructed IC (in terms of R^2) for latency jitter 1, 2 and 3 times the average FWHM of the sources for the Infomax, ERICA, JADE, fastICA, and SIMBEC algorithms that are implemented among others in GIFT and EEGIFT.

2. Method

2.1. Group tICA. The group ICA model is divided into the underlying data generation and mixing process, recording, pre-processing, reduction, component estimation and back-reconstruction (schematically illustrated in Figure 1). We assume that the scalp EEG signal is a gaussian mixture containing statistically independent non-gaussian source timeseries $s(t) = [s^1(t), s^2(t), \ldots, s^N(t)]^T$ indicated by $s_i(t)$ at time t for the ith source from N sources. The sources have weights that specify the contribution to each timepoint. The weights are multiplied by each source's fixed topography. Secondly, it is assumed that the N sources are linearly mixed so that a given time point contains a weighted mixture of the sources. The linear combination of sources is represented by the unknown mixing system A *where* $As(t) = u(t)$ where $u(t) = [u_1(t), u_2(t), \ldots, u_N(t)]^T$ and represents N ideal samples of the signal $u_i(t)$ at time t, for the ith source in the brain. The sampling of the electric activity on the scalp results in $y(t) = [y_1(t), y_2(t), \ldots, y_K(t)]^T$ where the EEG is sampled at K timepoints where $t \quad \{1, 2, \ldots, K\}$. A set of possible transformations during pre-processing, such as downsampling and filtering determine the effective sampling such that $y(j) = [y_1(j), y_2(j), \ldots, y_K(j)]^T$ where the effective temporal sampling is indexed by $j = 1, 2, \ldots, K$.

2.2. Data Reduction. For each individual separately, the preprocessed single trial data $y(j)$ are pre-whitened and reduced via PCA (Figure 1, $R_1^{-1}, \ldots, R_M{}^{-1}$) containing the major proportion of variance in the N uncorrelated timecourses of $x(j) = [x_1(j), x_2(j), \ldots, x_N(j)]^T$. PCA whitening preconditions the data and simplifies ICA estimation due to the orthogonal projection, reduction of complexity, and denoising, as well as compressing the data and thus reducing the computational load. Group data is generated by concatenating individual principal components in the aggregate data set G. In detail, let $X_i = R_i^{-1} Y_i$ be the L-*by*-V reduced data matrix from subject i where Y_i is the Q-*by*-V data matrix containing preprocessed EEG epochs from all channels,

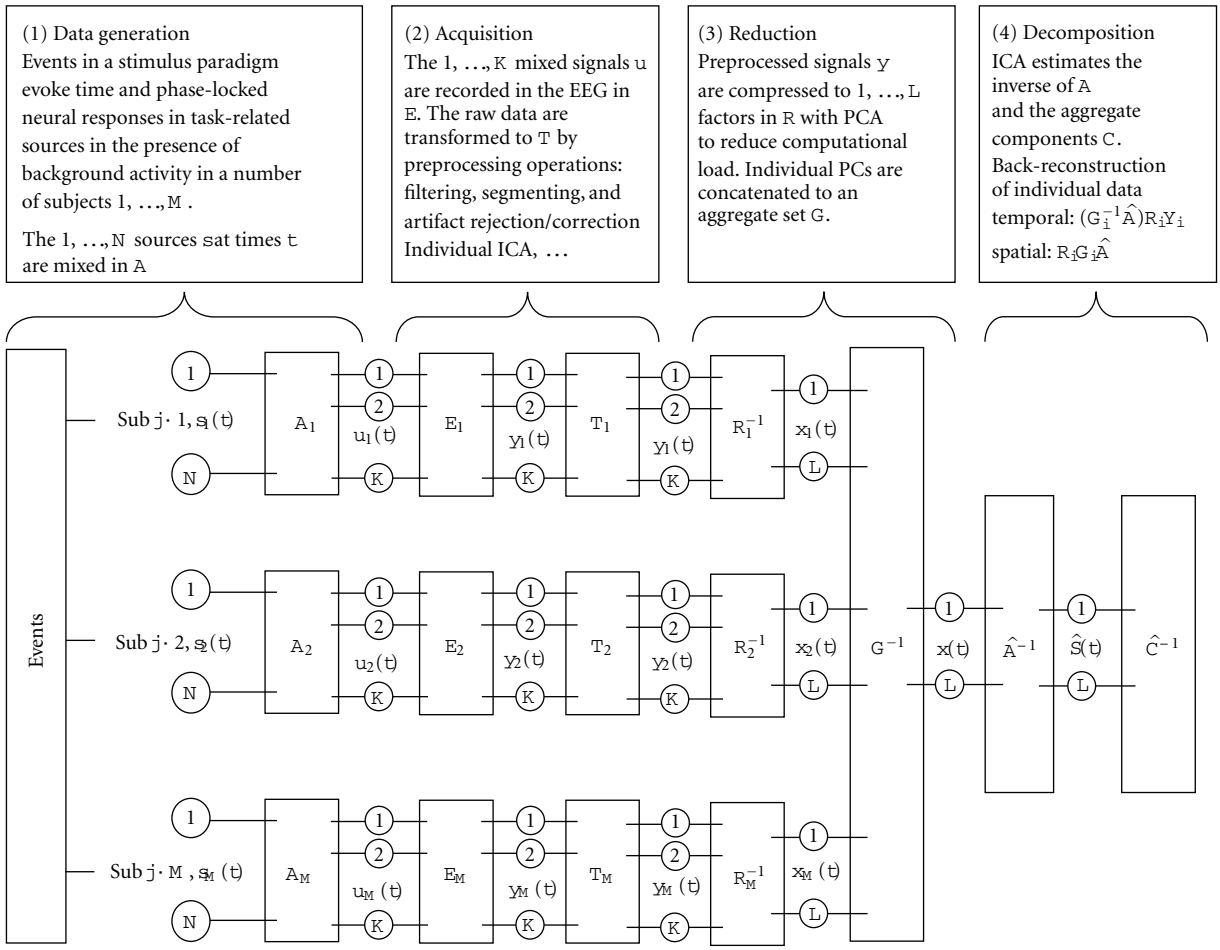


Figure 1: Group ICA. In the group ICA model, we assume that the EEG is a linear mixture of temporally independent sources in each subject $s(t)$. The linear combination of sources is represented by the unknown mixing matrix A, and yields the ideal samples of brain activity u(t), and the signals recorded with the EEG amplifier (E). Transformations (T) during preprocessing contain filtering, epoching, artefact rejection, individual ICA for additional artefact reduction and so forth, altering the effective temporal sampling and dimensionality of the data $y(i)$. For each individual separately, the pre-processed single trial data are pre-whitened and reduced to R via PCA. Group data is generated by concatenating individual principal components in the aggregate data set G. Temporal ICA is performed in this set, estimating aggregate components (C). From the aggregate components, the individual data are reconstructed (see text for details).

R_i^{-1} is the L-*by*-Q reducing matrix from the principal component decomposition, V is the number of timepoints (samples per epoch trials), Q is the number of scalp channels, and L is the size of the channel dimension following reduction. The next step is to concatenate the reduced data from all subjects into a matrix and reduce this matrix to N, the number of components to be estimated. The N-*by*-V reduced, concatenated matrix for the M subjects is

$$X = G^{-1}\begin{bmatrix} R_1^{-1}Y_1 \\ \vdots \\ R_M^{-1}Y_M \end{bmatrix}, \quad (1)$$

where G^{-1} is an N-*by*-LM reducing matrix from a second PCA decomposition and is multiplied on the right by the LM-*by*-V concatenated data matrix for the M subjects.

2.3. ICA Estimation. The idea is to find the mixing matrix A and compute the s_i sources for the group. After concatenation of individual principal components in the aggregate data set G, this matrix is decomposed by ICA, estimating the optimal inverse of the mixing matrix $\hat{A}$, and a single set of source timecourses ($\hat{S}$). Following ICA estimation, we can write $X = \hat{A}\hat{S}$, where $\hat{A}$ is the N-*by*-N mixing matrix and $\hat{S}$ are the N-*by*-V component timecourses. Substituting this expression for X into (1) and multiplying both sides by G results in

$$G\hat{A}\hat{S} = \begin{bmatrix} R_1^{-1}Y_1 \\ \vdots \\ R_M^{-1}Y_M \end{bmatrix}. \quad (2)$$

2.4. Partitioning and Single Subject Reconstruction. Partitioning the matrix G by subject provides the following expression

$$\begin{bmatrix} G_1 \\ \vdots \\ G_M \end{bmatrix} \hat{A}\hat{S} = \begin{bmatrix} R_1^{-1} Y_1 \\ \vdots \\ R_M^{-1} Y_M \end{bmatrix}. \quad (3)$$

We then write the equation for subject i by working only with the elements in partition i of the above matrices such that

$$G_i \hat{A}\hat{S}_i = [R_i^{-1} Y_i]. \quad (4)$$

The matrix $\hat{S}_i$ in (4) contains the single subject component timecourses for subject i, calculated from the following equation

$$\hat{S}_i = \left(G_i \hat{A}\right)^{-1} R_i^{-1} Y_i. \quad (5)$$

We now multiply both sides of (4) by R_i and write

$$Y_i \quad F_i G_i \hat{A}\hat{S}_i \quad (6)$$

yielding the ICA decomposition of the data from subject i contained in the matrix Y_i. The N-*by*-V matrix $\hat{S}_i$ contains the N component timecourses, and the Q-*by*-N matrix $F_i G_i \hat{A}$ is the single subject mixing matrix, yielding the scalp maps for N components.

2.5. Generation of Hybrid Data. In this simulation, 20 mutually uncorrelated hybrid EEG datasets were generated containing 63 channels, 256 timepoints and 500 trials. Three event-related sources (S1–S3) with variable topographies across datasets were mixed with real EEG data from 20 participants from an unrelated study. For each single trial, an event related response (ERR) was simulated with two Gaussians (7).

$$\mathrm{ERR}_x = a_1 e^{-((x-b)/3c_1)^2} - \frac{2}{3} a_2 e^{-((x-b)/2c_2)^2}. \quad (7)$$

The amplitudes a_1 and a_2 were varied randomly and independently from 0.5–2.5 and the widths c_1 and c_2 from 0.5–1.5, introducing additional jitter of the ERR amplitude and shape. Latency jitter is a relevant source of variability in single trials, affecting the accuracy of component estimation [21, 36, 37]. The three sources simulated here had non-overlapping peak latencies, and latency variability (within-"subject") in b was in a range of 20 samples in S1, corresponding to the FWHM of the ERR, 40 samples (2 FWHM) in S2, and 60 samples (3 FWHM) in S3. Sine waves with random phase and amplitude modulation were additionally entered as background activity into each source. Across individual datasets the average peak latency of each source dataset was varied by 20 samples (between-"subjects"). For each source, the scalp distributions were generated as dipolar maps covering six channels (of 63), with 50% overlap between S1-S2, and S2-S3, respectively. Across datasets, the location of each source was systematically varied. These sources were normalized to unit variance, and mixed with normalized real EEG data with the same dimensions from 20 participants. The resulting hybrid data are shown in the top half of Figure 2 for two datasets.

2.6. Independent Component Analysis. In order to generate a reference value for the performance of group ICA we computed individual ICA solutions in EEGLAB for each of the datasets, employing the Infomax algorithm [38]. For the group ICA, all subjects were analyzed at once, and principal component analysis (PCA) was used for compression to allow the datasets to be processed together. The number of components is estimated by doing singular value decomposition on the data and the resulting eigenvalues are passed to MDL method [34]. Here we selected 20 components as the top 20 components usually explain more than 95% of the variance in the data. In our experience, the exact choice of the number of components does not critically affect the results as long as this number is not much smaller than the true number of sources. In the PCA steps, data from each dataset was reduced over the spatial dimension, that is. from the number of channels to 20 principal components, concatenated across subjects, and again reduced to 20 components. Temporal ICA was then performed using the Infomax algorithm [38] with subsequent back-reconstruction into single subjects.In order to assess reconstruction accuracy of group ICA for different numbers of estimated components, we estimated the solutions for 10, 20, 30, 40 and 50 components using Infomax. For comparison between algorithms, we also estimated solutions using the fastICA [39], JADE [40], SIMBEC [41] and ERICA [42] algorithms. The reconstruction accuracy of group ICA was expressed as the variance of the simulated sources accounted for by the reconstructed ICs averaged across the 20 datasets (R^2), separately for the entire single trial images, the amplitude modulation across trials around the component peak latency (averaged in a 20 sample window), the component average timeseries, and topographies (Table 1). Results for two hybrid datasets with variable topographic and temporal representations of the three sources are illustrated in Figure 2.

2.7. Application to Real Data. In addition to the quantification of the model performance, we illustrate group ICA in the context of typical recording conditions and preprocessing steps that we employ with a decomposition of an auditory oddball dataset. 32 healthy participants took part in the experiment after providing a written statement of informed consent. Participants were sitting in an electro-magnetically shielded and sound-attenuated testing chamber (Rainford EMC Systems, Wigan, UK) and were fitted with 61 Ag/AgCl scalp electrodes mounted in an elastic cap (EasiCap, Falk Minow Services, Breitenbrunn, Germany) and two additional channels monitoring eye movements. All channels were referenced to the nose, and impedances were kept below 10 kΩ. EEGs were recorded continuously at 500 Hz sampling frequency with a band-pass from .01–250 Hz with BrainAmp DC amplifiers (BrainProducts, Munich, Germany). The experiment consisted of detecting an infrequent target sound

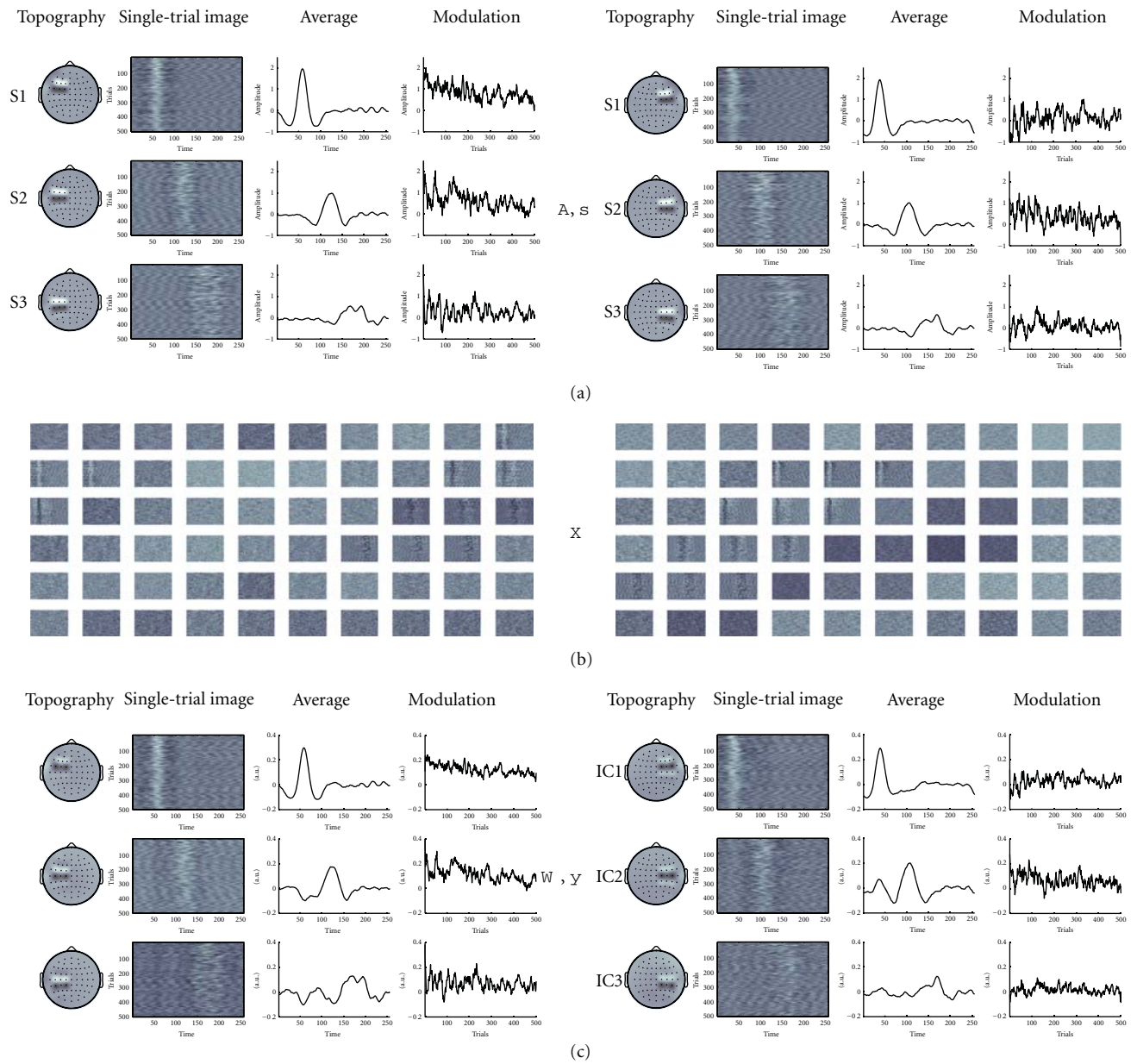


Figure 2: Hybrid Data. The figure shows two hybrid datasets with three source topographies (A) and timecourses (s) on top, the mixture of the sources with real EEG (X) in (b), and the reconstruction of the sources after group ICA (W, y) in (c) section.

within a series of frequent regular sounds and participants were asked to respond as quickly as possible by pressing a button with their right index finger. The standard stimulus was a center-panned 500 Hz tone, the target stimuli were left or right-panned location deviants, respectively. Targets occurred with a probability of 0.1 for each location. Stimulus duration was 75 ms and the inter-stimulus interval was 700 ms, with no immediate repetitions of targets. All stimuli were presented at approximately 65 decibels above threshold. EEGs were down-sampled to 250 Hz, filtered with a zero-phase Butterworth filter from 1–45 Hz, and re-referenced to common average reference. In the example application, the data were segmented from – 800 to 1200 ms around target stimuli and thus contain a sequence of standard-target-standard sounds, to dissociate between obligatory stimulus-related and target-related components, respectively. Trials with amplitudes exceeding ± 150 μV on any of the channels were excluded from further analysis. Concatenated single sweeps around target onset were subjected to single subject Infomax ICA in EEGLAB [23] running in MATLAB. Components with topographies and timecourses attributeable artifacts were identified and removed from the data [43]. For each dataset, we extract 20 components from the data based on the MDL method. Missing trials were padded with the mean from surrounding trials because there are gradual changes across trials. Single-trials were additionally denoised with a wavelet filter [44]. Hereafter, group ICA was computed, estimating 20 components.

Table 1: Reconstruction Accuracy. The table summarizes the reconstruction accuracy (RA, mean across datasets ± S.E.M.) of different group ICA models. RA stands for proportion of the source variability accounted for by the ICs averaged across the 20 datasets, and was computed separately for the entire single trial images, the amplitude modulation across trials around the component peak latency, the component average timeseries, and topographies.

	Single Trial			Peak Latency			Average			Topography		
Algorithm, Nr IC's	S1	S2	S3	S1	S2	S3	S1	S2	S3	S1	S2	S3
Infomax single, 20	0.98 ± .01	0.97 ± .02	0.98 ± .01	0.99 ± .01	0.99 ± .01	0.99 ± .02	1 ± .00	1 ± .00	1 ± .00	0.89 ± .03	0.89 ± .04	0.90 ± .03
Infomax, 10	0.87 ± .06	0.79 ± .08	0.66 ± .09	0.92 ± .03	0.89 ± .03	0.79 ± .05	0.99 ± .01	0.94 ± .03	0.84 ± .04	0.84 ± .04	0.82 ± .03	0.81 ± .03
Infomax, 20	0.88 ± .04	0.77 ± .07	0.54 ± .08	0.91 ± .03	0.87 ± .03	0.69 ± .06	0.99 ± .01	0.96 ± .02	0.84 ± .05	0.87 ± .04	0.83 ± .05	0.74 ± .05
Infomax, 30	0.87 ± .07	0.76 ± .07	0.50 ± .10	0.91 ± .04	0.86 ± .05	0.65 ± .05	0.99 ± .01	0.95 ± .02	0.86 ± .05	0.87 ± .04	0.81 ± .05	0.83 ± .04
Infomax, 40	0.87 ± .08	0.75 ± .09	0.47 ± .09	0.91 ± .03	0.86 ± .04	0.62 ± .08	0.99 ± .02	0.96 ± .01	0.87 ± .04	0.88 ± .03	0.79 ± .04	0.81 ± .04
Infomax, 50	0.87 ± .05	0.75 ± .06	0.44 ± .11	0.91 ± .05	0.85 ± .04	0.59 ± .07	0.99 ± .01	0.96 ± .01	0.84 ± .05	0.88 ± .04	0.81 ± .05	0.81 ± .04
ERICA, 20	0.88 ± .03	0.73 ± .05	0.46 ± .09	0.92 ± .05	0.85 ± .05	0.62 ± .08	0.99 ± .01	0.89 ± .04	0.77 ± .06	0.86 ± .03	0.76 ± .06	0.71 ± .06
JADE, 20	0.79 ± .06	0.76 ± .07	0.43 ± .10	0.85 ± .03	0.87 ± .05	0.59 ± .09	0.99 ± .02	0.96 ± .02	0.79 ± .06	0.87 ± .05	0.78 ± .08	0.77 ± .07
fastICA, 20	0.86 ± .09	0.77 ± .08	0.53 ± .09	0.90 ± .06	0.87 ± .07	0.68 ± .10	0.99 ± .01	0.96 ± .01	0.82 ± .05	0.87 ± .04	0.84 ± .06	0.74 ± .07
SIMBEC, 20	0.87 ± .08	0.78 ± .08	0.50 ± .09	0.91 ± .06	0.88 ± .06	0.65 ± .07	0.99 ± .01	0.92 ± .03	0.78 ± .06	0.87 ± .03	0.79 ± .04	0.75 ± .05

3. Results

Reconstruction accuracy (RA) for all analyses is summarized in Table 1. Individual ICA reconstructs the source timecourses with near-perfect accuracy and independent of latency jitter, the topographies are recovered with an accuracy of around 0.9. Group ICA models yield overall lower RA than individual ICA, coming closest to individual ICA in S1, where the entire source timecourse is reconstructed with 0.86 on the average, Infomax with 20 components yields the best performance at 0.88. RA for the amplitude modulation around the peak latency of S1 is 0.9 on the average, for the average timecourses RA is 0.99. These two latter features are typically most relevant for making inferences about electrophysiological data, that is. the overall shape of the waveform and the (single trial) peak amplitudes of components. The topographies of S1 are reconstructed with an accuracy of 0.87. The group ICA result for all features falls with increasing latency jitter regardless of algorithm choice. For the entire timecourse, average RA across algorithms is S1—0.86, S2—0.76, S3—0.50, with the most pronounced drop-off of all features. The peak RA for the peak is S1—0.90, S2—0.87, S3—0.65. RA for the timecourse average is S1—0.95, S2—0.95, S3—0.82, the topographies are reconstructed with S1—0.87, S2—0.80, S3—0.77 overall a less pronounced drop-off. This effect is more pronounced with increasing the number of estimated components as can be seen in the RA of all features of S3. In summary, all three sources were recovered with sufficient reconstruction accuracy of all four features.

The decomposition of the real data yielded a number of event related components that showed differential responses between standard and target sounds. We related these in terms of the topography and peak latency to the ERP components N1, T-complex, P2, N2 and subcomponents of the P3. In Figure 3, we show the group mean of one independent component that represents the auditory N1, together with the reconstructions for three representative subjects. The group mean of the N1 component is representative for the group level average, and as is expected somewhat smaller in amplitude than compared to a single subject's mean amplitude because of inter-individual latency differences. Parts of this dataset accompany the toolbox as tutorial material and the entire dataset is available upon request.

4. Discussion

This work presents an approach to perform a temporal independent component analysis on single-trial time domain EEG data for multiple subjects simultaneously. Our model uses a combination of principal component analysis for data reduction, subsequent independent component analysis on the aggregate data, and back-reconstruction of the aggregate mixing matrix in individual Subjects [20, 25]. This method is implemented in the freeware toolbox EEGIFT that runs in Matlab (R13 and newer) and is downloadable from http://mialab.mrn.org/ or http://icatb.sourceforge.net/. EEGIFT has a graphical user interface (GUI) that allow import of EEG data from multiple participants pre-processed in EEGLAB (http://sccn.ucsd.edu/eeglab/). Another GUI allows the user to specify of analysis details, such as PCA data reduction, model order, choice of ICA algorithm, and the respective parameters. EEGIFT also allow robust estimation with ICASSO [45]. The analysis output is stored after back-reconstruction as individual timecourses and topographies in Matlab format which can be used for specifying between-condition and/or between-groups statistical tests. Individual data, group averages, and population statistics can also be visualized in a GUI as topographies, grand mean timecourses and single trial images. Analyses can also be batch-scripted for convenience. A full documentation of functions in EEGIFT, including a tutorial walkthrough, accompanies the download package (http://icatb.sourceforge.net/gift/eegift_startup). User support is provided through the GIFT mailing list (Icatb-discuss).

In EEGIFT, the back-reconstructed timecourses and topographies are a function primarily of the variability within subjects, as opposed to representing a representation of the average across subjects. A simulation affords to create a situation of a known ground truth with sources and noise parameters, and is useful to illustrate of the concept of the group

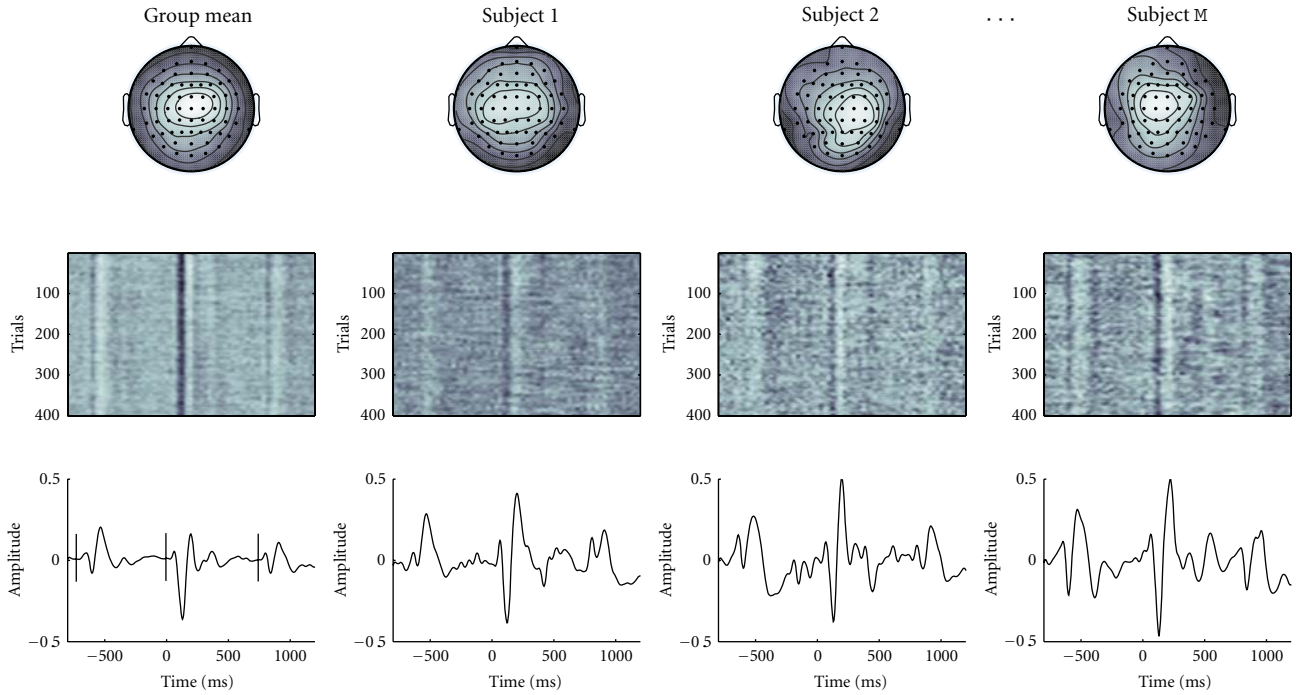


Figure 3: Real Data Decomposition. This figure is an illustration of one component from a group ICA decomposition of an auditory oddball dataset (n = 32 participants), and shows the group mean of one independent component in the leftmost column, and reconstructions for three subject s in the other columns. Note that the group mean component peaks have smaller amplitude due to latency jitter. Top: topography, Middle: single trial image, bottom: component event related average. Vertical lines in the lower left indicate the onset of the preceding standard, target, and succeeding standard, respectively. Topography and latency identify this component as the auditory N1, and typical N1-enhancement is clearly visible in response to targets.

model. Here, we generated hybrid data with realistic spatial, temporal and amplitude variability. The reconstruction of the solutions with different algorithms and numbers of components demonstrates that this approach offers a straightforward and computationally tractable solution to the problem of multi-subject analysis with ICA. The results from the decomposition of hybrid dataset illustrate that this group model is able to recover timecourses and topographies of event related responses on single trial level with overall sufficiently high accuracy. As laid out in the introduction, the critical determinant for the success of a group model computed for time-domain data is intra- and inter-subject latency jitter. There is considerable difficulty in reliably estimating latency jitter of event related responses in real EEG data; we assume here that physiologic jitter is roughly on the order of the full width at half maximum of sources, in which case the reconstruction accuracy of the group model reaches more than 90% of the accuracy of the individual ICA estimates. Individual ICA with subsequent ordering of components across subject would achieve the highest accuracy with an ideal clustering technique. We did not attempt to directly compare the performance of group-level ICA with that of individual ICA with subsequent clustering, apart from possible computational limitations, the challenge for existing algorithms is to identify and cluster components with poor between-subject correspondence of their topographies and timecourses [7, 29–31]. Group ICA avoids any such problems since the decomposition is computed for all datasets/subjects simultaneously, estimating a single set of components with the same order across datasets. Consequently, it is straightforward to perform random-effects population tests for the timecourses as well as the topographies. This can also be done in the statistical parametric mapping framework where testing within the Gaussian random field theory with adjustments for multiple comparisons (e.g., [28, 46, 47]). Apart from being used as a primary tool for inference, we believe that there is also a utility for group ICA in the mining stage of prediction-based analysis: For example, in cases where a-priori models of the event related responses can only be poorly specified, for example, where the selection criteria for appropriate electrodes and time-windows from the EEG data are unclear, group ICA results can be used for model specification.

As noted in Section 1, the limitation of the current method is that responses with poor time/phase-locking are not satisfyingly reconstructed [21, 36, 37]. This is a consequence of the data reduction and aggregation, which inherently limits the visibility of this method to evoked activity, both within and across subjects. Sources that have a loose relation to stimulus/response onset, if extracted and identified, are usually represented in more than one component. This is in clear contradistinction to the superior performance of temporal ICA on concatenated EEG epochs from single subjects, which is insensitive to trial-to-trial phase/latency

variability of sources. For detection of poorly time-locked processes in this framework one could consider time-domain data with frequency or time-frequency transformed data [35], or incorporate correction for latency jitter during preprocessing where feasible [8, 44]. We assume from testing the performance of this model extensively in simulated, hybrid, and real data that it is a very useful addition to the available ICA-toolbelt, in that it affords a straightforward possibility to perform group analysis of event related EEG responses. Since the underlying generative model is flexible and modality independent, and the software implementations are highly interrelated, a genuine advantage of this method is that the EEG components can be fused with results from diverse biomedical imaging modalities such as sMRI, fMRI, DTI and VBMas well as genetic information (SNP) within the same conceptual and computational framework [15, 18, 20, 23, 48, 49]. This then affords multimodal inferences which can bear novel insights about brain function.

Acknowledgment

The present study was financially supported with a BILAT-GRUNN Grant from the Research Council of Norway to Tom Eichele and by the National Institutes of Health, under Grants 1 R01 EB 000840 and 1 R01 EB 005846 to Vince Calhoun.

References

[1] E. Halgren and K. Marinkovic, "General principles for the physiology of cognition as suggested by intracranial ERPs," in *Recent Advances in Event-Related Brain Potential Research*, C. Ogura, Y. Koga, and M. Shimokochi, Eds., pp. 1072–1084, Elsevier, Amsterdam, The Netherlands, 1995.

[2] A. Arieli, A. Sterkin, A. Grinvald, and AD. Aertsen, "Dynamics of ongoing activity: explanation of the large variability in evoked cortical responses," *Science*, vol. 273, no. 5283, pp. 1868–1871, 1996.

[3] E. Halgren, K. Marinkovic, and P. Chauvel, "Generators of the late cognitive potentials in auditory and visual oddball tasks," *Electroencephalography and Clinical Neurophysiology*, vol. 106, no. 2, pp. 156–164, 1998.

[4] T. Eichele, K. Specht, M. Moosmann et al., "Assessing the spatiotemporal evolution of neuronal activation with single-trial event-related potentials and functional MRI," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 102, no. 49, pp. 17798–17803, 2005.

[5] K. A. Kiehl, M. C. Stevens, K. R. Laurens, G. Pearlson, V. D. Calhoun, and P. F. Liddle, "An adaptive reflexive processing model of neurocognitive function: supporting evidence from a large scale (n = 100) fMRI study of an auditory oddball task," *NeuroImage*, vol. 25, no. 3, pp. 899–915, 2005.

[6] S. Makeig, S. Debener, J. Onton, and A. Delorme, "Mining event-related brain dynamics," *Trends in Cognitive Sciences*, vol. 8, no. 5, pp. 204–210, 2004.

[7] J. Onton and S. Makeig, "Information-based modeling of event-related brain dynamics," in *Progress in Brain Research*, C. Neuper and G. Pfurtscheller, Eds., vol. 159, pp. 99–120, Elsevier, Amsterdam, The Netherlands, 2006.

[8] K. M. Spencer, "Averaging, detection, and classification of single-trial ERPs," in *Event Related Potentials*, T. C. Handy, Ed., pp. 209–228, The MIT, Cambridge, Mass, USA, 2005.

[9] S. Debener, M. Ullsperger, M. Siegel, and A. K. Engel, "Single-trial EEG-fMRI reveals the dynamics of cognitive function," *Trends in Cognitive Sciences*, vol. 10, no. 12, pp. 558–563, 2006.

[10] H. Eichele and H. T. Juvodden, "Mal-adaptation of event-related EEG responses preceding performance errors," *Frontiers in Human Neuroscience*, vol. 10, no. 4, p. 65, 2010.

[11] A. Hyvärinen and E. Oja, "Independent component analysis: algorithms and applications," *Neural Networks*, vol. 13, no. 4-5, pp. 411–430, 2000.

[12] J. V. Stone, "Independent component analysis: an introduction," *Trends in Cognitive Sciences*, vol. 6, no. 2, pp. 59–64, 2002.

[13] S. Makeig, T. P. Jung, A. J. Bell, D. Ghahremani, and T. J. Sejnowski, "Blind separation of auditory event-related brain responses into independent components," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 94, no. 20, pp. 10979–10984, 1997.

[14] T. P. Jung, S. Makeig, M. Westerfield, J. Townsend, E. Courchesne, and T. J. Sejnowski, "Analysis and visualization of single-trial event-related potentials," *Human Brain Mapping*, vol. 14, no. 3, pp. 166–185, 2001.

[15] V. D. Calhoun and T. Adali, "Unmixing fMRI with independent component analysis," *IEEE Engineering in Medicine and Biology Magazine*, vol. 25, no. 2, pp. 79–90, 2006.

[16] S. Debener, M. Ullsperger, M. Siegel, K. Fiehler, D. Y. Von Cramon, and A. K. Engel, "Trial-by-trial coupling of concurrent electroencephalogram and functional magnetic resonance imaging identifies the dynamics of performance monitoring," *Journal of Neuroscience*, vol. 25, no. 50, pp. 11730–11737, 2005.

[17] B. Feige, K. Scheffler, F. Esposito, F. Di Salle, J. Hennig, and E. Seifritz, "Cortical and subcortical correlates of electroencephalographic alpha rhythm modulation," *Journal of Neurophysiology*, vol. 93, no. 5, pp. 2864–2872, 2005.

[18] V. D. Calhoun, T. Adali, N. R. Giuliani, J. J. Pekar, K. A. Kiehl, and G. D. Pearlson, "Method for multimodal analysis of independent source differences in schizophrenia: combining gray matter structural and auditory oddball functional data," *Human Brain Mapping*, vol. 27, no. 1, pp. 47–62, 2006.

[19] V. D. Calhoun, T. Adali, G. D. Pearlson, and K. A. Kiehl, "Neuronal chronometry of target detection: fusion of hemodynamic and event-related potential data," *NeuroImage*, vol. 30, no. 2, pp. 544–553, 2006.

[20] T. Eichele, V. D. Calhoun, M. Moosmann et al., "Unmixing concurrent EEG-fMRI with parallel independent component analysis," *International Journal of Psychophysiology*, vol. 67, no. 3, pp. 222–234, 2008.

[21] M. Moosmann, T. Eichele, H. Nordby, K. Hugdahl, and V. D. Calhoun, "Joint independent component analysis for simultaneous EEG-fMRI: principle and simulation," *International Journal of Psychophysiology*, vol. 67, no. 3, pp. 212–221, 2008.

[22] T. Eichele, V. D. Calhoun, and S. Debener, "Mining EEG-fMRI using independent component analysis," *International Journal of Psychophysiology*, vol. 73, no. 1, pp. 53–61, 2009.

[23] A. Delorme and S. Makeig, "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis," *Journal of Neuroscience Methods*, vol. 134, no. 1, pp. 9–21, 2004.

[24] A. Cichocki, S.-I. Amari et al., *Adaptive Blind Signal and Image Processing*, Wiley, New York, NY, USA, 2002.

[25] V. D. Calhoun, T. Adali, G. D. Pearlson, and J. J. Pekar, "A method for making group inferences from functional MRI data using independent component analysis," *Human Brain Mapping*, vol. 14, no. 3, pp. 140–151, 2001.

[26] C. F. Beckmann and S. M. Smith, "Tensorial extensions of independent component analysis for multisubject FMRI analysis," *NeuroImage*, vol. 25, no. 1, pp. 294–311, 2005.

[27] T. W. Picton, S. Bentin, P. Berg et al., "Guidelines for using human event-related potentials to study cognition: recording standards and publication criteria," *Psychophysiology*, vol. 37, no. 2, pp. 127–152, 2000.

[28] S. J. Kiebel and K. J. Friston, "Statistical parametric mapping for event-related potentials: I. Generic considerations," *NeuroImage*, vol. 22, no. 2, pp. 492–502, 2004.

[29] F. Esposito, T. Scarabino, A. Hyvarinen et al., "Independent component analysis of fMRI group studies by self-organizing clustering," *NeuroImage*, vol. 25, no. 1, pp. 193–205, 2005.

[30] J. Onton, M. Westerfield, J. Townsend, and S. Makeig, "Imaging human EEG dynamics using independent component analysis," *Neuroscience and Biobehavioral Reviews*, vol. 30, no. 6, pp. 808–822, 2006.

[31] F. de Martino, F. Gentile, F. Esposito et al., "Classification of fMRI independent components using IC-fingerprints and support vector machine classifiers," *NeuroImage*, vol. 34, no. 1, pp. 177–194, 2007.

[32] V. J. Schmithorst and S. K. Holland, "Comparison of three methods for generating group statistical inferences from independent component analysis of functional magnetic resonance imaging data," *Journal of Magnetic Resonance Imaging*, vol. 19, no. 3, pp. 365–368, 2004.

[33] T. Eichele and V. Calhoun, "Parallel EEG-fMRI ICA decomposition," in *Integrating EEG and FMRI, Analysis and Application*, M. Ullsperger and S. Debener, Eds., Oxford University Press, New York, NY, USA, 2010.

[34] J. Rissanen, "A universal prior for integers and estimation by minimum description length," *Annals of Statistics*, vol. 11, no. 2, pp. 416–431, 1983.

[35] L. Wu, T. Eichele, and V. D. Calhoun, "Reactivity of hemodynamic responses and functional connectivity to different states of alpha synchrony: a concurrent EEG-fMRI study," *NeuroImage*, vol. 52, no. 4, pp. 1252–60, 2010.

[36] E. Donchin and E. F. Heffley, "Multivariate analysis of event-related potential data: a tutorial review," in *Multidisciplinary Perspectives in Event-Related Potential Research*, D. Otto, Ed., Government Printing Office, Washington, DC, USA, 1978.

[37] J. Mocks, "The influence of latency jitter in principal component analysis of event-related potentials," *Psychophysiology*, vol. 23, no. 4, pp. 480–484, 1986.

[38] A. J. Bell and T. J. Sejnowski, "An information-maximization approach to blind separation and blind deconvolution," *Neural Computation*, vol. 7, no. 6, pp. 1129–1159, 1995.

[39] A. Hyvärinen and E. Oja, "A fast fixed-point algorithm for independent component analysis," *Neural Computation*, vol. 9, no. 7, pp. 1483–1492, 1997.

[40] J. F. Cardoso and A. Souloumiac, "Blind beamforming for non Gaussian signals," *IEE-Proceedings-F*, vol. 140, no. 6, pp. 362–370, 1993.

[41] S. Cruces, A. Cichocki et al., "Criteria for the simultaneous extraction of arbitrary groups of sources," in *Proceedings of the International Conference on Independent Component Analysis and Blind Signal Separation*, San Diego, Calif, USA, December 2001.

[42] S. Cruces, L. Castedo, A. Cichocki et al., "Novel blind source separation algorithms using cumulants," in *Proceedings of the IEEE Interntional Conference on Acoustics, Speech, and Signal Processing*, pp. 3152–3155, Istanbul, Turkey, June 2000.

[43] T. P. Jung, S. Makeig, C. Humphries et al., "Removing electroencephalographic artifacts by blind source separation," *Psychophysiology*, vol. 37, no. 2, pp. 163–178, 2000.

[44] R. Quian Quiroga and H. Garcia, "Single-trial event-related potentials with wavelet denoising," *Clinical Neurophysiology*, vol. 114, no. 2, pp. 376–390, 2003.

[45] J. Himberg, A. Hyvärinen, and F. Esposito, "Validating the independent components of neuroimaging time series via clustering and visualization," *NeuroImage*, vol. 22, no. 3, pp. 1214–1222, 2004.

[46] K. J. Friston, A. P. Holmes, K. J. Worsley, J. P. Poline, C. D. Frith, and R. S. J. Frackowiak, "Statistical parametric maps in functional imaging: a general linear approach," *Human Brain Mapping*, vol. 2, no. 4, pp. 189–210, 1995.

[47] J. M. Kilner, S. J. Kiebel, and K. J. Friston, "Applications of random field theory to electrophysiology," *Neuroscience Letters*, vol. 374, no. 3, pp. 174–178, 2005.

[48] J. Liu, G. Pearlson et al., "Combining fMRI and SNP data to investigate connections between brain function and genetics using parallel ICA," *Human Brain Mapping*, vol. 30, no. 1, pp. 241–255, 2007.

[49] V. D. Calhoun, J. Liu, and T. Adali, "A review of group ICA for fMRI data and ICA for joint inference of imaging, genetic, and ERP data," *NeuroImage*, vol. 45, no. 1, pp. S163–S172, 2009.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 831409, 11 pages
doi:10.1155/2011/831409

Research Article

LIMO EEG: A Toolbox for Hierarchical LInear MOdeling of ElectroEncephaloGraphic Data

Cyril R. Pernet,[1] Nicolas Chauveau,[2,3] Carl Gaspar,[4] and Guillaume A. Rousselet[4]

[1] *Division of Clinical Neurosciences, SFC Brain Imaging Research Centre, SINAPSE Collaboration, University of Edinburgh, Western General Hospital, Edinburgh EH4 2XU, UK*
[2] *Inserm; Imagerie Cérébrale et Handicaps Neurologiques UMR 825, 31059 Toulouse, France*
[3] *Université de Toulouse, UPS, Imagerie Cérébrale et Handicaps Neurologiques UMR 825, 31059 Toulouse, France*
[4] *Centre for Cognitive Neuroimaging (CCNi), Institute of Neuroscience and Psychology, University of Glasgow, Glasgow G12 8QB, UK*

Correspondence should be addressed to Cyril R. Pernet, cyril.pernet@ed.ac.uk

Received 21 September 2010; Revised 23 November 2010; Accepted 31 December 2010

Academic Editor: Sylvain Baillet

Magnetic- and electric-evoked brain responses have traditionally been analyzed by comparing the peaks or mean amplitudes of signals from selected channels and averaged across trials. More recently, tools have been developed to investigate single trial response variability (e.g., EEGLAB) and to test differences between averaged evoked responses over the entire scalp and time dimensions (e.g., SPM, Fieldtrip). LIMO EEG is a Matlab toolbox (EEGLAB compatible) to analyse evoked responses over all space and time dimensions, while accounting for single trial variability using a simple hierarchical linear modelling of the data. In addition, LIMO EEG provides robust parametric tests, therefore providing a new and complementary tool in the analysis of neural evoked responses.

1. Introduction

LIMO EEG (https://gforge.dcn.ed.ac.uk/gf/project/limo_eeg/) is a toolbox for the statistical analysis of physiological data. The main goal of the toolbox is the analysis and formal testing for experimental effects at all electrodes/sensors and all time points of magneto- and electro encephalography (MEEG) recordings. This contrasts with traditional approaches that select peaks or mean amplitudes of averaged evoked responses. The toolbox is implemented in Matlab (http://www.mathworks.com/) and requires the Matlab statistical toolbox (free alternative to these functions can be found on the LIMO EEG server and corresponds to adapted version of Octave functions (http://www.gnu.org/software/octave/). The data structure and visualization makes use of the EEGLAB Matlab toolbox [1] (http://sccn.ucsd.edu/eeglab/); therefore LIMO EEG is better used as a plug-in of EEGLAB, although the statistical analyses can be performed independently. Similarly, the toolbox is primarily designed for EEG data although both EEGLAB and LIMO EEG can process MEG data.

The toolbox offers a comprehensive range of statistical tests (Table 1), including many popular designs (ANOVAs, linear regressions, ANCOVAs). Some of the statistical methods, that is, massive univariate general linear analyses [2, 3] and spatiotemporal clustering for multiple comparisons correction [4–6] have existed for several years whereas others like bootstrapping were introduced only recently [7–9].

Contrary to other toolboxes dedicated to the analysis of event related potentials (ERPs), LIMO EEG deals both with within-subject variance (i.e., single trial analyses) and between-subject variance (like in e.g., SPM [2, 3]). Using LIMO EEG, data are analyzed using a hierarchical general linear model where parameters of a GLM are estimated for each subject at each time point and each electrode independently (1st level analyses). Estimated parameters from the first level analyses are then integrated across subjects (2nd level analysis—Figure 1). This hierarchical modelling of the data is similar to the one used to analyze PET/fMRI data (SPM, FSL, BrainVoyager, etc.). Our general linear approach of analyzing MEEG data thus complements others which also rely on linear modeling but focus on

Table 1: Summary of statistical tests available in LIMO EEG via the GUI with the bootstrap procedures used at the univariate (one time frame on one electrode) and cluster levels.

Statistical tests available via the general user interface	Hypothesis tested at the univariate level using bootstrap (non corrected P values/significance testing)	Multiple comparisons correction (accounts for doing many tests) using cluster statistics
One sample t-test (Student T test)	H1 (resample subjects and use bootstrapped T values)	H0 (center data then resample subjects and use bootstrapped T values)
Paired t-test (Student T test)	H1 (resample subjects paired observations and use bootstrapped T values)	H0 (center data per condition then resample subjects and use bootstrapped T values)
Two samples t-test (Student T test)	H1 (resample subjects in each group and use bootstrapped mean differences between groups)	H0 (center data per group then resample subjects and use bootstrapped T values)
Regressions (Fisher F test)	H1 (resample subjects and use regression coefficients)	H0 (resample subjects and fit to the original design matrix and use bootstrapped F values)
One way ANOVA (Fisher F test)	H0 (center data per group then resample subjects and use bootstrapped F values)	H0 (center data per group then resample subjects and use bootstrapped F values)
One way ANCOVA (Fisher F test)	H0 (resample subjects and fit to the original design matrix and use bootstrapped F values)	H0 (resample subjects and fit to the original design matrix and use bootstrapped F values)
Repeated measures ANOVA (Hotelling T^2 test)	H0 (center data per conditions then resample subjects and use bootstrapped F values)	H0 (center data per conditions then resample subjects and use bootstrapped F values)

averaged event related data [2] rather than single trials, or factorize time [3, 8], or both, rather than using time as a natural dimension.

2. Method

2.1. Hierarchical General Linear Model for MEEG Data: 1st Level. MEEG data form 3 dimensional matrices. Following the EEGLAB convention, the 1st dimension is space (electrodes or sensors), the 2nd dimension is time and the 3rd and last dimension is trials. The analysis is performed electrode per electrode such that the data $\mathbf{Y}$ form a 2-dimensional $p \quad n$ matrix with p trials and n time frames (or time points). For each trial we define the experimental conditions by a 2 dimensional $p \quad m$ design matrix $\mathbf{X}$ with p rows (for trials) and m columns; each column codes for one condition or a covariate. In the current implementation, we consider each trial to be unique and therefore the model is similar to running a one-way ANOVA or ANCOVA. The model therefore follows (1) with $\mathbf{B}$ the estimated regression parameters (a $m \quad n$ matrix) and $\mathbf{E}$ the error term (a $p \quad n$ matrix). The solution of the normal equations is given by inverting $\mathbf{X}$. In practice we estimate the parameters following (2), by fitting simultaneously all frames, one electrode at a time, to obtain the parameters of the univariate model on the diagonal of the $\mathbf{B}$ matrix. Combining the columns of $\mathbf{X}$ (contrast weighting) allows testing for various effects at the individual level (t-tests, F-tests—for details see, e.g., [10])

$$\mathbf{Y} = \mathbf{XB} + \mathbf{E}, \tag{1}$$

$$\mathbf{B} = \operatorname{diag}\left(\left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{Y}\right) = \operatorname{diag}\left((\mathbf{X}) \quad \mathbf{Y}\right). \tag{2}$$

In LIMO EEG, the solution is given by (2) using a generalized Moore-Penrose pseudoinverse (pinv default function in Matlab [11, 12]). Thus, although the design matrices made up by LIMO EEG are almost always rank deficient (each condition is coded in one column of X), F or T tests are exact, that is, they give identical results to that obtained by applying a standard inverse to a full rank matrix.

2.2. Hierarchical General Linear Model for MEEG Data: 2nd Level. At the second level of analysis, beta coefficients from the different conditions (or their linear combinations) obtained from each subject are analyzed across subjects to test for statistical significance. Several robust methods have been implemented in LIMO EEG at this stage. Most of the techniques described below can be found in Wilcox [13] and correspond to tests performed under H1 to compute confidence intervals and under H0 to control for multiple comparisons. Compared to standard methods, robust methods provide better probability coverage for the confidence intervals and a tighter control of the type I error. Computations presented in this section are used in LIMO EEG to provide robust confidence intervals and uncorrected P values or a binary decision on significance. Computations for multiple comparisons correction are presented in the next section.

2.2.1. One-Sample t-Test. Whereas most ERP studies aim at comparing different experimental conditions, the GLM also allows testing for the covariation of single-trial ERPs with stimuli and cognitive factors (see e.g., Rousselet et al. [7] who tested the effect of image phase coherence across trials during a face discrimination task). A bootstrap-t approach in which subjects are drawn randomly with replacement is implemented in LIMO EEG. For each bootstrap, a one-sample t-test is performed on the bootstrap sample and the T value is stored. These T values provide an approximation of the T distribution under H0 and are used to estimate the alpha/2 and 1-alpha/2 quantiles. The P values are then simply the average number of times the T values obtained from the original data are above or below the bootstrapped

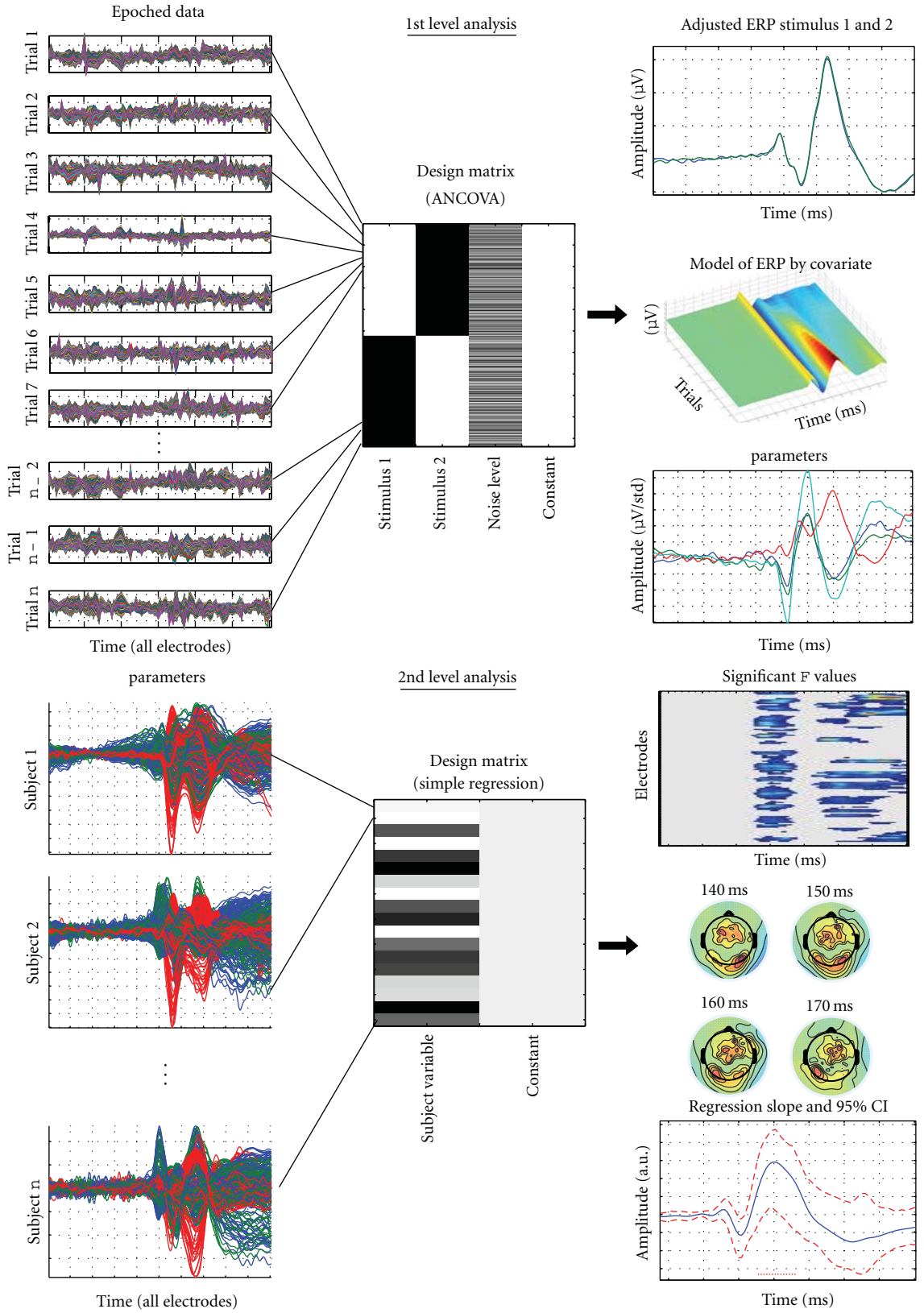


Figure 1: Illustration of the hierarchical procedure. At the 1st level of analysis (top), epoched data of each subject, comprising all trials, are analyzed to obtain the estimated beta parameters reflecting the effect of the various experimental conditions coded in the design matrix. Here the design is simplified from [7] and codes for the effect of stimulus 1, stimulus 2, and the noise level across all stimuli. At the 2nd level of analysis (bottom), the beta parameter(s) of experimental condition(s) coded at the 1st level are analyzed to test for significance across subjects. Here the 2nd level design matrix coded the subjects' age thus performing a regression of age on the estimated parameters that reflected the effect of noise level on visual evoked responses.

quantiles. Finally, confidence intervals around the mean can be computed following (3):

$$\text{CI} = \left[\text{Sample Mean} - T_{(U)} \quad \frac{s}{\text{sqrt}(n)};\right. \\ \left. \text{Sample Mean} - T_{(L)} \quad \frac{s}{\text{sqrt}(n)}\right], \tag{3}$$

where CI is the confidence interval, $T_{(L)}$ and $T_{(U)}$ are the critical T values obtained from the sorted bootstrapped T values with, $L =$ (alpha number of bootstraps/2) rounded to the nearest integer, $U =$ (number of bootstraps $- L$), s is sample standard deviation, sqrt(n) is the square root of the number of observations.

2.2.2. Two-Samples t-Test. To compare ERPs from two independent groups of subjects, we use a percentile bootstrap in which subjects from each group are sampled independently with replacement. For each bootstrap, we obtain 2 new independent samples and the mean difference between the two groups is computed. This method therefore tests differences under H1, that is, it tests that the mean of gp1 is different from the mean of gp2. After sorting these D differences in ascending order, confidence intervals take the values, $D_{(L+1)}$ and $D_{(U)}$ with L and U defined as above. If 0 is included in the CI, the difference between samples is not significant. Finally, the P value is the smallest value of either the averaged number of times the observed difference is above zero or, one minus this average.

2.2.3. Paired t-Test. The comparison of two sets of estimated parameters from the same group of subjects follows the procedure described for the two-sample t-test. However, because data are now paired, we sample subjects with replacement, to keep pairs of ERPs together and therefore preserve the intrasubject variance.

2.2.4. Regression Analysis. Regression analyses of ERP data allow assessing the inter-subject variability. Such variability is useful to test hypotheses about cognitive development, aging, various impairments, and individual differences in general (see e.g., Rousselet et al. [8, 9] for an example in normal aging). The method consists in sampling with replacement n matrices of ERPs (electrodes × time frames), n being the number of subjects. The link between subjects and predictors is maintained, so for simplicity we sample trial indices. The estimated regression parameters βs are computed for each bootstrap and sorted in ascending order. For a simple regression, 599 bootstraps are performed and the 95% confidence interval is $[\beta_{(a+1)}\ \beta_{(c)}]$, with a and c taking special values depending on the number of observations. For this simple case, 599 bootstraps have been shown to be enough to control the type I error rate [13]. For multiple regressions, a percentile bootstrap is used in conjunction with the Bonferroni inequality, and the confidence intervals are defined for each regressor as $[\beta_{(a+1)}\ \beta_{(c)}]$, with

$$a = \frac{(\text{alpha} \quad \text{number of bootstraps})}{(2 \quad \text{number of regressors})}, \\ c = \text{number of bootstraps-}a. \tag{4}$$

No P value can be computed with this technique but a binary decision on the statistical significance is obtained: a regression coefficient is significant if the confidence interval does not contain 0. Compared to other techniques, the modified bootstrap has been shown to perform well under heteroscedasticity but also if the data (subjects) are sampled from a nonnormal distribution [13].

2.2.5. ANOVA. Contrary to the other designs, only bootstraps under H0 are used to compute P values. The analysis relies on standard ordinary least squares (OLS) and, for repeated measure ANOVAs, sphericity is accounted for by a multivariate approach (Hotelling T^2 test for repeated factors and Hotelling generalized T^2 for within by between interactions; both transformed into F values). In a first analysis we obtain the observed F values. Then we build a data driven F table under H0. First we centre the data, independently for each group (N-way ANOVA) or condition (repeated measure ANOVA), so that each cell of the ANOVA has a mean of zero. Second, we use the centred data to estimate the F distributions under H0. We sample subjects with replacement, independently for each cell for N-ways ANOVAs, or keeping the association between observations in repeated measure ANOVAs. P values are obtained by sorting the bootstrap F values and counting how many times the observed F values are above the $F_{(U)}$ value. Using the same resampling as above but using the original data (i.e., under H1) we also compute the average difference between conditions allowing to construct robust confidence intervals using the techniques described for t-tests.

2.2.6. ANCOVA. The analysis of covariance follows the same strategy as the regression analysis: subjects' indices are sampled with replacement to keep data, group membership and predictors together. This resampling allows us to build robust confidence intervals around the predictors. Significance tests for the group differences and the covariate effects are obtained under H0. In this case, ERP data are sampled with replacement and fit to the original design matrix, thus breaking the relationship between the data and the predictors. We use this technique to estimate the distributions of the F values of group differences and covariates under H0. The P values are then obtained as for ANOVAs.

3. Multiple Comparisons Correction

Because tests are performed at many electrodes and time frames, multiple testing will give rise to a high number of false positives (type I error—see, e.g., Figure 2). This multiple comparison problem is independent of the type I error

rate obtained independently at one electrode and one time frame using the techniques describe above. Computations described above were performed mainly under H1 and used for robust confidence intervals and uncorrected P values. These techniques are complemented here by computations performed under H0, the null hypothesis of no effect, to correct for multiple testing.

This multiple testing problem is controlled in LIMO EEG using three methods, all relying on the same bootstrap procedure. For each technique described in the previous section (t-tests, regression, ANOVA, ANCOVA), we sample subjects with replacement under a true (t-tests, ANOVAs) or estimated (regression, ANCOVA) H0. This process is repeated B times and for each bootstrap we record (1) the maximum F value (= t^2 for t-tests) among all electrodes and time frames and (2) the maximum sum of significant temporal or spatial-temporal F clusters. These distributions of maximum Fs (Method 1) and maximum F clusters (Methods 2 and 3) under H0 can then be used to control the type I error rate across the entire data space [14].

Method 1 (maximum statistics). Uses the distribution of maximum bootstrap F (or t^2) values. The critical F value of the observed sample is corrected for multiple tests by using a probability distribution of the strongest F values obtained under H0 across all tests (across all electrodes and all time frames). This technique has the advantage of having an exact type I error rate [14]. However, this height threshold technique is conservative, similarly to Bonferoni and other familywise error corrections, because it is based on all the tests performed. One disadvantage of being too conservative is that, for instance, the size of a cluster of significant consecutive time frames will be smaller after correction (assuming extrema of a cluster have the lowest significant values before correction—see Figure 2) therefore possibly losing interesting information about the onsets and offsets of experimental effects. Another possibility is that this correction splits a cluster into smaller pieces because it does not take into account the spatial-temporal structure of the data. The second and third approaches use a correction based on cluster statistics and therefore overcomes this problem.

Method 2 (spatial-temporal clustering—2D). Uses the distribution of bootstrap clusters defined simultaneously in space and time (Figure 2). This clustering technique follows the philosophy presented in [6] and uses functions implemented in Fieldtrip (http://fieldtrip.fcdonders.nl/). An observed spatial-temporal cluster of F values is statistically significant if the sum of F values contained in the cluster is bigger than the threshold bootstrap cluster sum obtained under H0 (see e.g., [15, 16] for a similar approach with PET and fMRI data). Under H0, one can observe by chance clusters of significant electrodes and time frames. By recording the largest sum of cluster F values for each bootstrap, we can construct the distribution of the spatial-temporal cluster values under H0 and therefore test the significance of an observed cluster value. Because the H0 distribution is not specific to a particular location in space and time, this technique automatically controls for multiple testing. Note that at variance with the maximum statistics, the correction only applies to clusters already declared significant, making the cluster correction less conservative. Finally, because in MEEG a large effect in, for example, size (e.g., a P300 event) can mask a smaller one (e.g., N170), the control is not performed on the cluster size itself but on the sum of the F (or t^2) values inside each cluster [6]. The cluster sum statistics takes into account spatial extent and height information. Therefore, a spatially narrow cluster of effects around, for example, the N170, can survive the test by a greater density of F values.

Method 3 (temporal clustering—1D). Combines the cluster and maximum statistic approaches. For each bootstrap obtained under H0 we first take the largest temporal cluster value (sum of t^2 or F values) for each electrode and then only retain the largest one (Figure 2). By doing this for each bootstrap, we create an empirical distribution of temporal cluster values corrected in space. Again, an observed cluster will be significant if its sum is significantly bigger than the bootstrap threshold sum observed under H0. The advantage of this method over spatial-temporal clustering is the increased likelihood to reveal more spatially localized effects because temporal effects do not have to appear on groups of electrodes. It is also a convenient technique to test small groups of electrodes not necessarily spatially contiguous.

3.1. Bootstrap Computations under H0 for Multiple Comparisons Correction. The bootstrap procedures described here used the same resampling as before but often on centered data (H0 is thus true) and results are used to produce a corrected distribution (Method 1) or cluster distributions (Methods 2 and 3).

3.1.1. One-Sample t-Test. The bootstrap procedure used to adjust the individual type I error and construct robust confidence intervals for each electrode and each frame is performed under H1. The H0 version of this bootstrap consists in centering the data and then performing one-sample t-tests on centred data sampled with replacement. Because centered data have a zero mean, resampling allows us to measure variations around 0, the null hypothesis.

3.1.2. Two-Samples and Paired t-Tests. As for the one-sample t-test, the control of the individual type I error rates and CIs are calculated under H1 by computing differences between bootstrap group or pair samples. Therefore complementary tests under H0 are carried-out for each bootstrap. For each group or pair, data are centered and next resampled and t-tests computed. Because data are centered, no differences are expected (therefore testing under H0).

3.1.3. Regression Analysis and ANCOVA. Subjects are randomly sampled with replacement and data are fitted to same the design matrix. This procedure thus breaks the link between the data (subjects) and the model (design matrix), and therefore allows estimating the slope(s) of the various regressors under H0. The F values for the different regressors

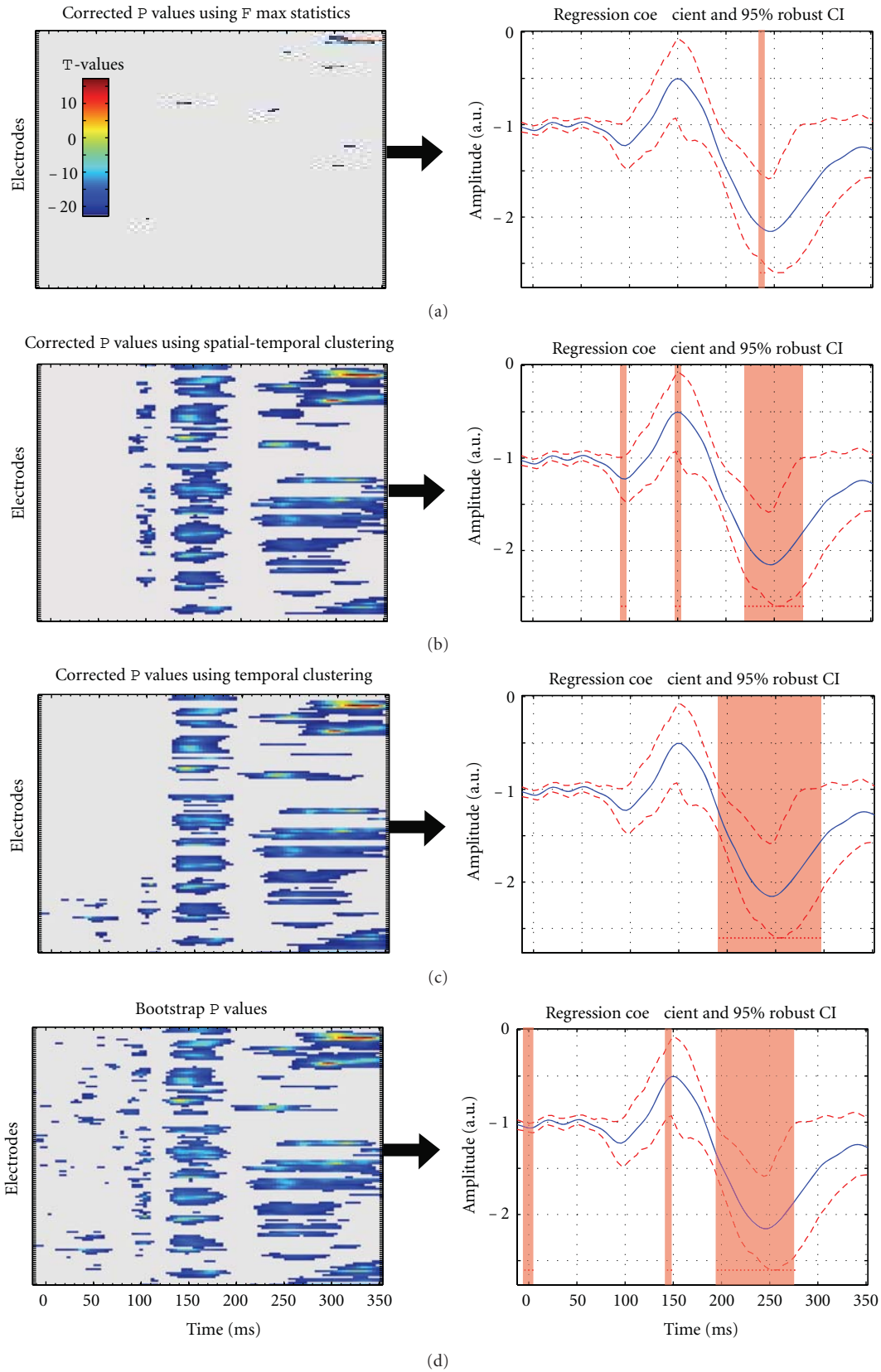


Figure 2: Illustration of the different multiple comparisons corrections (alpha 5%). At the top data are thresholded using a F max statistics (Method 1). In the middle, the same data are thresholded using spatial-temporal clustering (Method 2) or temporal clustering (Method 3). At the bottom, data are presented without any correction but a strict type I error rate (5%) for each electrode and frame separately is applied. Note that each method gives slightly different results.

or for the group effect in ANCOVAs are recorded for each bootstrap and used to compute the empirical distributions used to correct for multiple comparisons.

3.1.4. ANOVAs. Here only H0 computations are performed by centering each "cell" (each group or each condition—see above). Again, recording the F values for each effect at each bootstrap allows correcting for multiple comparisons using one of the methods described above.

4. Validation

4.1. Low-Level Functions. In order to test the validity of the code, all statistical functions (except bootstrap and multiple comparison procedures) have been tested against Statistica and the relevant information to use each function by itself is available in a downloadable document (validation_of_the_stats.pdf) on the LIMO EEG server. For each statistical test, several low dimentional data sets have been generated and analyzed using both LIMO EEG and Statistica to ensure that LIMO EEG returns the correct T, F, and P values. Because of the high dimensionality of MEEG data, such analysis can not be easily carried out by standard packages and the need for multiple comparisons correction does change the statistical significance of the effects. However, this simple testing allowed us and future users to easily test the low-level statistical functions of the toolbox and be certain of our implementation. Of importance, some tests return slightly different results. The main difference can be observed for the 2 sample t-test. Most software (e.g., Statistica & SPSS) assume variance homogeneity by default, which is fallacious because independent groups are likely to have different variances. LIMO EEG always assumes variance inhomogeneity thus returning slightly lower t values. The alternative ANOVA (limo_old_rep_anova.m) which is not accessible via the interface also returns slightly different values. By default, LIMO EEG computes repeated measures ANOVA using a Hotelling T^2 test to account for sphericity. However, a standard F test can also be computed by changing one parameter when calling the random_robust.m function. In this case, sphericity is accounted for by a Hyund-Feld correction. The correction value is different from that of Statistica or SPSS, which use the initial formulation [17], whereas our implementation follows the modified, corrected formula [18].

4.2. Multiple Comparisons Correction. Permutation combined with max cluster statistics have been shown to control in theory for multiple comparisons, maintaining the probability to commit a least one type I error across the entire search space at the nominal alpha level [6]. However, permutation has not yet been validated systematically in MEEG research. Thus, despite indications that permutation performs well under certain conditions involving the comparison of two groups [6], its performance remains to be tested more generally, and its application extended to other experimental designs. Bootstrap techniques are more versatile than permutation and have been developed to address many problems in psychology [13]. For instance, it is not clear how to implement a permutation test for an ANCOVA design; whereas a bootstrap test is easy to implement. Hence, bootstrap techniques offer more possibilities to MEEG researchers. However, bootstrap techniques, and their capacity to control the type I error rate, have not yet been validated in MEEG research, which is a limitation of our toolbox. Nevertheless, we report encouraging preliminary results suggesting that bootstrap techniques perform similarly to permutation in some conditions. We compared the familywise type I error rate of permutation and two bootstrap techniques associated with max cluster statistics in t-tests for independent samples.

Our simulation uses the 18 subjects of the dataset provided with LIMO and each subject was used as a "population" of about 1000 trials. This dataset is ideal to validate tests of differences under H0, because it contains ERP amplitudes spanning the whole continuum from face responses to noise responses. Thus, for each subject, we sampled with replacement from the total number of trials for that subject 100 trials twice to form fake condition 1 and fake condition 2 (level 1). Then we applied 3 tests on these 2 fake conditions. Each test involved 1000 random samples. In the first two tests, the 200 trials were pulled together and two sets of 100 trials were created either by random partitioning (permutation test), or by sampling with replacement (bootstrap test). Both tests estimate H0 by random resampling. In the third test, each group of 100 trials was mean centred and bootstrap samples with replacement drawn independently from each of them (technique implemented in LIMO and validated in [13]). For each test, and for each random sample, a t-test was performed to compare the groups of trials, followed by spatial-temporal clustering of F values (squared t values). We cluster the F values because a t-test is a special case of linear contrast, which is evaluated using an F statistics. Also, an F statistics is used for all the other GLM designs. Then we saved the maximum F cluster sum, and obtained a distribution of max cluster sums under H0, which was used to assess the significance of the original t-tests. So far we have conducted level 1 H0 analyses 200 times on each subject. The average number of positive tests is the type I error rate, after correction for multiple comparisons using cluster statistics. Across 18 subjects, and using 200 simulations, the type I error rate for permutation is 0.0506, with minimum 0.025 and maximum 0.085 across subjects. The type I error rate for the bootstrap test is 0.0489, min = 0.025, max = 0.08. The type I error rate for the bootstrap test with data centering is 0.0453, min = 0.025, max = 0.08. These results are very close to the nominal alpha results of 0.05. More simulations and more situations will need to be tested to compare precisely the behaviour of these techniques.

5. Graphical User Interface

LIMO EEG can be called directly in the Matlab command window or via the EEGLAB menu. It comes as a fully functional graphical user interface (GUI). Each of the main steps have there own GUI: General GUI (Figure 3(a)), import of epoched data and 1st level analysis (Figure 3(b)), 2nd

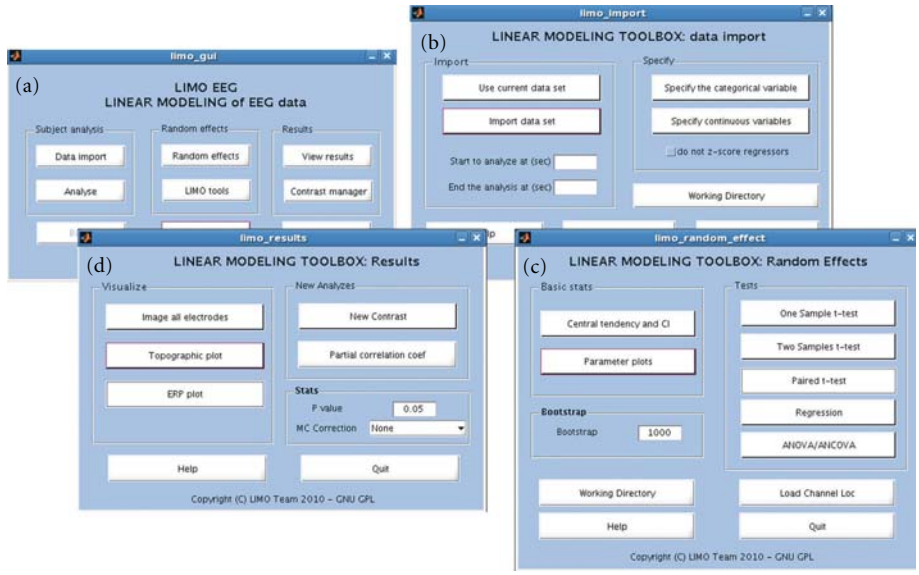


Figure 3: The four main GUI of LIMO EEG. All functions and plots are available via these user interfaces.

level analysis (Figure 3(c)), and visualization (Figure 3(d)). User do not have to call functions or type anything in the command window, everything can be obtained via interface. Each time a help button is also available for a description of each option in each GUI. In addition, we made available a data set on the LIMO EEG server which comes with a tutorial explaining how to analyse the data using there interfaces. A short example of results is given in the next section.

6. Application to EEG Data and Visualization Tools

In this section, we present some results from an analysis performed on 18 subjects to illustrate the various formats in which group data can be explored and presented. This data set is downloadable as a tutorial for LIMO EEG and results represent simplified analyses of what is presented in [8]. In short, subjects of various ages discriminated between pictures of two faces, face A and face B; the noise level in the images was varied parametrically (actually a manipulation of the phase of the image). Using such design one can therefore test for differences between ERP to the two faces using a paired t-test, test for an effect of the noise level using a one-sample t-test, or test for an age effect on ERP noise sensitivity using regression analysis.

6.1. 1st Level Analysis. For each subject we create a design matrix including face A, face B and the level of image noise (see Figure 1 top). The data are thus modelled as a weighted sum of three predictors (face A, face B and phase coherence effect) plus a constant and an error term.

6.2. 2nd Level Analysis. Using the estimated parameters from each subject one can test several effects. First, we looked for differences between faces A and B using a paired t-test (no significant effects, $P = .05$ corrected with spatial-temporal clustering) by entering into the analysis the estimated beta parameters for face A and face B from all subjects. As illustrated in Figure 4 (panel A), face stimuli evoked a typical ERP (A1, tools are provided to plot robust ERPs across subjects, here the average of 20% trimmed mean ERPs with 95% CI obtained using the bootstrap standard error) and no significant differences can be observed (A2). Second, we investigated a possible effect of the stimulus phase coherence on the visual evoked response. This was performed using a one-sample t-test ($P = .05$ corrected with spatial-temporal clustering) by entering the estimated beta parameters corresponding to this predictor. As illustrated (Figure 4—panel B), image phase coherence affects the evoked brain responses from 80 ms onward (full space/time map—B1) mainly over posterior lateral and central electrodes (topographic plot of F values—B2) with the strongest effect observed on electrode C1 between 110 ms poststimulus onset and 290 ms (B3). Finally, we also investigated an effect of age on ERP phase sensitivity by performing a simple regression with age as covariate (Figure 4 panel C, $P = .05$ corrected with spatial-temporal clustering). This analysis could be performed over the whole scalp by taking the same physical electrodes across subjects (data are presented Figure 2). The analysis can also be performed using an optimized electrode [19]. This strategy consists in selecting the electrode that shows the strongest model fit, so that we compare functionally similar electrodes across subjects. In this case, the analysis of the age effect on ERP sensitivity to noise was performed on the electrode that best modelled the data in each subject, as defined by the strongest R^2 (C1). This feature of LIMO EEG allows more flexibility in the way one combines data for group analyses. Here, results show the ERP sensitivity to image phase coherence is significantly modulated by age from 200 ms to 330 ms post stimulus onset (C2).

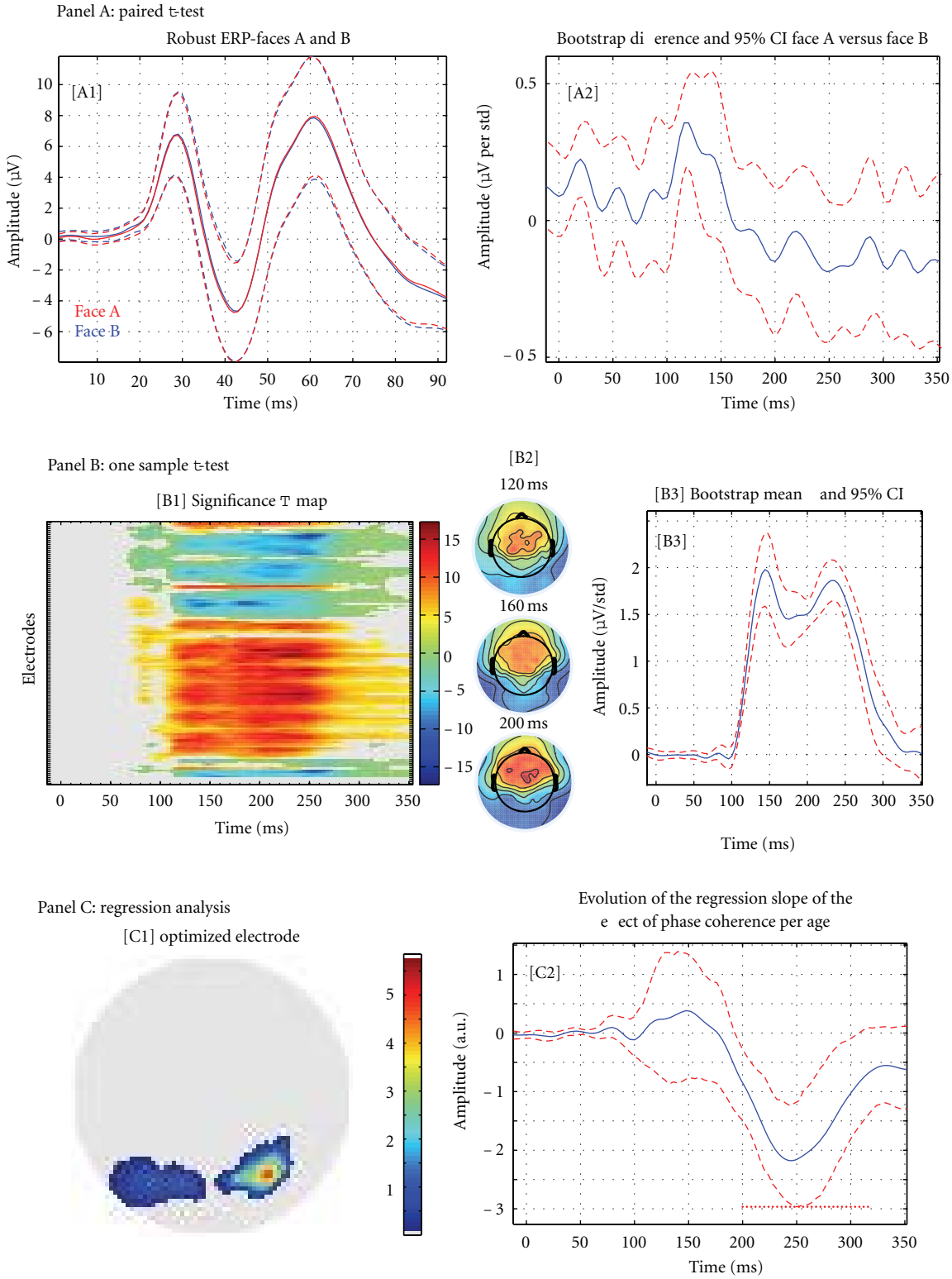


Figure 4: Examples of analyses/results obtain with LIMO EEG. Panel A presents results from a paired t-test between Face A and Face B (A1: average of the 20% trimmed means ERPs for face A and face B; A2: mean difference of estimated beta parameters and robust 95% confidence intervals). Panel B presents results from a one-sample t-test performed on the phase coherence regressor (B1: map of significance over all electrodes and frames; B2 topographic projection of the T values around the N170 event, B3: mean beta parameter (phase coherence) and robust 95% confidence interval for the electrode showing the strongest effect). Panel C presents results from the regression analysis of age on the phase coherence regressor (C1 map for the optimized electrode, that is, the map represents the location of the electrode chosen for each subject and the number of subjects; C2 results of the regression analysis, that is, plot the regression slope of the effect of phase coherence (estimated beta parameters) per subject age).

7. Discussion

7.1. Pros and Cons of a Massive Univariate Approach in MEEG. LIMO EEG relies on a massive univariate approach in which, like PET or fMRI, all possible measurements (voxels or electrode/time frames) are analyzed. This provides many advantages but also elicits some problems. On the positive side, the massive univariate approach is relatively easy to understand as it uses standard statistics, it is fully automatic, accommodates any design, and provides a full picture of electromagnetic events without having to hand pick electrodes or time frames. On the negative side, strict controls of statistical tests need to be implemented because of the multiple tests performed. Also, because analyses are performed on independent electrodes and time frames, one can miss more subtle effects that might develop over time or space, and would be picked up by multivariate [20] or multidimensional [21] approaches. However, the down side of these latter approaches is that they are much harder to interpret.

7.2. Robust and Parametric. In LIMO EEG, as in any parametric statistics package, we assume data come from a type of probability distribution, and makes inferences about the parameters of these distributions. In LIMO EEG, we assume that data come from a normal or nearly normal distribution, and make an inference about the mean values. Another important feature of LIMO EEG is the use of robust statistics. Here "robust" is used in the sense that the techniques implemented in LIMO EEG show overall more power than traditional tests when assumptions (e.g., normality) are violated and when experiment effects exist (H1) thus providing better probability coverage, especially when estimating confidence intervals. Using those techniques, we also ensure a tighter control of the type I error rate (H0). Our preliminary simulation results (18 times 200 Monte-Carlo) show that using 1000 bootstraps, the mean type I error rate of our 2 samples t-test is 0.0453, demonstrating that the cluster technique for multiple comparison correction offers a good control on false positives. Further simulations are needed to adequately test the type I error rate in various situations (designs/population) but this demonstrate, in principle, the validity of our method.

In LIMO EEG we limited the scope of most analyses to samples' means via bootstrap. In fact, robust statistics allow analyzing data using various distribution estimators other than the mean. The mean is not necessarily a good estimator of the central tendency of the data, and trimmed means, median, and M-estimators can provide more satisfactory results [22, 23] (there are trimmed means options in LIMO EEG and a few stand-alone functions to do, for example, t-tests on trimmed means). However, none of these estimators have been validated for MEEG data yet, hence the restriction to samples' means.

One current limitation of our parametric approach is that first level analyses, and the GLM designs at the second level, currently rely on an OLS solution. Ideally, one can make regressions more robust using weighted least squares (WLS). However, the problem of WLS is the computation of the covariance matrix. If one wants to properly estimate how trials/conditions (1st level) or subjects/conditions (2nd level) covary, new methods must be investigated in order to account for the spatial and temporal link between data points and not merely the covariation between conditions/subjects at each time point separately. Until such a method is available, an OLS solution seems the safest option.

7.3. Current Limits and Future Development. There is no real limit to the current implementation of LIMO EEG because it allows analyzing almost all kinds of designs. Limits are only related to various statistical aspects that deserve consideration. One current limit concerns the 1st level of analysis: all conditions are treated independently, which effectively corresponds to a 1 way ANOVA or a 1 way ANCOVA. However, experimental conditions could also be grouped in order to create a factorial design, thus pooling some variances together to account for interaction effects. Although our approach is valid because the estimated parameters of each condition can be combined via contrasts to reflect main effects and interactions as in a factorial design, it is likely to limit some analyses. Therefore, future versions of the toolbox will incorporate factorial variance pooling. A second limitation is the use of OLS. As mentioned above (*Robust and parametric section)* one would ideally use a WLS solution to allow non independence and heteroscedasticity between conditions. However current mathematical solutions do not exist to properly estimate the covariance matrix and until then the 1st level estimates will not be "robust".

8. Conclusion

Overall LIMO EEG provides a set of statistical tools allowing the analysis of many designs via GUI. It provides robust results which are unbiased by the selection of peaks or components. It also provides a new way to analyze data with an emphasis on effect size (robust confidence intervals), which we hope will help moving the field toward a more quantitative analysis of evoked neural responses [7].

Acknowledgments

This work is partially funded by ESRC: ESRC Grant RES-000-22-3209 supported G. A. Rousselet, ESRC Grant RES-062-23-1900 supported C. R. Pernet and G. A. Rousselet. Cyril R. Pernet is also funded by the SINAPSE collaboration—http://www.sinapse.ac.uk, a pooling initiative funded by the Scottish Funding Council and the Chief Scientist Office of the Scottish Executive.

References

[1] A. Delorme and S. Makeig, "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis," *Journal of Neuroscience Methods*, vol. 134, no. 1, pp. 9–21, 2004.

[2] S. J. Kiebel and K. J. Friston, "Statistical parametric mapping for event-related potentials: I. Generic considerations," *NeuroImage*, vol. 22, no. 2, pp. 492–502, 2004.

[3] S. J. Kiebel and K. J. Friston, "Statistical parametric mapping for event-related potentials (II): a hierarchical temporal model," *NeuroImage*, vol. 22, no. 2, pp. 503–520, 2004.

[4] J. M. Kilner, S. J. Kiebel, and K. J. Friston, "Applications of random field theory to electrophysiology," *Neuroscience Letters*, vol. 374, no. 3, pp. 174–178, 2005.

[5] J. Kilner and K. J. Friston, "Topological Inference for EEG and MEG," *Annals of Applied Statistics*, vol. 4, pp. 1272–1290, 2010.

[6] E. Maris and R. Oostenveld, "Nonparametric statistical testing of EEG- and MEG-data," *Journal of Neuroscience Methods*, vol. 164, no. 1, pp. 177–190, 2007.

[7] G. A. Rousselet, C. R. Pernet, P. J. Bennett, and A. B. Sekuler, "Parametric study of EEG sensitivity to phase noise during face processing," *BMC Neuroscience*, vol. 9, article 98, 2008.

[8] G. A. Rousselet, J. S. Husk, C. R. Pernet, C. M. Gaspar, P. J. Bennett, and A. B. Sekuler, "Age-related delay in information accrual for faces: evidence from a parametric, single-trial EEG approach," *BMC Neuroscience*, vol. 10, article 1471, p. 114, 2009.

[9] G. A. Rousselet, C. M. Gaspar, C. R. Pernet, J. S. Husk, P. J. Bennett, and A. B. Sekuler, "Healthy aging delays scalp EEG sensitivity to noise in a face discrimination task," *Frontiers in Perception Science*, vol. 19, no. 1, 2010.

[10] K. J. Friston, J. Ashburner, S. J. Kiebel, T. E. Nichols, and W. D. Penny, *Statistical Parametric Mapping: The Analysis of Functional Brain Images*, Academic Press, New York, NY, USA, 2007.

[11] E. H. Moore, "On the reciprocal of the general algebraic matrix," *Bulletin of the American Mathematical Society*, vol. 26, pp. 394–395, 1920.

[12] R. Penrose, "A generalized inverse for matrices," *Proceedings of the Cambridge Philosophical Society*, vol. 51, pp. 406–413, 1955.

[13] R. R. Wilcox, *Introduction to Robust Estimation and Hypothesis Testing*, Academic Press, New York, NY, USA, 2nd edition, 2005.

[14] T. Nichols and S. Hayasaka, "Controlling the familywise error rate in functional neuroimaging: a comparative review," *Statistical Methods in Medical Research*, vol. 12, no. 5, pp. 419–446, 2003.

[15] J. B. Poline and B. M. Mazoyer, "Analysis of individual positron emission tomography activation maps by detection of high signal-to-noise-ratio pixel clusters," *Journal of Cerebral Blood Flow and Metabolism*, vol. 13, no. 3, pp. 425–437, 1993.

[16] J. B. Poline, K. J. Worsley, A. C. Evans, and K. J. Friston, "Combining spatial extent and peak intensity to test for activations in functional imaging," *NeuroImage*, vol. 5, no. 2, pp. 83–96, 1997.

[17] H. Huynh, "Some approximate tests for repeated measurement designs," *Psychometrika*, vol. 43, no. 2, pp. 161–175, 1978.

[18] R. S. Chen and W. P. Dunlap, "A Monte Carlo study on the performance of a corrected formula for $\tilde{\varepsilon}$ suggested by Lecoutre," *Journal of Educational and Behavioural Statistics*, vol. 19, no. 2, pp. 119–126, 1994.

[19] J. J. Foxe and G. V. Simpson, "Flow of activation from V1 to frontal cortex in humans: a framework for defining "early" visual processing," *Experimental Brain Research*, vol. 142, no. 1, pp. 139–150, 2002.

[20] K. J. Friston, K. M. Stephan, J. D. Heather et al., "A multivariate analysis of evoked responses in EEG and MEG data," *NeuroImage*, vol. 3, no. 3, pp. 167–174, 1996.

[21] J. Onton and S. Makeig, "Why use ICA to decompose EEG/MEG data?" in *Event-Related Dynamic of Brain Oscillations*, C. Neuper and W. Klimesch, Eds., Progress in Brain Research, 159, Elsevier, 2006.

[22] R. R. Wilcox and H. J. Keselman, "Modem robust data analysis methods: measures of central tendency," *Psychological Methods*, vol. 8, no. 3, pp. 254–274, 2003.

[23] G. A. Rousselet, J. S. Husk, P. J. Bennett, and A. B. Sekuler, "Time course and robustness of ERP object and face differences," *Journal of Vision*, vol. 8, no. 12, 2008.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 938925, 14 pages
doi:10.1155/2011/938925

Research Article

Ragu: A Free Tool for the Analysis of EEG and MEG Event-Related Scalp Field Data Using Global Randomization Statistics

Thomas Koenig,[1] Mara Kottlow,[1] Maria Stein,[1] and Lester Melie-García[2]

[1] *Department of Psychiatric Neurophysiology, University Hospital of Psychiatry Bern, University of Bern, 3000 Bern 60, Bolligenstr. 111, Switzerland*
[2] *Neuroinformatics Department, Cuban Neuroscience Center, Havana 15202, Cuba*

Correspondence should be addressed to Thomas Koenig, thomas.koenig@puk.unibe.ch

Received 30 September 2010; Revised 7 December 2010; Accepted 31 December 2010

Academic Editor: Sylvain Baillet

We present a program (Ragu; Randomization Graphical User interface) for statistical analyses of multichannel event-related EEG and MEG experiments. Based on measures of scalp field differences including all sensors, and using powerful, assumption-free randomization statistics, the program yields robust, physiologically meaningful conclusions based on the entire, untransformed, and unbiased set of measurements. Ragu accommodates up to two within-subject factors and one between-subject factor with multiple levels each. Significance is computed as function of time and can be controlled for type II errors with overall analyses. Results are displayed in an intuitive visual interface that allows further exploration of the findings. A sample analysis of an ERP experiment illustrates the different possibilities offered by Ragu. The aim of Ragu is to maximize statistical power while minimizing the need for a-priori choices of models and parameters (like inverse models or sensors of interest) that interact with and bias statistics.

1. Introduction

Scalp field measurements represent activity of electrically active extended neural generators in the brain and offer a unique window to measure human information processing noninvasively and with a high time resolution. Today, EEG and MEG recording systems can record human scalp field data with high density in space (>100 sensors) and time (>1000 Hz), which improves the resolution of the results. However, the understanding of effects observed on the scalp has been severely hindered by the so-called inverse problem of EEG and MEG measurements, which prevents in the general case that effects observed on the scalp can be unambiguously attributed to a specific set of brain tissue. As a consequence, many of the results found in the literature depend at some point on some implicit or explicit model, and since these models vary considerably, unambiguous conclusions across studies and models are often difficult to draw.

The aim of the current paper is to present methods and software that allow users to analyze event-related scalp field data using methods that incorporate the physical underpinnings of scalp electromagnetic data but are model independent. The software should enable researchers to assess the significance of ERP effects globally, and without need of a-priori assumptions about the correct model, (i.e., about the location of "active" or "inactive" sensors, or about the correct parameters for a source model). Evidence for an effect based on such unbiased statistics can then entail further more model-based analyses implemented in other tools. In the remainder of the paper, we will develop the methodological background of the procedure, followed by a brief description of the software implementation, a sample analysis to illustrate the procedure, and a discussion of the implications.

2. Methodological Background

The physics that relates the intracranial brain-electromagnetic activity to the extracranial sensors is summarized by the so-called leadfield or forward solution of the EEG/MEG [1]. The leadfield of the EEG and MEG defines weighting factors

that linearly translate the activity of a given source to scalp potential differences; these weighting factors depend on the sensors position, the location and orientation of the source, and eventually (for EEG data) on the geometry and electromagnetic properties (i.e., conductivity, homogeneity) of the different tissues between source and sensor. The leadfield is a smoothing operator that blurs the measurements in space [1] and introduces correlations among the sensors depending (among other factors) on the distance between them. As a result, there are three important facts to take into account when dealing with EEG/MEG scalp field data:

(1) the activity of even a point-like source will produce a field that extends across the entire scalp, such that most sensors will pick up a signal from that source;

(2) a single sensor can pick up signals from many different and eventually remote sources;

(3) for the case of EEG, since all measurements are potential differences, the signals recorded at a given electrode are always dependent on, at some point, an arbitrary choice of reference.

In our opinion, a major part of the publications that have employed scalp potentials to investigate the effects of some experimental manipulations have not taken these facts sufficiently into account, such that the interpretability of the obtained results is seriously limited. As a consequence, the impact of ERP studies is probably below the original potential of the measured data.

2.1. Statistical Assessment of Topographic Effects for One Time Point. In general, the aim of a statistical comparison of scalp field maps between two or more conditions at a given time point is to test whether some of these conditions consistently differed in active sources. Interestingly, such arguments can be made without estimating the location of those sources. This is so because scalp fields are additive; if two sources are active at the same moment in time, the data measured is the sum of the two scalp fields produced by the two sources. This implies that we can also interpret the difference of scalp fields observed during different conditions. This difference scalp field is identical to the scalp field of those sources that were different between the two conditions. (All sources that were identical in the two conditions cancel out when the difference is computed).

In order to test whether some conditions differ in active sources, it is thus sufficient to show that there are scalp field differences between these conditions, and that these differences are unlikely to have occurred by chance. To avoid any biases, this evidence can be based on quantifying the overall amount of difference of activity, that is the overall strength of scalp field differences. Once such a quantifier is available, it can be used to test the measurements against the null hypothesis. The suggested quantifier and the suggested statistical testing rely on previously reviewed and published papers [2, 4, 5] and are briefly explained below.

A global and well-established quantifier of scalp field strength is the Global Field Power (GFP, [6, 7]). As shown in formula (1), the computation of GFP is analogous to the computation of the standard deviation across all sensors

$$\mathrm{GFP} = \sqrt{\frac{\sum_{j=1}^{n}\left(v_j - \overline{v}\right)}{n}}, \tag{1}$$

where v_j is the voltage measured at sensor j, n is the number of sensors, and $\overline{v}$ is the mean measurement across all sensors. The GFP can be shown to be reference independent. Given that the sensor array covered a sufficient part of the scalp, using the GFP of scalp field differences to quantify the effect of an experimental manipulation is thus compatible with the three important facts about EEG/MEG scalp data mentioned in the introduction:

(i) since all sensors are taken into account, the scalp field produced by difference source(s) is taken into account to its largest possible extend (no problem with fact 1);

(ii) since all sensors are being used, false negatives based on partially overlapping scalp fields are unlikely (no problem with fact 2);

(iii) there is independence of the reference (no problem with fact 3).

As previously outlined [3], the usage of the GFP of a difference map can be generalized to cases with more than two conditions and/or two or more groups using a global measure s of scalp field differences as defined below

$$s = \sum_{i=1}^{c}\sqrt{\frac{\sum_{j=1}^{n}\left(\overline{v}_{ij} - \overline{\overline{v}}_j\right)^2}{n}}, \tag{2}$$

where c is the number of conditions and group, n is the number of sensors, $\overline{v_{ij}}$ is the voltage of the grand mean across subjects of condition and/or group i at sensor j, and $\overline{\overline{v}}_j$ is the grand mean across subjects and conditions of the voltage at sensor j. All data has to be against the average reference [3].

If instead of a group/condition membership a predictor is available that is assumed to be linearly related to the activity of an unknown set of sources, the scalp field produced by this set of sources can be estimated using the so-called covariance maps β_j [4]. Given a set of scalp field maps v_{ij}, where i is the index of the observation and j is the sensor, and given for each map v_{ij} the predictor b_i, the covariance map of v_{ij} and b_i is given as follows:

$$\beta_j = \sum_{i=1}^{m} v_{ij} \cdot b_i. \tag{3}$$

As mentioned above, the quantification of the overall strength of the sources that account for the predictor b_i, the GFP of the covariance map can be used. In this case, s is defined as

$$s = \sqrt{\frac{\sum_{j=1}^{n}\left(\beta_j - \overline{\beta}\right)^2}{n}}, \quad \text{where } \overline{\beta} = \sum_{j=1}^{n}\beta_j. \tag{4}$$

As argued in previous papers [3], the value of s depends on the amplitudes of the mean differences among conditions and/or groups, and on some random variance across subjects and conditions. For the assessment of the significance of an effect, we are interested in whether the value of s is solely due to random variance across conditions and/or groups, or whether it is at least partially caused by a certain consistency of an effect across the measurements obtained in the different groups and/or conditions. This can be tested by randomly shuffling the group and/or condition assignments in each subject and recompute s. The resulting value of s will then depend only on the random variance across subjects and conditions, but an eventual consistency of differences among groups and/or conditions across subjects (i.e., an effect of group and/or condition) has been eliminated by the randomization procedure. Any value of s obtained after random shuffling is thus an instance of s under the null hypothesis, stating that some differences are due to noise alone. By repeating the random shuffling and computation of s many times, one can obtain an estimate of the distribution of s under the null hypothesis and compare the value of s in the real data against this distribution. The significance of the effect, that is the probability of the null hypothesis, is then given by the percent of randomly obtained values of s that are larger than or equal to the value of s obtained with the real data. In the literature, the procedure to compare groups and/or conditions has been called TANOVA (topographic analysis of variance); if a linear predictor is used, the proposed term is TANCOVA (topographic analysis of covariance).

In general, nonparametric randomization statistics as those described above are known to have similar statistical power as classical parametric tests if the assumptions made by the parametric tests hold, and have better power otherwise [8]. One important additional point has, however, to be taken into account when applying randomization statistics, which is exchangeability. Exchangeability means that the distribution of the effect sizes remains the same after the shuffling. This may become a problem in fMRI data, where the spectrum of the physiological data is at or below the spectrum of the experimental design [9]; for EEG and MEG, this is, however, not a problem, because the events to be analyzed are typically very short, and instantaneously measurable at the sensor level. Furthermore, it is obvious that the number of observations sets a limit to the number of possible permutations, which set a natural lower limit on the possible level of significance.

2.2. Statistical Assessment of Significance across a Time Interval. In ERP experiments, it is often not a priori clear at what latency window an effect can be expected, and the analysis needs to explore the data across many time frames. This may obviously inflate the possibility of false positive findings due to multiple testing, and some test for the overall significance of an effect is necessary. In previous papers [2, 3], we have proposed to obtain such indices of overall significance by estimating how likely it was that the overall count of significant time points (at a given threshold of significance) could have been observed by chance, or how likely it was that the observed duration of a period of significant effects would have been observed by chance.

If randomization statistics have been computed, such overall statistics can be directly derived from a further analysis of the results of the randomization runs. Following the description in Koenig and Melie-Garcia 2010, we illustrate the procedure for the overall count of significant time periods. First, a threshold for significance is chosen, and the count of the number of significant time points in the data is established, which will serve as the overall measure of effect size. As before, this effect size needs to be compared to the distribution of the count of false positives assumed to occur under the null hypothesis.

In the present case, we can estimate the distribution of the count of false positives from the randomization runs. We assess, for each randomization run r and time point t, a "pseudo-P-value" P defined as the percentage of cases where the measure s of r was larger than the measures s obtained in the remaining randomization runs. For a given randomization run r, we thus obtain P values at each moment t, and we can establish the count of P values that are lower than the above chosen threshold of significance. This count is thus an instance of the number of time points with P values below the critical threshold while the null hypothesis still holds on a global level. If this count of false positives is assessed for all randomization runs, an estimate of its distribution under the null hypothesis is obtained. The count of significant time points obtained in the original data is then compared against this distribution, yielding an overall estimate of the significance of the difference. This test is similar to cluster-size statistics used in MRI data analysis and have been described elsewhere [9].

For the assessment of significance of the duration of an effect, the analogous procedure can easily be inferred.

2.3. Data Normalization. For the interpretation of significant differences between two or more scalp fields, it is sometimes useful to make a distinction between two specific cases. In one case, the distribution of the active intracranial sources is the same in all conditions, and the differences among conditions can be explained by a scaling factor that is common for all these active sources. Functionally, one would thus interpret such a difference as a quantitative difference of activation in the presence of apparently similar brain functions. Because of the above-outlined linear relation between intracerebral sources and scalp field measurements, the same argument can be made if the measured scalp fields differ merely by a scaling factor that is common for all sensors. If (and this is the alternative case) differences between scalp fields cannot be solely explained by a scaling factor common for all sensors, the active intracerebral sources must have had at least a partially different location and/or orientation, which can be considered as a qualitative difference and indicates that at least partially different brain functions have been recruited. In order to distinguish these two cases, the program offers the possibility to normalize the variance of the scalp fields across sensors before the statistical tests are computed. This eliminates the effect of potential differences in scaling, such that significant results obtained

with normalized data can be taken as evidence of qualitative difference, or evidence for the recruitment of at least partially different brain functions. Therefore, to complete an analysis that was based on normalized data, it is thus suggested to run separate univariate statistics on the spatial variance of the scalp field measurements, which is identical to an analysis of the Global Field Power (GFP) [6, 7] of the data. GFP analyses are currently not implemented in the software but will follow.

2.4. Visualization of Scalp Field Differences. Mean scalp field differences between two conditions or groups can easily be displayed using difference maps. If more than two conditions need to be compared simultaneously, this gets, however, increasingly complex, because the differences between all possible pairs all may have a different spatial distribution. A classical way to deal with such problems is multidimensional scaling (MDS) that allows to downscale high-dimensional result spaces into lower dimensional ones that can be easier visualized. Based on a matrix of similarities among all observations, multidimensional scaling represents each observation as a point in a lower-dimensional space, such that the closeness of the observation points optimally represents the original similarities.

In the current case, the number of sensors defines the original amount of dimensions of the result space; this has to be reduced to a two-dimensional space in order to be displayed on a computer screen. The similarities between the mean scalp fields of the different conditions and/or groups can be assessed using the covariance between these maps. In this case, the two-dimensional space that optimally represents the entire matrix of covariances is spanned between the first two eigenvectors obtained from this covariance matrix [10, 11]. For the purpose of visualization, each mean scalp field is projected onto these two eigenvectors, which yields the two-dimensional coordinates of each mean different scalp field in this optimized two-dimensional result space. These coordinates of the different scalp fields are then displayed as points in a scatterplot. If two points are relatively close, this indicates that the corresponding scalp fields were relatively similar; if two points are relatively far apart, the scalp fields were relatively different. The direction of the difference between two points in the scatterplot and the scalp distribution of the firsts and second eigenvectors furthermore give an approximate account of the distribution of the scalp field difference between the two corresponding scalp field distributions.

3. Implementation

The program presented here implements the above described statistical procedures for the statistical comparison of event-related EEG and MEG multichannel scalp field data across a broad range of experimental designs. It is called Ragu (RAndomization Graphical User interface), making an allegation to the preparation of a ragout. A good ragout is obtained by slowly cooking many different ingredients until they are undistinguishable; this cooking is similar to the programs' procedure of increasing the data's entropy by randomizing until its constituents form an unstructured mixture.

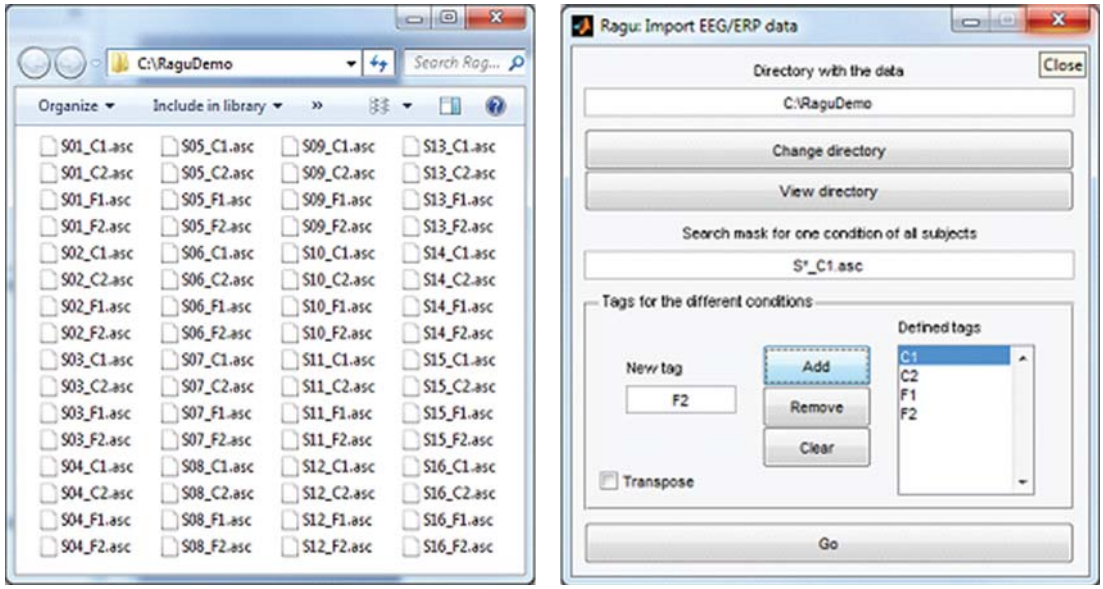
The program offers the possibility to compute these statistics either time point by time point, or on data averaged over some specified time interval. If time point by time point statistics are used, it further offers to compute the above-described overall statistics that prevent problems of multiple testing across time. Once the randomization statistics have been computed, the program displays all the effects (main effects and interactions) as line graphs showing the probability P of the null hypothesis as a function of time. If duration threshold statistics have been applied, periods of significance exceeding the critical duration are additionally marked. Results can then be interactively explored by displaying the mean scalp fields belonging to those within or between factors that constitute an effect. Additionally, these mean that scalp fields are displayed using multidimensional scaling.

Apart from the procedures described above, Ragu serves as a platform for the implementation of further statistical tools, such as microstate statistics. However, since these methods still await validation, an independent review, and publication, they are not further discussed here.

Ragu was developed under Matlab (http://www.mathwork.com/) and Windows 7. The program is available in the form of a downloadable standalone Matlab graphical user interface compiled for Windows using MS Visual Studio 2005; a Matlab license is therefore not necessary. The source code can be made available upon request, and the Matlab background should ensure cross-platform portability. The program is freeware; we attempt, but do not guarantee, support for bugs and questions that are not obvious from the manual or the papers. We, however, request users who publish results based on the output of the program to quote some of the conceptual papers [2–4] or the current paper.

The program uses standard, plain text-based ASCII input files that contain time x sensor matrices of scalp field potentials. This should avoid problems of incompatible data format but offers little control over possible mistakes in channel sequences and so forth. Checking the correctness of the imported data is therefore part of the user's responsibilities.

The program allows saving and loading previously imported data, definitions of designs, and obtained results. The program always saves the entire information to standard Matlab files: The data, the analysis parameters, and the results are thus always within the same container, ruling out uncertainties about what results have been obtained with what data and parameters. These files contain a structure with all the information used by the program. Users with Matlab skills can open these files in Matlab (V7.10 or above) and extract or modify the data according to their needs, but care must be taken not to corrupt the internal consistency of the information or false results may be obtained. Furthermore, Ragu can save and load Matlab figures files (V6.0 or higher); users with Matlab skills can thus use these figure files as basis for their figures. Output to metafiles and bitmaps is also available, as well as a tab-delimited text output to be used with spreadsheet applications.



(a) (b)

Figure 1: Ragu data import. (a) shows the directory containing data to be imported. The first 3 characters code the subject ("S01", "S02",...), characters 6 and 7 ("C1", "C2", "F1", and F2) code for the 4 conditions. The dialog (invoked by Data -> Import command) with the parameters to import the data is shown on the right side. With the provided search mask, a list of the expected file names of one condition and all subjects is constructed (note that the search mask must find exactly one among all conditions). This list is then extended to the remaining conditions using the specified tags. Thus, as a crosscheck, one of the tags for the conditions has to appear somewhere in the search mask. Then, all the data is read based on the expected file names. If the importing of the data was successful, a confirmation is given informing the user about the dimensions of the imported data matrix. Otherwise, the output window displays error messages that may help identifying the problem.

4. Usage and Sample Analysis

4.1. Installation and Update. The Ragu installation package can be downloaded at http://www.thomaskoenig.ch/Ragu_pkg.exe. This package contains the Ragu program, the installer of the required Matlab runtime library, and a history of the changes made to the program across time. The installation of the runtime library is necessary only once, to later install newer versions of Ragu, it is sufficient to download and run the file http://www.thomaskoenig.ch/Ragu_pkg_NoMCR.exe, that is much smaller. Upon request to the first author, users can be put on a mailing list that alerts you whenever a new version of the program has been compiled and uploaded. The source code is also made available upon request.

4.2. Sample Data. As an example, Ragu has been applied on a dataset from Stein et al. [12]. For this experiment, 16 healthy English speaking exchange students living in Switzerland for the duration of one year were recruited. Subjects were recorded twice, once at the beginning of their stay when they had basic German language skills (day 1) and in the middle of their stay with improved German language skills (day 2). During the experiment, they read German sentences with either semantically correct (The wheel is ROUND) or false (The garden is SHY) endings. It is known from previous studies that the violation of the semantic expectancy generated by the first part of the sentence (The garden is) produces an ERP scalp field called N400 in response to the last word (SHY) which is proportional to the degree of violation of the individual semantic expectancy [13].

The data analyzed here consists thus of four conditions: sentences with correct endings at day 1, with false endings at day 1, and with correct and false sentence endings at day 2. This represents a two-factorial design consisting of the factors "day" (day 1 and day 2) and "expectancy" (correct or false). The EPRs were recorded from 74 scalp locations with a 250 Hz sampling rate, were low-pass filtered at 8 Hz, and lasted from the onset of the last sentence word to 1000 ms after stimulus. Additionally, all subjects performed language tests at day 1 and day 2; thus, an overall score of language proficiency increase from day 1 to day 2 was available.

4.3. Data Import. Ragu stores all scalp field data to be analyzed internally in a single four-dimensional matrix (number of subjects x number of conditions x sensors x time points). To import data, the user has to provide, for each condition and subject, a plain ASCII file with only the measurements, one row for each time point, and one column for each sensor. The naming of the files has to be such that each file contains a tag that is unique for each subject, and a tag that is unique for each condition; the remaining parts of the filename must be identical for all files. All files have to be in the same directory (also if there are several groups of subjects), and missing data is not allowed.

According to Figure 1, in our example, we search the different files with S _C1.asc and define 4 conditions: C1 (correct sentence ending day one), C2 (correct sentence

Figure 2: Specification of the within-subject design of the sample analysis in Ragu. In the upper left list box, all conditions are shown. For each factor, the levels of a condition can be set with the + or – button. To choose which factor to define, the "Set" buttons of the two possible factors are used. Additionally, the factors and factor levels can be labelled, and conditions can be excluded.

ending day two), F1 (false sentence ending day one), and F2 (false sentence ending day two).

Once the data have been successfully imported, one can optionally specify further parameters such as the sampling rate and the latency of the event onset, and the montage (the possible formats are simple and specified in the online help), which helps for the later interpretation of the results. After the data and its additional parameters have been defined, it is recommended to briefly verify with the View->View data command whether the program represents the data as expected.

4.4. Defining and Understanding Within-Subject Designs. The experimental design is specified separately for within- and between-subject factors. Within subjects, it is possible to define up to two factors, and each factor can have several levels. If two factors are defined, the levels of the two factors must be orthogonal. Figure 2 shows the dialog where users can enter the within-subject design of their experiment (invoked by Design->Within Subject Design). Users can name each factor and assign a label to each level of each factor, which will help for the later interpretation of results using multidimensional scaling. It is also possible to exclude some conditions from the analysis for post-hoc comparisons.

As visible in Figure 2 and introduced before, our sample consists of the factor "expectancy" containing the two levels "correct" and "false" and the factor "day" with the levels day 1 and day 2.

Once all the data has been imported, the data parameters have been set, and the within-subject design has been defined, the program is ready to compute the corresponding TANOVA. For these computations, a number of options are available (Analysis->Randomization options).

Most importantly, and as discussed above, it can be specified if and how the data is normalized before the statistics are computed. If the L2 norm of the raw data is chosen, each individual scalp field of each condition is scaled to unity variance. This is the recommended type of normalization. For backward compatibility, it is also possible to normalize on the level of group/condition grand means; this is invoked by choosing dissimilarity [7] for normalization.

Furthermore, the number of randomization runs can be chosen. The recommended number for an accurate estimate of the significance at the 5% level is 1000 runs, for the 1% level, it is 5000 runs [8], but as the computation of so many runs is lengthy, lower number can also be sufficiently informative for exploratory purposes.

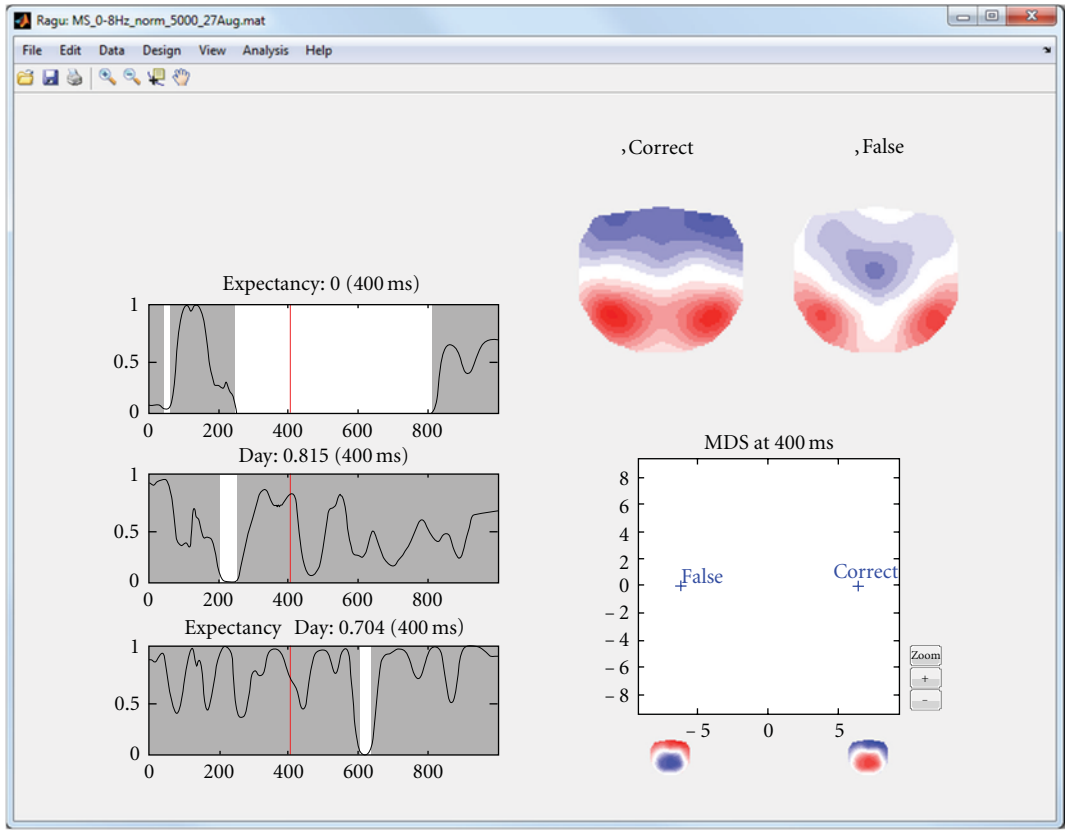
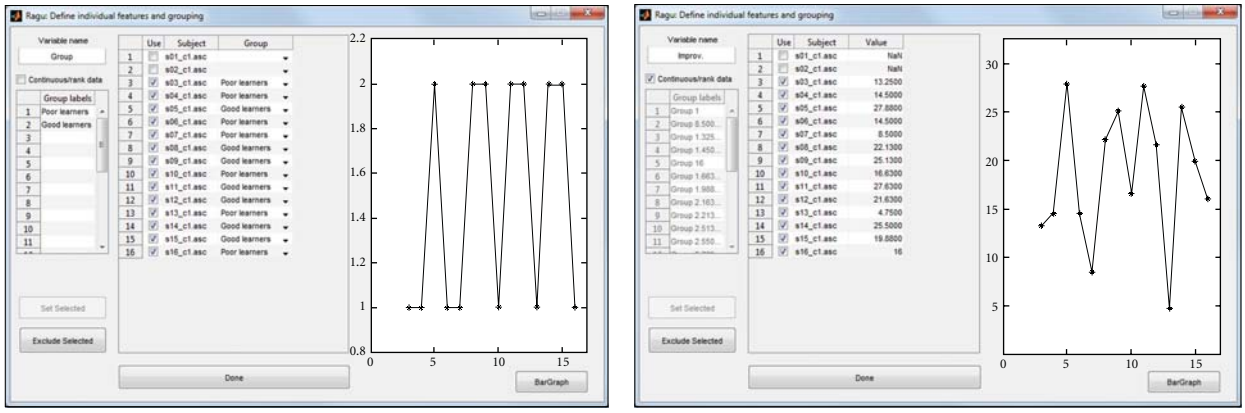


Figure 3: Display of the results of the analysis of the within-subject factors in the sample dataset. The left part of the display shows the significance of the TANOVA's main effects (expectancy and day) and their interaction as line graphs showing the probability P (y-axis) of the null hypothesis as a function of time (x-axis). Significant time points ($P < .05$) are marked in white. By clicking in a graph, a cursor is set, the P value of the respective time point is displayed besides the graph title, and the effect is mapped on the right side of the display. In the current figure, the main effect of expectancy (correct versus false sentence endings) at 400 ms is displayed. The upper right part of the display shows the mean topographic maps of all factor levels from the graph where the cursor has been set. In the present display of the main effect of expectancy, these are the mean maps across subjects and days of correct and false sentence endings at 400 ms. For the figure in the lower right part, these two mean maps have been fed into an MDS analysis. For this purpose, all mean maps were submitted to a spatial PCA. The x-axis of the figure represents the projection of mean maps onto the first eigenvector. The spatial distribution of the eigenvector is represented by two topographic maps below the x-axis. The graph indicates that the "false" condition is more negative and the correct condition is more positive at parietal electrodes.

Finally, it is possible to adjust the threshold for the acceptance of significance; this affects the display of results and the statistics on temporal cluster-size thresholds.

The first analysis of the sample data that we presented above is based on a purely within-subject design; all subjects are expected to show comparable effects, and no between-subject factor has been defined. After running a TANOVA with this design, the program displays a graph with the significance of each within-subject factor as a function of time (main effects), and the interactions of the factors (Figure 3, left part). In the case of our sample dataset, we therefore obtain a main effect of day, a main effect of expectancy, and an interaction of expectancy and day. The user can click into these graphs; this will display the obtained P values at the selected time period. In addition, the mean scalp field distributions of all groups and factor levels that form part of the effect are shown (Figure 3, right part). Finally, those mean scalp field distributions are submitted to a multidimensional scaling and projected upon the first two resulting eigenvectors. Those projections are shown in a scatterplot as shown in Figure 3 (lower right part). This scatter plot allows an intuitive first interpretation of an effect; the further two points are apart, the larger the difference among the corresponding mean maps is. Figure 3 illustrates the display based on the sample data.

4.5. Defining and Understanding Between-Subject Designs. Apart from being able to investigate up to two within-subject factors, it is also possible to define a between-subject design to run analyses that account for individual or group differences. When invoking the between-subject design dialog (Design->Between Subject Design), the program lists the data files of all subjects of one (arbitrary) condition, and the user can assign each subject to a specific group. Alternatively, when checking the "Continuous/rank data" box, each subject can be assigned an individual value that quantifies some



(a) (b)

Figure 4: Specification of the between-subject design of the sample analysis in Ragu. This figure shows the mask for the definition of the between-group design. The variable name appears later on the output of the results. As seen in both examples, no behavioural measures exist for subjects 1 and 2. These subjects are excluded from the analyses by unchecking the "use" checkbox. The line graphs on the right of each example show the value filled in for each subject. (a) The division of subjects in a group of low language proficiency improvement and a group of high language proficiency improvement. The values 1 or 2 are given to each subject as shown in the line graph. (b) For the computation of a TANCOVA, the "continuous/rank data" box has to be checked. Then, the value of proficiency increase from day one to day two can be entered individually. The line graph shows the level of increase of each subject.

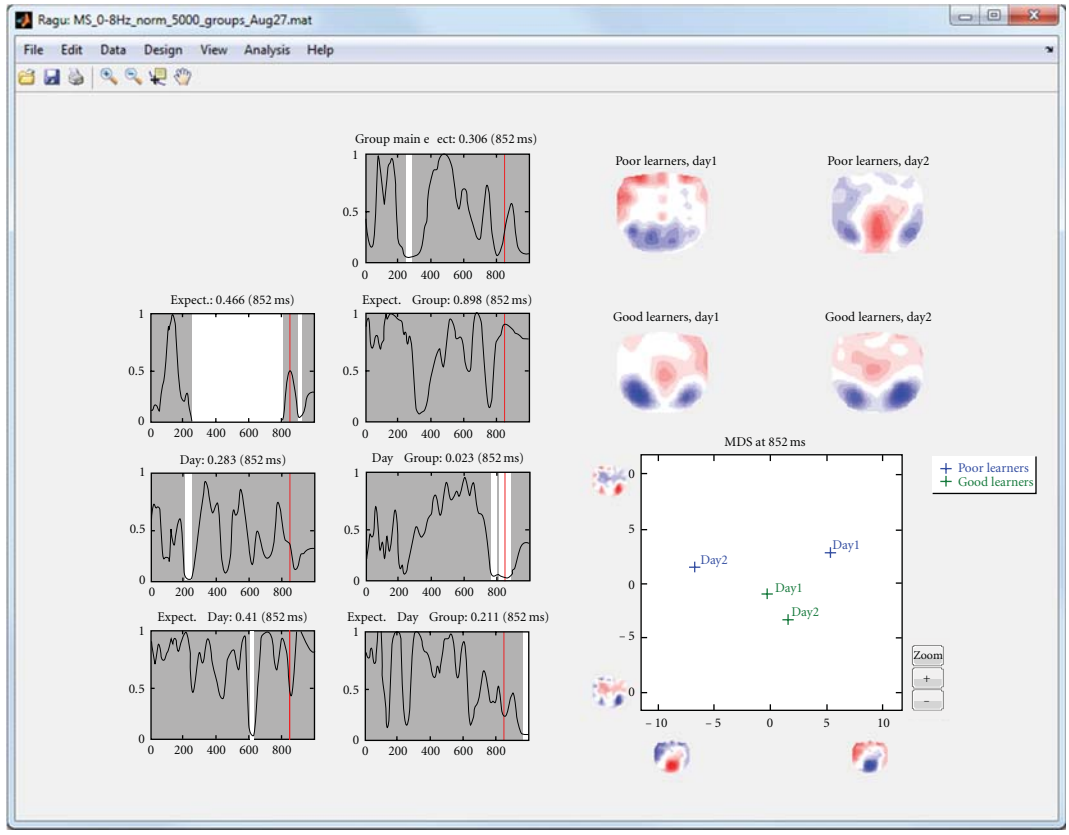


Figure 5: Display of the results of the analysis of the sample dataset when subjects were divided into two groups with low and high language improvement. The left-most row of line graphs shows the same information as the line graphs of Figure 3 (except for 2 less subjects); the additional line-graphs show all effects including the new factor group. The interaction day by group shows a significant effect around 850 ms. Because there are more than two mean maps, the MDS figure now also contains a y-axis that shows the projection of the mean maps onto the second eigenvector. The MDS indicates that the day by group interaction is mainly due to the differences between days 1 and 2 in poor learners; good learners show a much smaller change. The result suggests that from day 1 to day 2, poor learners have changed in late, feedback-related processing, while good learners have maintained their initial processing strategies. The same conclusion is also deducible from the mean maps shown in the upper right part of the display.

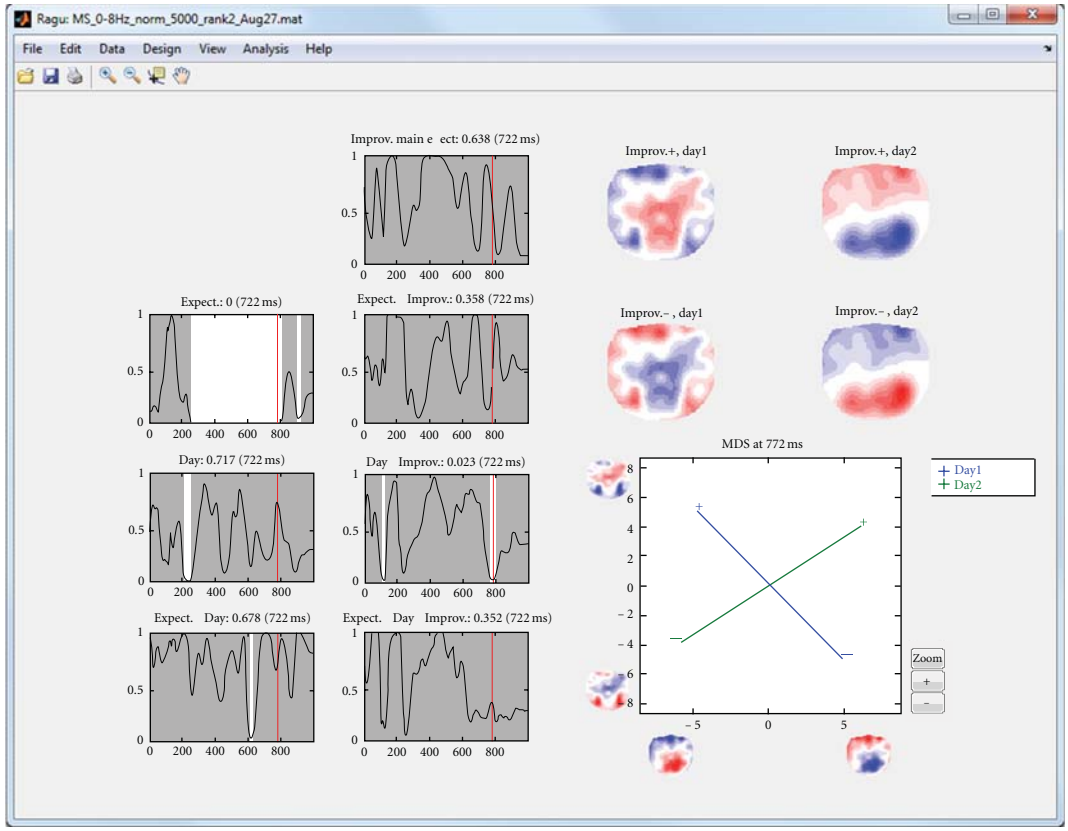


Figure 6: TANCOVA of the ERPs with the increase of language proficiency. The second row of graphs shows the same information as in Figure 5, with the exception that the effect of language improvement has been taken into account as a linear predictor. Again, the interaction day by improvement shows a late effect now somewhat before 800 ms. The mapping of this interaction on the right side shows the positive and negative covariance maps separately for days 1 and 2. These covariance maps are again entered into an MDS. It appears that there is an almost an orthogonal relation among the language improvement and ERP topography on day 1 and day 2 at this time range.

interindividual factor. This factor has to be interval or rank scaled and will be considered as covariate for a TANCOVA [4]. If necessary, individual subjects can also be excluded from the analysis.

In the following analysis of the sample data, we divided the subjects into a group with above median German proficiency increase from day 1 to day 2 ("good learners") and a group with below median German proficiency increase (Figure 4(a)). When computing a TANOVA, in addition to the effects already known from the pure within-subject analysis (Figure 3), we obtain the interaction of group membership with day, the interaction of group membership with expectancy, and the triple interaction of group membership, day, and expectancy (Figure 5).

The output of this group ANOVA shows an effect of day group in a late time interval around 800 ms.

Alternatively, instead of subdividing the subjects into groups based on their performance, it is possible to investigate whether there is evidence for components that are linearly related to performance across subjects. This approach is called TANCOVA and is also available in the program. By checking the "continuous/rank data" box in the between-subject design dialog, the individual performance (learning rates in the present example) can be entered (Figure 4(b)), and the program will compute a TANCOVA.

In our sample, we investigated whether the ERPs at day 1 have a predictive value for the increase in language proficiency from day 1 to day 2. Figure 6 shows the results of computing moment-by-moment TANCOVAs of the ERPs with the increase of language proficiency.

4.6. Further Statistics. Using a global measure of differences across all channels eliminates the problem of multiple testing across sensors, but since the previous analyses have been conducted time point by time point, false positive results may have been obtained due to multiple testing across time. To protect against these, it is possible to compute statistics on the overall count of significant time points and the duration of significant effects as discussed above. Figure 7 shows and explains how such overall thresholds can be obtained for the duration of continuous periods of significance of the time point by time point analyses. For the simplicity of the example, we used only the correct sentence endings in this analysis. In Figure 8, the obtained duration threshold has been applied to the TANOVA results.

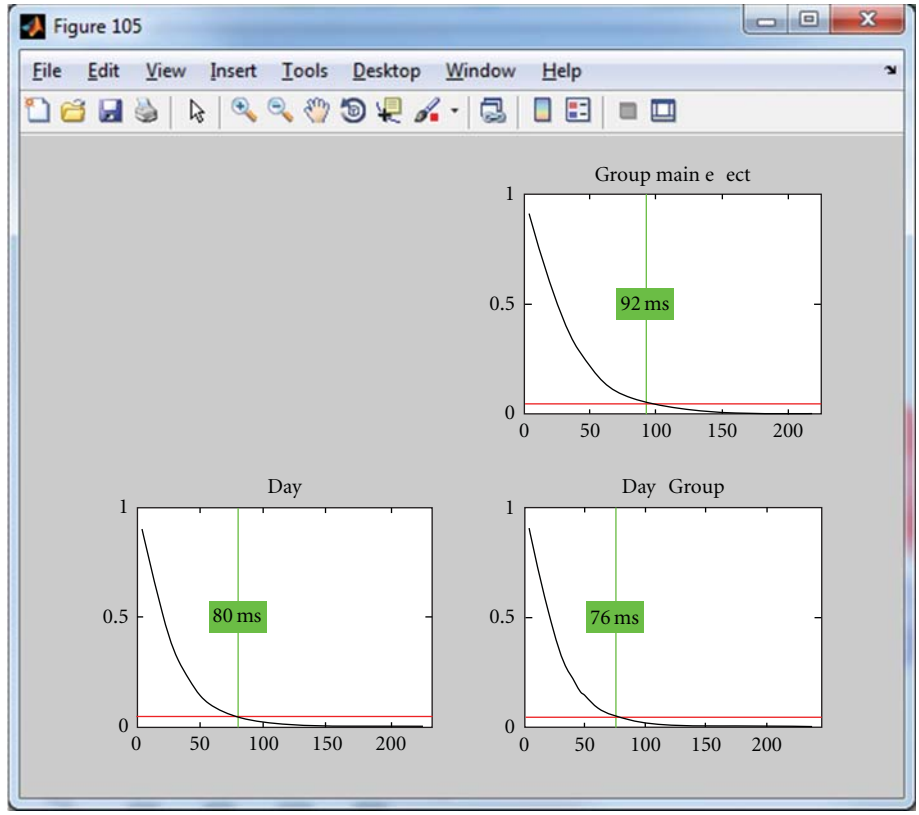


Figure 7: Estimation of a duration threshold of significant TANOVA effects. The threshold is estimated for each effect separately. The horizontal axis indicates the duration of continuous epochs with local significances of the TANOVAs below the selected threshold. The vertical axis indicates the probability of encountering a certain effect duration under the null hypothesis. These durations are obtained by "testing" the results of the randomization runs against each other [2, 3]. The red line indicates the chosen threshold for overall significance, in the current case, $P = .05$. The vertical green line indicates the duration that is longer than $(1 - P)$ percent (in the present case 95%) of all randomly obtained effect durations under the null hypothesis. This threshold can then be applied to the previously obtained TANOVA results. Thresholds have been computed based on the result of the time point-wise sample TANOVA analysis of Figure 8.

If there is an a-priori hypothesis about a time window where some effect should be tested, one can also compute the analyses outlined above based on topographies averaged across a time interval. As an example, we took the results of the group analysis with the factors day, expectancy, and group as described above. Based on the results of the cluster duration test (Figure 8), we wanted to know whether the effect is consistent across time points and stable if we average the signal over the time points between 780 and 890 ms. We thus computed the TANOVA again over the averaged time frame. The results are displayed in Figure 9. They show that the effect is indeed stable across time points as it persists when averaging.

In addition, and independently of comparisons among groups and conditions, the program contains a module to compute the topographic consistency test (TCT, [2]) that assesses, for each group and condition, during which time points there is evidence for a consistent pattern of active sources across subjects (invoked with Analysis->Topographic Consistency Check); a detailed explanation of this method is found in [2]. This test can optionally be computed at the beginning of the analysis to define the analysis window.

As Figure 10 shows, there is evidence for common activations across subjects over prolonged time periods. The first period of consistent topography lasts until about 600 to 700 ms and continues after an interruption until the end of the data. It is also evident that the significance level of the test is inversely related to the GFP of the ERP.

4.7. Summary of Results. The results of our sample analysis showed an interaction effect of expectancy day from 600 to 650 ms, mainly due to the difference of topographies of correct word endings from day 1 to day 2. Additionally, in the group analysis, an interaction effect day group from around 780 to 890 ms was seen. This interaction effect was mainly due to the change of topographies from day 1 to day 2 in the group of bad learners. Since we saw that the interaction expectancy day was due to differences in correct

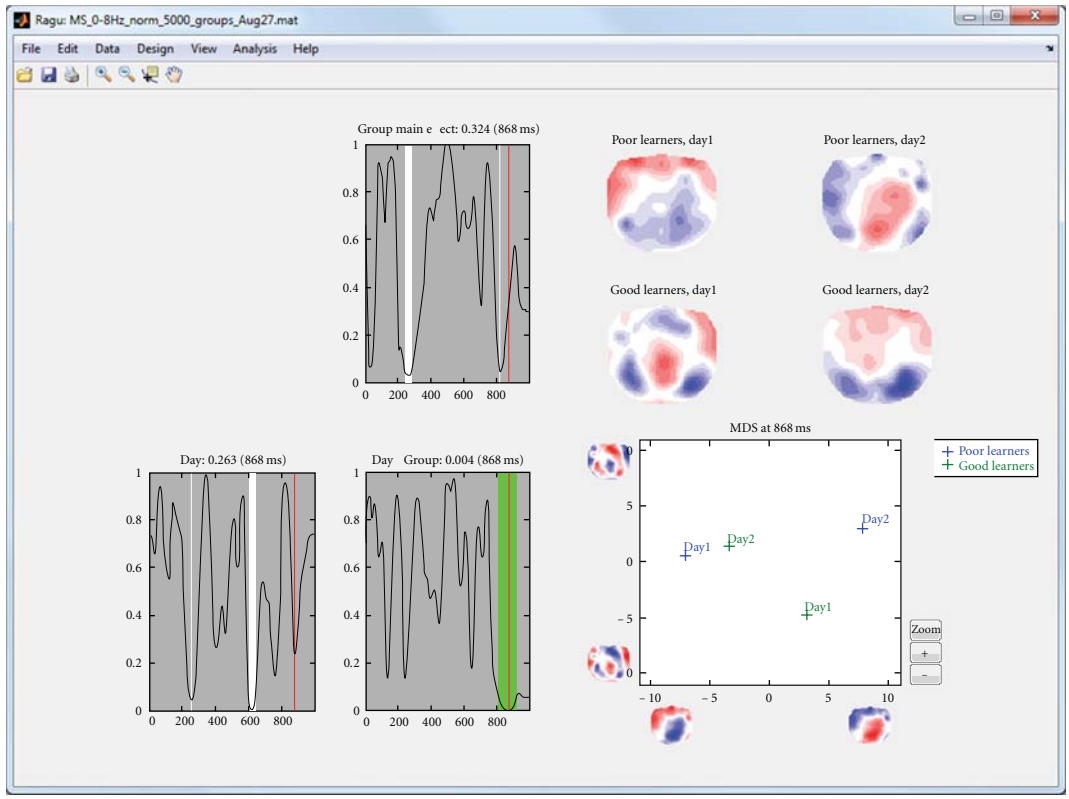


Figure 8: Results of a TANOVA computed for correct sentence ending on days 1 and 2, with good and poor learners as groups. The duration threshold estimate from Figure 7 has been applied. Periods meeting the duration criterion are shown in green. The late group by day interaction meets the duration criterion, whereas the other effects do not.

word endings, we assumed that this may also play a role in the interaction effect day group. Thus, bad learners should show a change of processing of correct words from day 1 to day 2.

We tested our assumption as formulated above in a new design with the factors group and day, with day containing only correct sentence endings at day 1 and day 2. This TANOVA resulted in a more stable interaction effect in the same time frame as indicated by the cluster size test. Computing this new TANOVA again averaged over the important time frame resulted in a significant interaction group day, indicating that the effect is stable and consistent over the respective time points.

Finally, the consistent topography test supported our results showing that the processing duration of correct sentence endings was shorter at day 2 than at day 1, whereas the duration of the consistent topography did not differ between false sentence endings on day 1 and day 2.

This sample highlights the advantage of an analysis without the need of a-priori decisions. With a-priori choices we would have limited the analysis to search effects around 400 ms due to previous studies reporting about the N400 effect. We would have missed the results found around 800 ms mainly due to different topographies in response to correct sentence endings.

5. Discussion

In the current paper, we present software designed to compute statistical analyses on scalp field data using methods and algorithms based on randomization techniques that are custom tailored to the specific properties and problems of such data. The methods, user interface, and display of the results implemented in the program should accommodate most of the experimental designs that maintain an acceptable degree of complexity (two within-subject factors with multiple levels each, and one between-subject factor, also with multiple levels). The paper is thought as an introduction for researchers using EEG/MEG data that want to understand the basic concepts of the methods and make use of the software. For a more thorough discussion of the underlying concepts, we refer to other publications [2–4].

In terms of the "flow" of an analysis of event-related scalp field data, the methods and tools presented here offer a good starting point, but typically not the end point of an exhaustive analysis of a data set. The main advantage of beginning an analysis with the methods proposed here is that they offer robust, powerful, and physiologically meaningful statistics on the entire, untransformed, and unbiased set of measurements. Thus, without the need to select sensors,

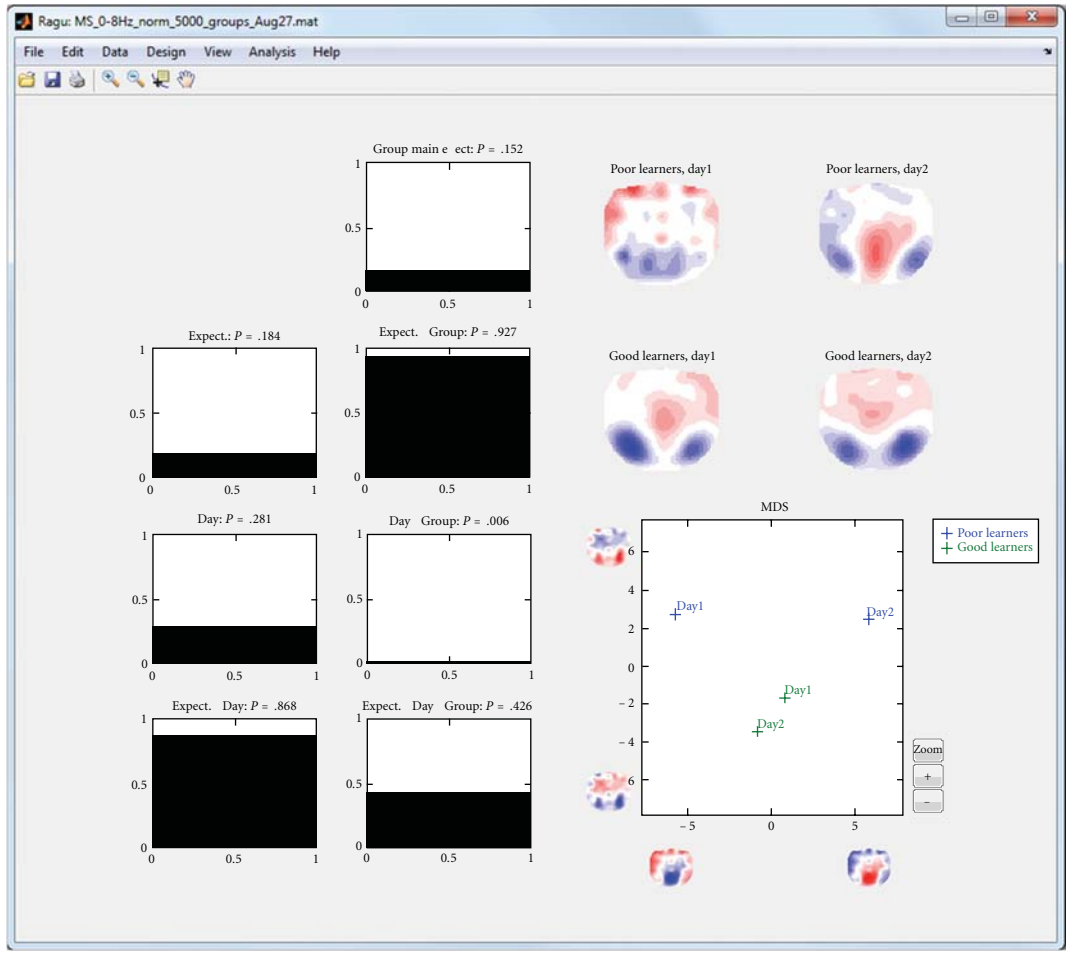


Figure 9: Group TANOVA over averaged time points between 780 and 890 ms. When averaged across multiple time points, the graphs on the left each shows significance levels for the whole averaged time span as bar graphs instead of line graphs. The y-axis shows again the level of significance (P), which is now written above each graph for the whole time span.

time windows of interest, type and parameters of inverse solutions, or other a-priori choices, the data informs the researcher about whether and when there is a significant effect of some experimental manipulation. At the same time, significance indicates that the conditions and groups involved in the effect activated at least partially different sources and thus assumingly different brain functions. Once such a global statistical basis has been established, the data can be further manipulated to be explored more locally in sensor or inverse space. In other words, we hope that the methods and tools introduced here can help to minimize the dependence of statistical evidence from a-priori choices of specific models.

A further remark to be made here is on the general difference of assumptions when doing statistics on the scalp compared to the source level. Consistent scalp fields indicate consistent source localization and source orientation, while source orientation is typically not considered in voxel-wise statistics of inverse solutions. As argued before [3], source orientation appears to be a very robust and sensitive feature of ERP data; all results obtained by averaging evoked scalp potentials imply that not only the amplitude of the sources of the evoked potential was constant, but also their orientation. The interpretation of what a consistent change of orientation means remains less clear. On the other hand, statistics based on inverse solutions obviously depend on the correctness of the assumptions of the inverse model. Furthermore, at least for distributed inverse solutions, the result space is drastically inflated without an increase of degrees of freedom of the data, and heavy corrections for multiple testing across voxels need then to be applied post-hoc to correct for this somewhat artificial problem.

A disadvantage of the program (common to all programs that are based on randomization and resampling techniques) is that computation time increases linearly with the amount of randomization runs, which can make computation time lengthy for larger datasets. In its current implementation, the program runs as a single-thread process, such that it

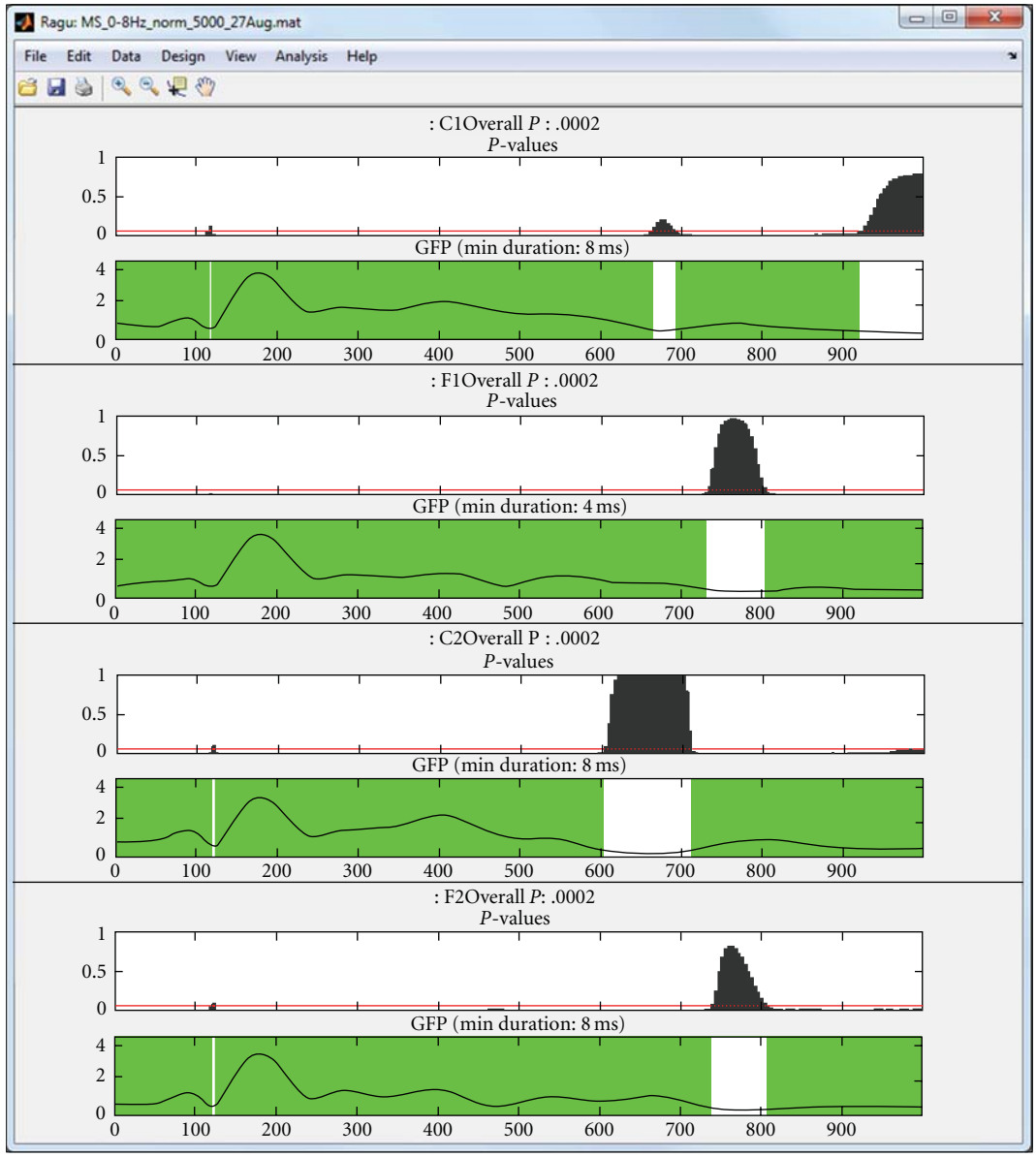


Figure 10: Topographic consistency test (TCT) applied to the four within conditions of the sample data (C1: correct sentence ending day 1; F1: false sentence ending day 1; C2: correct ending day 2; F2: false ending day 2). For each condition, two graphs are shown. The upper one displays the P value obtained by the TCT (vertical axis) and the chosen threshold (red line). The x-axis displays the time from 0 to 1000 ms from the word onset on. The second graph shows the Global Field Power (GFP), with periods of consistent topographies marked in green. The y-axes of the lower graphs indicate the GFP amplitude in μV.

creates limited interference with performance when running in the background. Parallelization is, however, planned in future releases.

Another limitation is that the program is academic software and under constant development. Since there are no separate alpha and beta releases, it may contain undocumented, more experimental options that are not yet meant for the general public (e.g., analyses in the frequency domain). So, unless you do not know precisely what to expect, please do not use them. And finally, the program has been developed and is maintained with limited resources; careful crosschecking of the plausibility of the results is mandatory; user support may become limited, but suggestions, problem reports, and criticism are always welcome.

References

[1] J. C. Mosher, R. M. Leahy, and P. S. Lewis, "EEG and MEG: forward solutions for inverse methods," *IEEE Transactions on Biomedical Engineering*, vol. 46, no. 3, pp. 245–259, 1999.

[2] T. Koenig and L. Melie-García, "A method to determine the presence of averaged event-related fields using randomization tests," *Brain Topography*, vol. 3, pp. 233–242, 2010.

[3] T. Koenig and L. Melie-Garcia, "Statistical analysis of multichannel scalp field data," in *Electrical Neuroimaging*, C. M. Michel, T. Koenig, D. Brandeis, L. R. R. Gianotti, and J. Wackermann, Eds., pp. 169–189, Cambridge University Press, Cambridge, UK, 2009.

[4] T. Koenig, L. Melie-García, M. Stein, W. Strik, and C. Lehmann, "Establishing correlations of scalp field maps with other experimental variables using covariance analysis and resampling methods," *Clinical Neurophysiology*, vol. 119, no. 6, pp. 1262–1270, 2008.

[5] W. K. Strik, A. J. Fallgatter, D. Brandeis, and R. D. Pascual-Marqui, "Three-dimensional tomography of event-related potentials during response inhibition: evidence for phasic frontal lobe activation," *Electroencephalography and Clinical Neurophysiology*, vol. 108, no. 4, pp. 406–413, 1998.

[6] D. Lehmann and W. Skrandies, "Reference-free identification of components of checkerboard-evoked multichannel potential fields," *Electroencephalography and Clinical Neurophysiology*, vol. 48, no. 6, pp. 609–621, 1980.

[7] D. Lehmann and W. Skrandies, "Spatial analysis of evoked potentials in man-a review," *Progress in Neurobiology*, vol. 23, no. 3, pp. 227–250, 1984.

[8] B. F. J. Manly, *Randomization, Bootstrap and Monte Carlo Methods in Biology*, Chapman & Hall, Boca Raton, Fla, USA, 2007.

[9] T. E. Nichols and A. P. Holmes, "Nonparametric permutation tests for functional neuroimaging: a primer with examples," *Human Brain Mapping*, vol. 15, no. 1, pp. 1–25, 2002.

[10] E. R. John, P. Easton, L. S. Prichep, and J. Friedman, "Standardized varimax descriptors of event related potentials: basic considerations," *Brain Topography*, vol. 6, no. 2, pp. 143–162, 1993.

[11] J. Wackermann, "Towards a quantitative characterisation of functional states of the brain: from the non-linear methodology to the global linear description," *International Journal of Psychophysiology*, vol. 34, no. 1, pp. 65–80, 1999.

[12] M. Stein, T. Dierks, D. Brandeis, M. Wirth, W. Strik, and T. Koenig, "Plasticity in the adult language system: a longitudinal electrophysiological study on second language learning," *NeuroImage*, vol. 33, no. 2, pp. 774–783, 2006.

[13] M. Kutas and S. A. Hillyard, "Reading senseless sentences: brain potentials reflect semantic incongruity," *Science*, vol. 207, no. 4427, pp. 203–205, 1980.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 935364, 12 pages
doi:10.1155/2011/935364

Research Article

BioSig: The Free and Open Source Software Library for Biomedical Signal Processing

Carmen Vidaurre,[1] Tilmann H. Sander,[2] and Alois Schlögl[3,4]

[1] *Machine Learning Group, Berlin Institute of Technology, Department of Software Engineering and Theoritical Computer Science, 10587 Berlin, Germany*
[2] *Physikalisch-Technische Bundesanstalt Institut Berlin, 10587 Berlin, Germany*
[3] *Institute of Human Computer Interfaces, Graz University of Technology, Krenngasse 37, 8010 Graz, Austria*
[4] *Institute of Science and Technology Austria (IST Austria), Am Campus 1, 3400 Klosterneuburg, Austria*

Correspondence should be addressed to Alois Schlögl, alois.schloegl@ist.ac.at

Received 24 September 2010; Revised 24 November 2010; Accepted 31 December 2010

Academic Editor: Sylvain Baillet

BioSig is an open source software library for biomedical signal processing. The aim of the BioSig project is to foster research in biomedical signal processing by providing free and open source software tools for many different application areas. Some of the areas where BioSig can be employed are neuroinformatics, brain-computer interfaces, neurophysiology, psychology, cardiovascular systems, and sleep research. Moreover, the analysis of biosignals such as the electroencephalogram (EEG), electrocorticogram (ECoG), electrocardiogram (ECG), electrooculogram (EOG), electromyogram (EMG), or respiration signals is a very relevant element of the BioSig project. Specifically, BioSig provides solutions for data acquisition, artifact processing, quality control, feature extraction, classification, modeling, and data visualization, to name a few. In this paper, we highlight several methods to help students and researchers to work more efficiently with biomedical signals.

1. Introduction

The area of biomedical signal processing has to deal with a large variety of topics. Artifact contamination, low signal-to-noise ratios, different data formats, classification, and statistical evaluation are general challenges of the field. Furthermore, a large number of different data processing methods for different signal modalities (EEG, ECG, etc.) and for different applications has to be considered. Moreover, software development itself is an important part of biomedical signal processing.

In the 1990s, the use of Matlab became popular to process biosignals. However, the algorithms were rarely available and the reimplementation of methods was common, even within the same research group. The field of software development was characterized by providers that offered closed (proprietary) solutions. This caused incompatibilities, and the same algorithms were implemented again and again. Another side effect was that each equipment provider defined its own data format for storing biosignals. These data could then be analysed only with the proprietary software of the vendor. Data export, if possible, was difficult and resulted usually in loss of information (e.g., metadata about the recording conditions, like filter settings or sampling rate, were not preserved).

These facts made the development and validation of new methods difficult. Additionally, the success of free and open source software in the field of operating systems (e.g., Linux) and server software did encourage the development of a free software library for biomedical signal processing.

Despite its focus on EEG data, BioSig can be used for general signal processing tasks related to a variety of measurement modalities. One example is the calculation of event-related averages in MEG. Another one is the calculation of spectral estimates of individual channels or time segments in functional near-infrared spectroscopy data. BioSig covers many EEG and polygraphic data formats. Furthermore, data loading is accomplished by a simple command.

BioSig consists of some (more or less) coherent parts, that are summarized as follows.

(i) *BioSig for Octave and Matlab (biosig4octmat)*. A toolbox for Octave and Matlab with powerful data import and export filters, feature extraction algorithms, classification methods, and a powerful viewing and scoring software.

(ii) *BioSig for C/C++ (biosig4c++)*. A C/C++ library that provides reading and writing routines for different biosignal data formats.

(iii) *rtsBCI (rtsbci)*. A real-time Brain Computer Interface (BCI) system implemented in Matlab and Simulink.

Most functions implemented in BioSig can be used with both Matlab and Octave and are installed through the package "biosig4octmat". This is also the main module of the project. Within this library many data formats are supported, and the toolbox provides a common interface for reading [1] different formats. An automated detection of the file format eases the use, making the detection transparent to the user. The writing of several common file formats is also supported. Additionally, useful algorithms for artifact detection and correction are available. Many algorithms for stochastic model parameters (autoregressive, multivariate, time-varying, etc.) are accessible in the time series analysis (TSA) [2] toolbox. These and other functions from the NaN-toolbox [3] are able to handle data with missing values (caused by, e.g., artifacts), too.

BioSig software is available "on-line" (cf. [4]) and under the terms of the "General Public License" (GPL) v3 [5]. The GPL guarantees to the users that the BioSig library can be freely used, studied, modified, and distributed. Having a library for biomedical signal processing provides a summary of prior art in the field and might be helpful against the detrimental effects of software patents.

2. Structure of BioSig

2.1. Toolbox Components. Matlab is a widespread numerical programming language used for biosignal processing, therefore BioSig started being developed for this proprietary platform. However, in order to provide a really free and open library, a special effort was undertaken to provide compatibility with Octave [6], a free and largely compatible alternative to Matlab. All functions are tested for their compatibility with both platforms. Although BioSig supports other programming languages such as C/C++ or Python, the main module of BioSig is for Matlab and Octave and we will focus on this in the following.

BioSig covers many aspects of biomedical signal processing. Therefore, the toolbox is divided into subcategories which depend on the functionally of the algorithms contained in them. After installing BioSig, the following folders are available and ordered by subtasks:

(i) file access, data input and output (loading and saving routines), path: biosig/t200/ ,

(ii) preprocessing, quality control, and artifact processing, path: biosig/t250/ ,

(iii) signal processing and feature extraction, path: biosig/t300/ ,

(iv) event-related synchronization/desynchronization (ERS/D) maps, path: biosig/t310/ ,

(v) classification and statistics (single trial analysis), path: biosig/t400/ ,

(vi) statistical tests, path: biosig/t450/ ,

(vii) evaluation criteria, path: biosig/t490/ ,

(viii) visualization, path: biosig/t500/ ,

(ix) time series analysis, path: tsa/ ,

(x) statistics of data with missing values encoded as NaN (not a number), path: nan/ ,

(xi) interactive viewer and scoring (requires Matlab), path: biosig/viewer/ ,

(xii) documentation and help, path: biosig/doc/ .

Figure 1 represents a scheme of the toolbox and how its different elements are interrelated.

The module "data input and output" is a common interface for accessing the various formats including an automated format detection. It supports reading of about 40 and writing of 10 different data formats including some audio formats. The preprocessing module provides tools for triggering (segmenting) signal data, for artifact detection, artifact reduction and quality control. The signal processing module includes several specialized biosignal processing functions, but also interfaces to standard signal processing functions and wrapper functions for more complex analyses. The classification module includes support for different classification methods. A number of classifiers including linear, quadratic and regularized discriminant analysis, several methods of Support Vector Machines (SVM), Naive Bayes Classifiers, Perceptron Learning, Partial Least Squares/Regression analysis as well as some sparse classifiers are supported; cross-validation procedures are supported to prevent overfitting. More recently, these methods were extended for the use with missing values and are now also distributed as part of the NaN-toolbox [3]. The module on evaluation criteria contains several functions for performance metrics as used in the field of Brain-Computer Interface (BCI) research [7]. The visualization module contains a simple viewer for biomedical data as well as a wrapper function to visualize the results of several standard analysis procedures. The interactive viewing and scoring software (SViewer) is based on the graphical user interface of Matlab, which is currently not supported by Octave. An alternative is the free stand-alone viewing and scoring software "SigViewer".

Other important modules of BioSig are the Time Series Analysis (TSA) toolbox [2] and the NaN-toolbox [3], which are also part of the Octave-forge repository. The TSA toolbox provides a unique variety of coupling measures based on a multivariate autoregressive modeling routine. The NaN toolbox handles data with missing values, which are commonly caused by artifacts and encoded by not-a-number (NaN). BioSig contains also several demonstration examples

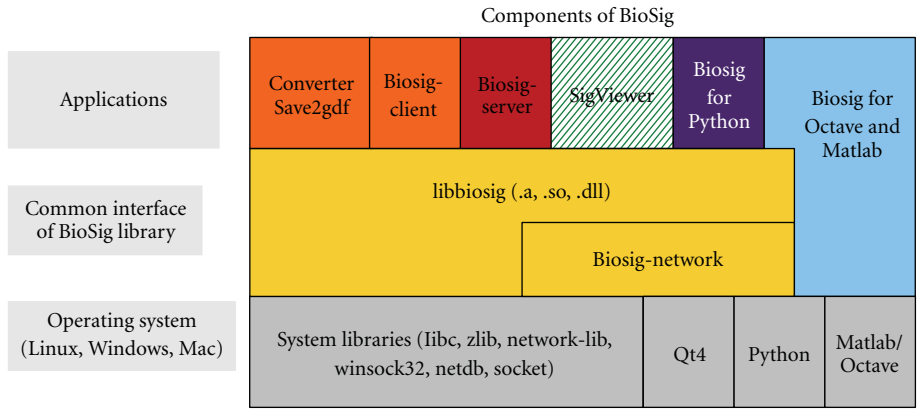


Figure 1: Architecture of the BioSig toolbox and its elements.

and a benchmark function for comparing the performance of different platforms. The benchmark functions perform some typical processing steps for calculating the classifier of BCI experiment. First some data is loaded; then several features are extracted; the features are used to compute a classifier; a cross-validation procedure (in this case a leave-one-out method) is used for validating the classifiers. The benchmark can be used to compare different hardware platforms as well as different versions of Octave and Matlab.

Contemporary high-density EEG and MEG can result in very large multidimensional signal vectors. As a consequence their processing might require large amounts of memory only available within 64 bit operating systems. To facilitate work with simpler hardware, BioSig can read data in blocks and the user has to control that parameter extraction is consistent across blocks of data. Reading of individual channels is possible even if the full data set does not fit into memory. This is accomplished by consecutive reading of blocks and concatenation of single channel data slices.

2.2. Compatibility between Octave and Matlab. Making BioSig compatible to Octave and Matlab was not self-evident. In the past, a number of core functions present in Matlab were missing from Octave, and had to be provided by BioSig for full compatibility. Luckily, the number of missing functions has been strongly reduced with newer versions of Octave (v3.2 and higher).

In addition, another problem with proprietary Matlab was addressed, notable not all Matlab users have all toolboxes available. Additional effort was spent to replace the dependency on add-on toolboxes (like statistics and signal processing) with free alternatives. This effort resulted in the release of "free toolboxes for Matlab" (freetb4matlab), which makes toolbox from Octave and Octave-forge available for the use with Matlab.

In general, the attempt to make BioSig (in particular biosig4octmat) fully compatible to Octave as well as Matlab was widely successful. Currently, only the interactive scoring software (a desirable but not mission-critical component) cannot be used with Octave. BioSig demonstrates that also a large-scale project can be programmed in such a way that it can run on Matlab as well as Octave without any code modifications.

2.3. Compatibility of BioSig with Other Toolboxes. Clearly the BioSig is one of several toolboxes designed for biomedical signal analysis. This suggests that interdependency between toolboxes and avoiding redundancy are quite important, but so far this topic has been rather neglected. A promising example is the reliance of FieldTrip (http://fieldtrip.fcdonders.nl/) and SPM (http://www.fil.ion.ucl.ac.uk/spm) on the BioSig for specific tasks. Examples are the reading of various file formats only supported in the BioSig (cf. FieldTrip: ft_read_data.m) and the multivariate autoregressive modeling implemented in the BioSig (cf. FieldTrip: ft_mvaranalysis.m). Additionally, Biosig is also included in EEGlab, a widely used interactive Matlab toolbox for EEG and MEG processing.

Dependencies like these need to be better addressed as no single toolbox provides all possible types of analysis. The range of processing steps appropriate for bioelectric and biomagnetic signals as summarized in [8] clearly exceeds the scope of a single toolbox. Unlike for proprietary software, Toolbox design is not "one against the others race", but rather a cooperative effort to create "scratch an itch" of developers and users, and eventually build the "super tool" for everyone.

A related question is the proper acknowledgment of a toolbox and its authors in the scientific literature. A toolbox typically implements several tens to at most a few hundred published algorithms. An interesting idea, is that the toolbox provides a "log of methods" used by an application to the user. Currently, BioSig cites the publications in the documentation of each function. In this way, the authors of the original works can be acknowledged. It would be desirable, that also the published software and its authors are properly cited.

3. Data Formats

Biomedical signals are stored in many different data formats. Most formats have been developed for a specific purpose of a specialized community (ECG research, EEG analysis,

Overview of open data formats for biomedical signals

Format	Multiple sampling rates and scaling factors	Multiple data types	Supports automated overflow detection	Physical units	Patient info, recording equipment, researcher	Events, markers, annotations/ predefined codes	Random data access, streaming	Electrode position/ orientation	Software tools available
ASTM E14676							No		
BCI2000	No	Yes	No	"uV"	−(1)	Yes/no	Yes	No	Yes
BDF	Yes	int24	No	char[8]	−(1)	Yes/no	Yes	No	Yes
BKR	No	int16	No	"uV"	−(1)	No/no	Yes	No	Yes
EBS	No	No			−(1)		Yes		
EDF	Yes	int16	No	char[8]	−(1)	No/no	Yes	No	Yes
EDF+	Yes	int16	No	char[8]	−(1)	Yes/no	Yes	No	Yes
FEF	Yes	Yes	Yes	Yes (2)	Yes		No		No
GDFv1.x	Yes	Yes	Yes	char[8]	−(1)	Yes/yes	Yes	No	Yes
GDFv2.0	Yes	Yes	Yes	Yes (2)	+(1)	Yes/yes	Yes	Yes	Yes
GDFv2.1	Yes	Yes	Yes	Yes (2)	Yes	Yes/yes	Yes	Yes	Yes
HL7aECG	Yes	Yes	No	Free text	Yes		No		Yes
MFER		Yes		23 predefined				No	Yes
SCP-ECG	No	No	No	Fixed	Yes	Ecg only	Yes	No	Yes
unisens							No		
SIGIF	Yes	Yes		Free text	−(1)		Yes	No	Yes

Figure 2: Properties of open and vendor independent data formats.

sleep research, etc.), by companies, research groups, and standardization organizations. A detailed comparison of about 20 biomedical data formats with publicly available specifications is shown in Figure 2 (for more details see [9]).

Although BioSig supports over 40 different data formats and can ease the problem, still the definition of a general purpose format was needed. In order to overcome the proliferation of data formats a "General Data Format for biosignals" (GDF) [1] has been developed with the aim to combine the best features of different formats into a single data format. BioSig provides a common interface for different data formats including an automated identification of the file format. This provides a seamless user interface, specifically the user can utilize the same functions for reading different formats.

Version 1 of the General Data format (GDF), [10], has been developed and successfully implemented and used in BCI research. GDF provides many useful features (different sampling rates and calibration values for different channels, an automated overflow detection, support of different data types, encoding of filter settings, etc.), that are only partly implemented in other formats. A key idea is also to define a fixed coding scheme for events, which supports compatibility of event information across different studies and laboratories. GDF is the first data format that addresses this topic.

Within recent years, new requirements became apparent. The new Version 2 of the GDF addresses the need for:

(i) subject-specific information (gender, age, impairment, etc.),

(ii) recording location, identification of recording software, and so forth,

(iii) possibilities for storing the electrode positions in spatial coordinates, electrode impedances, and so forth,

(iv) more efficient encoding of date and time, physical dimensions, and filter information,

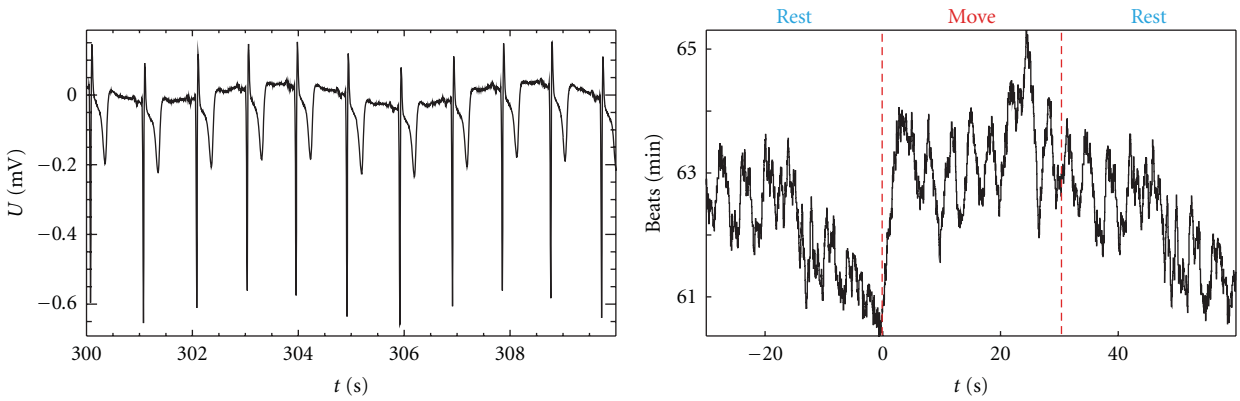
(v) nonequidistant (sparse) sampling.

The structure of GDF v2.0 is similar to EDF [11], GDF1.x [10], and EDF+ [12].

Briefly, an GDF file consists of the following five components: the fixed header or header 1 (with 256 bytes) is mandatory, the variable header or header 2 containing channel-specific information (number-of-channels times 256 bytes), the tag-length-value (TLV) header or header 3 contains optional information, the data section, and the table of events. Header 2 can be empty, in case that no channel information is stored (e.g., in pure event files).

Data is stored in little endian format. However, BioSig supports also big-endian platforms by converting the data internally. The Version field is of type char [8] and is stored at the beginning of the file. This field is used to provide upwards compatibility with past and future versions of GDF.

The format definition of GDF is nearly as simple as the definition of EDF. The use of binary encodings enables a more compressed representation; accordingly, more information can be stored within the header information. This enables a higher accuracy (e.g., in date and time information) and additional information can be stored without extending the header size.



(a) Section of raw ECG

(b) Task dependent heart rate

Figure 3: (a) Section of 10 s of raw ECG from a measurement lasting 1800 s. (b) The heart rate determined from the ECG was averaged with respect to the onset of the finger movement task performed by the subject. The changes in the averaged heart rate are within a range of 4 beats/min. This is small in relation to the 63 beats/min mean value and it can be concluded that the subject was relaxed.

The proposed format specification was successfully implemented in C/C++ as well as an *M*-file which can be used with Octave (>2.9.12) and Matlab (>6.5). The software implementation requires only minor changes to upgrade from EDF, BDF, or earlier GDF to GDF 2.20. In BioSig different data formats can be simultaneously supported.

GDF provides a superset of features from many other data formats. GDF v2.10 includes support for, user-specified event description (like in EDF+ and BrainVision format), manufacturer information (like in SCP [13] and MFER [14]), and the orientation of MEG sensors. Accordingly, GDF v2.x is (upwards) compatible with most other data formats; this means that biosignal data from other formats can be converted to GDF without loss of information. Routines for reading and writing GDF files in Octave and Matlab, as well as in C, are implemented in the open source package BioSig. For more details, please refer to [1].

4. BioSig in Biomedical Research

4.1. Heart Rate Extraction. Even the apparently simple task of extracting the heart rate from an electrocardiographic (ECG) signal is appropriate for a biosignal toolbox because the user is not distracted by coding a heart rate extractor on the fly. In BioSig, two well-tested and published algorithms [15, 16] are implemented in a single routine. The first algorithm determines the envelope of the ECG using a Hilbert transform and the positions of the R-peaks are determined by thresholding. The second algorithm uses a bank of filters, and it incorporates an ectopic beat correction. The resulting heart rates are therefore suitable for advanced heart rate variability studies.

The first algorithm [15] was used in the example shown in Figure 3. During a session lasting 1800 s, the MEG, the fNIRS (functional near-infrared spectroscopy), and the ECG were recorded from a subject. For the whole duration the subject had to alternate between 30 s of finger movements and 30 s of rest. The heart rate was extracted offline from the ECG. Subsequently, event-related averages were calculated for MEG, fNIRS, and heart rate over the 30 epochs of finger movements using trigger points related to the onset of finger movements. The extraction of heart rate, determination of trigger time points, and the averaging were performed using appropriate routines from BioSig. The acquisition of the ECG followed by the extraction of an event-related heart rate, as shown in Figure 3, established that the subject was in a relaxed state throughout the measurement. The oscillations in the averaged heart rate in Figure 3(b) indicate a certain degree of synchrony between respiration and the finger movement task. Results for MEG and fNIRS are discussed in [17].

4.2. Artifact Processing. Several artifact processing methods are included in BioSig. The performance of the methods has been demonstrated in research and published in several papers. In the following we briefly explain some of them.

The first method consists of a "histogram-based" quality control of the biomedical signals (see Figure 4). In [18] it was found that the header information of the EEG recordings does not always provide the real saturation values of the recording equipment, therefore an automated saturation detection was not possible. A quality control method based on histogram analysis was developed and its performance was successfully demonstrated. The amplitude histograms and entropies of all-night sleep recordings from 8 different sleep laboratories were calculated. This method is provided by BioSig in order to support the visual identification of the thresholds needed for the saturation detection.

Also, algorithms for detection of muscle artifacts are implemented. For example, the method described in [19] is available in BioSig. In that paper the authors used time domain and frequency domain methods for the detection of muscular noise in awake EEG. For time domain detection, they used slope and maximum/minimum amplitude. The parameters in the frequency domain were absolute and relative "high beta" power (>25 Hz) and spectral edge

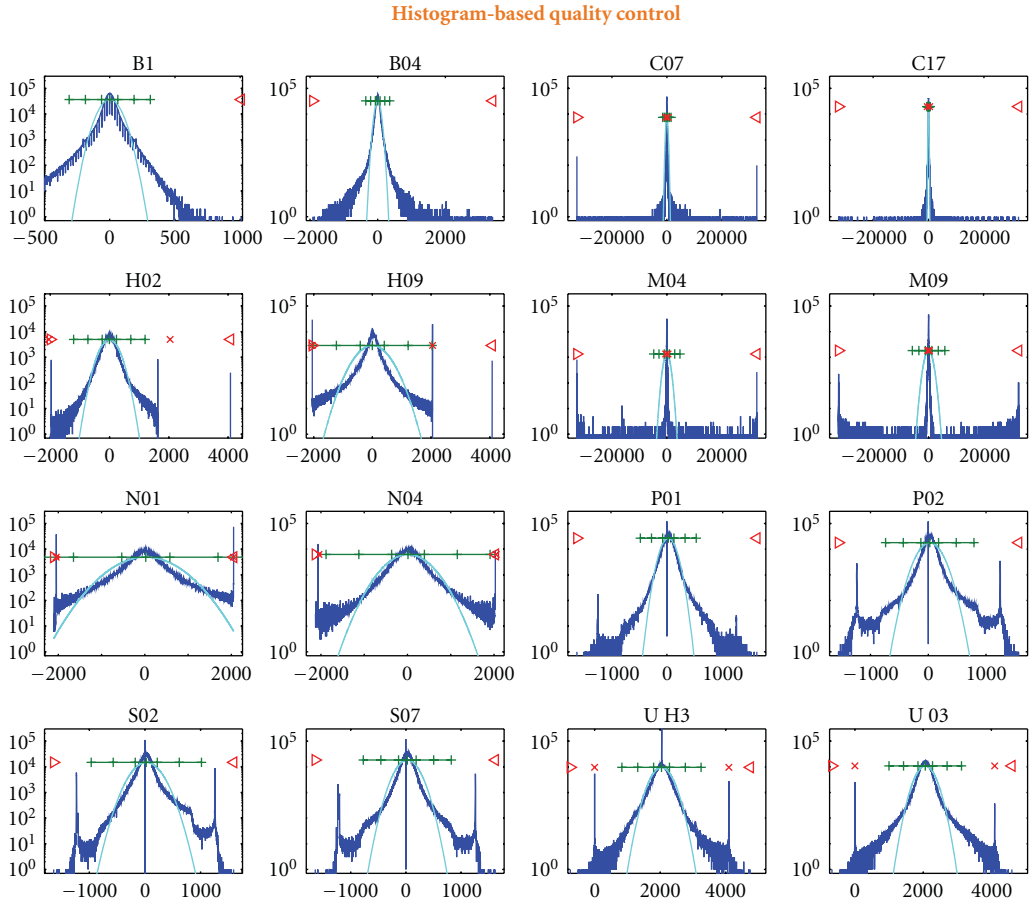


FIGURE 4: Histograms of 16 all-night sleep EEG (modified from [18]). Thresholds for overflow detection can be obtained through BioSig's eeg2hist.m tool.

frequency. The detection thresholds were calculated from subject distributions calculated from a reference period. This method consistently outperforms the use of constant empirical thresholds.

Finally, a method for artifact removal of electrooculographic (EOG) artifacts in EEG is also available in BioSig. It is a very powerful algorithm based on simple linear regression. Its suitability has been demonstrated in two papers, [20, 21].

The observed EEG can be considered as a linear superposition of EEG and EOG components. This can be written in the form of a regression model:

$$\vec{\mathbf{Y}}_t = \vec{\mathbf{E}}_t + \mathbf{b}_{N\times M} \cdot \vec{\mathbf{O}}_t. \tag{1}$$

Accordingly, the observed EEG at time t is a vector $\vec{\mathbf{Y}}_t$ with N elements, and the observed EOG activity $\vec{\mathbf{O}}_t$ at time t has M elements. The observed EEG data $\vec{\mathbf{Y}}_t$ consists of a linear superposition of the true EEG activity $\vec{\mathbf{E}}_t$ and the ocular activity $\vec{\mathbf{O}}_t$ that propagates through the mechanism of volume conduction to each EEG electrode. The propagation factors are described by the model parameters $\mathbf{b}_{N\times M}$, which describe the influence of M components of the ocular dipoles to each of the N EEG channel. Because the propagation mechanism is simple volume conduction [22–24], it depends only on the geometry and the conductivity of the head tissue. It is reasonable to assume that these are constant during the whole EEG recording time T, $0 < t \quad T$ and independent of frequency. It should be noted that the regression model can be also written in the following form:

$$\begin{bmatrix} \vec{\mathbf{Y}}_t \\ \vec{\mathbf{Z}}_t \end{bmatrix} = \begin{bmatrix} \mathbf{I}_{N\times N} & \mathbf{b}_{N\times M} \\ \mathbf{0}_{M\times N} & \mathbf{I}_{M\times M} \end{bmatrix} \cdot \begin{bmatrix} \vec{\mathbf{E}}_t \\ \vec{\mathbf{O}}_t \end{bmatrix}, \tag{2}$$

where $\vec{\mathbf{Z}}_t = \vec{\mathbf{O}}_t$ represents the observed EOG channels.

If the EOG activity is measured, its contribution can be removed using the least squares solution of (1). A right multiplication of (1) with $\vec{\mathbf{O}}_t^T$ and applying the expectation operator $\cdot_t$ over time t yields

$$\left\langle \vec{\mathbf{Y}}_t \cdot \vec{\mathbf{O}}_t^T \right\rangle = \left\langle \vec{\mathbf{E}}_t \cdot \vec{\mathbf{O}}_t^T \right\rangle + \mathbf{b}_{N\times M} \cdot \left\langle \vec{\mathbf{O}}_t \cdot \vec{\mathbf{O}}_t^T \right\rangle. \tag{3}$$

Because EEG and EOG can be considered uncorrelated, the term $\vec{\mathbf{E}}_t \cdot \vec{\mathbf{O}}_t^T$ becomes zero, the true model coefficients are

$$\mathbf{b} = \left\langle \vec{\mathbf{Y}}_t \cdot \vec{\mathbf{O}}_t^T \right\rangle \cdot \left\langle \vec{\mathbf{O}}_t \cdot \vec{\mathbf{O}}_t^T \right\rangle^{-1} \quad (4)$$

and the EEG data can be corrected according to

$$\vec{\mathbf{E}}_t = \vec{\mathbf{Y}}_t - \mathbf{b} \cdot \vec{\mathbf{O}}_t. \quad (5)$$

The method is also known as "least squares approach" or "multiple least squares approach" in case that more than one EOG component is removed. $\mathbf{b}$ is chosen in such a way that the mean square of $\vec{\mathbf{E}}$ is minimized.

This model takes into account only EEG and EOG sources. In practice, other noise sources (e.g., amplifier and impedance noise, electric and magnetic interferences, and muscle activity) occur, too. In order to analyze the possible influence of these noise sources on the reduction method, a noisy model has to be considered. One consequence of this analysis is the fact that the model estimates $\hat{\mathbf{b}} = \boldsymbol{\beta}$ are least biased if the signal-to-noise ratio between EOG and other noise sources is as large as possible. Therefore, we estimated the model coefficients from data with large ocular activity. Furthermore, it can be advantageous to filter the data (e.g., for removing the very low frequency components of the $1/f$ amplifier noise, and the very high frequency activity). In other words, the correction coefficients are most accurate if the influence of other noise sources can be avoided.

The difference between the regression approach (linear superposition model) and the component-based approaches such as blind source separation remains in the manner how the signals describing the EOG activity are obtained. While the regression approach uses the observed EOG activity, the component-based approaches decompose the data into a number of independent (and uncorrelated) components, and different heuristics are used for identifying the EOG components. An advantage of component-based methods could not be demonstrated within a study [20]. It is important to use bipolar EOG channels with EOG electrodes located close to the eyes as regressors. The results suggest that it is more difficult to identify the artifact components with blind source separation methods than with the dedicated channels (like EOG) recording the artifact.

Figure 5 illustrates the regression method to correct EOG artifacts in EEG. On the left the raw EEG data is visible, on the right the corrected EEG is displayed. More detailed information is available in [20, 21].

4.3. Coupling and Connectivity with EEG and Multivariate Autoregressive Models. One of the most striking problems in neuroscience is the study of brain areas that interact with each other, and how they interact during the performance of a certain task.

The (auto-)spectrum of a single channel and the cross-spectrum of two different channels have been used for a while to analyze the connectivity of different brain areas [25, 26].

An often used measure related to the cross-spectrum is the coherence, which is defined as the power of the cross-spectrum of two channels normalized with the corresponding power autospectra. Therefore, its magnitude varies from 0 to +1. The normalized cross-spectrum before taking the power is called coherency, and it is a complex number, so it has a real and imaginary part. As a complex number it can be represented by its amplitude and phase.

Nolte et al. [27] proposed to investigate the imaginary part of the coherency as a connectivity measure, because a nonzero imaginary part of coherency can not be explained by volume conduction alone, but is an indicator for a functional coupling between different brain areas. By computing the phase of the coherency (using the real and imaginary parts), the time delay between signals present in two channels can be estimated. Another measure defined to remove bias due to volume conduction is the partial coherence. It is computed between a pair of channels, partializing out the activity of the remaining channels.

The measures mentioned so far are antisymmetric or symmetric and therefore cannot represent the direction of the information flow. Kaminisky and Blinowska [28] proposed the directed transfer function to detect whether the coupling between brain areas is forwards, backwards, or both. The partial directed coherence (PDC), motivated by the partial coherence, was also defined for this purpose in [29]. Only the PDC and the generalized PDC have the potential to identify the causal relationships and the underlying structure of an observed system.

All these measures have in common that they can be estimated from a multivariate autoregressive (MVAR) model, so that MVAR models can be considered a common basis for the comparison of different coupling measures. BioSig does integrate a complete toolbox for MVAR modeling in the folder *tsa* (time series analysis) [2].

Examples for the application of coupling measures to EEG using BioSig can be found in [30–33], and methodological issues are addressed in more detail by the works [8, 34].

4.4. Brain Computer Interfacing. The purpose of a BCI system is to identify the user's intention by observing and analyzing brain activity without relying on signals from muscles or peripheral nerves. BioSig contains many useful tools for BCI research, most of them designed for EEG signals (although certain functions can be used to process other signals). The reason why BioSig is focused on EEG is that it is noninvasive, portable, can be used in almost any environment, and it has excellent time resolution.

Figure 6 illustrates a typical BCI. An online data-processing system controls devices in real time and provides feedback to the user. To generate the control signal, the BCI must extract and classify EEG features. The feature extraction method is typically based on the type of neurophysiological activation, and the classifier is usually obtained by offline analyses of previous data records from the same subject (subject-selected feature parameters as well, cf. [35]).

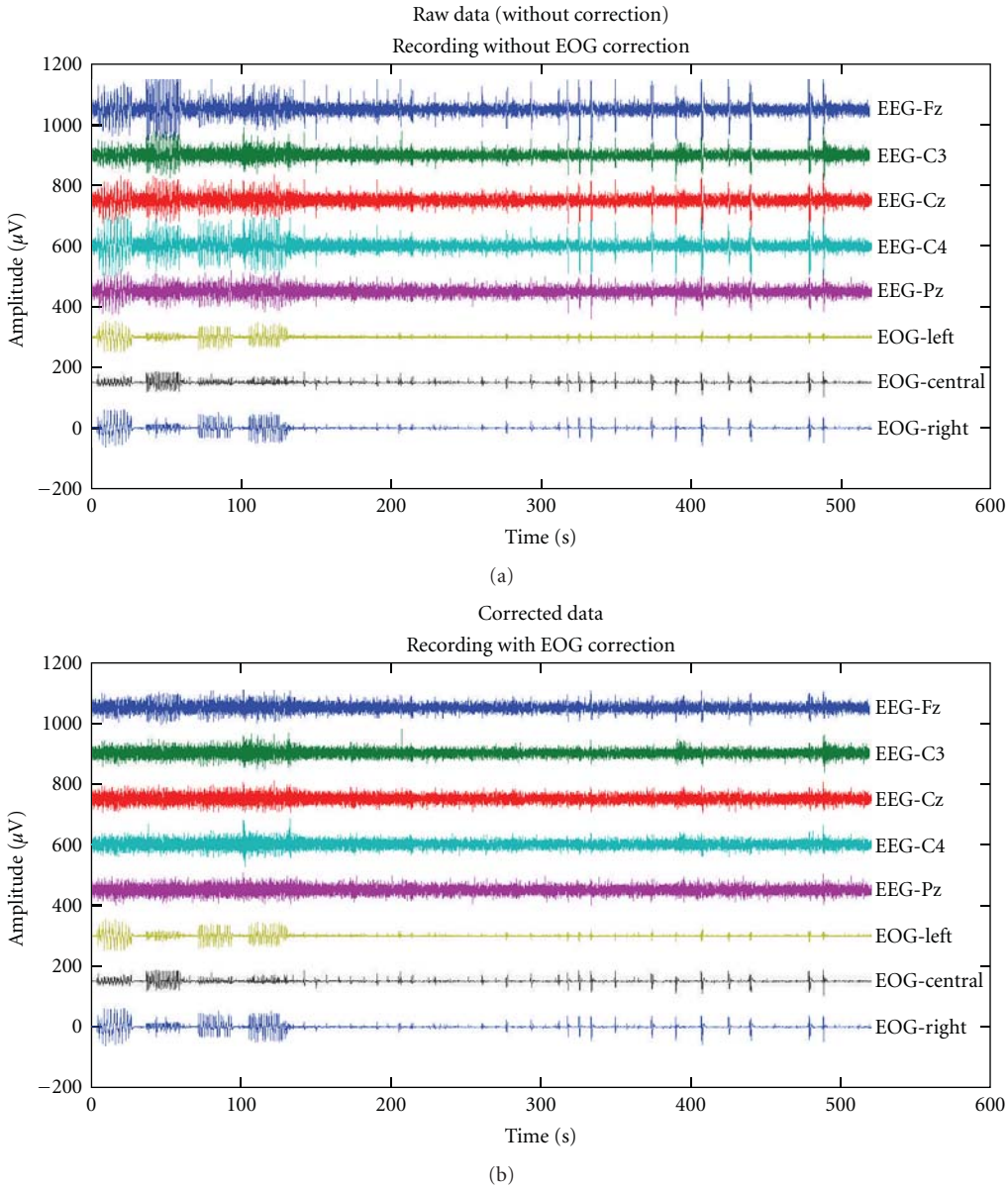


Figure 5: (a) Raw EEG data, contaminated with ocular artifacts. (b) Corrected data using regression analysis.

The rtsBCI toolbox is a real-time Brain Computer Interface (BCI) system implemented in Matlab and Simulink that can serve the purpose of designing an online system, however it is currently not supported.

Some BCIs use primarily spectral analysis (e.g., frequency band power or autoregressive spectra) to characterize spontaneous oscillatory EEG activity. It can also use autoregressive parameters directly to describe the entire spectral density function [36]. Alternatively, a BCI can analyze the user's response to visual or acoustic stimuli, which can be presented one by one or in a steady-state (repetitive) mode.

As already mentioned in Section 4.2, data preprocessing is important to remove the influence of technical artifacts and nonbrain activity such as electrical signals caused by eye movements or facial muscles. Section 4.2 offers examples of methods that are implemented in BioSig to remove artifacts. Additionally, in the case of EEG recordings, spatial filters can also focus on a specific brain area or identify particular signal components [35, 37–39]. These are available in BioSig as well.

A BCI uses offline analysis for several purposes. The most common is the estimation of a reliable classifier. But when the classifier and/or features contain hyperparameters, such as adaptation speed or regularization coefficients, they need to be tuned off-line [40–43]. For the evaluation of methods, the application of cross-validation and may be resampling procedures might be necessary. All tools for offline analysis are as well available in BioSig.

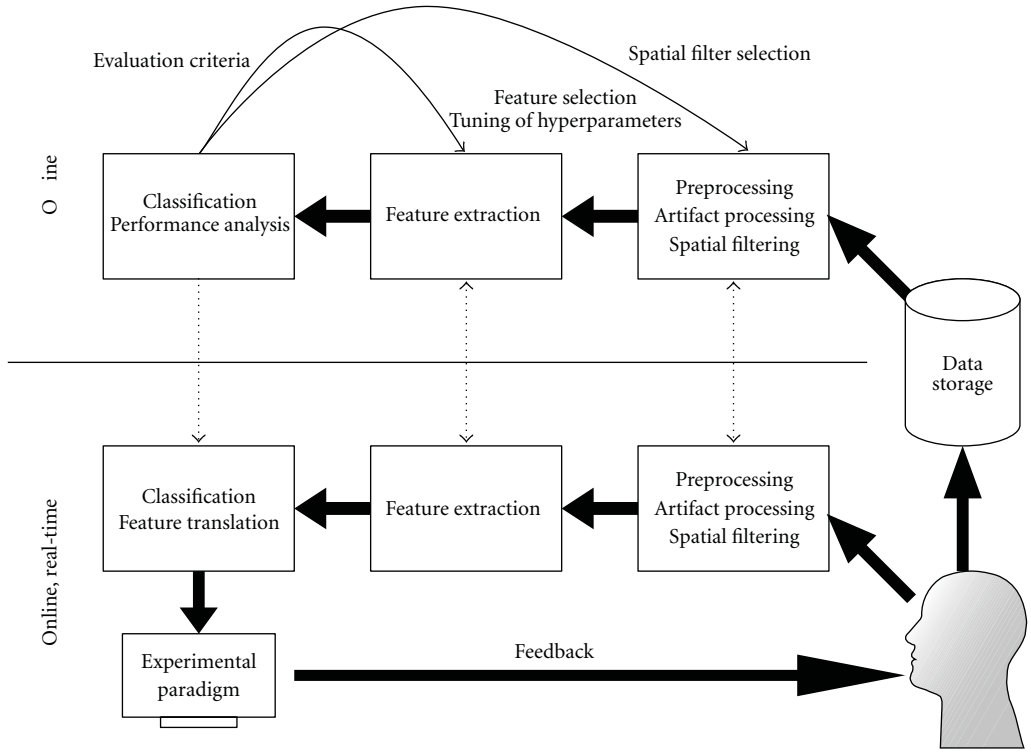


Figure 6: Elements of a brain computer interface.

An overview of general methods for BCI research implemented in BioSig is presented in Table 1.

Evaluation Criteria. Traditionally, BCI performance is quantified by classification accuracy or error rate. For the multiclass (when the user performs more than 2 tasks), there are other metrics that might be preferable as for example, the Cohen's kappa coefficient, which is derived from the confusion matrix [44].

In some cases, it is desirable to quantify BCI performance in terms of the information transfer rate [45]. Other metrics are the correlation coefficient, mean square error, and area under the receiver operating characteristic curve. Support for these and some more criteria is provided within BioSig [46–48].

5. BioSig for C/C++ and Libbiosig

BioSig for C/C++ (short biosig4c++) provides some command line tools for data conversion, a library to access a number of data formats (libbiosig), and some experimental code for network transfer of biosignal data. The motivation for a C/C++-based library is performance issues, flexibility in terms of supported platforms (e.g., Matlab can be hardly installed on some embedded devices), and interfacing to existing libraries. When there was a need for a converter between XML-based HL7aECG and the SCP-ECG data format, it was first implemented in C/C++ [49], nowadays about 30 data formats can be read and 10 are written. Moreover, biosig4c++ can be used now with Octave and Matlab through a MEX-interface yielding a much better performance than the traditional m-scripts. biosig4c++ provides also an interface for Python (it enables reading of 30 biosignal data formats into Python) and might be also useful for other software platforms. The free viewing and scoring software "SigViewer" uses libbiosig for accessing biosignal data. An experimental implementation of a network-based data transfer is included in biosig4c++, which might be useful for biosignal recorders (embedded devices) and for network-based biosignal archives.

6. Conclusions and Future Work

BioSig provides a whole tool chain of data processing methods for BCI research. These tools are useful in other application areas, like seizure detection and seizure prediction in epileptic EEG. Connectivity analysis with MVAR methods is another expanding application topic of BioSig.

Data analysis with biosig4octmat emphasizes the almost fully automated data analysis. However, a major limitation to this goal is the manual scoring of artifacts. Here, we see a need to validate promising artifact processing methods. The results on EOG artifacts show that a two-channel regression analysis can reduce about 80% of EOG artifacts in EEG recordings [21], similarly to blind source separation methods combined with some heuristics for component selection, [20]. Similar results are expected for raw MEG data in channel space. For EMG artifacts, there are methods of inverse filtering [36] and high pass filtering implemented [50], but

Table 1: List of BCI-related task that can be performed using BioSig.

Data preprocessing	Triggering, partitioning of data
	Artifact processing
	Quality check of data through histogram analysis
	Spatial filters
	Detection of EMG artifacts
	Common spatial patterns
Feature extraction Band power	Adaptive autoregressive parameters
	Adaptive multivariate autoregressive parameters
	(Adaptive) Hjorth
	(Adaptive) Barlow
	(Adaptive) Wackermann
	(Adaptive) time-domain parameters
	Adaptive brain rate,
	spectral edge frequency
Feature classification	Linear discriminant analysis (LDA)
	Quadratic discriminant analysis (QDA)
	Support vector machines
	Naive Bayesian classifier (NBC)
	Augmented NBC
	Sparse LDA
	Generalized discriminant analysis
Evaluation criteria Classification accuracy	Cohen's kappa coefficient
	Receiver operating characteristics (ROC)
	Area under the ROC curve
	Mutual information, information transfer rate
	Correlation coefficient
Metafunctions	findclassifier, cross-validation (xval),
	standardized analysis
	(demo2 is an example of a standardized
	offline analysis)

currently only limited results about their performance are available. However, the validation of the performance of artifact processing methods is crucial for a number of possible applications, including BCI but also seizure detection and prediction.

Up to now, controlled signal conditioning under BioSig is not included in order to simplify the design and also because during the experiments the conditions are usually fixed. Otherwise it is not clear whether changes are due to the recording system or they occurred in the observed system.

biosig4octmat is an application that can be used with Octave and is available from Sourceforge. Currently, the monthly download range is about 500 per month. Its installation in Octave is similar as in Matlab. However, the BioSig benchmark shows that Octave is somewhat slower than Matlab. In the future, the BioSig development will stay committed to compatibility to Octave, and will work actively to support compatibility with both.

For certain applications, the support of hardware platforms beyond personal computers is of interest. For example, embedded devices are important for online and real-time applications. Here, a C/C++ library like biosig4c++/libbiosig can be very useful. Biosig supports so far different programming languages including C/C++, Octave/Matlab and Python. Experimental support for other programming languages (e.g., Java, PhP, Perl, Ruby, Tcl, etc.) using the SWIG tool is currently investigated. Standardization (of data formats as well as data analysis methods) is also an important area, and due to its free software development model BioSig provides a suitable platform for these topics.

So far, the emphasis of BioSig was in providing a library of high-quality methods, useful algorithms, and reference implementations for all kind of biomedical signal processing problems, rather than a "user-friendly" environment for nonexperts. However, a folder with several Demos is available after installation (biosig/demo/). The future of BioSig is open, and the development and future direction of BioSig depends on each contributor to the BioSig project.

Acknowledgments

Helpful discussions with Alain de Cheveigne, Robert Oostenveld, and Vladimir Litvak are gratefully acknowledged. This work has been supported by FP7-ICT-248326, MEIF-CT-2006-40666. This publication only reflects the authors' views. Funding agencies are not liable for any use that may be made of the information contained herein.

References

[1] A. Schlögl, "GDF—a general dataformat for biosignals," Tech. Rep., 2004–2009, http://arxiv.org/pdf/cs/0608052v6.

[2] A. Schlögl, "Time series analysis toolbox," 2010, http://octave.sourceforge.net/tsa/index.html.

[3] A. Schlögl, "The NaN-toolbox v2.0: a statistics and machine learning toolbox for Octave and Matlab for data with and w/o MISSING VALUES encoded as NaN's," 2010, http://octave.sourceforge.net/nan/index.html and https://mloss.org/software/view/206/.

[4] A. Schlögl, "The BioSig project 2003–2009," 2009, http://biosig.sf.net/.

[5] The General Purpose License, http://www.fsf.org/.

[6] J. W. Eaton, Octave, http://www.octave.org/.

[7] A. Schlögl, J. Kronegg, J. E. Huggins, and S. G. Mason, "Evaluation criteria in BCI research," in *Towards Brain-Computer Interfacing*, J. Dornhege, R. Millan, T. Hinterberger, D. J. Mc-Farland, and K.-R. Müller, Eds., pp. 327–342, MIT Press, Cambridge, Mass, USA, 2007.

[8] T. H. Sander, T. R. Knösche, A. Schlögl et al., "Recent advances in modeling and analysis of bioelectric and biomagnetic sources," *Biomedizinische Technik*, vol. 55, no. 2, pp. 65–76, 2010.

[9] A. Schlögl, "An overview on data formats for biomedical signals," in *Image Processing, Biosignal Processing, Modelling and Simulation, Biomechanics*, pp. 1557–1560, World Congress on Medical Physics and Biomedical Engineering, Munich,

Germany, 2009.

[10] A. Schlögl, O. Filz, R. Ramoser, and G. Pfurtscheller, "GDF—a general dataformat for biosignals," Tech. Rep., 2004, http://pub.ist.ac.at/ schloegl/matlab/eeg/gdf4/TR_GDF.pdf.

[11] B. Kemp, A. Varri, A. C. Rosa, K. D. Nielsen, and J. Gade, "A simple format for exchange of digitized polygraphic recordings," *Electroencephalography and Clinical Neurophysiology*, vol. 82, no. 5, pp. 391–393, 1992.

[12] B. Kemp and J. Olivan, "European data format 'plus' (EDF+), an EDF alike standard format for the exchange of physiological data," *Clinical Neurophysiology*, vol. 114, no. 9, pp. 1755–1761, 2003.

[13] Standard Communications Protocol for computer-aided Electrocardiography (SCP-ECG). EN1064-2007 and ISO/DIS 11073/91064.

[14] Medical Waveform Format (MFER), ISO/TS 11073/92001: 2007.

[15] M. E. Nygards and L. Sornmo, "Delineation of the QRS complex using the envelope of the e.c.g," *Medical and Biological Engineering and Computing*, vol. 21, no. 5, pp. 538–547, 1983.

[16] V. X. Afonso, W. J. Tompkins, T. Q. Nguyen, and S. Luo, "ECG beat detection using filter banks," *IEEE Transactions on Biomedical Engineering*, vol. 46, no. 2, pp. 192–202, 1999.

[17] T. H. Sander, S. Leistner, H. Wabnitz, B. M. MacKert, R. MacDonald, and L. Trahms, "Cross-correlation of motor activity signals from dc-Magnetoencephalography, near-infrared spectroscopy, and electromyography," *Computational Intelligence and Neuroscience*, vol. 2010, Article ID 785279, 2010.

[18] A. Schlögl, B. Kemp, T. Penzel et al., "Quality control of polysomnographic sleep data by histogram and entropy analysis," *Clinical Neurophysiology*, vol. 110, no. 12, pp. 2165–2170, 1999.

[19] M. Van De Velde, G. Van Erp, and P. J. M. Cluitmans, "Detection of muscle artefact in the normal human awake EEG," *Electroencephalography and Clinical Neurophysiology*, vol. 107, no. 2, pp. 149–158, 1998.

[20] A. Schlógl, A. Ziehe, and K.-R. Müller, "Automated ocular artifact removal: comparing regression and component-based methods," available from *Nature Precedings*, http://precedings.nature.com/documents/3446/version/1/files/npre20093446-1.pdf.

[21] A. Schlögl, C. Keinrath, D. Zimmermann, R. Scherer, R. Leeb, and G. Pfurtscheller, "A fully automated correction method of EOG artifacts in EEG recordings," *Clinical Neurophysiology*, vol. 118, no. 1, pp. 98–104, 2007.

[22] P. L. Nunez, *Electric Fields of the Brain. The Neurophysics of EEG*, Oxford University Press, New York, NY, USA, 1981.

[23] J. Malmivou and R. Plonsey, *Bioelectromagnetism: Principles and Applications of Bioelectric and Biomagnetic Fields*, Oxford University Press, New York, NY, USA, 1995.

[24] J. J. M. Kierkels, G. J. M. van Boxtel, and L. L. M. Vogten, "A model-based objective evaluation of eye movement correction in EEG recordings," *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 2, pp. 246–253, 2006.

[25] P. L. Nunez, R. Srinivasan, A. F. Westdorp et al., "EEG coherency I: statistics, reference electrode, volume conduction, Laplacians, cortical imaging, and interpretation at multiple scales," *Electroencephalography and Clinical Neurophysiology*, vol. 103, no. 5, pp. 499–515, 1997.

[26] P. L. Nunez, R. B. Silberstein, Z. Shi et al., "EEG coherency II: experimental comparisons of multiple measures," *Clinical Neurophysiology*, vol. 110, no. 3, pp. 469–486, 1999.

[27] G. Nolte, O. Bai, L. Wheaton, Z. Mari, S. Vorbach, and M. Hallett, "Identifying true brain interaction from EEG data using the imaginary part of coherency," *Clinical Neurophysiology*, vol. 115, no. 10, pp. 2292–2307, 2004.

[28] M. J. Kaminski and K. J. Blinowska, "A new method of the description of the information flow in the brain structures," *Biological Cybernetics*, vol. 65, no. 3, pp. 203–210, 1991.

[29] L. A. Baccalá and K. Sameshima, "Partial directed coherence: a new concept in neural structure determination," *Biological Cybernetics*, vol. 84, no. 6, pp. 463–474, 2001.

[30] A. Schlögl and G. Supp, "Analyzing event-related EEG data with multivariate autoregressive parameters," in *Event-Related Dynamics of Brain Oscillations. Analysis Ofdynamics of Brain Oscillations: Methodological Advances*, C. Neuper and W. Klimesch, Eds., Progress in Brain Research 159, pp. 135–147, Elsevier, Berlin, Germany, 2006.

[31] G. G. Supp, A. Schlögl, T. C. Gunter, M. Bernard, G. Pfurtscheller, and H. Petsche, "Lexical memory search during N400: cortical couplings in auditory comprehension," *NeuroReport*, vol. 15, no. 7, pp. 1209–1213, 2004.

[32] G. G. Supp, A. Schlögl, C. J. Fiebach et al., "Semantic memory retrieval: cortical couplings in object recognition in the N400 window," *European Journal of Neuroscience*, vol. 21, no. 4, supplement, pp. 1139–1143, 2005.

[33] G. G. Supp, A. Schlögl, N. Trujillo-Barreto, M. M. Müller, and T. Gruber, "Directed cortical information flow during human object recognition: analyzing induced EEG gamma-band responses in brain's source space," *PLoS ONE*, vol. 2, no. 8, article e684, 2007.

[34] A. Schlögl, "A comparison of multivariate autoregressive estimators," *Signal Processing*, vol. 86, no. 9, pp. 2426–2429, 2006.

[35] B. Blankertz, R. Tomioka, S. Lemm, M. Kawanabe, and K. R. Müller, "Optimizing spatial filters for robust EEG single-trial analysis," *IEEE Signal Processing Magazine*, vol. 25, no. 1, pp. 41–56, 2008.

[36] A. Schlögl, *The Electroencephalogram and the Adaptive Autoregressive Model: Theory and Applications*, Shaker, Aachen, Germany, 2000.

[37] J. Müller-Gerking, G. Pfurtscheller, and H. Flyvbjerg, "Designing optimal spatial filters for single-trial EEG classification in a movement task," *Clinical Neurophysiology*, vol. 110, no. 5, pp. 787–798, 1999.

[38] H. Ramoser, J. Müller-Gerking, and G. Pfurtscheller, "Optimal spatial filtering of single trial EEG during imagined hand movement," *IEEE Transactions on Rehabilitation Engineering*, vol. 8, no. 4, pp. 441–446, 2000.

[39] H. Ramoser, J. R. Wolpaw, and G. Pfurtscheller, "EEG-based communication: evaluation of alternative signal prediction methods," *Biomedizinische Technik*, vol. 42, no. 9, pp. 226–233, 1997.

[40] C. Vidaurre, A. Schlögl, R. Cabeza, R. Scherer, and G. Pfurtscheller, "A fully on-line adaptive BCI," *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 6, pp. 1214–1219, 2006.

[41] C. Vidaurre, A. Schlögl, R. Cabeza, R. Scherer, and G. Pfurtscheller, "Study of on-line adaptive discriminant analysis for EEG-based brain computer interfaces," *IEEE Transactions on Biomedical Engineering*, vol. 54, no. 3, pp. 550–556, 2007.

[42] C. Vidaurre, R. Scherer, R. Cabeza, A. Schlögl, and G. Pfurtscheller, "Study of discriminant analysis applied to motor

imagery bipolar data," *Medical and Biological Engineering and Computing*, vol. 45, no. 1, pp. 61–68, 2007.

[43] C. Vidaurre, N. Krämer, B. Blankertz, and A. Schlögl, "Time Domain Parameters as a feature for EEG-based Brain-Computer Interfaces," *Neural Networks*, vol. 22, no. 9, pp. 1313–1319, 2009.

[44] J. Cohen, "A coefficient of agreement for nominal scales," *Educational and Psychological Measurement*, vol. 20, pp. 37–46, 1960.

[45] A. Schlögl, C. Neuper, and G. Pfurtscheller, "Estimating the mutual information of an EEG-based Brain-Computer Interface," *Biomedizinische Technik*, vol. 47, no. 1-2, pp. 3–8, 2002.

[46] A. Schlögl, J. Kronegg, J.E. Huggins, and S. G. Mason, "Evaluation criteriain BCI research," in *Towards Brain-Computer Interfacing*, G. Dornhege, J. R. Millan, T. Hinterberger, D. J. Mc-Farland,, and K.-R. Müller, Eds., pp. 327–342, MIT Press, Cambridge, Mass, USA, 2007.

[47] A. Schlögl and C. Brunner, "BioSig: a free and open source software library for BCI research," *Computer*, vol. 41, no. 10, pp. 44–50, 2008.

[48] G. Dornhege, J. R. Millan, T. Hinterberger, D. J. McFarland, and K.-R. Müller, Eds., *Towards Brain-Computer Interfacing*, MIT Press, Cambridge, Mass, USA, 2007.

[49] A. Schlögl, F. Chiarugi, E. Cervesato, E. Apostolopoulos, and C. E. Chronaki, "Two-way converter between the HL7 aECG and SCP-ECG data formats using BioSig," in *Proceedings of the Computers in Cardiology Conference (CAR '07)*, pp. 253–256, October 2007.

[50] M. Van de Velde, G. Van Erp, and P. J. M. Cluitmans, "Detection of muscle artefact in the normal human awake EEG," *Electroencephalography and Clinical Neurophysiology*, vol. 107, no. 2, pp. 149–158, 1998.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 363565, 13 pages
doi:10.1155/2011/363565

Research Article

Craniux: A LabVIEW-Based Modular Software Framework for Brain-Machine Interface Research

Alan D. Degenhart,[1] John W. Kelly,[2] Robin C. Ashmore,[3] Jennifer L. Collinger,[3,4] Elizabeth C. Tyler-Kabara,[1,5] Douglas J. Weber,[1,3,4] and Wei Wang[1,3]

[1] *Department of Bioengineering, University of Pittsburgh, Pittsburgh, PA 15219, USA*
[2] *Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA*
[3] *Department of Physical Medicine and Rehabilitation, University of Pittsburgh, Pittsburgh, PA 15213, USA*
[4] *Department of Veterans Affairs, Human Engineering Research Laboratories, Pittsburgh, PA 15206, USA*
[5] *Department of Neurological Surgery, University of Pittsburgh, Pittsburgh, PA 15213, USA*

Correspondence should be addressed to Wei Wang, wangwei3@pitt.edu

Received 1 October 2010; Revised 7 December 2010; Accepted 24 January 2011

Academic Editor: Sylvain Baillet

This paper presents "Craniux," an open-access, open-source software framework for brain-machine interface (BMI) research. Developed in LabVIEW, a high-level graphical programming environment, Craniux offers both out-of-the-box functionality and a modular BMI software framework that is easily extendable. Specifically, it allows researchers to take advantage of multiple features inherent to the LabVIEW environment for on-the-fly data visualization, parallel processing, multithreading, and data saving. This paper introduces the basic features and system architecture of Craniux and describes the validation of the system under real-time BMI operation using simulated and real electrocorticographic (ECoG) signals. Our results indicate that Craniux is able to operate consistently in real time, enabling a seamless work flow to achieve brain control of cursor movement. The Craniux software framework is made available to the scientific research community to provide a LabVIEW-based BMI software platform for future BMI research and development.

1. Introduction

Brain-machine interface (BMI) technology aims to establish a direct link for transmitting information between the brain and external devices. It offers a rich and natural assistive device control interface for individuals with disabilities [1, 2] and is a rapidly-progressing, extremely active research area in the field of neuroscience and neural engineering. Various neural signal modalities, including electroencephalography (EEG) [3], magnetoencephalography (MEG) [4], electrocorticography (ECoG) [5], intracortical local field potentials (LFPs) [6], and neuronal firing rates [7–9], have been used for BMI research. Regardless of the input modality, all BMI systems require an essential suite of software capable of acquiring neural signals continuously and converting them in real time or near real time into specific BMI control commands for an external device, such as a prosthetic hand, in order to accomplish a specific task.

To conduct innovative and unique BMI studies, researchers very often need to implement new signal processing techniques, neural decoding algorithms, or experimental paradigms in a BMI software package. Given the rapid progression of the field, it is desirable to reduce the time it takes from the conception of a new idea to software implementation, data collection, and data analysis. However, the increasing complexity of BMI systems has made this problematic. For example, sophisticated neural decoding algorithms previously studied in offline analysis are now being investigated for real-time BMI control [10]. Additionally, more advanced external devices are being controlled by BMI systems, such as the dexterous prosthetic arm and hand system developed by the Revolutionizing Prosthetics project [11, 12]. These advancements call for an open-source software framework that enables BMI researchers to better focus on the essential engineering and scientific questions they are investigating and to develop advanced

BMI features more efficiently. This framework should be able to manage the basic software operations common to many BMI studies and should be easily extendable in a high-level programming environment that offers the ease and flexibility for programming new BMI modules.

One successful open-source general purpose BMI software package is BCI2000, a modular C++-based system for neural signal acquisition, data saving, stimulus presentation, and more [13], which has been widely distributed among academic institutions and used in numerous research studies [14]. Source code as well as binary executable files are freely available for download, allowing end users to either use the software as is or modify it to suit their own needs. One of the greatest advantages of BCI2000 is its modular, lightweight, and portable design, making it extremely popular and successful in the BMI research community. Recently, the BCPy2000 open-source framework [15] has been made available as a user contribution package to BCI2000. This framework follows the same system architecture as BCI2000, but it allows BMI researchers to develop new modules in Python, a high-level language that greatly reduces software programming complexity for fast prototyping of new BMI software.

This paper presents an open-source open-access real-time BMI software framework inspired by BCI2000 termed "Craniux," developed using LabVIEW (National Instruments, Inc.), a high-level multiplatform graphical development environment. Craniux implements a core framework for BMI operation, including modular architecture, network communications between modules, data flow control, data visualization, data storage, and graphical user interfaces. Craniux offers a unique set of advantages that can greatly facilitate BMI software development and research. First, it enables BMI researchers to develop and share new BMI modules in the LabVIEW development environment and take full advantage of many features inherent to this environment, such as

(i) high-level graphical programming for fast development and run-time debugging,

(ii) a rich set of data visualization options and graphical user interface elements,

(iii) ease of multithreading and parallel processing programming, including automatic parallelism and multicore processor support,

(iv) a large number of high-quality LabVIEW function libraries for signal processing and stream-lined integration with a wide range of engineering hardware (e.g., national instruments controller cards),

(v) reuse and sharing of custom-made LabVIEW modules as sub-VI (virtual instrument) blocks.

Second, facilitated by the above-mentioned LabVIEW features, we have further implemented functionality critical for BMI research

(i) Real-time operation in which the system is capable of acquiring a block of neural data, processing this data, and generating an output before the next block of data is received [13, 16].

(ii) Online neural decoder training capability accomplished through data sharing between real-time operations and parallel decoder training.

(iii) "On-the-fly" data visualization and online experiment parameter control.

(iv) Deterministic control of system execution, including parameter updates and display of visualization data.

(v) Streaming and storage of raw neural data, various intermediate processing data, and experimental parameters to disk for offline analysis.

(vi) Distribution of BMI modules across computers using well-defined generic network communication protocols optimized for data transmission between software modules.

Finally, Craniux has been developed to be a lightweight, extendable, and portable software framework. Its modular architecture, well-defined user interfaces, and generic network communication protocol make it very easy to maintain and develop BMI engines. The existing engines and standard template engines provide a starting point for new engine development.

In the following sections, we will first introduce the basic system architecture of the Craniux software. We will then provide system performance testing results based on both simulated and real experimental data. The last section will further discuss the uniqueness of this software framework as compared to other existing BMI software tools, its advantages and limitations, and future directions.

2. System Architecture

The Craniux software package has been designed to be a highly modularized system, capable of operating across both a distributed network of computers and on a single computer. To accomplish this, and to make data transfer between engines as reliable as possible, all data communication is conducted using the TCP/IP protocol. Data saving is implemented using the LabVIEW TDMS (Technical Data Management Streaming) framework [17], ensuring all system data are streamed to disk as quickly as possible in order to maximize system performance. The following sections describe the system framework, engine execution, GUI operation, communication protocols, and data saving operation in further detail.

2.1. Distributed Engine Framework. Figure 1 depicts the design of the Craniux system. Inspired by the BCI2000 framework, this system consists of five distinct components: the system launcher, acquisition engine, signal processing engine, application engine, and data saving manager and may be distributed across as many as four computers. Furthermore, each engine has an associated graphical user interface (GUI), through which the user interacts with the engine. The main system components perform the following functions.

System Launcher. The system launcher is the initial interface the user is presented with when running the software and allows the user to specify system-level parameters at runtime. It is here that the specific engines, their network locations, and high-level experimental parameters (e.g., subject ID, date, investigators, and session number) are specified. Additionally, the system launcher controls the start and stop of execution though it itself is not a part of the real-time operation of the system.

Acquisition Engine. Acquisition engines are responsible for the acquisition and initial preprocessing (e.g., spectral estimation) of neural data from some signal source such as an amplifier or user datagram protocol (UDP) connection.

Signal Processing Engine. Signal processing engines receive data from the acquisition engine and are responsible for the processing of this data, such as the generation of a control signal.

Application Engine. Application engines receive data from the signal processing engine and are responsible for the control of interaction between the subject and the BMI.

Data Saving Manager. The data saving manager is responsible for the saving of Craniux data and receives input from the acquisition, signal processing, and application engines.

In order to ensure sequential processing of data through each engine, system execution proceeds from the acquisition engine to the signal processing engine, then to the application engine, and finally from the application engine back to the acquisition engine; only one of each type of engine may be running at a given time. This cyclical data flow guarantees that each block of data received by the acquisition engine is processed and a system output is generated before processing of the next block of data begins.

At any point in operation, the system may be suspended and any of the engines replaced with another of the same type, preserving the state of those engines that remain running. This is desirable for BMI operation, as system parameters such as neural decoder weights obtained during operation with a specific application (e.g., a center-out computer cursor task) may be retained and immediately used for a new application (e.g., the control of a robotic arm). Table 1 provides a list of the current engines available in the Craniux system.

2.2. Engine Execution. Each engine in the system operates in a basic sequence, first receiving data from a previous engine, processing the received data, and sending the relevant results of that processing to the next engine in the signal chain. Figure 2 outlines the basic flow of the execution of an individual engine. Engine execution first begins with the initialization of all parameters, including the loading of user-specified parameter files, and the identification of those engine-specific parameters to be saved. From here, execution proceeds into the main sequence of the engine, where the engine (1) waits for data from the previous engine in the signal chain, (2) processes the received data (or performs some other action), and (3) sends the results of this processing to the next engine in the signal chain. Execution then proceeds back to (1), where the engine waits for the next block of input data. Operating in parallel to this main sequence are a number of additional threads, such as data saving, engine-specific processes not capable of or not requiring real-time operation (e.g., neural decoder training), and communication with the engine's GUI. A detailed description of engine execution, including the enforcement of deterministic execution within engine components, is provided in the supplemental materials (see supplementary materials available online at doi:10.1155/2011/363565).

2.3. Graphical User Interface (GUI) Elements. The GUI for each engine is responsible for both on-the-fly control of engine-specific parameters as well as the visualization of engine-specific data. Permitting on-the-fly control is essential to successful BMI operation, as during real-time closed-loop BMI operation it is often necessary to dynamically adjust parameters such as the computer assist level or computer cursor speed [8]. As opposed to a traditional graphical user interface, which simply serves as a front end user interface for a LabVIEW application, GUIs in the Craniux system exist as stand-alone applications. It is through these applications that the user interacts with each engine. GUIs and their associated engines maintain reciprocal two-way communication; parameter value change events are monitored by the GUI and transmitted to its associated engine via TCP/IP, while data to be visualized is transmitted from the engine to the GUI. It should be noted that parameter value changes are instantaneously transmitted from the GUI to the engine and are accessed by the engine at the beginning of its main sequence in order to ensure the consistency of all parameter values throughout the processing of a single block of data. Parameter value changes are also index stamped and saved to disk, allowing the complete reconstruction or replay of the full system state during offline analysis. In order to allow for data visualization on the fly, all data elements are transmitted from the engine to the GUI, providing the experimenter with the most accurate representation of the state of the engine. This occurs in parallel with the real-time main sequence execution, so that this communication does not interfere with the timing of the execution of the main sequence of the engine.

2.4. Communication between Components. Communication between Craniux components utilizes self-establishing and self-repairing network connections that provide efficient, reliable data flow robust to any data type or combination of variables that is sent over them. For communication between engines, these connections take the form of a ring that maintains data flow and controls program execution. For communication between engines and their GUIs or the data saving manager, a single TCP connection is established. When creating new engines, this ring requires no input and single connections only require the developer to provide a network host name. The only user input necessary is the

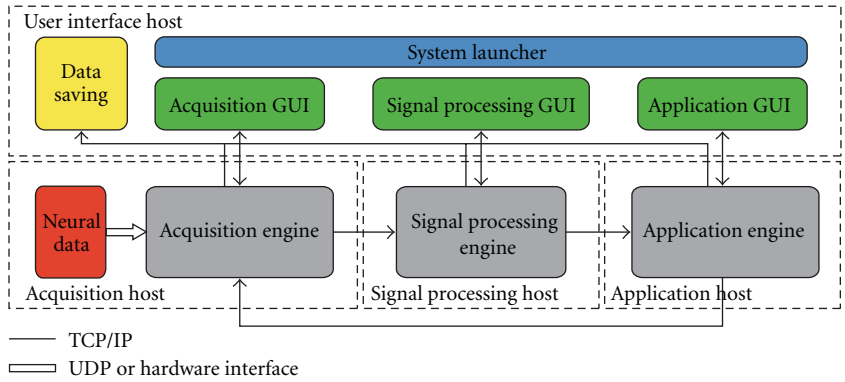


Figure 1: *Craniux system framework.* The Craniux system is comprised of the acquisition, signal processing, and application engines, their associated GUIs, the system launcher, and the data saving manager. Engines and user interface elements are spread across four network hosts: the acquisition host, the signal processing host, the application host, and the user interface host, though the same computer may serve as multiple hosts. Network communication between system engines, as well as communication between engines and GUIs, is performed using the TCP/IP protocol. A block of neural data enters the system through the ACquisition engine, which sends preprocessed data to the Signal processing engine. The signal processing engine generates a control signal, which is then sent to the application engine. The application engine then communicates any relevant application-specific data (e.g., target information used for neural decoder training) back to the acquisition engine, which reads the next block of neural data. Bidirectional data transfer occurs between engine-specific GUIs and their associated engines, with system parameters transferred from the GUI to the engine and visualization data transferred from the engine to the GUI. Finally, the system launcher is responsible for loading the desired engines, tracking general experimental parameters, and experimental control.

Table 1: List of current acquisition, signal processing, and application engines.

Engine name	Engine type	Description
Acquisition template	Acquisition	"Empty" acquisition engine generating random data used to maintain system dataflow
Read UDP binary	Acquisition	Reads raw neural data transmitted via UDP
SimECoG	Acquisition	Generates synthetic ECoG data
Signal processing template	Signal processing	"Empty" signal processing engine used to maintain system dataflow
Linear decoder	Signal processing	Generates a control signal using linear combinations of input features
Population vector	Signal processing	Generates a control signal using the population vector algorithm [18]
OLE	Signal processing	Generates a control signal using the optimal linear estimator algorithm [19]
Application template	Application	"Empty" application engine used to maintain system dataflow
Center-out cursor control	Application	Two or three-dimensional cursor control application
Threshold Crossing	Application	Sends UDP commands to an external device when control signals cross user-defined thresholds
Circle drawing	Application	Circle/ellipse-drawing application [20]
Biofeedback	Application	Displays real-time feedback of a neural control signal to the subject

IP address of each engine, which is specified on the system Launcher. Available ports are automatically selected for each connection.

All network connections use the TCP protocol. TCP was chosen over UDP because its superior reliability is important in a ring structure responsible for the control of program execution; a dropped packet between engines would break the ring and leave each engine waiting for data that will never arrive. It is also important to note that Nagle's algorithm [21] was disabled for all connections used in the Craniux system. The Nagle algorithm attempts to reduce TCP packet overhead and bandwidth usage by intentionally delaying transmission, so that multiple packets can be combined before being sent. Here, the latency introduced by this algorithm is unacceptable, and bandwidth usage is not a concern. The concept behind Nagle's algorithm is retained in our system; however, as all data to be sent simultaneously is combined into a single packet before transmission.

To send variables over the network, the developer must only create a list of the variable names on the sending side of the connection. No information on variable type or size is needed. The provided variable names are packed

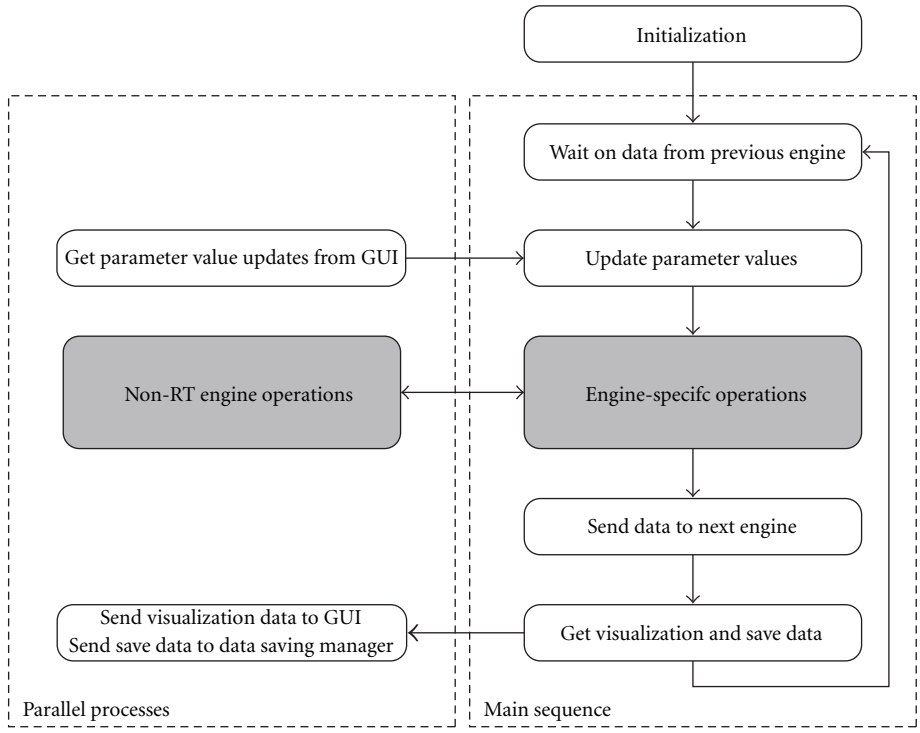


Figure 2: *Engine execution.* After initialization, each engine proceeds into the main sequence loop, in which core engine processes are executed sequentially. Data are first received from the previous engine in the signal chain, and any parameter value changes received from the engine's GUI are updated locally. The system next performs any actions specific to the individual engine (e.g., calculation of a control signal or updating of a display), and sends the results of these actions to the next engine in the signal chain. Current values of any data items to be visualized are placed in a queue, and data is sent to the data saving manager. The engine then proceeds to the beginning of the main sequence loop to await the arrival of the next input. Parallel to the main sequence loop are any parallel processes designed to operate asynchronously. These processes will always include receiving parameter value updates from the GUI and sending visualization data to the GUI, and may include individual engine-specific operation such as decoder training or monitoring for events. Shaded blocks represent those areas to be modified by the developer during the creation of new engines, while white blocks represent sections of code providing core functionality.

together with their values into a single variable of LabVIEW's "variant" data type, which is then sent over the network. On the receiving side of the connection, the data is read and parsed into the correct values, which are written to those existing variables on the receiving side with the same name and data type as the sent variables. Additional information on the transfer of information between components has been provided in the supplemental materials.

2.5. Data Saving. The Craniux framework for saving data is a reliable process that minimizes latency introduced by saving and creates highly accessible data. Data saving is conducted by an independent data saving manager, which receives data from all engines. This data is initially saved in LabVIEW's TDMS format, which was specifically created for quickly and continuously streaming large amounts of data to the hard drive to help eliminate data-saving bottlenecks in speed normally introduced by slow writes to disk [17]. When saving data, a packet containing all the variable values to be saved and the data packet index is placed into a first-in-first-out (FIFO) buffer. Parallel to the main execution of Craniux, these packets are removed from the buffer and sent to the data saving manager, located on the user interface host, via the communications framework described in Section 2.4. Upon receiving a packet, the data saving manager streams the data to a TDMS file. When creating new engines, it is only necessary to provide a list of variable names to be saved; these items will be automatically identified and their values saved accordingly.

A single TDMS file is saved for each experimental run; stopping or suspending system execution closes all references to the current data file. Within each data file, data saved by each engine is separated into two groups: sampled variables and controls. Sampled variables are data sampled continuously at each update of the BMI system, such as cursor position during a brain-controlled cursor movement task. As controls are normally parameter settings that are infrequently updated (e.g., the number of targets), these values are only saved when changed. The current data packet number is included in every save operation, so that the experiment can be reconstructed afterwards with the data properly aligned in time. A separate LabVIEW VI has been created to convert Craniux TDMS files into the MATLAB (Mathworks, Inc.) MAT format. These MAT files

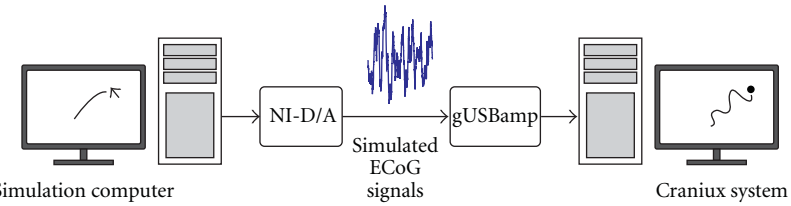


Figure 3: *Simulated ECoG experimental setup.* Experimenter-controlled mouse position on the simulation computer modulates the high-gamma power of simulated directionally tuned ECoG signals. These signals are output at 2400 Hz using a National Instruments D/A card and are read into the Craniux system using the g.USBamp amplification system and BCI2000. The Craniux system then decodes the desired cursor position from the simulated signals using the population vector algorithm.

contain structures for each engine paired with each data type (sampled variables and controls). The saved values for each variable are stored in an array, with the data packet number array providing the time index for each element. Array variables are stored as cell arrays, allowing them to be aligned with their associated data packet numbers and enabling the data structure to handle dynamic changing of array sizes during a BMI session.

3. System Validation

3.1. Closed-Loop Cursor Movement Control Using Simulated ECoG Signals. The experimental setup used for validation of the Craniux system is shown in Figure 3. An electrocorticographic (ECoG) signal simulator in which experimenter-controlled mouse cursor movement was used to modulate the high gamma-band activity of a number of synthetic signals. This simulator is capable of generating 32 channels of analog signals with directionally tuned high-gamma band (70–120 Hz) activity emulating ECoG signals recorded from human subjects [2, 22] according to the following [6, 18],

$$S = S_1 + d\cos(\theta)S_2, \tag{1}$$

where S is a single simulated directionally modulated ECoG signal, S_1 is a pink noise signal with a 1/frequency power falloff [23]. S_2 is a second pink noise signal band-pass filtered between 70 and 120 Hz, d controls the depth of modulation of the high-gamma band, and θ is the angle between the preferred direction of the simulated ECoG signal and the vector pointing from the center of the computer screen to the current mouse cursor position on the computer screen. The preferred directions of the 32 simulated signals were uniformly distributed over two-dimensional (2D) space. Simulated signals were generated at 2400 Hz using a National Instruments NI PCI-6723 32 channel analog output board on a simulation computer (Windows XP x86 operating system, AMD Athlon 64 FX-62 Dual Core CPU @ 2.81 GHz, 3.5 GB RAM, NVIDIA GeForce 7900 GS video card) and then stepped down to match the amplitude of typical ECoG signals recorded from human subjects.

Simulated ECoG signals were then sampled at 1200 Hz using the g.USBamp amplification system (Guger Technologies, OEG) on a separate computer (Windows XP x86 operating system, Intel Core i7 CPU 920 @ 2.67 GHz, 2.49 GB RAM, 2 NVIDIA GeForce 9800 GT video cards) and sent to the Craniux system as binary UDP packets using a simplified version of the BCI2000 software package. BCI2000 was used in this case due to its reliability and efficiency in interfacing with the g.USBamp amplification system. These raw time-domain signals entering the Craniux system were first converted into the frequency domain using LabVIEW's built-in autoregressive (AR) spectral estimation function (10 Hz bins, 500 ms window) in the read UDP binary acquisition engine and then passed to the population vector signal processing engine. Here, signals were normalized to pseudo-Z-scores based on the following [24, 25],

$$f_{\text{norm},i,j} = \frac{f_{i,j} - \overline{f}_{i,j}}{\sigma_{i,j}}, \tag{2}$$

where $f_{\text{norm},i,j}$, $f_{i,j}$, and $\overline{f}_{i,j}$ are the normalized, raw, and mean power of the ith channel and jth frequency band, respectively, and $\sigma_{i,j}$ is the standard deviation of the raw band power of the ith channel and jth frequency band. Mean and standard deviation values were calculated based on data collected during a baseline condition in which the computer cursor on the simulation computer remained in the center of the screen (i.e., no modulation of high-gamma band activity).

The brain control task used was a typical 2D center-out design, with the movement direction of a cursor controlled by multiple ECoG signal features across 32 channels according to the population vector algorithm [18]

$$\begin{aligned} f_i &= b_{0,i} + b_{x,i}m_x + b_{y,i}m_y, \\ P &= \sum_i^N (d_i - b_{0,i})\mathbf{C}_i, \end{aligned} \tag{3}$$

where f_i is the activity of individual feature i, m_x and m_y are the desired movement in the x and y direction, $b_{0,i}$, $b_{x,i}$, and $b_{y,i}$ are coefficients found using linear regression relating desired movement to the activity of feature i, P_i is the trajectory vector predicted by the activity of feature i, d_i is the instantaneous activity of feature i, and $\mathbf{C}_i = [b_{x,i}\ b_{y,i}]/(b_{x,i}^2 + b_{y,i}^2)^{1/2}$ is a vector representing the preferred direction of feature i.

The standard workflow used to achieve ECoG-controlled 2D cursor movement with the Craniux framework is described below. Though simulated ECoG signals were used

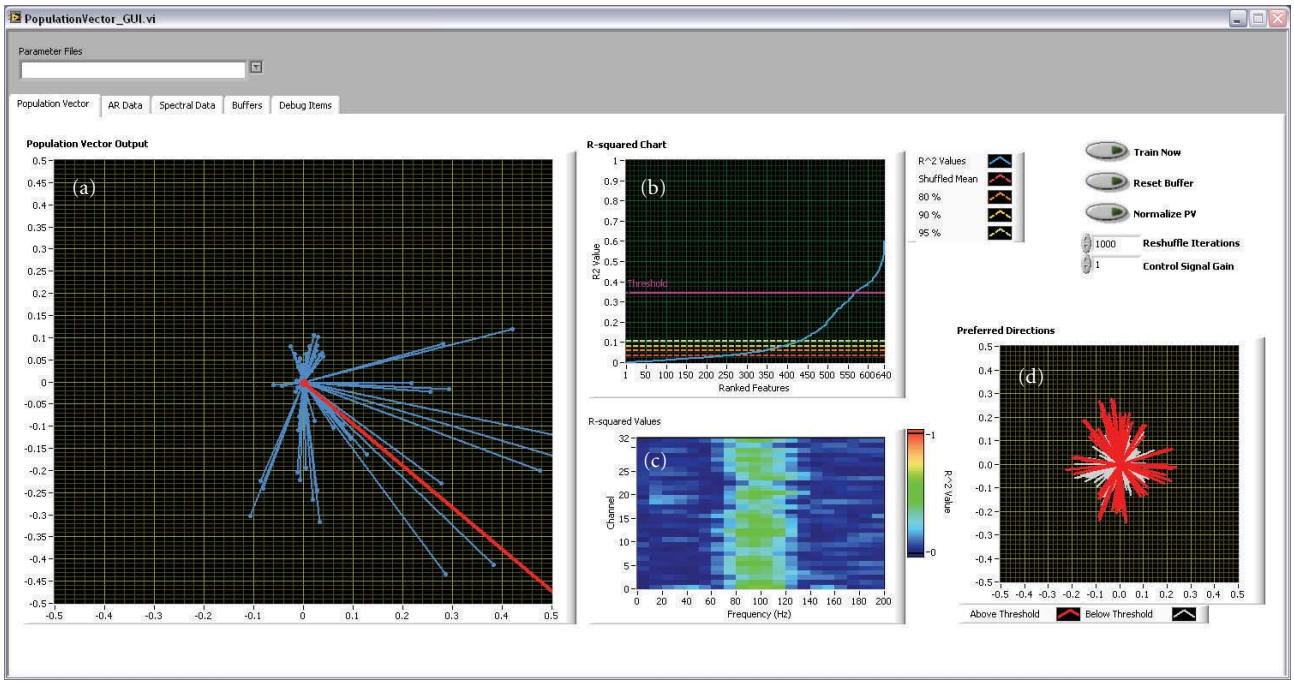


Figure 4: *Craniux system screenshot during population vector-based control.* (a) Plot of the instantaneous activity of each feature used for cursor control along its preferred direction (blue) and the resultant population vector (red). (b) R^2 value plot indicating the distribution of R^2 values obtained during population vector training (blue) compared to the mean, 80th, 90th, and 95th percentile R^2 values obtained after training on 1000 iterations of target-shuffled data (red, dark orange, light orange, and yellow lines). The threshold above which features are chosen for use in the decoder is shown by the pink line. (c) R^2 values obtained during population vector training arranged by channel and frequency band. Note that the 70–120 Hz frequency band features show high R^2 values across all channels, consistent with the method used to generate the simulated ECoG signals. (d) The preferred direction distribution of all features. Red lines correspond to those features with R^2 values above the user-determined threshold, while white lines are those features falling below the threshold.

here to validate the system, this workflow will be similar for real neural signals.

(1) *Collection of Baseline Data.* Once the Craniux system is started, approximately 3 minutes of baseline data is collected, from which the Craniux system will calculate feature mean and standard deviation values. These will then be used in the calculation of pseudo-Z-scores for all ECoG signal features in real time.

(2) *Collection of Training Data for the Neural Decoder.* During this period, the experimenter will use the ECoG signal simulator to generate modulated ECoG signals based on the target position (i.e., desired cursor movement direction). The ECoG data along with target position are automatically buffered by Craniux for neural decoder training.

(3) *Training of the Neural Decoder.* During this period, the buffered data is used to train the neural decoder. A multiple linear regression procedure is used to determine the degree of directional tuning and preferred direction for each ECoG signal feature as mentioned above [26, 27]. The resulting R-squared values and preferred directions are displayed by the population vector GUI, allowing experimenters to visualize the results on the fly and interactively select a subset of directionally tuned ECoG signal features for brain control.

(4) *Real-Time Brain Control.* Activities of ECoG signal features selected during step (3) are then used to generate the population vector, a 2D velocity control signal that drives the cursor. Figure 4 shows the population vector GUI during closed-loop brain control, illustrating the user interface elements provided to the user during this process.

It is worth noting that all the above procedures are conducted in a continuous BMI session without stopping and restarting the Craniux system. This streamlined workflow allows BMI studies to be conducted smoothly and efficiently. Furthermore, steps (2) and (3) can be conducted at any time during a BMI session in parallel with step (4). This allows the neural decoder to be recalibrated on the fly to adapt to any potential changes or nonstationarities of input neural signals, a key element for achieving and maintaining reliable brain control [28]. Figure 5 shows an example of directionally modulated normalized time-frequency data for one ECoG signal saved by the Craniux system, as well as trajectories of the cursor during real-time brain control.

3.2. Brain-Controlled Cursor Movement Using Real ECoG Signals Recorded from a Human Subject. Further validation of the Craniux system was conducted in a human subject undergoing subdural epilepsy monitoring. Informed consent was obtained from the subject prior to testing; all experimental procedures were approved by the University of Pittsburgh Institutional Review Board and followed all guidelines for human subject research. Experimental methods used were similar to those presented in [29], with the exception that the Craniux system was used for data collection and brain control. Standard ECoG electrodes exhibiting high-gamma band modulation in response to overt movement screening tasks were chosen for use in closed-loop control. High-gamma band power (70–110 Hz) of two neighboring ECoG electrodes was used to control the vertical movement of a cursor with a push-pull scheme, with the cursor control signal calculated according to:

$$c_y = a(s_1 - s_2) - b, \tag{4}$$

where c_y is the one-dimensional control signal, s_1 and s_2 are the high-gamma band power of the two neighboring electrodes used for control, and a and b are gain and offset terms used to normalize the control signal to zero mean and unit variance. Thus, in order to achieve satisfactory brain control, the subject had to decorrelate the activity of the two electrodes to generate the desired cursor control signal. Brain control sessions began with the collection of baseline data for normalization purposes as described in the previous section. Individual trials began with the placement of the cursor at the center of the computer screen along with the presentation of one of two peripheral targets located in the vertical plane of the workspace (e.g., a "center-out" task). Trials in which the subject was able to hit the presented target within the maximum trial length of 10 seconds were deemed successful; failure to do so resulted in an unsuccessful trial. All trials were followed by an inter trial interval of 2 seconds in which neither the cursor nor the target were visible. Figure 6 shows the results of one brain control session, during which the subject was able to achieve an 88% success rate.

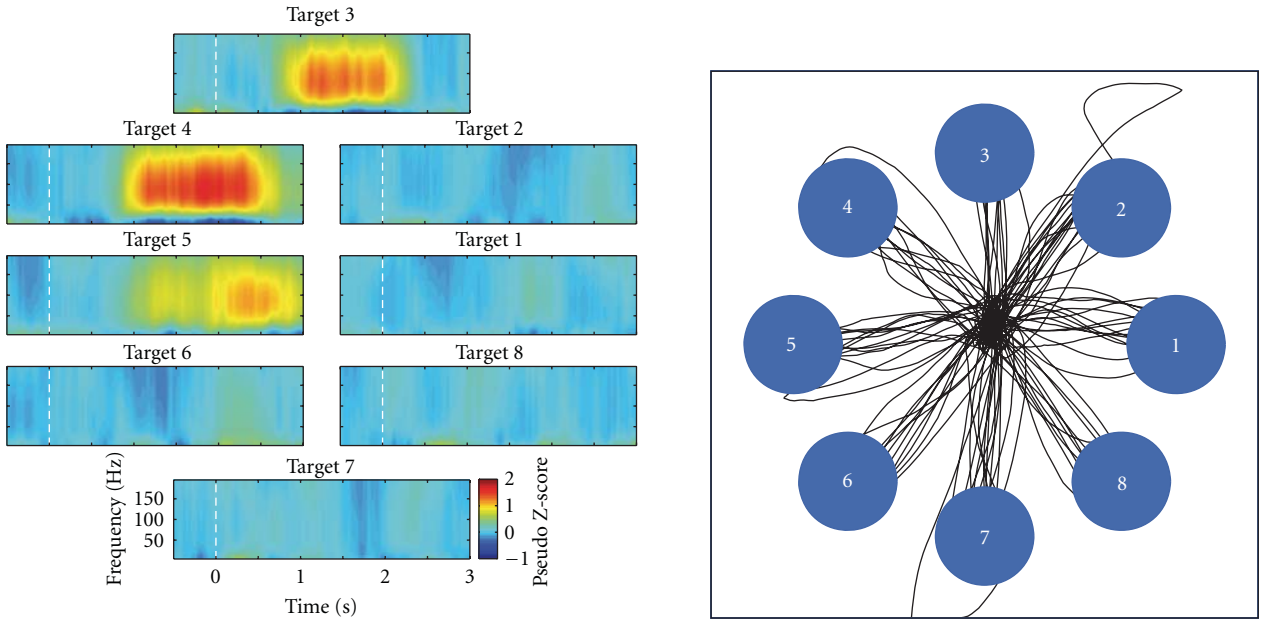
3.3. System Timing. To evaluate the consistency of system performance, timing characteristics were analyzed for a typical Craniux setup using 15th order autoregressive (AR) spectral estimation of 10 Hz frequency bins over 0.5 second windows of simulated neural data (see Section 3.1), the linear decoder signal processing engine, and the center-out cursor control application engine. Analog and digital data were sampled by the g.USBamp amplification system at 1200 Hz and acquired directly by Craniux at a 33.3 ms frame rate. To trigger timing events, a digital signal was sent from Craniux back to the digital input of the amplifiers, so that the timing events could be acquired precisely and synchronously with the raw analog input signal at 1200 Hz.

Three different timing tests were conducted: a system processing test, a display update rate test, and an overall system latency test. The first two tests (system processing and display update rate) were performed on both a single computer (Windows XP x86 operating system, Intel Core i7 CPU 920 @ 2.67 GHz, 2.49 GB RAM, 2 NVIDIA GeForce 9800 GT video cards) and with Craniux distributed across the network, so that data acquisition, spectral estimation, and GUIs were hosted on one computer (the same as that used for local timing test, see above) while signal processing and Application engines, including the 3D render window, were hosted on a separate computer (Windows XP x86 operating system, AMD Athlon 64 FX-62 Dual Core CPU @ 2.81 GHz, 3.5 GB RAM, NVIDIA GeForce 7900 GS video card). For both configurations, tests were conducted using 16, 32, and 64 channels of data. The third test (system latency) was run only on the single-computer configuration with processing performed on 32 channels of data.

The first test used 5,000 consecutive frames of collected data to measure the system processing time, the time between the arrival of a block of data from the amplifier, and the time when the Craniux system had finished all processing on the data and begun waiting on the next block. These results are shown in the second column of Table 2. As expected, processing time was found to increase with the number of processed channels but remained below the 33.3 ms time required to maintain a consistent frame rate and prevent the loss of data. Distributing Craniux across the network showed improvements in processing time for all channel configurations. Since processing time is only required to remain below the frame rate, running Craniux as a distributed system is not necessary unless the system is under a heavy load. AR spectral estimation was found to require the most processing time, especially as the number of channels increased. These results indicate the extra processing time made available when Craniux is run as a distributed system could easily be utilized to run more complex signal processing algorithms or to decrease the frame rate.

The second test also used 5,000 consecutive frames of data but now measured the refresh rate, the amount of time between consecutive display updates on the center-out cursor control engine. The results are shown in the third column of Table 2. The refresh time was found to be 33.3 ms for all configurations, precisely what would be expected given the system frame rate. Furthermore, the low variability of this timing indicates that the user would experience a consistent cursor update with no noticeable jitter.

The final test measured system latency, the elapsed time between a neural signal event, and the point in time when the Craniux system can generate an action in response to this event. A 10 Hz sine wave with zero offset was input to 1 channel of the amplifier; this channel of data was fed through the Craniux system to the point at which the display was updated in the center-out cursor control engine, occurring just before processing fully completes and the system begins waiting on the next data block. At this point, if a zero crossing was detected on the sine wave, the digital output bit being written back to the amplifier was flipped. In this case, the elapsed time between a zero crossing of the sine wave (a simulated neural event) and the bit value change (the time of the system response) indicates the system latency. Data was collected for 5,000 consecutive sine wave zero-crossings, with zero crossing events symmetrically distributed about the center of each 33.3 ms data frame. In distributing the zero



(a) (b)

Figure 5: *Closed-loop brain control using simulated ECoG data.* (a) Time-frequency plots of a single simulated ECoG signal averaged across all repetitions of an 8-target center-out cursor control task. Plots are aligned to target presentation at time $t = 0$ (dashed white line). In all, a total of 32 channels of simulated directionally tuned ECoG signals were generated. (b) Real-time cursor trajectories controlled by simulated ECoG signals using the population vector algorithm.

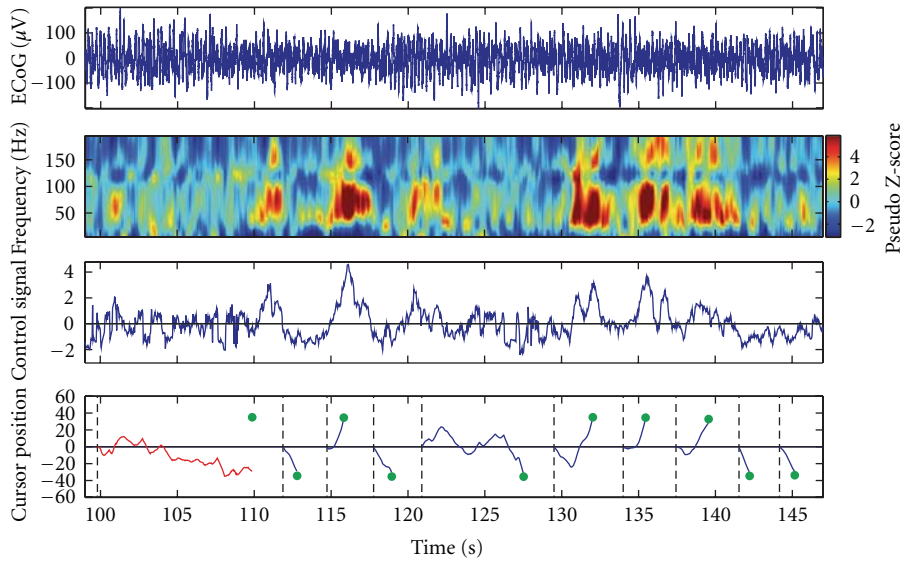


Figure 6: *Closed loop ECoG-based computer cursor control.* One-dimensional computer cursor control using the Craniux system in a subject implanted with ECoG electrodes. Top. Raw time-domain ECoG signal for one of two electrodes used for cursor control. Top-middle. Time frequency data saved by the Craniux system for the same electrode. Bottom-middle. Control signal generated by the Craniux system to control computer cursor movement. Positive control signal values move the cursor in the up direction, while negative control signal values will move the computer cursor down. Note that control signal values are unitless as they have been normalized to zero-mean and unit variance. Bottom. Vertical cursor positions generated by the neural control signal. Dashed black lines represent target onset, green circles indicate the position of presented targets, blue lines indicate cursor trajectories for successful trials, and red lines indicate cursor trajectories for unsuccessful trials.

Table 2: *Characterization of system timing.* Craniux system processing time, refresh rate, and latency for local and network system configurations under various processing loads. Values shown are mean timing values plus or minus one standard deviation from the mean.

System configuration	Processing time (ms)	Refresh rate (ms)	Latency (ms)
Local, 16 channels	12.8 ± 0.7	33.3 ± 0.5	N/A
Network, 16 channels	9.9 ± 0.6	33.3 ± 0.5	N/A
Local, 32 channels	17.8 ± 0.8	33.3 ± 0.7	33.2 ± 9.6
Network, 32 channels	15.2 ± 1.0	33.3 ± 0.4	N/A
Local, 64 channels	28.0 ± 1.3	33.3 ± 0.7	N/A
Network, 64 channels	24.9 ± 0.9	33.3 ± 0.4	N/A

crossings in this way, it is known that the latency should have an average of slightly less than half the frame length plus the mean processing time (33.9 ms for this configuration) and a range nearly equal to the frame length. The latency was found to have a mean and standard deviation of 33.2± 9.6 ms, meeting all expectations.

4. Discussion

Craniux is a powerful, yet simple and easily extendable, open-source framework for BMI studies that require high-performance real-time BMI software. Currently, a number of open-source software solutions for BMI research are available for academic use. These software packages include extremely specialized, high-overhead systems used in nonhuman primate BMI research [8, 28, 30], highly-modular, visual-programming-based software platforms such as OpenViBE [31], as well as portable, lightweight systems for human BMI research [13]. Software tools for more specific BMI research applications have also been made available, from toolboxes allowing for the interfacing of MEG systems in real time for BMI use [32, 33] to real-time brain mapping software capable of quickly identifying signals from electrocorticographic electrodes related to cortical activity corresponding to overt movement, speech, and sensory stimulation [34, 35].

The Craniux framework is inspired by the system architecture design of BCI2000, and we believe that it takes advantage of several unique features of the LabVIEW graphical development environment for developing real-time BMI software. By making it an open-access and open-source software framework, we hope to serve the research community on at least two fronts. First, at the basic level, Craniux is a BMI software solution with an easy-to-use graphical user interface. Those researchers interested in BMIs can use this software to conduct research without writing custom software. Second and most importantly, we hope this framework will facilitate the development of new BMI paradigms and signal processing algorithms by the research community through providing the basic functionalities of BMI system operation, allowing researchers to focus on the development of their specific research questions. Finally, as this framework is set up in the LabVIEW environment, it naturally inherits the many advantages offered by the high-level graphical nature of LabVIEW programming.

In its current form, the Craniux framework demonstrates the benefits and ease with which it can be used and modified to develop new BMI paradigms and algorithms. The simplicity of the LabVIEW programming language makes the creation of new BMI engines accessible to individuals who may not be familiar with object-oriented programming. Would-be developers can simply take one of the provided engine templates, implement their desired operation, and save the engine under a new name (this process is described in greater detail in the supplemental materials). This new engine will then be available for use in the Craniux framework, without the need for the compiling of code down to executables as required by programming languages such as C/C++. The debugging of newly created engines can also be easily performed during run time through the use of LabVIEW's built-in debugging tools. The dataflow-driven nature of Craniux further simplifies debugging, allowing system execution to be halted and resumed at any point during operation without the loss of the current state of the system. These tools, along with advanced data visualization options, make the rapid prototyping of highly sophisticated neural signal processing techniques possible.

Craniux currently offers a rich set of options to visualize BMI data on the fly at multiple processing stages in various formats. Neural signals, such as EEG, MEG, or ECoG, can be viewed as scrolling time-frequency plots or dynamic spatiotemporal plots in the frequency domain. This is beneficial for online examination of neural signal quality, as certain features may be difficult to view in a simple plot of time-domain raw neural signals. The results of calculations performed during the training and application of neural decoding algorithms can also be visualized on the fly, providing researchers with the opportunity to select neural signal features, visualize decoding weights, and examine decoder outputs without suspending operation of the system. For example, our implementation of the population vector algorithm allows researchers to dynamically change the value of the R-squared threshold used for feature selection, view the preferred direction distribution of the currently selected features, and view the instantaneous contribution of all features to the control signal output by the algorithm. This visualization capability is of particular importance when using and developing sophisticated decoding algorithms, as it allows BMI researchers to judge the validity of the decoding weights on the fly and make adjustments of neural signal

processing and other BMI experiment parameters accordingly.

We have also shown the potential for enhancement in Craniux performance through its distribution across multiple network hosts, as assigning individual engines to separate computers eliminates the possibility of competition between engines for system resources. The separation of graphical user interface elements from real-time engines further improves system performance by ensuring the real-time engine execution is not affected by user interface interaction events or data visualization. Furthermore, the capability to distribute the Craniux system across multiple network hosts could prove especially useful in long-term human BMI studies. Experimental sessions could be run remotely on a daily basis, eliminating the need for either subjects or investigators to travel to participate in these sessions. This will become important as BMI technology moves into preclinical and clinical trials.

Craniux also offers a streamlined workflow for BMI research. It allows for on-the-fly control of specific experimental parameters, offering experimenters great flexibility for BMI user training. For example, an experimenter can quickly adjust the output gain of a neural decoder if it is deemed that a brain-controlled cursor is moving in the correct direction but with a very low speed. In our experience, this flexibility is critical for effective BMI training. Meanwhile, the Craniux system is capable of capturing all changes in experimental parameters along with BMI data, allowing researchers to perform offline analysis of BMI sessions. Furthermore, various BMI procedures, including the collection of baseline data for the normalization of neural signals and the training of neural decoding algorithms, may be performed without the cessation of system operation. This provides both BMI researchers and experimental subjects with a seamless experience in which system parameters can be continuously updated to improve BMI performance.

It should be noted that the timing of the Craniux system is dependent on the timing of the acquisition engine, which currently can be driven by UDP packets sent from neural acquisition hardware or controlled explicitly by the acquisition engine itself (e.g., the "SimECoG" engine). Any number of neural recording hardware solutions may be used for BMI operation provided that data recorded by these devices can be packaged and transmitted via UDP. Additionally, hardware-specific acquisition engines can also be created within the Craniux framework.

In addition, it is important to mention that editing or developing new BMI engines in the Craniux framework requires the purchase of LabVIEW. However, it is not uncommon for open-source research tools to be built upon commercial software; two such examples are the EEGLAB [36] and FieldTrip packages for neural data analysis. These packages are both built upon MATLAB, an extremely powerful commercial data analysis software package. Just as many researchers are now using MATLAB instead of custom-written C programs for data analysis, we believe that the time and effort saved by the use of the Craniux system in BMI software development will outweigh the cost of the LabVIEW software. Furthermore, if the Craniux software is to be used as a self-contained out-of-the-box software package, all engines can be compiled down to binary executable files and run using the freely available LabVIEW runtime engine, eliminating the need for the LabVIEW software. Finally, as demonstrated in Section 3, given the computing capability of current personal computers and the code optimization inherently performed by the LabVIEW environment, the overall performance of the Craniux system is comparable to BMI systems developed using other programming languages. Hence, the gain from using the high-level LabVIEW programming environment does not come at the expense of significant sacrifices in system performance.

The Craniux software package, including in-depth documentation and detailed operation instructions for all engines, has been made available free of charge to academic institutions and can be accessed at http://hrnel.pitt.edu/Software.html. The Craniux software package can be downloaded as a library of LabVIEW virtual instruments (VIs), and all stable system updates will be made available for download.

5. Conclusion

While other open-access open-source BMI software solutions are currently available, we feel that the Craniux software package fills a specific need in the realm of BMI research. Powerful yet lightweight, this system allows experimenters to rapidly develop and test cutting-edge technology in an online environment, whether it is new neural signal processing techniques, new neural decoders, or advanced prosthetic devices. This system offers an easy-to-use "out-of-the-box" solution for BMI research as well as other neural data visualization and processing purposes. Additionally, the Craniux system provides an extendable framework through the provision of template engines. The provided framework possesses the basic fundamental architecture for running closed-loop BMI experiments and enables other researchers to take advantage of LabVIEW functionality to design and conduct novel experimental paradigms without the need to implement their own core system framework. It is also worth noting that functionality offered by the Craniux framework also lends itself useful for other neuroscience research and even neurorehabilitation applications that could benefit from real-time processing and visualization of neural data, such as cortical source imaging using EEG or MEG recordings. It is with these characteristics in mind that we feel the Craniux software package will prove an important addition to the BMI research community.

Acknowledgments

The authors thank Dr. Gerwin Schalk for helpful discussions regarding BCI2000 and real-time BMI software in general. They would also like to thank the subject whose data is presented here for volunteering to participate in this study. This work was supported by Telemedicine and Advanced Technology Research Center (TATRC) of the US Army Medical Research and Material Command Agreement W81XWH-07-1-0716. It is supported by a special grant from the Office

of the Senior Vice Chancellor for the Health Sciences at the University of Pittsburgh. This publication was also made possible by Grant nos. 5 UL1 RR024153 and KL2 RR024154 from the National Center for Research Resources (NCRR), a component of the National Institutes of Health (NIH), and NIH Roadmap for Medical Research and was supported by a National Defense Science and Engineering Graduate Fellowship, sponsored by the Air Force Office of Scientific Research, a National Science Foundation (NSF) Graduate Research Fellowship, and the Quality of Life Technology Center, under NSF Grant no. EEEC-0540865. Its contents are solely the responsibility of the authors and do not necessarily represent the official view of NCRR or NIH. Information on NCRR is available at http://www.ncrr.nih.gov/. Information on Re-engineering the Clinical Research Enterprise can be obtained from http://nihroadmap.nih.gov/clinicalresearch/overview-translational.asp. This project is also supported by the Craig H. Neilsen Foundation. Additional funding support was provided by NIH grants from the NIBIB (1R01EB007749) and NINDS (1R21NS056136). This material is supported in part by the Office of Research and Development, Rehabilitation Research & Development Service, VA Center of Excellence in Wheelchairs and Associated Rehab Engineering, Grant no. B6789C. The contents of this publication do not represent the views of the Department of Veterans Affairs or the United States Government.

References

[1] A. B. Schwartz, X. T. Cui, D. Weber, and D. W. Moran, "Brain-controlled interfaces: movement restoration with neural prosthetics," *Neuron*, vol. 52, no. 1, pp. 205–220, 2006.

[2] W. Wang, J. L. Collinger, M. A. Perez et al., "Neural interface technology for rehabilitation: exploiting and promoting neuroplasticity," *Physical Medicine and Rehabilitation Clinics of North America*, vol. 21, no. 1, pp. 157–178, 2010.

[3] D. J. McFarland, W. A. Sarnacki, and J. R. Wolpaw, "Electroencephalographic (EEG) control of three-dimensional movement," *Journal of Neural Engineering*, vol. 7, no. 3, Article ID 036007, 2010.

[4] J. Mellinger, G. Schalk, C. Braun et al., "An MEG-based brain-computer interface (BCI)," *NeuroImage*, vol. 36, no. 3, pp. 581–593, 2007.

[5] G. Schalk, K. J. Miller, N. R. Anderson et al., "Two-dimensional movement control using electrocorticographic signals in humans," *Journal of Neural Engineering*, vol. 5, no. 1, pp. 75–84, 2008.

[6] D. A. Heldman, W. Wang, S. S. Chan, and D. W. Moran, "Local field potential spectral tuning in motor cortex during reaching," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 14, no. 2, pp. 180–183, 2006.

[7] L. R. Hochberg, M. D. Serruya, G. M. Friehs et al., "Neuronal ensemble control of prosthetic devices by a human with tetraplegia," *Nature*, vol. 442, no. 7099, pp. 164–171, 2006.

[8] M. Velliste, S. Perel, M. C. Spalding, A. S. Whitford, and A. B. Schwartz, "Cortical control of a prosthetic arm for self-feeding," *Nature*, vol. 453, no. 7198, pp. 1098–1101, 2008.

[9] K. Ganguly and J. M. Carmena, "Emergence of a stable cortical map for neuroprosthetic control," *PLoS Biology*, vol. 7, no. 7, Article ID e1000153, 2009.

[10] S. Koyama, S. M. Chase, A. S. Whitford, M. Velliste, A. B. Schwartz, and R. E. Kass, "Comparison of brain-computer interface decoding algorithms in open-loop and closed-loop control," *Journal of Computational Neuroscience*, vol. 29, pp. 73–87, 2010.

[11] S. Adee, "Dean Kamen's 'luke arm' prosthesis readies for clinical trials," *IEEE Spectrum*, February 2008, http://spectrum.ieee.org/biomedical/bionics/dean-kamens-luke-arm-prosthesis-readies-for-clinical-trials.

[12] S. Adee, "Winner: the revolution will be prosthetized," *IEEE Spectrum*, January 2009, http://spectrum.ieee.org/robotics/medical-robots/winner-the-revolution-will-be-prosthetized.

[13] G. Schalk, D. J. McFarland, T. Hinterberger, N. Birbaumer, and J. R. Wolpaw, "BCI2000: a general-purpose brain-computer interface (BCI) system," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 6, pp. 1034–1043, 2004.

[14] G. Schalk, "BCI2000 provided the basis for experiments in the following peer-reviewed journal papers," *BCI2000 Website*.

[15] "Bcpy2000," August 2010, http://bci2000.org/downloads/BCPy2000/BCPy2000.html.

[16] J. A. Wilson, J. Mellinger, G. Schalk, and J. Williams, "A procedure for measuring latencies in brain computer interfaces," *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 7, pp. 1785–1797, 2010.

[17] "The NI TDMS file format," National Instruments, August 2010, http://zone.ni.com/devzone/cda/tut/p/id/3727.

[18] A. P. Georgopoulos, A. B. Schwartz, and R. E. Kettner, "Neuronal population coding on movement direction," *Science*, vol. 233, no. 4771, pp. 1416–1419, 1986.

[19] E. Salinas and L. F. Abbott, "Vector reconstruction from firing rates," *Journal of Computational Neuroscience*, vol. 1, no. 1-2, pp. 89–107, 1994.

[20] A. B. Schwartz, D. W. Moran, and G. A. Reina, "Differential representation of perception and action in the frontal cortex," *Science*, vol. 303, no. 5656, pp. 380–383, 2004.

[21] J. Nagle, "Congestion control in IP/TCP internetworks," January 1984, http://tools.ietf.org/html/rfc896.

[22] E. C. Leuthardt, G. Schalk, J. R. Wolpaw, J. G. Ojemann, and D. W. Moran, "A brain-computer interface using electrocorticographic signals in humans," *Journal of Neural Engineering*, vol. 1, no. 2, pp. 63–71, 2004.

[23] M. S. Keshner, "1/f NOISE," *Proceedings of the IEEE*, vol. 70, no. 3, pp. 212–218, 1982.

[24] C. Tallon-Baudry, O. Bertrand, M. A. Hénaff, J. Isnard, and C. Fischer, "Attention modulates gamma-band oscillations differently in the human lateral occipital cortex and fusiform gyrus," *Cerebral Cortex*, vol. 15, no. 5, pp. 654–662, 2005.

[25] E. Edwards, S. S. Nagarajan, S. S. Dalal et al., "Spatiotemporal imaging of cortical activation during verb generation and picture naming," *NeuroImage*, vol. 50, no. 1, pp. 291–301, 2010.

[26] A. B. Schwartz, R. E. Kettner, and A. P. Georgopoulos, "Primate motor cortex and free arm movements to visual targets in three-dimensional space. I. Relations between single cell discharge and direction of movement," *Journal of Neuroscience*, vol. 8, no. 8, pp. 2913–2927, 1988.

[27] W. Wang, S. S. Chan, D. A. Heldman, and D. W. Moran, "Motor cortical representation of position and velocity during reaching," *Journal of Neurophysiology*, vol. 97, no. 6, pp. 4258–4270, 2007.

[28] D. M. Taylor, S. I. H. Tillery, and A. B. Schwartz, "Direct cortical control of 3D neuroprosthetic devices," *Science*, vol. 296, no. 5574, pp. 1829–1832, 2002.

[29] W. Wang, A. D. Degenhart, J. L. Collinger et al., "Human motor cortical activity recorded with micro-ECoG electrodes

during individual finger movements," in *Proceedings of the 31st Annual International Conference of the IEEE Engineering in Medicine and Biology Society: Engineering the Future of Biomedicine (EMBC '09)*, pp. 586–589, September 2009.

[30] D. Bacher, J. McFerron, N. Krishnamurthy, and A. Batista, "An experimental rig for closed-loop neuroprosthetics," in *Poster Presented as Part of the Society for Neuroscience Conference*, pp. 1–5, Washington, DC, USA, September 2008.

[31] Y. Renard, F. Lotte, G. Gibert et al., "OpenViBE: an open-source software platform to design, test, and use brain-computer interfaces in real and virtual environments," *Presence: Teleoperators and Virtual Environments*, vol. 19, no. 1, pp. 35–53, 2010.

[32] G. Sudre, W. Wang, T. Song et al., "rtMEG: a real-time software toolbox for brain-machine interfaces using magnetoencephelography," in *Proceedings of the 17th International Conference on Biomagnetism Advances in Biomagnetism (Biomag '10)*, vol. 28, pp. 362–365, March 2010.

[33] "The FieldTrip buffer for real-time access to EEG/MEG data," October 2010, http://fieldtrip.fcdonders.nl/development/realtime/buffer.

[34] G. Schalk, E. C. Leuthardt, P. Brunner, J. G. Ojemann, L. A. Gerhardt, and J. R. Wolpaw, "Real-time detection of event-related brain activity," *NeuroImage*, vol. 43, no. 2, pp. 245–249, 2008.

[35] G. Schalk, P. Brunner, L. A. Gerhardt, H. Bischof, and J. R. Wolpaw, "Brain-computer interfaces (BCIs): detection instead of classification," *Journal of Neuroscience Methods*, vol. 167, no. 1, pp. 51–62, 2008.

[36] A. Delorme and S. Makeig, "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis," *Journal of Neuroscience Methods*, vol. 134, no. 1, pp. 9–21, 2004.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 327953, 7 pages
doi:10.1155/2011/327953

Research Article

rtMEG: A Real-Time Software Interface for Magnetoencephalography

Gustavo Sudre,[1] Lauri Parkkonen,[2] Elizabeth Bock,[3] Sylvain Baillet,[4] Wei Wang,[5] and Douglas J. Weber[5]

[1] *Program in Neural Computation, Carnegie Mellon University, Pittsburgh, PA 15213, USA*
[2] *Brain Research Unit, Low Temperature Laboratory, Aalto University School of Science, 00076 Espoo, Finland*
[3] *Department of Neurology, Froedtert & The Medical College of Wisconsin, Milwaukee, WI 53226, USA*
[4] *Departments of Neurology and Biophysics, Froedtert & The Medical College of Wisconsin, Milwaukee, WI 53226, USA*
[5] *Department of Physical Medicine and Rehabilitation, University of Pittsburgh, Pittsburgh, PA 15260, USA*

Correspondence should be addressed to Douglas J. Weber, djw50@pitt.edu

Received 1 October 2010; Revised 18 January 2011; Accepted 28 February 2011

Academic Editor: Robert Oostenveld

To date, the majority of studies using magnetoencephalography (MEG) rely on off-line analysis of the spatiotemporal properties of brain activity. Real-time MEG feedback could potentially benefit multiple areas of basic and clinical research: brain-machine interfaces, neurofeedback rehabilitation of stroke and spinal cord injury, and new adaptive paradigm designs, among others. We have developed a software interface to stream MEG signals in real time from the 306-channel Elekta Neuromag MEG system to an external workstation. The signals can be accessed with a minimal delay (45 ms) when data are sampled at 1000 Hz, which is sufficient for most real-time studies. We also show here that real-time source imaging is possible by demonstrating real-time monitoring and feedback of alpha-band power fluctuations over parieto-occipital and frontal areas. The interface is made available to the academic community as an open-source resource.

1. Introduction

Off-line analysis of magnetoencephalography (MEG) data has been applied to a wide spectrum of basic and clinical neuroscience questions (see, e.g., [1, 2]). The ability to process and analyze MEG data in real time would potentially open new opportunities for neuroscientific research and innovative clinical applications. For example, adaptive paradigms (or optimal experiment designs [3, 4]) would benefit from the possibility of capturing MEG measurements in real time, for example, to select the most efficient stimulus type, or to determine which stimulus classes necessitate the collection of more repetitions in order to increase classification accuracy in the context of a cognitive-state decoding task. Moreover, real-time neurofeedback could be used to train subjects to modulate some specific spatial and dynamic features of their neural activity in the context of brain-machine interface (BMI) applications. From a clinical standpoint, neurofeedback training may help promote neuroplasticity to reinforce spared corticospinal pathways after stroke or spinal cord injury [5, 6].

While systems that use real-time feedback with different MEG machines have been previously described [7–11], this work presents a software interface ("rtMEG") designed to acquire signals from an Elekta Neuromag device in real time. It provides the following additional features with respect to the software that was described previously [12].

(i) This version of the software interface is more robust and is better integrated into the standard MEG acquisition system. For example, it performs data acquisition using the set of parameters specified through the regular acquisition software interface. Furthermore, data are streamed with proper channel calibration and ordering. In the near future, users will also have the option to stream the data with online signal-space projection (SSP) [13] noise reduction being applied, while currently this transformation should be performed on the client workstation.

(ii) The rtMEG interface now writes data to the Fieldtrip buffer [14], instead of being integrated into the BCI2000 pipeline. The Fieldtrip buffer consists of an open-source server program that runs continuously, providing a shared memory buffer to which rtMEG writes the data. While it was possible to stream the data out of BCI2000 in the previous implementation, that software was still required to run rtMEG. With the current implementation, researchers have the freedom to use whatever solutions they favor by running the Fieldtrip buffer implemented within rtMEG and using the code freely available online [15] to read from the buffer. Moreover, researchers have the option to work with any of the Fieldtrip tools used for off-line analysis in an on-line setting. Another advantage of using the Fieldtrip buffer is the independence on the operating system. While the buffer has been implemented within rtMEG, the user still has the option to run it externally under Windows, MacOS, and Linux/Unix using the software provided by the Fieldtrip developers (in contrast, BCI2000 is mostly run on Windows). Finally, the Fieldtrip buffer provides the flexibility to interact with other commonly used software packages (BCI2000 [16], Brainstream [17], among others), and because the code to read from the buffer is freely available online [15], researchers can easily integrate it to their own custom solutions.

(iii) rtMEG can be modified and compiled using open source software.

It is important to note that although the rtMEG interface does not depend on BCI2000 anymore [12], it is still able to interact with the latter. Indeed, BCI2000 can read from the Fieldtrip buffer either by using the Fieldtrip buffer source module or the Remote Data Access streaming interface.

We describe the system setup and the tests that were performed to assess the delay in accessing the data stream. We then show results regarding acquisition delays and illustrate the technique with real-time source estimation in a neurofeedback experiment. We conclude with a discussion of several scenarios where we foresee that the rtMEG interface may prove useful.

2. Methods

The interface was developed to function in conjunction with the standard MEG acquisition, without affecting the normal workflow. In a typical scenario (Figure 1), a dedicated computer runs the main acquisition software and saves the acquired data on the MEG filesystem. The rtMEG interface runs on this acquisition workstation and operates in parallel with the standard acquisition software.

In a typical experimental setting, a separate computer controls stimulus delivery to the subject. Stimuli may comprise multiple categories (auditory, visual, etc.). For synchronization, the stimulus computer sends event-related trigger pulses through the parallel port to mark the onsets of stimuli in the recorded files.

Figure 1: Illustration of a typical setup with rtMEG. The acquisition computer controls the acquisition, stores the data, and runs the rtMEG interface. Another computer drives the experimental paradigm by providing stimuli to the subject and sending trigger events that eventually go to the MEG data file. The rtMEG interface writes data to the Fieldtrip buffer, which can be run by rtMEG or by any other computer in the network. The application(s) reading from this buffer can run on the stimulus computer or on any other computer connected in the network.

rtMEG writes data to a Fieldtrip buffer that can be either run by rtMEG itself or hosted by any other computer located in the same network as the acquisition computer (e.g., the stimulus computer). This buffer can then be read using Matlab [18] (with Fieldtrip scripts) or another preferred solution (see the code openly available on the Fieldtrip website [15]). Similarly, the computer reading from the buffer can be the same as the computer hosting the buffer, or any other computer in the same network.

2.1. Details of Implementation and Distribution. In the usual setup, each Digital Signal Processor (DSP) unit manages 12 channels in the MEG machine, and packets comprising 28 samples per channel are sent by each DSP to the real-time computer, which reorders and synchronizes the data and attaches metainformation, such as calibration coefficients and sampling rate, to them. The acquisition computer, which also runs rtMEG, receives the data from the real-time computer. When using typical sampling rates (<1.5 kHz), the data are sent to the acquisition computer in chucks of about 1 s, which corresponds to a considerable and often unacceptable delay for any real-time application. However, rtMEG can optionally reduce the size of the chunk, down to a lower bound of 28 samples, by reconfiguring the real-time computer and thereby substantially diminishing the average transit delay of the data.

The data received from the real-time computer are then stored in a local shared memory buffer that is used by different Neuromag programs, such as the on-line visualization. rtMEG taps into this local buffer, reads the data, and writes them to a Fieldtrip buffer, which can then be easily read by several different clients using an open-source format. This Fieldtrip buffer can be run by rtMEG itself in a separate thread, or by a separate computer in the network.

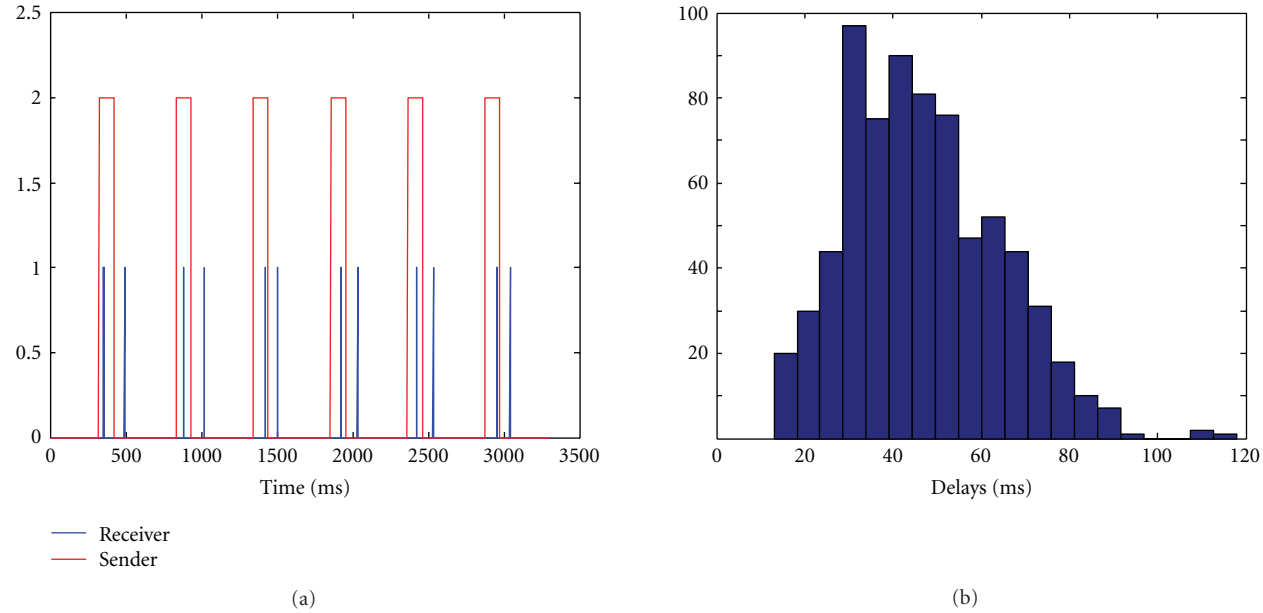


Figure 2: Measuring the data-access delay in real time. (a) Red pulses correspond to the signal created by the acquisition workstation; blue traces show the time when the computer reading from the Fieldtrip buffer detected the pulse and sent an acknowledgment pulse back to the acquisition computer. The delay was measured by calculating the time difference between all vertical red bars and the corresponding blue bars. (b) Histogram of the observed delays during a 5-minute measurement with buffer size 29.

rtMEG was written in C, and all network communication is done using TCP/IP. The source code is made available to the research community under Gnu Public License (GPL) and stored in the Fieldtrip source control repository. Documentation [19] has been written in the Fieldtrip Wiki. Binary files for HP-UX and Linux platforms have also been provided for the users' convenience.

2.2. Assessing Delays to Data Access. Real-time MEG applications often rely on minimal system delays, and the rtMEG interface needs to be carefully assessed in this respect. We measured the delay associated to complete feedback loop as follows. We recorded 306 MEG channels and 3 stimulus channels at 1 kHz. These data were written to a Fieldtrip buffer implemented inside rtMEG and then read over the network by a separate Linux computer. Data were written to and read from the buffer every 29 samples. The acquisition software was set to generate a pulse (square wave) in one of the stimulus trigger channels every 500 ms (rise from zero level to value "2", hold on for 100 ms, and then return to zero). The Linux computer ran a simple C program that was designed to write a logical "1" to the parallel port every time a change was detected on the trigger channel, and a logical "0" otherwise. The parallel port was mapped on to a different stimulus trigger channel in the data. Because the MEG system acquires all signals synchronously, this form of testing using the trigger input is indicative of the data-access delays in the system. Delays were measured as the time difference between the occurrences of "1"—when the Linux computer responded to a change in the trigger—and "2"—marking the actual occurrences of the change—in the data; see Figure 2(a).

2.3. Real-Time Feedback and MEG Source Imaging. The primary goal of real-time operations is to provide the subject with a measure of his/her brain activity. To prove and evaluate this technical concept, an experiment was designed to report on variations of ongoing regional brain activity related to behavior. This objective was challenging because it implied that both (1) data acquisition and formatting, and (2) source modeling of ongoing brain activity, were achievable in real time. To our knowledge, this latter feature had not been demonstrated with EEG or MEG so far. Here, we designed a simple paradigm in which the subject was alternating 20-s segments of rest with his eyes either closed or open. An auditory cue was provided to the subject to let him know when to open or close his eyes. It is a very well-documented and robust phenomenon that the amplitude of alpha (8–13 Hz) oscillations is stronger over the dorsal parietal and posterior occipital brain regions with the eyes being closed versus open.

Real-time estimation of ongoing alpha power was performed over a set of cortical regions of interest (ROIs) that were predefined from the individual brain anatomy of one subject. The ROIs covered the dorsal parietal and posterior occipital (PO) cortex and were delineated using BrainStorm [20] (Figure 4(a)). An additional ROI was defined over the anterior and dorsolateral prefrontal cortex, for comparison with the levels of alpha power changes observed in the parieto-occipital region. The cortical surface was obtained

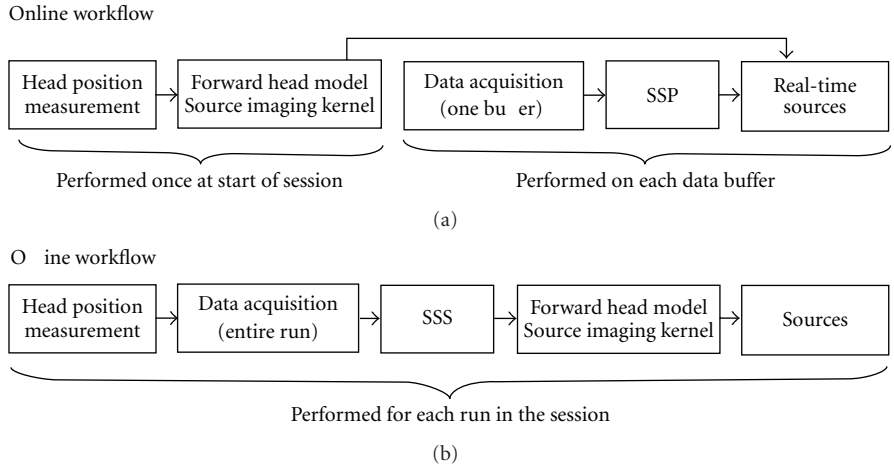


Figure 3: Workflows of real-time data collection and processing using rtMEG (a) and of the optimal, off-line processing chain (b). For real-time data analysis, the forward head model and inverse imaging kernel were precomputed and applied online on all subsequent recordings. This differs from the optimal off-line pipeline, where forward and inverse modeling is completed for each individual run. In addition, interference suppression was performed in the client workstation using SSP on each data buffer during real-time data collection and analysis, whereas SSS was used over the entire duration of each individual run during the off-line analysis.

from the T1-weighted volume MRI (1.5 T, SPGR sequence, voxel size: $0.9 \times 0.9 \times 1.5\,\text{mm}^3$; field of view: 240×240 mm) using BrainVISA [21]. MEG data acquisition and analysis were performed at Froedtert & the Medical College of Wisconsin (Milwaukee, USA) using a 306-channel Elekta Neuromag MEG system.

The entire recording session lasted 10 minutes and consisted of a short 10-s baseline run, followed by 3 runs of 130 seconds each. The subject's head position was measured at the beginning of each run by the software provided with the MEG system. The head location from the short baseline run was used by the forward head modeling and inverse source modeling steps necessary to access cortical source estimates from ongoing MEG data. Both steps were completed in approximately 2 minutes using BrainStorm after the baseline run was acquired. Head modeling was performed using the overlapping-sphere analytic approach [22]. The linear imaging kernel from BrainStorm's weighted and cortically constrained minimum-norm estimate (WMNE) [23] was subsequently obtained and stored in memory. Because the WMNE is a linear, stationary source estimation approach, source signals can be readily accessed from each real-time buffer data by simply completing the matrix multiplication of the imaging kernel with either the sensor data time series or Fourier coefficients. In our study, this was further reduced to the extraction of the elementary sources within the targeted ROIs, which amounted to about 750 current dipoles.

For each 500-ms segment, the power in the alpha range across the PO ROI was computed from the Fourier coefficients of each of the 750 elementary sources. These were obtained by applying the imaging kernel to the fast-Fourier transform (FFT) coefficients of the running segment of sensor data. The power in each ROI therefore consisted of the sum of the magnitude of the resulting Fourier coefficients in the 8–13 Hz range across the entire set of elementary sources forming the ROI. The cumulative time taken to perform this operation—magnitude of the product of a 750×306 imaging kernel by $306 \times 1,000$ Fourier coefficients of MEG sensor data—was about 100 ms on a conventional workstation running Matlab.

The overall benefits of the imaging kernel and Fourier-domain approach were that the time-consuming steps of the forward and inverse modeling were performed offline. The downside was the suboptimal accuracy of these models due to cumulative head movements during the session. These movements were evaluated from the measurements of the head positions collected at the beginning of each of the 3 feedback runs.

State-of-the-art MEG acquisition may also include active denoising techniques, requiring both on-line and off-line processing steps to be performed. In the case of the MEG installation used for this study, the standard data acquisition pipeline consists of (1) the on-line application of signal-space projection (SSP) to compensate for the spatial pattern of some environmental interference sources and (2) the off-line application of the signal-space separation (SSS) technique [24], to fully benefit from the latest generation of single-layer magnetically shielded rooms. Figure 3 details the approach we used in the present study to assess the deviations of the outcome of the real-time data acquisition and source analysis from the conventional, optimal pipeline that is only accessible offline.

Real-time visual feedback on the level of alpha power within the target ROIs was provided to the subject after the processing of each 500-ms data segment by the stimulus computer that was hosting the FieldTrip buffer (see Figure 4(a)). These measurements of brain activity were saved to a disk file and converted to a visual display that was provided to the subject via a video projection system (60-Hz refresh rate). During the segments with eyes open, the

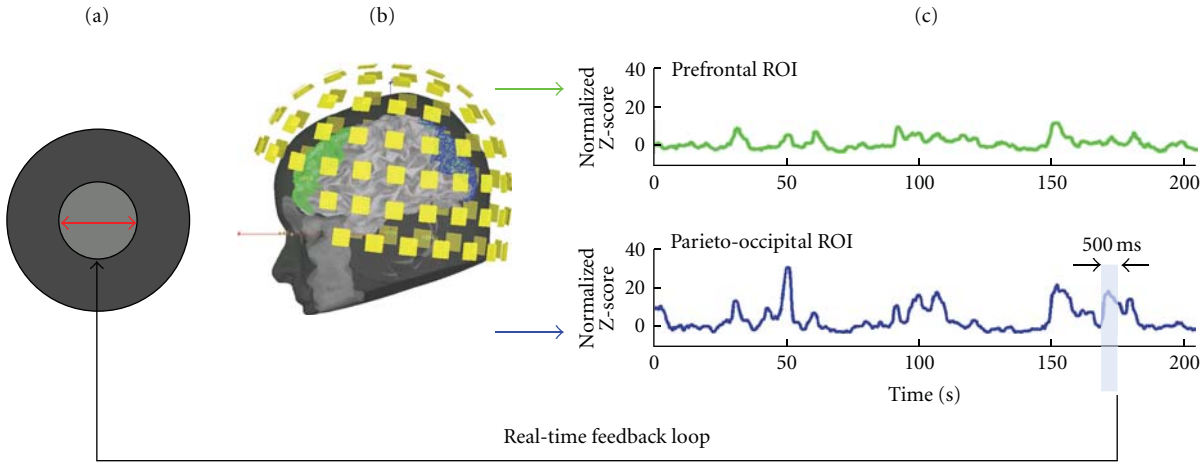


Figure 4: Real-time visual feedback on the power of alpha oscillations in brain regions of interest. (a) The subject was provided with a visual gauge of the real-time level of alpha power within the parieto-occipital (PO) region of interest shown in blue in (b). The radius of the light-grey disk in (a) evolved every 500 ms and increased as alpha power decreased during the eyes-open segment of the experimental run. The static, dark-grey disk was an incentive target for the subject. Its radius was indexed to 2 times the average PO alpha power captured during the baseline run acquired at the beginning of the session. (c) shows the ongoing, respective levels of alpha power variation in the two ROIs: prefontal (in green in b) and PO (in blue in b).

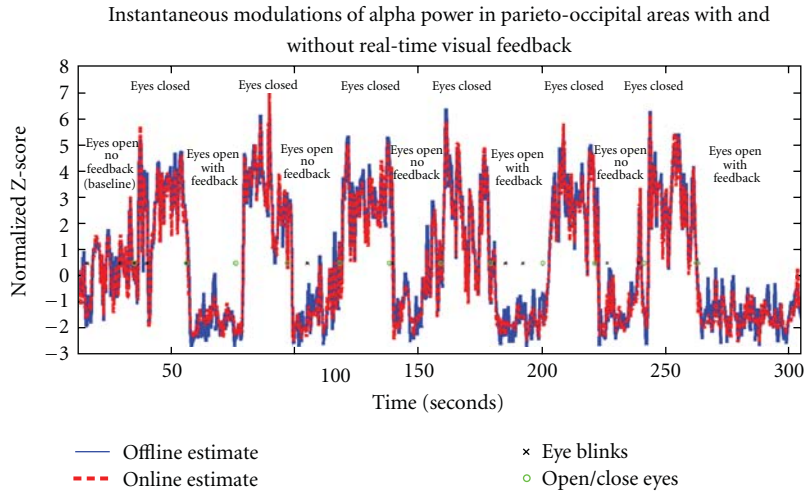


Figure 5: Comparison of off-line (in blue) versus on-line (in red) estimates of alpha power modulations over the PO ROI. The traces of the time-resolved power estimates are plotted over time and were standardized over the 20-s baseline period immediately preceding the first eyes-closed segment. As expected, eye blinks (marked with an "x") were detected from the EOG channel during the segments where the subject had his eyes open. The transitions from open to closed eyes are marked at the time the subject was given an auditory stimulus (marked with an "o") every 20 s. In this particular run (300 s), the root-mean-square (RMS) difference between the online and offline time series reached 29.2%. Both time series indicate lower and more sustained decreases of alpha power in the PO region with respect to baseline, when feedback was provided to the subject than when no feedback was shown.

subject was instructed to try to maximize the level of the visual gauge, which was indexed to the inverse of the power of alpha oscillation in the targeted ROIs (Figures 4(b) and 4(c)).

3. Results

The following sections describe the results obtained while measuring the data-access delays introduced by the rtMEG interface to the data stream, and the results observed while providing real-time feedback of alpha-band power modulation.

3.1. Delay Measurements. The average delay to access the data was measured to be 44 ± 17 ms, and it was insensitive to the number of channels being simultaneously transmitted over the network. No changes were noticed after the system

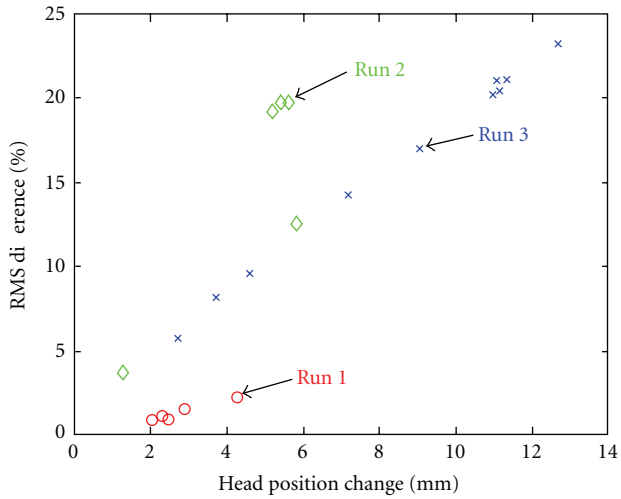


Figure 6: RMS differences in the alpha power estimates resulting from changes in the head position across three different sessions, each session consisting of multiple runs of 130-s duration each. The differences in head position are with respect to the one captured during an initial baseline run acquired at the beginning of the first session. The on-line calculations of alpha power modulations using the reference head position were compared to those obtained offline but with optimal noise attenuation and forward head and inverse source modeling. Colors represent different sessions, and each marker represents a run within a session.

continuously collected data for several minutes. A histogram of the observed delays during a representative measurement is shown in Figure 2(b). The variability of the results is attributed to the asynchrony between the change in the trigger channel and the boundaries of the 29-sample buffer. Hence, the theoretical distribution should show the mean data-access delay time ± 29 ms (1000-Hz sampling rate). However, the program that read the data from the buffer was designed to run in an infinite loop, and whenever there was no new data in the buffer since the last read action, it paused for a predetermined amount of time. This sleep time is responsible for the subtle dissimilarities between the theoretical distribution and the histogram shown in Figure 2(b). The overall results show that the interface introduced only modest delays to the measured signal, which are likely to be short enough for most real-time MEG applications. The distribution of delays was also consistent over time.

The delay values reported here are slightly higher than what was reported before [12], which is justified because of the different ways in which the two implementations access the MEG data. While the previous implementation collected the data directly from the DSPs, the current implementation reads the data from the local buffer in the acquisition computer. Moreover, the previous implementation did not sort and calibrate the channels as is now done by the real-time computer. The current implementation is preferred because it provides a more intuitive and robust interface to the user without repeating processing steps that are already reliably implemented in the real-time computer while still keeping the data-access delay at an acceptable level.

It is important to reiterate that this experiment measured the delay to access the data; more complex real-time processing will likely increase the overall system delay.

3.2. Real-Time Source Imaging. Both the on-line and off-line source analyses revealed modulations of oscillatory alpha power within the PO region (Figure 5). These measures were standardized (Z-score) with respect to a baseline data segment of reference obtained in the first 20 seconds of each feedback run (subject resting with eyes open, fixating at a crosshair on the screen). As shown in Figure 5, excursions under the baseline alpha levels were stronger and more sustained during the segments with eyes open and feedback than when no feedback was provided, indicating an encouraging trend that feedback indeed drove the subject towards lower alpha levels than during baseline, and during segments where no feedback was present.

Comparison of the off-line and on-line estimates of alpha power modulations in the PO regions qualitatively demonstrated that the data were not altered or significantly delayed by the transfer from the acquisition to the analysis workstation, and/or by the optimal denoising techniques applied and more accurate head/source models (Figure 5). The discrepancies observed—reaching up to 24.5% RMS error as in Figure 5—showed strong dependence on the fluctuations in the subject's head position over time, reaching a maximum of 12.5 mm (see Figure 6).

4. Conclusions

The analysis of MEG signals in real time opens up new possibilities for the study of brain function. Potential applications include the following.

(i) *Basic Research.* Real-time visualization of MEG data in source space (on the brain surface) for quality assurance and rapid interpretation of the measurement. Dynamic and adaptive paradigms where subject's brain state could be a condition to stimulus delivery.

(ii) *Brain-Machine Interfaces.* Our previous off-line MEG studies have shown that we can decode intended movement direction from MEG signals and accurately localize cortical areas representing such information for real-time BMI operation [25]. With the real-time capability, it will be beneficial to use MEG as a presurgical tool to localize the optimal placement site for an ECoG grid for obtaining real-time BMI control. Furthermore, researchers may test various neural processing, decoding, and user training paradigms "on the fly" within a single MEG session.

(iii) *Clinical.* Real-time neurofeedback training can be used to promote neuroplasticity [5, 6]. Through the operation of an rtMEG-BMI system, users can learn to voluntarily modulate or change their brain activity [7, 8], inducing neuroplasticity for recovery of motor function or to improve control of neuroprosthetic devices.

This paper described a software solution that enables easy real-time access to the MEG signals from any computer connected to the local network. We demonstrated that the delay to access the data by this software was minimal, and that the access mechanism easily lends itself to real-time source modeling.

Acknowledgments

The authors thank the University of Pittsburgh Medical Center (UPMC), Center for Advanced Brain Magnetic Source Imaging (CABMSI) for providing the scanning time for MEG data collection. They specifically thank Mrs. Anna Haridis and Dr. Anto Bagic at UPMC CABMSI for assistance in MEG setup and data collection. They also would like to thank Stefan Klanke and Robert Oostenveld for assistance with the interaction with the Fieldtrip buffer. This work was partially supported by NSF Cooperative Agreement EEC-0540865, US Army (TATRC) Agreement W81XWH-07-1-0716, and Grant Number 5 UL1 RR024153 and KL2 RR024154 from NCRR of NIH, and a special grant from the Office of the Senior Vice Chancellor for the Health Sciences at University of Pittsburgh, as well as a student travel fund from Center for Neural Basis of Cognition. Additional funding support was provided by NIH grants from the NIBIB (1R01EB007749) and NINDS (1R21NS056136) for D. J. Weber and the French National Research Agency (Agence Nationale pour la Recherche) through the ViMAGINE project (ANR-08-BLAN-0250) for S. Baillet.

References

[1] S. M. Stufflebeam, N. Tanaka, and S. P. Ahlfors, "Clinical applications of magnetoencephalography," *Human Brain Mapping*, vol. 30, no. 6, pp. 1813–1823, 2009.

[2] J. W. Wheless, E. Castillo, V. Maggio et al., "Magnetoencephalography (MEG) and magnetic source imaging (MSI)," *Neurologist*, vol. 10, no. 3, pp. 138–153, 2004.

[3] D. J. C. MacKay, "Information-based objective functions for active data selection," *Neural Computation*, vol. 4, no. 4, pp. 590–604, 1992.

[4] K. Chaloner and I. Verdinelli, "Bayesian experimental design: a review," *Statistical Science*, vol. 10, no. 3, pp. 273–304, 1995.

[5] N. Birbaumer and L. G. Cohen, "Brain-computer interfaces: communication and restoration of movement in paralysis," *Journal of Physiology*, vol. 579, no. 3, pp. 621–636, 2007.

[6] W. Wang, J. L. Collinger, M. A. Perez et al., "Neural interface technology for rehabilitation: exploiting and promoting neuroplasticity," *Physical Medicine and Rehabilitation Clinics of North America*, vol. 21, no. 1, pp. 157–178, 2010.

[7] J. Mellinger, G. Schalk, C. Braun et al., "An MEG-based brain-computer interface (BCI)," *NeuroImage*, vol. 36, no. 3, pp. 581–593, 2007.

[8] E. Buch, C. Weber, L. G. Cohen et al., "Think to move: a neuromagnetic brain-computer interface (BCI) system for chronic stroke," *Stroke*, vol. 39, no. 3, pp. 910–917, 2008.

[9] T. N. Lai, M. Schröder, N. J. Hill et al., "A brain computer interface with online feedback based on magnetoencephalography," in *Proceedings of the 22nd International Conference on Machine Learning (ICML '05)*, pp. 465–472, ACM, August 2005.

[10] H. Rongen, V. Hadamschek, and M. Schiek, "Real time data acquisition and online signal processing for magnetoencephalography," *IEEE Transactions on Nuclear Science*, vol. 53, no. 3, pp. 704–708, 2006.

[11] C. W. Hesse, R. Oostenveld, T. Heskes, and O. Jensen, "On the development of a brain- computer interfacesystem using high-density magnetoencephalogram signals for real-time control of a robot arm," in *Proceedings of the 4th Annual Symposium of the Benelux Chapter of the IEEE Engineering in Medicine and Biology Society (EMBS '07)*, pp. 1–4, 2007.

[12] G. Sudre, W. Wang, T. Song et al., "rtMEG: a real-time software toolbox for brain-machine interfaces using magnetoencephelography," in *Proceedings of the 17th International Conference on Biomagnetism Advances in Biomagnetism (Biomag '10)*, pp. 362–365, April 2010.

[13] M. A. Uusitalo and R. J. Ilmoniemi, "Signal-space projection method for separating MEG or EEG into components," *Medical and Biological Engineering and Computing*, vol. 35, no. 2, pp. 135–140, 1997.

[14] Fieldtrip buffer, http://fieldtrip.fcdonders.nl/development/realtime/buffer.

[15] Fieldtrip website, http://fieldtrip.fcdonders.nl/.

[16] G. Schalk, D. J. McFarland, T. Hinterberger, N. Birbaumer, and J. R. Wolpaw, "BCI2000: a general-purpose brain-computer interface (BCI) system," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 6, pp. 1034–1043, 2004.

[17] Brainstream software, http://www.brainstream.nu/.

[18] Mathworks Matlab, http://www.mathworks.com/.

[19] rtMEG documentation, http://fieldtrip.fcdonders.nl/development/realtime/neuromag.

[20] Brainstorm software, http://www.neuroimage.usc.edu/brainstorm/.

[21] Brainvisa software, http://www.brainvisa.info/ .

[22] M. X. Huang, J. C. Mosher, and R. M. Leahy, "A sensor-weighted overlapping-sphere head model and exhaustive head model comparison for MEG," *Physics in Medicine and Biology*, vol. 44, no. 2, pp. 423–440, 1999.

[23] S. Baillet, J. C. Mosher, and R. M. Leahy, "Electromagnetic brain mapping," *IEEE Signal Processing Magazine*, vol. 18, no. 6, pp. 14–30, 2001.

[24] S. Taulu, M. Kajola, and J. Simola, "Suppression of interference and artifacts by the signal space separation method," *Brain Topography*, vol. 16, no. 4, pp. 269–275, 2004.

[25] W. Wang, G. P. Sudre, Y. Xu et al., "Decoding and cortical source localization for intended movement direction with MEG," *Journal of Neurophysiology*, vol. 104, no. 5, pp. 2451–2461, 2010.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 747290, 12 pages
doi:10.1155/2011/747290

Research Article

BrainNetVis: An Open-Access Tool to Effectively Quantify and Visualize Brain Networks

Eleni G. Christodoulou,[1] Vangelis Sakkalis,[1] Vassilis Tsiaras,[2] and Ioannis G. Tollis[1, 2]

[1] *Institute of Computer Science (ICS), Foundation for Research and Technology—Hellas (FORTH), N. Plastira 100, GR-70013 Heraklion, Greece*
[2] *Department of Computer Science, University of Crete, GR-71409 Heraklion, Greece*

Correspondence should be addressed to Eleni G. Christodoulou, echristo@ics.forth.gr

Received 17 September 2010; Revised 25 November 2010; Accepted 31 December 2010

Academic Editor: Sylvain Baillet

This paper presents BrainNetVis, a tool which serves brain network modelling and visualization, by providing both quantitative and qualitative network measures of brain interconnectivity. It emphasizes the needs that led to the creation of this tool by presenting similar works in the field and by describing how our tool contributes to the existing scenery. It also describes the methods used for the calculation of the graph metrics (global network metrics and vertex metrics), which carry the brain network information. To make the methods clear and understandable, we use an exemplar dataset throughout the paper, on which the calculations and the visualizations are performed. This dataset consists of an alcoholic and a control group of subjects.

1. Introduction

One of the major issues in neuroscience is to describe how different brain areas communicate with each other during perception, cognition, and action as well as during spontaneous activity in the default or resting state. Mainly two different approaches for capturing and localizing brain activity motifs have been proposed; univariate spectrum based analysis and functional connectivity analysis [1]. Friston [2] defined functional connectivity as the statistical dependence between the activations of distinct and often well-separated neuronal populations.

Network models and graph theory provide a common framework for describing brain functional connectivity [3–5]. The interdependence between brain areas is estimated using multivariate neurophysiological signals (EEG, MEG, ECoG) and/or haemodynamic response images (fMRI). Then, a network is formed by corresponding either brain areas or channels to vertices and by considering an edge between two vertices if and only if the estimated interdependence is above a threshold. Regarding threshold selection, it is important to notice that it is a rather tricky part and there is currently no established way of favouring a specific threshold value. In practice, a broad range of threshold values is used to characterize the network. However, the authors propose two alternative approaches in selecting a threshold value based either on group statistics between specific graph-theoretic measures of the populations under analysis [6] or utilizing a signal-based technique of selecting the optimal visualization threshold using surrogate (artificially generated ensemble of data aiming at revealing the most significantly coupled brain regions) datasets to correctly identify the most significant correlation patterns [7]. The next step in the analysis, after edge identification, is to measure some networks statistics and characterize the network. Then, using the network characterization, one can draw conclusions on the effect of illnesses or of cognitive loads on functional connectivity [6–11].

In this study, we briefly refer to pairwise (bivariate) and multivariate interdependence measures, as well as linear and nonlinear ones, that have been successfully used as indices of cerebral engagement [12]. This information is important for the correct usage of the tool, especially for nonexpert users, as the application of these measures on the raw EEG data produces the input to our tool. The BrainNetVis tool provides a dynamic snapshot of

the highly complex underlying neural mechanisms by means of graph visualization [13]. BrainNetVis is an open-access multiplatform tool, provided by ICS-FORTH, for graph representation and brain network visualization. Please note that BrainNetVis calculates the following presented metrics on the *synchronization matrices* (adjacency matrices) that the user should calculate in advance! However, the preprocessing section (Section 3.2) briefly presents some widely used techniques to assess functional brain connectivity and form the adjacency matrix.

At this point, we refer to some already existing tools on the field. These tools capture different kinds of EEG information than BrainNetVis and they may be used complementary to it. One of them is EEGLAB [14], which we have been using extensively for better perception of the brain area. EEGLAB is an interactive Matlab toolbox for processing continuous and event-related EEG, MEG, and other electrophysiological data incorporating independent component analysis (ICA), time/frequency analysis, artifact rejection, event-related statistics, and several useful modes of visualization of the averaged and single-trial data. EEGlab offers also dipole localization functions. Some of the metrics that we implement have also been implemented in the Brain Connectivity toolbox (a matlab toolbox) by Rubinov and Sporns [15]. Other related toolboxes include MEA-Tools [16] and ERPWAVELAB [17]. In these toolboxes, however, the measures for quantifying channel interactions are mainly confined to the temporal crosscorrelation [16] and the coherence spectrum [17, 18]. However, more sophisticated interdependence techniques addressing not only linear but also nonlinear synchronization and causality are also available and applied in certain pathologies like Epilepsy [12]. Such measures can act complementary to graph theoretic indices that characterize brain networks as discussed in [19] and can be used as input to BrainNetVis.

The paper is organized as follows. Section 2 presents essential information on the different ways of graph modelling and manipulations, using BrainNetVis. Section 3 refers to the preprocessing needed (Section 3.2), the most commonly used menu calls and the GUI (Section 3.3), and the possible graph visualization options (Section 3.4). Our conclusion is given in Section 4.

2. Network Analysis

Before presenting BrainNetVis, it is important to provide here some basic definitions from graph theory.

A *graph* $G = (V, E)$ defined on a set of *vertices* $V = \{v_1, \ldots, v_n\}$ and *edges* $E = \{e_1, \ldots, e_m\}$, where each edge $e \in E$ is an ordered or unordered pair of vertices. An ordered pair $e = (u, v) \in V \times V$ is called a *directed edge*, while an unordered pair $e = \{u, v\}$, where $u, v \in V$, is called an *undirected edge*. In case $u = v$, e is called a *self-loop*. In our study, we consider *simple* graphs, that is, graphs without self-loops. Also the cardinality of V is denoted by n (i.e., $n = |V|$).

A *weighted network* $G = (V, E, \omega)$ consists of a graph with vertex set V and edge set E augmented with an edge value function $\omega : E \to \mathbb{R}$ that assigns to each edge $e \in E$ a real value $\omega(e)$. Every weighted network $G = (V, E, \omega)$ corresponds to a real $n \times n$ matrix $W = (w_{ij})$, $i, j \in \{1, 2, \ldots, n\}$, where w_{ij} is equal to value $\omega(e)$ of edge $e = (v_i, v_j)$ if $e \in E$, or to 0 otherwise. If we reserve value 0 to mean the absence of an edge, then the correspondence between G and W is one to one. In this work, we consider a subset of weighted networks, which we call *synchronization networks*, where edge values are restricted to interval $(0, 1]$ and interpreted as strength of dependence between vertices.

In synchronization networks, higher edge values indicate stronger dependencies. To define the length of an edge, we should at least reverse the order of edge values by applying, for example, the inverse function $g : (0, 1] \to [1, +\infty)$, that is,

$$g(x) = \frac{1}{x}. \tag{1}$$

We also propose another function $g : (0, 1] \to [1, +\infty)$, where

$$g(x) = 1 - \log_2(x). \tag{2}$$

These are definitions on how to transform the edge lengths in the case of synchronization networks. Which of the two functions performs better depends on the graph structure and on the metric or the visualization method that uses these functions. When choosing the appropriate formulation, one should consider that the function $1/x$ tends to $+\infty$ faster than the function $1 - \log_2(x)$ when $x \to 0^+$. Therefore, the edges with small values are assigned longer lengths with the $1/x$ function than those with the $1 - \log_2(x)$ function.

The length of a path from vertex u to vertex v is the sum of the lengths of the edges of the path. The shortest path distance from vertex u to vertex v is denoted by $d_G(u, v)$. If vertex v is unreachable from vertex u, then $d_G(u, v) = +\infty$.

3. Methods and Results

3.1. Exemplar Case. In what follows, we are using the data of a specific use case, consisting of alcoholic and control subjects, in order to provide concrete examples of use of the application. Briefly, the specific study included 30 control subjects and 30 alcoholic subjects. Each subject was fitted with a 61-lead electrode cap (ECI, Electro-Cap International). All scalp electrodes were referred to C_z. In this experiment, each subject was exposed to pictures of objects chosen from the 1980 Snodgrass and Vanderwart picture set [20]. The stimuli in each trial were randomized (but not repeated) and were presented on a white background for 300 ms at the center of a computer monitor. Their size was approximately 5–10 cm × 5–10 cm, thus subtending a visual angle of 0,05 –0,1 . Ten trials were shown, with the interval between trials fixed to 3.2 s. The participants were instructed to memorize the pictures in order to be able to identify them later. For each subject and for each trial and frequency band (0.5–4 Hz, 4–8 Hz, 8–13 Hz, 13–30 Hz, 30–45 Hz) the interdependence for each channel pair (there are 61 (61 – 1)/2

channel pairs since the number of active EEG channels is 61) was calculated using the coherence and the RIM methods. The results were stored in 61×61 interdependence matrices W with elements ranging from 0 to 1. The main finding of this study, using BrainNetVis, was that the alcoholic subjects have impaired synchronization of brain activity and loss of lateralization during the rehearsal process as compared to control subjects.

3.2. Preprocessing. In order to create a graph, a matrix containing the EEG channel pairwise correlations is required. Thus, one needs to calculate the correlations among all pairs of electrodes and deduce the respective adjacency matrix, called *synchronization matrix*. There exist a number of measures that capture the linear and the nonlinear links between time-series in a frequency band in order to calculate the required correlations (in the EEG analysis context they are called synchronization indices). Three measures have been chosen after an extensive study in linear and nonlinear synchronization measures [12]: the typical magnitude squared coherence method (MSC) [21], a nonlinear bivariate measure for generalized synchronization (RIM) [22] and Partial Directed Coherence (PDC) [23]. The advantage of magnitude squared coherence is that it is well known and widely accepted. The advantage of RIM is that it is able to capture nonlinear patterns available in the signals, whereas PDC can measure causality.

(1) Magnitude Squared Coherence (MSC). MSC (or simply coherence) has been a well-established and traditionally used tool to investigate the linear relation between two signals or EEG channels. Let us suppose that we have two simultaneously measured discrete time series x_i and y_i, $i = 1 \ldots N$. MSC is the cross-spectral density function $S_{xy}(f)$, which is simply derived via the fourier transform of the crosscorrelation, normalized by their individual autospectral density functions. Hence, MSC is calculated using the Welch's method as

$$\gamma_{xy}(f) = \frac{\left|\left\langle S_{xy}(f)\right\rangle\right|^2}{\left|\left\langle S_{xx}(f)\right\rangle\right|\left|\left\langle S_{yy}(f)\right\rangle\right|}, \tag{3}$$

where $\langle \cdot \rangle$ indicates window averaging. The estimated MSC for a given frequency f ranges between 0 (no coupling) and 1 (maximum linear interdependence).

(2) A Robust Interdependence Measure (RIM). Given two scalar time series $\{x(t)\}_{t \in \mathbb{T}}$ and $\{y(t)\}_{t \in \mathbb{T}}$ with $\mathbb{T} = \{1, \ldots, N\}$, which have been measured from dynamical systems X and Y, the dynamics of the systems are reconstructed using delay coordinates [24]

$$\mathbf{x}(t) = [x(t), x(t+\tau), \ldots, x(t+(m-1)\tau)]^T \tag{4}$$

and similarly we reconstruct $\mathbf{y}(t)$ from $\{y(t)\}_{t \in \mathbb{T}}$, with an embedding dimension m and a delay time τ for $n \in \mathbb{T} = \{1, \ldots, N\}$, where $N = N - (m-1)\tau$.

Regarding τ and m, they are parameters of Arnhold's method [25]. Taken's [24] embedding theorems and their sequels (e.g., [26]) are existence proofs but they do not directly show how to get a suitable time delay τ or embedding dimension m from a finite time series. Empirical and heuristic criteria are employed for selecting τ and m. Usually, a choice of τ is the value for which the autocorrelation function first passes through zero, while m is determined using variations of false nearest neighbour statistics [27–29]. Parameter τ can also be calculated using the method of Fraser [30].

Let $r_{t,j}$ and $s_{t,j}$, $j = 1, \ldots, k$, denote the time indices of the k nearest Euclidean neighbors of $\mathbf{x}(t)$ and $\mathbf{y}(t)$, respectively. Temporally correlated neighbors are excluded by means of a Theiler correction: $|r_{t,j} - t| > m \cdot \tau$ and $|s_{t,j} - t| > m \cdot \tau$. For each $t \in \mathbb{T}$, the average square distance of $\mathbf{y}(t)$ to all remaining points in $\{\mathbf{y}(j)\}_{j \in \mathbb{T}}$ is given by

$$R_t(Y) = \frac{1}{N-1} \sum_{j=1, j \neq t}^{N} |\mathbf{y}(t) - \mathbf{y}(j)|^2. \tag{5}$$

For each $\mathbf{y}_t$, the X-conditioned mean squared Euclidean distance is defined as

$$R_t^{(k)}\left(\frac{Y}{X}\right) = \frac{1}{k}\sum_{j=1}^{k} \left|\mathbf{y}(t) - \mathbf{y}\left(r_{t,j}\right)\right|^2. \tag{6}$$

Quiroga et al. [25] defined the dependence measure

$$N\left(\frac{Y}{X}\right) = \frac{1}{N}\sum_{t=1}^{N} \frac{R_t(Y) - R_t^{(k)}(Y/X)}{R_t(Y)}. \tag{7}$$

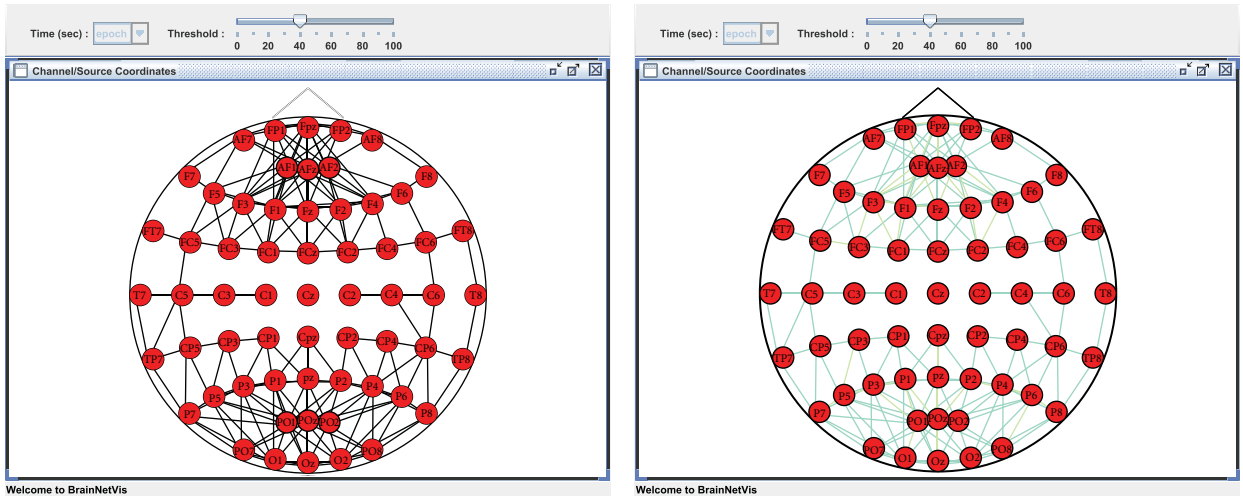
The measure $N(X/Y)$ is defined in complete analogy, and as interdependence measure between X and Y, we use the mean value $(N(X/Y) + N(Y/X))/2$.

(3) Partial Directed Coherence (PDC). Let $\{\mathbf{x}(t)\}_{t \in \mathbb{N}}$ with $\mathbf{x}(t) = [x_1(t), \ldots, x_n(t)]^T$ be a stationary n-dimensional time series with mean zero. Then, a vector autoregressive model of order p for $\mathbf{x}$ is given by

$$\mathbf{x}(t) = \sum_{r=1}^{p} \mathbf{A}(r)\mathbf{x}(t-r) + \varepsilon(t), \tag{8}$$

where $\mathbf{A}(r)$ are the $n \times n$ coefficient matrices of the model and $\varepsilon(t)$ is a multivariate Gaussian white noise process with covariance matrix $\mathbf{\Sigma}$. In this model, the coefficients $A_{ij}(r)$ describe how the present values of x_i depend linearly on the past values of the components x_j. In order to provide a frequency domain measure for Granger-causality, Baccala and Sameshima [23] introduced the concept of PDC. This measure is based on the Fourier transform of the coefficient series

$$\overline{\mathbf{A}}(\omega) = I - \sum_{r=1}^{p} \mathbf{A}(\omega) e^{-i\omega r}. \tag{9}$$



(a) Binary network using threshold = 0.4

(b) Greyscale network using colormap scale. The *warmer* the color of the edge, the stronger the coherence of its adjacent vertices

FIGURE 1: Example of weighted networks for a virtual alcoholic patient. Both pictures are produced with the Arnhold's method for broadband activity.

More precisely, the PDC from x_j to x_i is defined as

$$\pi_{i\ j}(\omega) = \frac{\left|\overline{A}_{ij}(\omega)\right|}{\sqrt{\sum_{l=1}^{n}\left|\overline{A}_{lj}(\omega)\right|^2}}. \tag{10}$$

The PDC $\pi_{i\ j}(\omega)$ takes values between 0 and 1 and vanishes for all frequencies ω if and only if the coefficients $A_{ij}(r)$ are zero for all $r = 1, \ldots, p$.

The synchronization matrix created using one of the above methods serves as input to the BrainNetVis tool thus, it should be calculated separately and a priori. Please note that the presented tool currently implements only graph characterization measures and visualization schemes. It can be used with a variety of inputs in the form of the adjacency matrix. However, we provide the preprocessing section mostly for the interested but not expert user that wishes to investigate how graph analysis may be applied to the neuroscience field. In this sense, even if signal processing techniques are outside of the scope of the tool, we do describe the most widely used methods that provide the input for the further graph analysis. Nevertheless, it is true that most of the methods presented, linear (i.e., PDC) but mostly nonlinear ones (i.e., RIM), assume some kind of *stationarity*. Generally EEG distribution is considered as a multivariate Gaussian process even if the mean and covariance properties generally change from segment to segment. Therefore, strictly speaking, EEG meets quasistationarity because it can be considered stationary only within short intervals. Hence, the user should somehow test the stationarity assumptions prior to using these methods. Hopefully, a novel and prosperous technique capable of decomposing a multivariate time series into its stationary and nonstationary part is known as stationary subspace analysis (SSA) [31] and can be utilized to overcome the implicit stationarity constraints.

3.2.1. Binary and Greyscale Networks on BrainNetVis. BrainNetVis provides the option of using either a *binary* or a *greyscale* network by adjusting, respectively, the *Network Metrics Options* under the *View* drop down menu. In our use case, we provided as input to the tool a synchronization matrix describing the brain network of a *virtual* alcoholic patient. This virtual patient has been created by taking the means across the node and edge values over *all* 30 alcoholic subjects. We underline that this subject does not actually exist. We applied a binary network, using *threshold* = 0.4 and a greyscale network which we visualized using colormap scale. The edge length transformation function can also be selected under the same menu. We used

$$\ell(e) = \frac{1}{x}. \tag{11}$$

The results are depicted in Figure 1.

3.2.2. Data Structure. Two types of files are required for the algorithms that BrainNetVis encapsulates to run properly

(1) A square synchronization matrix with the data from the EEG study (*required for the algorithms to function*).

(2) A file containing a matrix of the labels and the coordinates of each electrode. The rows of the table correspond to the electrodes. The first column contains the electrodes' labels, and the other columns contain the coordinates of the electrodes. These will be either 2 columns (for 2D data, respective to x and y coordinates) or three columns (for 3D data, respective to x, y, and z coordinates). (*required for the visualization options*)

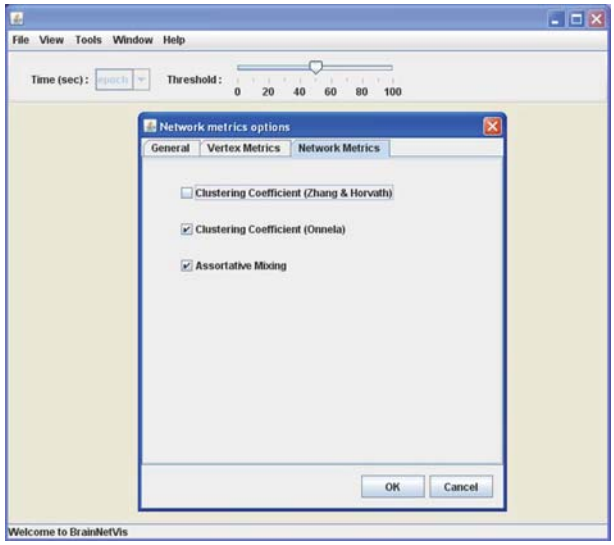


Figure 2: A menu screenshot depicting the selection of global network metrics.

3.3. Menu Calls (GUI). The network metrics available in BrainNetVis will be presented here, in a way that follows the tool's structure.

3.3.1. Global Network Metrics. Networks are often classified into unifying categories in order to obtain a better understanding of their structure and function. *Network measures* are numbers which capture reduced information for graphs and describe essential properties. Network measures should catch the relevant and needed information, they should differentiate between certain classes of networks and be easily computed in order to be useful in algorithms and applications.

A very important global network metric is *clustering coefficient*. The clustering coefficient has been introduced by Watts and Strogatz [32] in 1998. For a vertex v, the clustering coefficient $c(v)$ measures the connectivity of its direct neighborhood. The clustering coefficient $C(G)$ of a graph is the average of $c(v)$ taken over all vertices.

In the BrainNetVis application, we implement two different kinds of clustering coefficients, proposed by Zhang and Horvath (the first) and Onnela (the second). Zhang and Horvath proposed a definition which uses only the network values, in the context of gene coexpression networks. On the other hand, Onnela proposed a version of local clustering coefficient based on the concept of subgraph intensity, defined as the geometric average of subgraph edge values. Both metrics are defined in Table 1. It has to be noticed that the Onnela clustering coefficient definition suffers from the drawback that it requires an underlying binary network; if this is not available as a separate set of data, then presumably it must be obtained by discretizing the weighted edges.

The other important global network metric, included in the tool, is *assortative mixing*. This feature captures the similarity between properties of adjacent network vertices. Intuitively, this measure captures the tendency of network vertices to connect either to vertices with similar degrees (high degrees connected with high degrees and low degrees connected with low degrees) or to vertices that have dissimilar degrees (high degrees connected with low degrees). Newman [33] proposed an interesting measure to quantify the degree of similarity (dissimilarity) between adjacent vertices in a network using assortative mixing, which is given as the correlation between properties of every pairs of adjacent vertices. Each vertex may have assigned a single scalar, such as a centrality measure of the vertex position in a network, or a set of scalar properties. Then, the assortativity coefficient for an undirected graph is defined as the (sample) Pearson product-moment correlation coefficient. The formula of this computation is given in Table 1, and it is written in a symmetrical form. This equation can also be used for directed graphs by simply ignoring the direction of edges.

The value of the assortativity coefficient, r, lies in the range $-1 \quad r \quad 1$, with $r = 1$ indicating perfect assortativity and $r = -1$ indicating perfect disassortativity (perfect negative correlation between the properties of the vertices of the edges under consideration). Brain functional networks tend to be assortative [34, 35]. From computational studies, it has been observed that information gets easily transferred through assortative networks as compared to that in disassortative networks [36].

Global network metrics on BrainNetVis. BrainNetVis allows the calculation of the mentioned global network metrics by following the *Tools* menu (see Figure 2). Continuing the previous example on an alcoholic patient, we applied the simple *Clustering Coefficient* and the *Assortative Mixing*.

3.3.2. Vertex Metrics-Centrality Measures. The above concerned global network metrics. There exists a significant interest in local network properties as well, which concentrates on one node of interest. These properties are very important since at the local scale we can detect which vertices are the most relevant for the organization and functioning of a network. These local measures are commonly named *centrality measures* (or centrality indices) and have proved of great value in analysing the role played by individuals in social networks and in identifying essential proteins, keystone species, and functionally important brain regions.

Centrality Measures Based on Neighbourhoods. The simplest and most basic centrality measure is *degree centrality* $c_D(v)$ of a vertex v. In practice, this is the number of neighbours of the node of interest. In spite of the simplicity of this concept, degree is the most fundamental network measure and most other centrality measures are linked to it. The definitions of degree centrality, both for directed and for undirected networks are provided in Table 1.

In the case of greyscale networks, instead of using the term *degree centrality*, we use the term *strength centrality*. The formulas for strength centrality are defined correspondingly (Table 1). In BrainNetVis, strength centrality is presented as *normalized degree centrality*. This is accessed when the user chooses the *Normalized Metrics* on the Tools Network

Table 1: Network and vertex metrics available in BrainNetVis.

Zhang and Horvath	$c_Z(v) = \sum_{i\neq j\ V\setminus\{v\}} \hat{w}_{vi}\hat{w}_{ij}\hat{w}_{jv} / \sum_{i\neq j\ V\setminus\{v\}} \hat{w}_{vi}\hat{w}_{jv}$ $c_Z(v) = (1/\max_{i,j}(w_{ij})) \cdot (\sum_{i\neq j\ V\setminus\{v\}} w_{vi}w_{ij}w_{jv} / \sum_{i\neq j\ V\setminus\{v\}} w_{vi}w_{jv})$ The weights have been normalized by $\max_{i,j}(w_{ij})$. The above definition uses only the network values, in the context of gene coexpression networks.
Onnela	$c_O(v) = (1/\binom{\deg(v)}{2}) \sum_{i\neq j\ V\setminus\{v\}} (\hat{w}_{vi}\hat{w}_{ij}\hat{w}_{jv})^{1/3}$ $c_O(v) = (1/\max_{i,j}(w_{ij})\binom{\deg(v)}{2}) \sum_{i\neq j\ V\setminus\{v\}} (w_{vi}w_{ij}w_{jv})^{1/3}$ Here, the edge values are normalized by the maximum value in the network, $\hat{w}_{ij} = w_{ij}/\max_{l,k} w_{lk}$.
Assortative mixing	
Symmetrical weighted networks	$r = (4m\sum_{\{u,v\}\ E}\rho(u)\rho(v) - [\sum_{\{u,v\}\ E}(\rho(u)+\rho(v))]^2) / (2m\sum_{\{u,v\}\ E}(\rho(u)^2+\rho(v)^2) - [\sum_{\{u,v\}\ E}(\rho(u)+\rho(v))]^2)$
Directed weighted networks	$r = (H\sum_{(u,v)\ E}\omega(u,v)\rho(u)\rho(v) - AB) / (\sqrt{H\sum_{(u,v)\ E}\omega(u,v)\rho(u)^2 - A^2}\sqrt{H\sum_{(u,v)\ E}\omega(u,v)\rho(v)^2 - B^2})$ $A = \sum_{(u,v)\ E}\omega(u,v)\rho(u)$ $B = \sum_{(u,v)\ E}\omega(u,v)\rho(v)$ $H = \sum_{e\ E}\omega(e)$ is the sum of all values of edges in E.
Degree centrality $c_D(v)$ of vertex v	
Undirected binary network	Degree $\deg(v)$ of vertex v
Directed binary network	In-degree $c_{iD}(v) = \deg^-(v)$ Out-degree $c_{oD}(v) = \deg^+(v)$
Strength centrality $c_S(v)$	
Greyscale symmetric network	Strength $s(v)$ of vertex v
Greyscale assymetric network	In-strength: $c_{iS}(v) = s^-(v)$ Out-strength: $c_{oS}(v) = s^+(v)$
Shortest-path Efficiency	$c_{Ef}(v) = (1/n_{Ef})\sum_{u\neq v} 1/d_G(v,u)$, where $n_{Ef} = n - 1$
Shortest-path Betweeness centrality $c_B(v)$ of a vertex v V	$c_B(v) = (1/n_B)\sum_{s\ V\setminus\{v\}}\sum_{t\ V\setminus\{v,s\}}(\sigma_{st}(v)/\sigma_{st})$, where σ_{st} is the number of shortest (s,t)-paths $\sigma_{st}(v)$ is the number of shortest (s,t)-paths passing through some vertex v other than s, t and $n_B = (n-1)(n-2)$ is a normalizing constant.
Bonacich's eigenvector centrality	$\lambda c(v_i) = \sum_{j=1}^{n} w_{ji}c(v_j)$ In matrix notation with $\mathbf{c} = [c(v_1), c(v_2), \ldots, c(v_n)]^T$, this yields: $\lambda\mathbf{c} = W^T\mathbf{c}$. This type of equation is well known and solved by the eigenvalues and eigenvectors of W^T. We call the eigenvector $\mathbf{s} = [s_1, \ldots, s_n]^T$ of the maximal eigenvalue of $\lambda\mathbf{c} = W^T\mathbf{c}$ principal eigenvector. Then, the eigenvector centrality of node v_i is defined as: $c_{EV}(v_i) = \lvert s_i\rvert / \lVert\mathbf{s}\rVert_p$, where the centrality vector $\mathbf{s}$ is normalized by dividing it by its p-norm $\lVert\mathbf{s}\rVert_p = (\sum_{i=1}^{n}\lvert s_i\rvert^p)^{1/p}$ 1 $p <$, and $\lVert\mathbf{s}\rVert_p = \max_{i=1,\ldots,n}\{\lvert s_i\rvert\}$ $p =$ to produce centrality scores $c(v_i)$ 1.
Hubbell's centrality	$\mathbf{c} = \alpha W^T\mathbf{c} + \mathbf{e}$ where $\mathbf{c} = [c(v_1), c(v_2), \ldots, c(v_n)]^T$ and $\mathbf{e} = [e_1, e_2 \ldots, e_n]^T$. In order to get meaningful results, α should be chosen according to restriction $\lvert\alpha\rvert < 1/\lambda_1$, where λ_1 is the maximum value of an eigenvalue of W. This restriction is not mentioned in the literature.
Subgraph centrality of vertex v_i	It is given by the ith diagonal entry of the kth power of the adjacency matrix, A $c_{SG}(v_i) = \sum_{k=0}\mu_k(i)/k!$ with number of closed walks: $\mu_k(i) = (A^k)_{ii}$. This measure generalizes to greyscale networks by substituting matrix W for A.

Table 1: Continued.

Network entropy	$H(\hat{P}) = -\sum_{i,j} \pi_i \hat{p}_{ij} \log \hat{p}_{ij} = \sum_i \pi_i H_i$ To produce the above equation, we have set a Markov matrix $P = [p_{ij}]$ be the stochastic process which defines the information source and its stationary distribution $\pi : \pi P = \pi$.

Metrics Options General tab and normalizes the edge values to range from 0 to 1 accordingly.

Centrality Measures Based on Distances. Another set of informative measures are the *Centrality Measures Based on Distances*, implying distances that information has to cover in order to be transferred through the network. The first metric that falls in this category is *closeness centrality*. Closeness can be regarded as a measure of how long it will take the information to spread from a given vertex to others in the network. Setting $G = (V, E)$ as an undirected graph, the shortest path closeness centrality of vertex $v \quad V$ is defined as the inverse of the mean geodesic distance from vertex v to every other vertexe. A serious drawback of this metric is that it can only be used for connected graphs. A new measure, called *shortest path efficiency*, is proposed in Latora and Marchiori [37] and implemented in BrainNetVis application.

For a vertex v, Latora and Marchiori defined efficiency as

$$ef(v) = \frac{1}{n-1} \sum_{u \neq v} \frac{1}{d_G(v,u)}. \quad (12)$$

The formula for that is provided in Table 1.

Note that (12) can also be used for disconnected graphs. If some vertices v and u are not connected, then they do not contribute to $ef(v)$. In this case, $d_G(v,u) = +$ $1/d_G(v,u) = 0$. The global efficiency, $ef(G)$, of a graph is the average of $ef(v)$ taken over all vertices [37]

$$ef(G) = \frac{1}{n} \sum_{v} ef(v) = \frac{1}{n(n-1)} \sum_{v} \sum_{V} \sum_{u \neq v} \frac{1}{d_G(v,u)}. \quad (13)$$

In addition to *shortest path efficiency*, we are interested in *shortest-path betweenness centrality*. In this metric, two other nodes, apart from the central vertex v, are involved. We call these nodes s and t, respectively. This metric intuitively refers to the number of shortest paths which connect vertices s and t that pass through vertex v. In the formula provided in Table 1, the relative numbers $\sigma_{st}(v)/\sigma_{st}$ are interpreted as the extent to which vertex v controls the communication between vertices s and t. A vertex is considered central, if it is between many pairs of other vertices. Shortest-path betweenness centrality can be generalized to greyscale networks where the length of a path is equal to the sum of the lengths of its edges.

Centrality measures based on Neighborhoods and on Distances in BrainNetVis. We applied the above types of centrality measures on our synchronization matrix of the alcoholic patient's EEG. Figure 3 depicts the visualization of the individual's brain network using the *Static Visualization Method*. The *Binary Network* using threshold = 0.4 has been selected. The centrality measures calculated are the *Degree Centrality*, *Shortest Path Efficiency* and *Shortest Path Betweenness Centrality*. They are depicted on the respective table, shown in the same figure. Both the figure and the table with the metrics can be created by the following the *View* menu.

Spectral Centrality Measures. Another set of network metrics is based on the calculation of the *eigenvectors* of the adjacency matrix of the network, produced at the preprocessing step. Most of them are calculated by solving a linear equation system. These measures are called *Spectral Centrality Measures. Bonacich's eigenvector centrality* is one of them according to which the centrality of each vertex is proportional to the sum of the centralities of the vertices to which it is directly connected. The respective formula is presented in Table 1.

Expanding the simple Bonacich's eigenvector centrality, Hubbell [38] suggested yet another centrality measure based on the solution of a system of linear equations. *Hubbell's centrality* uses an approach based on directed weighted graphs where the weights of the edges may be real numbers. The general assumption of Hubbell's centrality is similar to the idea of Bonacich, but the centrality of a vertex depends both on its connection to other vertices and to exogenous input which sometimes is called boundary conditions. In this case, we include one more input to the equation $\lambda \mathbf{c} = W^T \mathbf{c}$ which describes Bonacich's eigenvector centrality. The result is shown on Table 1. This formula encapsulates the relative importance of endogenous versus exogenous factors in the determination of centrality.

The next spectral centrality measure, *subgraph centrality*, has been introduced by Estrada et al. [39]. It is calculated as the weighted sum of the number of closed walks in a graph, where longer walks receive lower weight than shorter ones. Very relative to the subgraphs of the network is the number of short walks of length k, starting and ending on vertex v_i. This number is symbolized with $\mu_k(i)$ on Table 1.

Last but not least, a very interesting idea was suggested by Demetrius et al. [40], describing *network entropy*. Evidence has been presented that this quantity is related to the capacity of the network to withstand random changes in the network structure. Network entropy is based on the Kolmogorov-Sinai (KS) entropy, which is a generalization of the Shannon entropy in that it describes the rate at which a stochastic process generates information. In our context, information corresponds to a sequence of vertices visited by an assumed Markov process on the network. Network entropy takes into account the impact of a vertex's removal on the network. This is captured by the product $\pi_i H_i$ of the respective definition on Table 1. The interested reader could find more detailed information in [41].

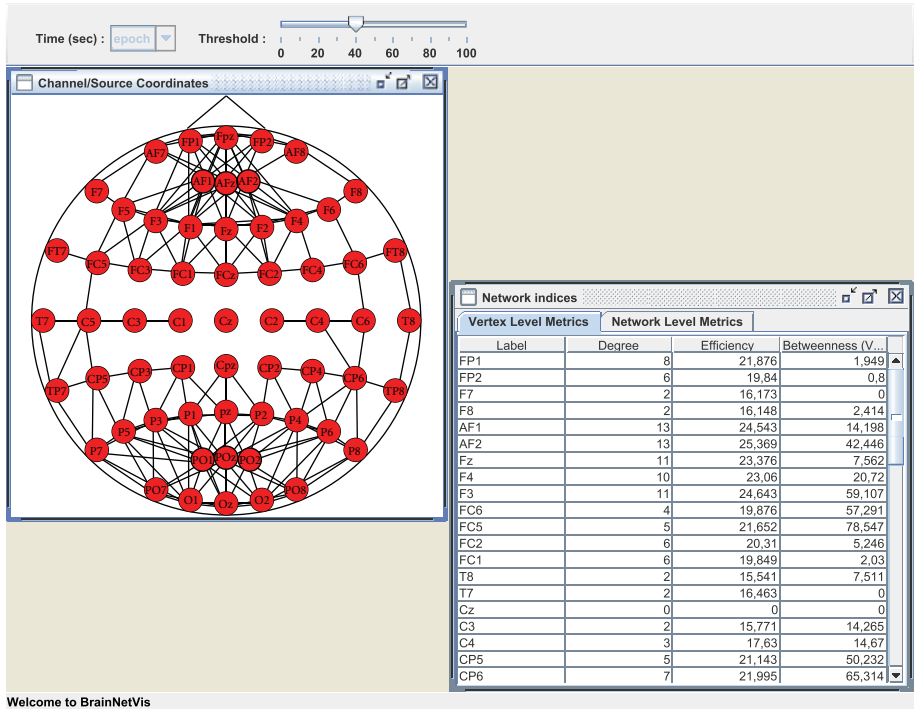


Figure 3: Centrality measures for the virtual alcoholic patient based on neighborhoods and on distances in BrainNetVis. The graph has been calculated by the Arnhold's method for broadband activity.

Spectral Centrality Measures in BrainNetVis. We applied the above types of centrality measures on our synchronization matrix of the alcoholic patient's EEG. Using links from the *Tools* menu, we calculated the *Bonacich's Eigenvector Centrality*, *Hubbell's Centrality*, *Subgraph Centrality*, and *Network Entrophy*. One can define the type of networks with which he wishes to work (binary or greyscale) and also select the threshold value.

3.4. Graph Drawing Techniques. Regarding the way in which the brain is depicted, BrainNetVis tool incorporates three different kinds of visualization as the follows.

3.4.1. Static Visualization Method. In this method, in order to visualize the topology of the emerged network, we create a static framework where each electrode is depicted by a node placed in a position similar to the actual electrode's position on the human cortex. Depending on the number of the electrodes of each experiment, an oval shape is outlined (which corresponds to the scalp) and inside this oval shape, a number V of circles exist that correspond to the electrodes placed on the subjects' head during the experiments.

3.4.2. Multidimensional Scaling. Multidimensional Scaling (MDS) is a family of techniques for analysis and visualization of complex data. The "beauty" of MDS is that we can analyze any kind of distance or similarity matrix, in addition to correlation matrices. Objects in a data set are represented as points in a geometric space; distance in this space represents proximity or similarity among objects. In our case, the objects are the electrodes and the distances among them are respective to their correlation in the synchronization matrix. In general, the goal of the analysis is to detect meaningful underlying connections among the electrodes which reflect the connections among different brain functional regions. In *BrainNetVis*, we incorporated a 2D visualization of the connections among electrodes. At this point, it has to be noticed that the more dimensions we use in order to reproduce the distance matrix, the better the fit of the reproduced is matrix to the observed matrix (i.e., the smaller the stress is). In fact, if we use as many dimensions as there are variables, then we can perfectly reproduce the observed distance matrix. Of course, our goal is to reduce the observed complexity of nature, that is, to explain the distance matrix in terms of fewer underlying dimensions. Some exemplar views of multidimensional scaling are shown in Figure 4

3.4.3. Force-Based or Force-Directed Algorithms. These are a class of algorithms for drawing graphs in an aesthetically pleasing way. Their purpose is to position the nodes of a graph in two-dimensional or three-dimensional space so that all the edges are of more or less equal length and there are as few crossing edges as possible. The force-directed algorithms achieve this by assigning forces amongst the set of edges and the set of nodes; the most straightforward method is to assign forces as if the edges were springs (see Hooke's law), and the nodes were electrically charged particles (see Coulomb's law). The entire graph is then simulated as if it were a physical system. The forces between its nodes change the dynamics and the layout of the system which at

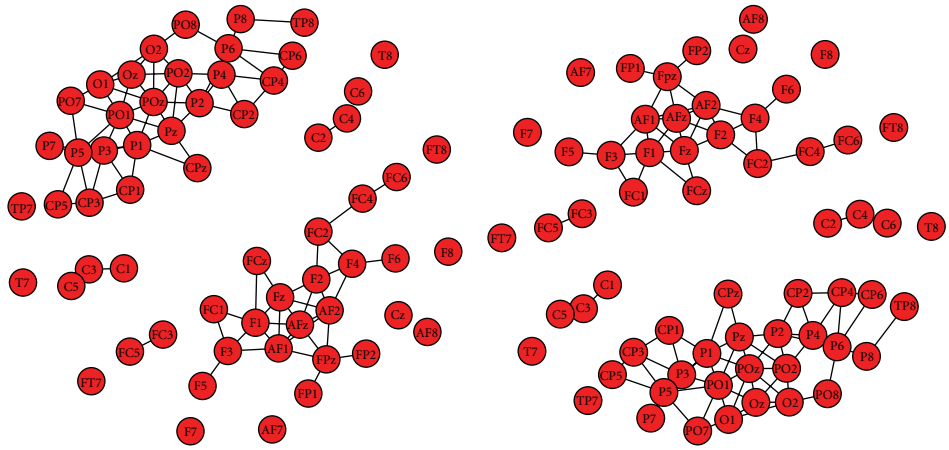


Figure 4: Multidimensional scaling.

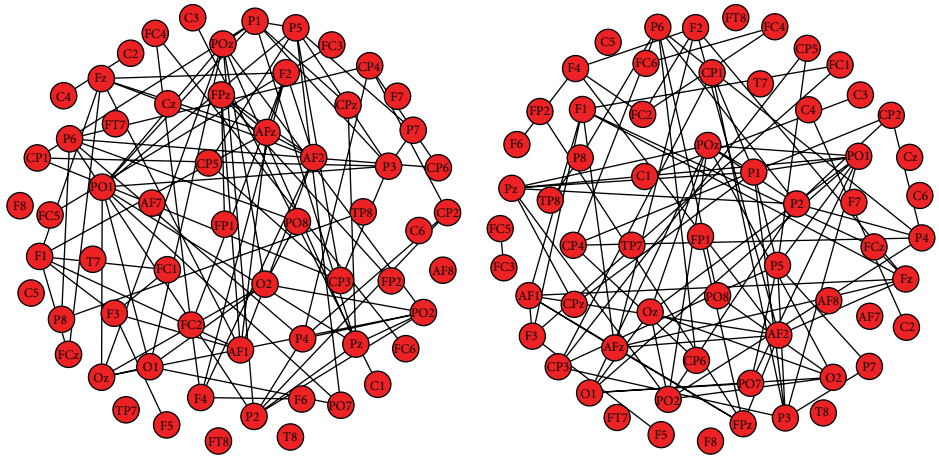
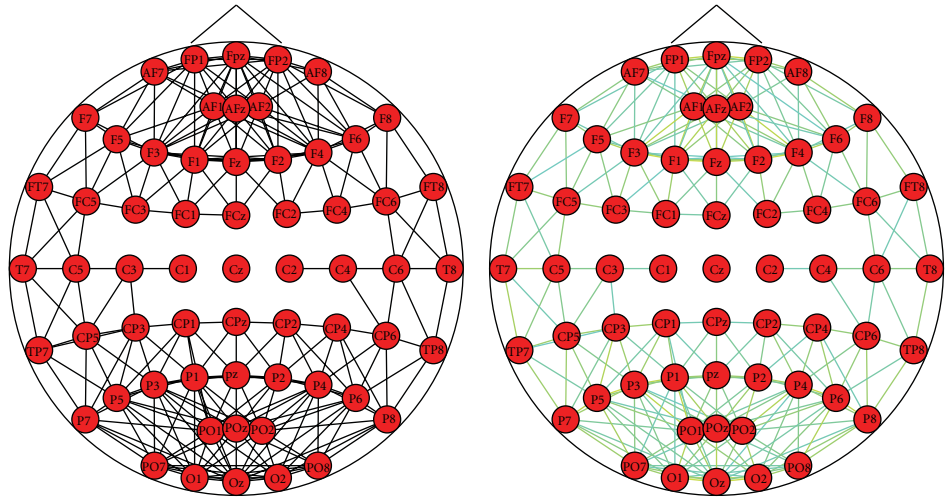


Figure 5: Binary stress.

some point reaches its equilibrium state: at that moment, the graph is drawn. For force-directed graphs, it is also possible to employ mechanisms that search more directly for energy minima, either instead of or in conjunction with physical simulation. One of these mechanisms is binary stress (bStress), and it is the one we have incorporated in our tool. This model bridges the two most popular force directed approaches—the stress and the electrical-spring models—through the binary stress cost function, which is a carefully defined energy function with low descriptive complexity allowing fast computation via a Barnes-Hut scheme. Both electric-spring and stress approaches enjoy successful implementations and offer pleasing layouts to many graphs. Electric-spring models have the advantage of a lower descriptive complexity compared to the stress model. On the other hand, the stress function has a mild landscape, which allows utilizing powerful optimization techniques such as majorization. This way, good minima are usually achieved regardless of the initial positions. As far as the binary stress model is concerned, computationally, it is able to merge the advantages of both the electric-spring model and the stress model. Namely, it offers a low descriptive complexity, while at the same time, it is similar in its form to the known stress function, thus enabling the use of the majorization optimization scheme. More than other models, bStress emphasizes uniform spread of the nodes within a circular drawing area. In addition, bStress is suitable for drawing large graphs, not only because of its improved scalability, but also because it achieves good area utilization. Some exemplar views of binary stress visualization scaling are shown in Figure 5

More information on graph drawing techniques can be found in [13].

When we choose to visualize our graphs using the static visualization method, a change in the network metrics is not depicted on the output panel; this is because the electrode positions are stable and set from the beginning. Nevertheless, the changes in the calculations are saved in a



(a) (b)

Figure 6: Static visualization for the synchronization matrix of the virtual control subject using (a) binary network and (b) greyscale network. Instead of scales of grey, the edge weights are depicted in colormap scale. Both pictures are produced with the Arnhold's method for broadband activity.

matrix which is accessible by the end user. On the other hand, in multidimensional and binary stress modeling, the effects that take place when a network metric changes its value are depicted immediately after the change.

One can then set up the display options of his/her preference, for example, set up the way the graph vertices and edges will be displayed. As far as the nodes of the network are concerned, one can arrange their size, their color (*uniform* or *colormap*)and the depiction of the node labels. Regarding the edges, there exist three options for the color: *uniform* for directed networks, *greyscale* for greyscale networks (the intensity of the shadows of grey corresponds to the strength of the respective edge), and *colormap*. *Colormap* is also used in the case of greyscale networks but in this case colors are used: the closer the tint is to red color, the larger the strength of the respective edge is and the closer the tint is to blue color, the smaller the strength of the edge is. Moreover, one can adjust the size of the edge and whether this will be directed or not. Figure 6 depicts the brain of the virtual control subject using both binary and colormap networks. In both cases, the threshold was set to 0.5.

4. Conclusion

Using BrainNetVis, one can visualize and quantify the connections of the brain, based on EEG or MEG acquired signals. The inner brain connectivity is depicted as a graph; different sensor locations (electrodes) are visualized as nodes and their interconnections as edges. Therefore, scientists and clinicians will be able to get a better insight regarding brain connectivity and functionality and deduce more accurate results. We tested the tool using EEG data from alcoholic patients [7]. We were thus able to investigate some structural brain features that EEG and clinical data alone would not reveal. This tool can be easily used by the interested researcher, and it is accessible via http://www.ics.forth.gr/bmi/tools.html. It runs in every operating system that has JRE installed. Future work includes the support of the preprocessing methods mentioned in the same intuitive environment and the support of the binary European Data Format (EDF). Currently, simple ASCii text format is supported for simplicity and flexibility reasons.

Appendix

We present here a summary of the metrics used at BrainNetVis and their placement under the tools menu. The main menu when the GUI opens contains the options: *File, View, Tools, Window, and Help.*

File. This drop-down menu includes the following tabs.

(i) *Import*. Following this tab, the user can give as input the greyscale matrix that corresponds to the network of interest along with the vertex coordinates. He can browse his computer for these required files.

(ii) *Export*. It is used to export the produced visualizations to a file with various formats (.eps,.pdf,.jpg, etc)

(iii) *Exit*. It is used to quit the GUI.

(iv) *Output*. One can export all the metrics of the examined network at a.*txt* file, which is saved in the same directory with the tool executable.

View. Under the *View* drop-down menu, one can find the following.

(i) *Network Visualization*. One can choose among the three supported visualization techniques: Channel/Source coordinates, Multidimensional Scaling and Binary Stress, described in detail in Section 3.4

(ii) *Network Metrics*. Following this tab, the user can ask either for the *Vertex level metrics* table, which contains the values of the vertex metrics that interest the user (and which he chooses under the *Tools* drop-down menu), or for the *Network level metrics*, which contains the values of the global network metrics.

Tools. This menu contains the following.

(i) *Display Options*. Following this tab, the user can set up the display of the graphs. He can set his preferences concerning the nodes (size, color, label, font) and/or the edges (size, color, direction, arrow size).

(ii) *Network Metrics Options*. Three tabs appear in this sub-menu. The first one is named *General* and contains options like if the network is directed or not, binary or not and synchronization network or not. In the latter case, the tool provides an option on the normalization of the edge length. The second tab is named *Vertex Metrics* and contains options for all the vertex metrics described in Section 3.3.2. Finally, the last tab is named *Network Metrics* and contains options for the network metrics described in Section 3.3.1.

Window. Here, the user can change the size of the window of the GUI.

Acknowledgment

The authors wish to thank Dimitris Andreou for the development of the supportive software of the tool's different versions.

References

[1] V. Sakkalis, "Applied strategies towards EEG/MEG biomarker identification in clinical and cognitive research," *Biomarkers in Medicine*, vol. 5, no. 1, pp. 93–105, 2011.

[2] K. J. Friston, "Functional and effective connectivity in neuroimaging: a synthesis," *Human Brain Mapping*, vol. 2, no. 1-2, pp. 56–78, 1994.

[3] E. Bullmore and O. Sporns, "Complex brain networks: graph theoretical analysis of structural and functional systems," *Nature Reviews Neuroscience*, vol. 10, no. 3, pp. 186–198, 2009.

[4] C. J. Stam and J. C. Reijneveld, "Graph theoretical analysis of complex networks in the brain," *Nonlinear Biomedical Physics*, vol. 1, article 3, 2007.

[5] F. De Vico Fallani, L. Astolfi, F. Cincotti et al., "Brain network analysis from high-resolution EEG recordings by the application of theoretical graph indexes," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 16, no. 5, pp. 442–452, 2008.

[6] V. Sakkalis, T. Oikonomou, E. Pachou, I. Tollis, S. Micheloyannis, and M. Zervakis, "Time-significant wavelet coherence for the evaluation of schizophrenic brain activity using a graph theory approach," in *Proceedings of the 28th IEEE-EMBS, Engineering in Medicine and Biology Society (EMBC '06)*, vol. 1, pp. 4265–4268, New York, NY, USA, 2006.

[7] V. Sakkalis, V. Tsiaras, M. Zervakis, and I. Tollis, "Optimal brain network synchrony visualization: application in an alcoholism paradigm," in *Proceedings of the 29th Annual International Conference of IEEE-EMBS, Engineering in Medicine and Biology Society (EMBC '07)*, pp. 4285–4288, 2007.

[8] C. J. Stam, B. F. Jones, G. Nolte, M. Breakspear, and P. Scheltens, "Small-world networks and functional connectivity in Alzheimer's disease," *Cerebral Cortex*, vol. 17, no. 1, pp. 92–99, 2007.

[9] N. Situ, R. Rezaie, A. Papanicolaou, L. Pollonini, U. Patidar, and G. Zouridakis, "Functional connectivity networks in the autistic and healthy brain assessed using granger causality," in *Proceedings of the 32nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 2010.

[10] M. Massimini, F. Ferrarelli, R. Huber, S. K. Esser, H. Singh, and G. Tononi, "Neuroscience: breakdown of cortical effective connectivity during sleep," *Science*, vol. 309, no. 5744, pp. 2228–2232, 2005.

[11] M. Valencia, M. A. Pastor, M. A. Fernández-Seara, J. Artieda, J. Martinerie, and M. Chavez, "Complex modular structure of large-scale brain networks," *Chaos*, vol. 19, no. 2, Article ID 023119, 2009.

[12] V. Sakkalis, C. Doru Giurcăneanu, P. Xanthopoulos et al., "Assessment of linear and nonlinear synchronization measures for analyzing EEG in a mild epileptic paradigm," *IEEE Transactions on Information Technology in Biomedicine*, vol. 13, no. 4, pp. 433–441, 2009.

[13] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis, *Graph Drawing: Algorithms for the Visualization of Graphs*, Prentice Hall, Upper Saddle River, NJ, USA, 1999.

[14] http://sccn.ucsd.edu/eeglab/.

[15] M. Rubinov and O. Sporns, "Complex network measures of brain connectivity: uses and interpretations," *NeuroImage*, vol. 52, no. 3, pp. 1059–1069, 2010.

[16] U. Egert, TH. Knott, C. Schwarz et al., "MEA-Tools: an open source toolbox for the analysis of multi-electrode data with MATLAB," *Journal of Neuroscience Methods*, vol. 117, no. 1, pp. 33–42, 2002.

[17] M. Mørup, L. K. Hansen, and S. M. Arnfred, "ERPWAVELAB: a toolbox for multi-channel analysis of time-frequency transformed event related potentials," *Journal of Neuroscience Methods*, vol. 161, no. 2, pp. 361–368, 2007.

[18] A. Delorme and S. Makeig, "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis," *Journal of Neuroscience Methods*, vol. 134, no. 1, pp. 9–21, 2004.

[19] V. Sakkalis, V. Tsiaras, and I. Tollis, "Graph analysis and visualization for brain function characterization using EEG data," *Journal of Healthcare Engineering*, vol. 1, no. 3, pp. 435–460, 2010.

[20] J. G. Snodgrass and M. Vanderwart, "A standardized set of 260 pictures: norms for name agreement, image agreement, familiarity, and visual complexity," *Journal of Experimental Psychology: Human Learning and Memory*, vol. 6, no. 2, pp. 174–215, 1980.

[21] S. M. Kay, *Modern Spectral Estimation*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1988.

[22] J. Arnhold, P. Grassberger, K. Lehnertz, and C. E. Elger, "A robust method for detecting interdependences: application to intracranially recorded EEG," *Physica D*, vol. 134, no. 4, pp. 419–430, 1999.

[23] L. A. Baccalá and K. Sameshima, "Partial directed coherence: a new concept in neural structure determination," *Biological Cybernetics*, vol. 84, no. 6, pp. 463–474, 2001.

[24] F. Takens, "Detecting strange attractors in turbulence," in *Proceedings of the Dynamical Systems and Turbulence Symposium*, vol. 898 of *Lecture Notes in Mathematics*, pp. 366–381, 1981.
[25] R. Q. Quiroga, A. Kraskov, T. Kreuz, and P. Grassberger, "Performance of different synchronization measures in real data: a case study on electroencephalographic signals," *Physical Review E*, vol. 65, no. 4, Article ID 041903, 14 pages, 2002.
[26] T. Sauer, J. A. Yorke, and M. Casdagli, "Embedology," *Journal of Statistical Physics*, vol. 65, no. 3-4, pp. 579–616, 1991.
[27] M. B. Kennel, R. Brown, and H. D. I. Abarbanel, "Determining embedding dimension for phase-space reconstruction using a geometrical construction," *Physical Review A*, vol. 45, no. 6, pp. 3403–3411, 1992.
[28] L. Cao, "Practical method for determining the minimum embedding dimension of a scalar time series," *Physica D*, vol. 110, no. 1-2, pp. 43–50, 1997.
[29] R. Hegger, H. Kantz, and T. Schreiber, "Practical implementation of nonlinear time series methods: the TISEAN package," *Chaos*, vol. 9, no. 2, pp. 413–435, 1999.
[30] A. M. Fraser and H. L. Swinney, "Independent coordinates for strange attractors from mutual information," *Physical Review A*, vol. 33, no. 2, pp. 1134–1140, 1986.
[31] P. Von Bünau, F. C. Meinecke, F. C. Király, and K. R. Müller, "Finding stationary subspaces in multivariate time series," *Physical Review Letters*, vol. 103, no. 21, Article ID 214101, 2009.
[32] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.
[33] M. E. J. Newman, "Assortative mixing in networks," *Physical Review Letters*, vol. 89, no. 20, Article ID 208701, 4 pages, 2002.
[34] C. H. Park, S. Y. Kim, Y. H. Kim, and K. Kim, "Comparison of the small-world topology between anatomical and functional connectivity in the human brain," *Physica A*, vol. 387, no. 23, pp. 5958–5962, 2008.
[35] V. M. Eguíluz, D. R. Chialvo, G. A. Cecchi, M. Baliki, and A. V. Apkarian, "Scale-free brain functional networks," *Physical Review Letters*, vol. 94, no. 1, Article ID 018102, 2005.
[36] R. Xulvi-Brunet and I. M. Sokolov, "Reshuffling scale-free networks: from random to assortative," *Physical Review E*, vol. 70, no. 6, Article ID 066102, 6 pages, 2004.
[37] V. Latora and M. Marchiori, "Efficient behavior of small-world networks," *Physical Review Letters*, vol. 87, no. 19, Article ID 198701, 4 pages, 2001.
[38] C. H. Hubbell, "An input-output approach to clique identification," *Sociometry*, vol. 28, pp. 377–399, 1965.
[39] E. Estrada and J. A. Rodríguez-Velázquez, "Subgraph centrality in complex networks," *Physical Review E*, vol. 71, no. 5, Article ID 056103, pp. 1–9, 2005.
[40] L. Demetrius, V. M. Gundlach, and G. Ochs, "Complexity and demographic stability in population models," *Theoretical Population Biology*, vol. 65, no. 3, pp. 211–225, 2004.
[41] V. L. Tsiaras, *Algorithms for the analysis and visualization of biomedical networks*, Ph.D. thesis, Computer Science Department, University of Crete, Heraklion, Greece, 2009.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 598206, 11 pages
doi:10.1155/2011/598206

Research Article

fMRI Artefact Rejection and Sleep Scoring Toolbox

Yves Leclercq,[1] Jessica Schrouff,[1] Quentin Noirhomme,[1] Pierre Maquet,[1,2] and Christophe Phillips[1,3]

[1] *Cyclotron Research Centre, University of Liège, 4000 Liège, Belgium*
[2] *Department of Neurology, University of Liège, 4000 Liège, Belgium*
[3] *Department of Electrical Engineering and Computer Science, University of Liège, 4000 Liège, Belgium*

Correspondence should be addressed to Christophe Phillips, c.phillips@ulg.ac.be

Received 30 August 2010; Revised 2 December 2010; Accepted 17 January 2011

Academic Editor: Sylvain Baillet

We started writing the "fMRI artefact rejection and sleep scoring toolbox", or "FA$\mathbb{S}$T", to process our sleep EEG-fMRI data, that is, the simultaneous recording of electroencephalographic and functional magnetic resonance imaging data acquired while a subject is asleep. FA$\mathbb{S}$T tackles three crucial issues typical of this kind of data: (1) data manipulation (viewing, comparing, chunking, etc.) of long continuous M/EEG recordings, (2) rejection of the fMRI-induced artefact in the EEG signal, and (3) manual sleep-scoring of the M/EEG recording. Currently, the toolbox can efficiently deal with these issues via a GUI, SPM8 batching system or hand-written script. The tools developed are, of course, also useful for other EEG applications, for example, involving simultaneous EEG-fMRI acquisition, continuous EEG eye-balling, and manipulation. Even though the toolbox was originally devised for EEG data, it will also gracefully handle MEG data without any problem. "FA$\mathbb{S}$T" is developed in Matlab as an add-on toolbox for SPM8 and, therefore, internally uses its SPM8-meeg data format. "FA$\mathbb{S}$T" is available for free, under the GNU-GPL.

1. Introduction

"FA$\mathbb{S}$T" stands for "fMRI artefact rejection and sleep scoring toolbox". We, researchers from the Cyclotron Research Centre, University of Liège, Belgium, started writing this set of tools to analyze our sleep EEG-fMRI data, that is, both electroencephalographic (EEG) and functional magnetic resonance imaging (fMRI) data acquired simultaneously while the subject is asleep. The joint acquisition of EEG and fMRI data allows the integration of electric and haemodynamic information about brain activity [1]. This is a requirement, for example, in neuroimaging sleep studies as sleep activity can only be derived from the EEG signal, while fMRI allows the localization of haemodynamic signal variation throughout the brain volume [2, 3]. Nevertheless, when processing such data, one has typically to tackle three crucial issues.

Handling Large Data Sets. Reviewing long multichannel continuous recording of M/EEG (magneto- and/or electroencephalographic) activity is cumbersome as it usually involves displaying and manipulating (exploring, comparing, chunking, appending, etc.) large data sets, up to several gigabytes (Gb) for hour long recordings.

fMRI Artefact Rejection. When recording EEG and fMRI data simultaneously, the EEG signal is contaminated by artefacts induced by the gradient switching and high static field of the MR scanner. The rejection of these artefacts is very challenging. If the EEG data are not averaged afterwards, that is, for continuous or single trial analysis, then any inaccuracies in this rejection may have a large and negative impact on the results.

Scoring Data. Scoring continuous M/EEG recordings, such as is common with sleep recordings, is a tedious task, as the scorer has to manually browse through the entire data set and give a "score" to each time-window displayed.

As far as we know, FA$\mathbb{S}$T is the only freely available software package that can efficiently deal with those three issues. Moreover, the tools provided can also be tailored for one's own need, and new features can easily be added

(as additional Matlab functions), leading to a flexible toolbox for anyone dealing with (long) M/EEG recordings, EEG-fMRI data and/or (sleep) signal scoring.

We chose to implement our ideas as an add-on toolbox for SPM8 ([4] and Litvak et al. in this issue) and not, for example, an add-on for EEGlab ([5] and Delorme and Makeig in this issue), for two practical reasons, stemming from our original sleep EEG-fMRI data.

A Single Processing Platform. We found it more convenient to process both EEG and fMRI data within the same software suite. Since we were already using SPM to process our fMRI data, we decided to add EEG tools (continuous recording visualization/handling, fMRI artifact rejection and sleep scoring) to the SPM8 package.

The Data Format. Thanks to the memory mapping feature of the SPM8-`meeg` data format (see Section 2.1), whatever the size of the data set saved on the computer hard disk, only the bits required for the current operation are actually loaded in memory. This offers a quick and transparent access to data set up to several gigabytes, even on a standard computer (32 bits machine and less than 4 Gb of RAM).

The latter point is absolutely crucial for us, and only SPM8 provides this feature at the moment. FA$\mathbb{S}$T thus internally uses the open SPM8-`meeg` data format. The conversion from the original data format to that of SPM8 can be performed directly by FA$\mathbb{S}$T or through SPM8, see Section 2.1. Note that it is not necessary to master the whole SPM8 package to use FA$\mathbb{S}$T. Once in the appropriate format, M/EEG data can then be easily visualized and manipulated; see Section 3.1.

Specifically, for EEG-fMRI acquisitions, FA$\mathbb{S}$T can operate directly on the raw data acquired with a "brainamp MR" system (BrainProducts Gmbh, Gilching, Germany) and includes the well-known "averaged artefact substraction" (AAS) method [6] for the "gradient artefact" rejection, as well as the recently published "constrained independent component analysis" or cICA method [7] for the rejection of the "pulse artefact", see Section 3.2. Other classic methods for the "pulse artefact" rejection are also available. Finally, an easy GUI is available for the manual scoring of continuous M/EEG data: sleep stages, for sleep recordings, or any other "stage"; for other types of data, see Section 3.3. Some statistics and sleep-specific features can also be automatically extracted.

FA$\mathbb{S}$T can be operated in 2 ways: via user-friendly GUIs or the command line. The GUIs let the user select the data to process, and tune various parameters and options for each tool. The "default" parameter values can also be modified by editing a single "default" file, allowing user- or site-specific settings. The command line approach allows the scripting of operations, for example, to automatically process several recordings with one execution of a Matlab script.

FA$\mathbb{S}$T is distributed for free, under the GNU-GPL, and available for download at the following address: http://www.montefiore.ulg.ac.be/ phillips/FASST.html. It comes without any warranty: you should use it at your own risk. A manual detailing FA$\mathbb{S}$T features and possibilities is also available here: http://www.montefiore.ulg.ac.be/ phillips/FASST_manual.pdf.

2. Software Characteristics

FA$\mathbb{S}$T is an add-on to the popular SPM8 software and is written in Matlab, with a few routines written in C/C++ but interfaced with Matlab. Since Matlab is a high level multiplatform computing language, only the few routines written in C/C++ need to be compiled for a specific operating system. So far, those routines were compiled for Windows XP only, but some operating systems (like Windows 7 and Mac OS X) will be directly supported by distributing the compiled routines. For the other OSs (like Unix) a simple compilation script will be made available in the next release. In order to work properly, FA$\mathbb{S}$T, therefore, needs to have the following 2 softwares installed:

(i) a recent version of Matlab. We used version 7.5 (R2007b) (any later Matlab version should *in theory* work too) to develop FA$\mathbb{S}$T, as well as Matlab "signal processing toolbox" (for some filtering functions though this requirement will be lifted in the next release),

(ii) the latest SPM8 version. FA$\mathbb{S}$T relies on the SPM8-`meeg` data format (see Sections 2.1 and 2.2) and also uses some M/EEG-specific routines.

Note also that FA$\mathbb{S}$T, on top of relying on Matlab and SPM8, includes a few routines from the following three other freely available Matlab toolboxes:

(i) EEGLAB ([5] and Delorme and Makeig in this issue), mainly for a few functions used by the constrained ICA "pulse artefact" rejection, available from http://sccn. http://sccn.ucsd.edu/eeglab/,

(ii) the FMRIB plugin for EEGLAB [8], for the electrocardiographic (ECG) peak detection and classic "principal component analysis" (PCA, which is sometimes refered as the "optimal basis set" (OBS) approach) and "Gaussian mean" (a variation of the AAS method) pulse artefact rejection methods, available from http://www.fmrib.ox.ac.uk/eeglab/fmrib plugin/index.html,

(iii) the "mutual information computation" package [9], for the selection of correction matrices during cICA "pulse artefact" rejection, available from http://penglab.janelia.org/proj/mRMR/,

These additional routines are already included in FA$\mathbb{S}$T and do not need to be downloaded separately. We, therefore, thank their authors for letting us use and distribute their work.

2.1. Data Format. FA$\mathbb{S}$T internally works with SPM8-`meeg` data format which stores M/EEG data in two separate files: a `.mat` header file and a `.dat` binary file. contains all the information about the data (channel names and types,

sampling rate, stimuli, etc.), and the binary file stores the data themselves as a raw list of numbers. For a thorough description of SPM8-`meeg` data format, one should have a look at SPM8 documentation, but it is worth mentioning here the key feature of SPM8-`meeg` data format used by FA$T: the whole data set is not loaded into memory, but only the header content. Then only the "window", over time and/or channels, of data required for the current operation (e.g., displaying the signal from 10 channels over 20 seconds) is loaded. The trick is that the data file on disk is memory mapped into Matlab, such that it can be accessed transparently, as a regular variable, without eating up all the memory. For example, an EEG-fMRI sleep recording of 4 hours weights about 9,5 Gb of EEG data (72 channels × 4 hours × 3600 seconds/hour × 5000 Hz sampling rate × 2 bytes/sample) but can be displayed and manipulated without any problem. Any new data file generated by FA$T is of course stored in the same format. Additional information generated and used by FA$T, such as the sleep score encoded by one or more users, are simply added to the header data structure and do not interfere with SPM8 machinery.

Data conversion or importation is always an issue in EEG and MEG, since each company enjoys his own specific (and usually proprietary) format. With FA$T, there are 2 ways to import the data in the right format, either with the SPM8 "convert" function or directly in FA$T.

2.1.1. SPM8 Data Conversion. SPM8 relies on the "fieldtrip" (FT) toolbox [10] and Oostenveld et al. in this issue to read in and convert pretty much any existing EEG/MEG data formats. SPM8-FT generally reads in the original header information/file(s) and creates the `.mat` header file, then goes through the data file(s) and creates the associated binary file. This approach is very robust and probably the safest.

Nevertheless, SPM8-FT conversion may take some time for large data sets, as data are read in and then written on disk in the new `.dat` file. Moreover, depending on the way raw original data are stored, SPM8-FT can more than double the size of your original data on disk: for example, raw EEG data in INT16 format, that is, 2 bytes per sample, are written in FLOAT, that is, 4 bytes per sample, and will thus double the disk space usage.

2.1.2. FA$T Data Conversion. FA$T can use its own specific importation routines, but this approach is much less exhaustive than SPM8-FT in terms of supported data formats. So far only Brain Products (Brain Products GmbH, Gilching, Germany) is directly supported (Raw-EGI (Electrical Geodesics Inc., Eugene, OR, USA) and `.edf` data formats import are only in beta version.) Yet, FA$T specific approach has 2 advantages. First, these data can be directly selected via the FA$T GUI, and they will be converted "on the fly", that is, no need to launch SPM8. Second, brain products EEG data are imported without generating a second data file. Practically, the `.vhdr` and `.vmrk` header files are directly read in by FA$T and translated into SPM8-`meeg` `.mat` file, which is then directly linked to the original binary data file, without reading-converting-writing the data themselves. Moreover, FA$T will also (try to) recover and save the "real-world" beginning time of the recording, that is, the computer clock time of the recording, from the original data. This is useful when comparing, appending and chunking files (see Sections 3.1.2 and 3.1.3).

There is, thus, no "import" button in FA$T and EEG data acquired with a Brain Products amplifier can be directly selected in the GUI: the (header) conversion will take place automatically. This is very fast, as only "administrative" bits of information about the data (including the triggers) are converted, and disk space efficient, as the data binary file is not copied. These two features are particularly useful for simultaneous EEG-fMRI recordings where data files easily reach several Gbs. Other data formats should first be converted using SPM8-FT functionalities.

2.2. Channel Definition. Channel definition (name, type, and 2D-location) is in line with that of SPM8, which also offers GUI facilities to easily edit channel and data information. The goal of FA$T's "channel definition" is not to import subject specific information, such as importing channel location file (this actually can be done within SPM8), but rather to add some features to the standard SPM8 format: this allows mainly the on-the-fly display of simple bipolar montage (like horizontal or vertical EOG's) alongside M/EEG channels and the use of different channel scalings, for example, for M/EEG and ECG/EMG/EOG channels. Most common channel names are already available within the toolbox, but any laboratory or experiment specific setup can be added: The default electrode/sensor setup are defined in a "electrode defaults" file, which is a simple Matlab script easy to edit.

3. Main Functions of the Toolbox

The following tools and features are available in the current version of FA$T:

(i) handling tools: displaying one M/EEG data file, comparing multiple M/EEG data files channel by channel, appending M/EEG data files, chunking a time window from an M/EEG data file, and computing and displaying the spectrogram of one M/EEG data file,

(ii) EEG-fMRI artefact rejection tools: gradient artefact rejection (AAS method) and pulse artefact rejection (AAS, "Gaussian mean" and cICA methods),

(iii) sleep specific tools: manual sleep scoring, spectral power calculations, slow wave detection and propagation, and sleep statistics.

3.1. General Tools. As mentioned previously, these are tools to display, review, compare, process, manipulate, and handle (long) continuous data files, containing EEG, MEG, EOG, ECG, EMG, or any other sampled signal.

3.1.1. Displaying One M/EEG Data File. One continuous M/EEG recording (of any length) can be easily displayed

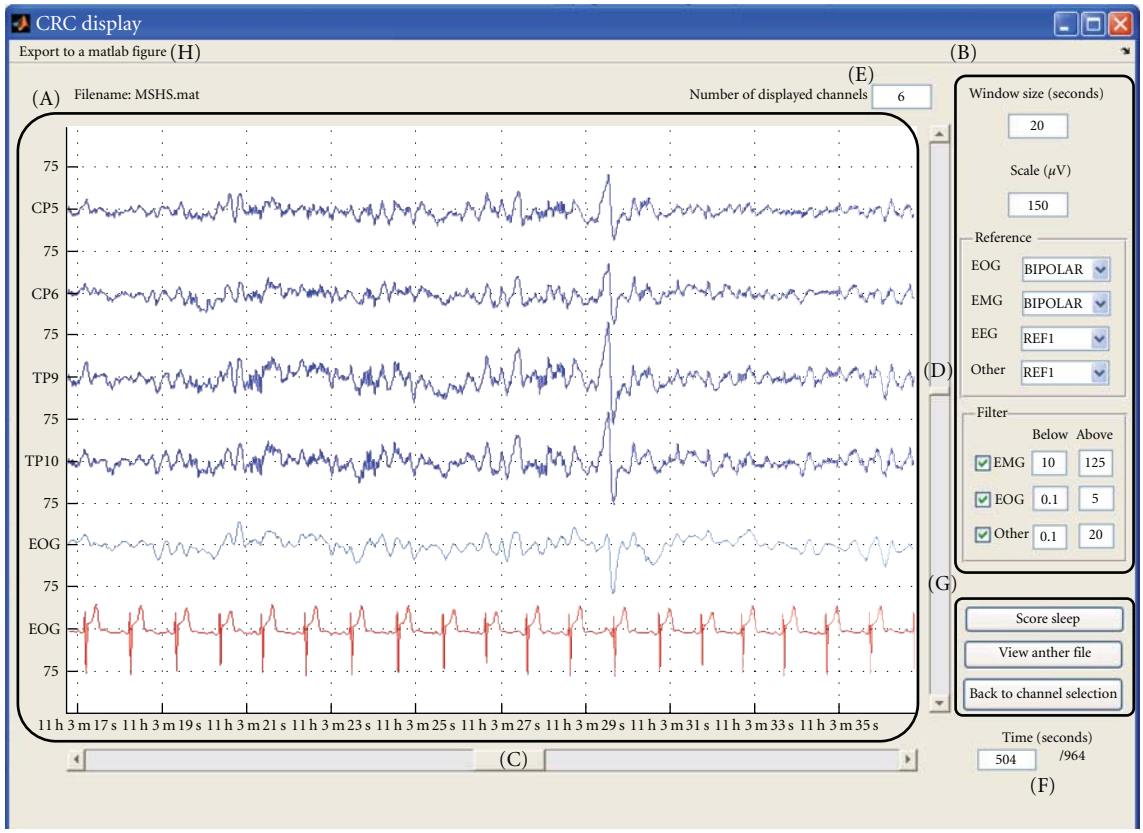


Figure 1: Main signal display window: (A) main display, (B) display options (see text for details), (C) time scrolling bar, (D) channel scrolling bar, (E) number of channels to display at once, (F) time in seconds at the beginning of display, (G) change channels or file to display, or start "sleep scoring", and (H) exportation of the current main display to a new Matlab figure.

and rapidly browsed through; see Figure 1. All or any subset of channels can be selected, then the channel signals are displayed in the main central box. There are two scrolling bars: one to quickly browse throughout the data over time and another one to browse through subsets of selected channels.

To help visualization, standard unipolar channels are displayed in blue, bipolar channels in green, and automatically rescaled channels are shown in red. The list of channels that are automatically rescaled and the bipolar montages are specified in the "electrode defaults" file. For example, electrocardiographic (ECG) signal has a much larger amplitude than EEG, and thus ECG channels should be scaled differently for a convenient display. Several other options are available through the GUI.

(i) The number of channels per screen, time window (in seconds) displayed, and channel scale (in μV) can be modified at any time. The numbers on the side of the main display (75 in Figure 1) indicate the scale used for the EEG channels in μV, except for the EOG/EMG/ECG channels which have a fixed scale defined in the default file.

(ii) Reference can be modified through a pulldown menu. For the EEG channels, this reference may be any other channel, the mean of all the EEG channels or the mean of the two mastoids (called M1 and M2). For the EOG and EMG channels there is an additional "Bipolar" choice for the reference.

(iii) A different bandpass filter can be applied to the different types (EMG, EOG or "Other") of channels.

(iv) The power spectrum of the displayed signal can be directly computed for one channel via a "right-click" pulldown menu in the main display over the specific channel. The resulting spectrum is then shown in a separate Matlab figure.

Note that the rereferencing and filtering are performed only "on-display" and the original data stored on disk are left untouched. There is therefore no need to perform these pre-processing steps before visualizing the data. In fact these features let the user explore the effect of filtering or rereferencing on the displayed data.

3.1.2. Comparing Multiple M/EEG Data Files. This tool is designed to display the same channel from multiple M/EEG files. It can obviously be used to compare the results of different artefact correction methods applied to the same data set or to visually check the effect of any processing applied to a continuous data set. For example, Figure 2 shows the same signal before (bottom) and after (top) pulse artefact correction. Only one channel can be displayed at one time and the displayed channel is selected via a pulldown menu.

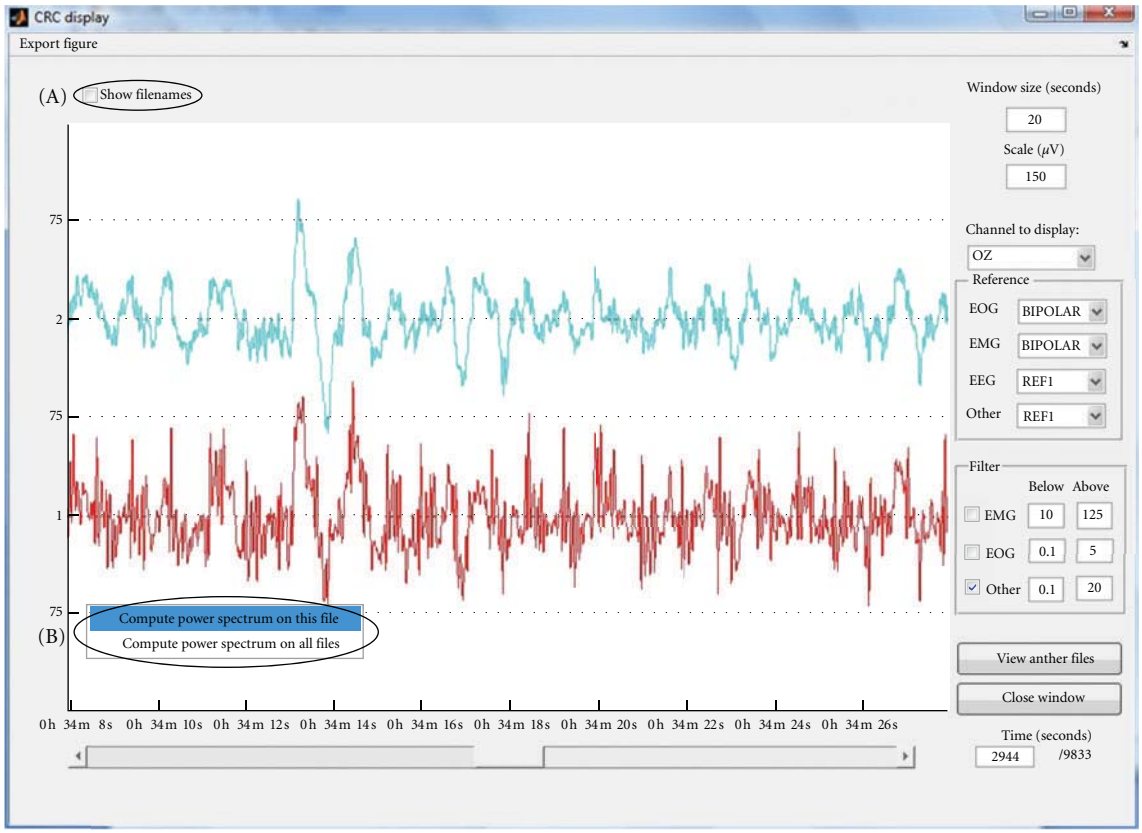


Figure 2: Multiple files comparison GUI: (A) toggle filename display and (B) compute power spectrum of one or all channels.

The routine also checks the "real-world" beginning time, that is, the computer clock time of the recording, of each data set and aligns the different M/EEG time series in consequence. If the beginning time was not imported from the raw data; all files are assumed to begin at the same time with the first sample. The power spectrum of the displayed signals can also be directly estimated and shown in a separate window.

3.1.3. Appending and Chunking M/EEG Data Files. The appending tool is designed to append two separate M/EEG files into a single one. This is particularly useful if recording was (accidentally) interrupted but the different data sets should be considered as one single "recording session". If the "real-world" recording time is available, then the file order is automatically determined and any time gap between the end of the first file and the beginning of the second is filled with zeros. Otherwise the data will simply be appended one directly after the other.

"Chunking" is the opposite of "appending", and lets the user cut out an episode out of a large M/EEG file to save it as a separate data file. This can be useful if one wants to study a specific episode of activity such as sleep stages and epileptic discharge. The beginning and end of the new file can be defined by markers (or triggers) or by time (relative to the beginning of the file or in "real-world" time).

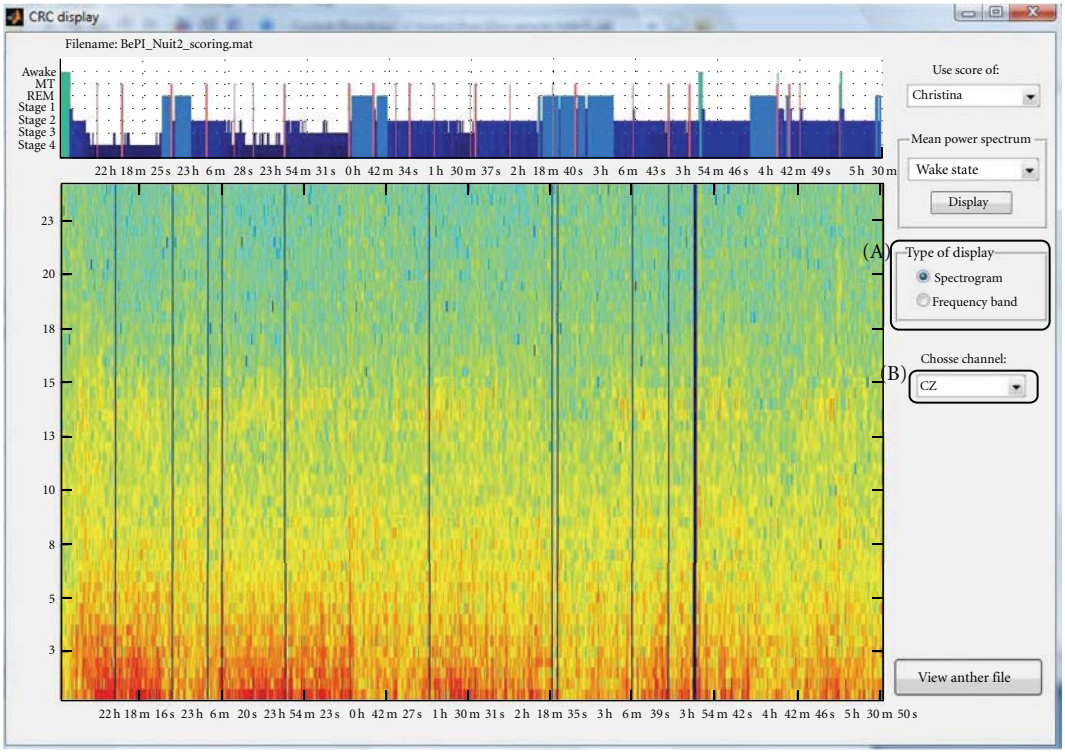
3.1.4. Computing and Displaying the Spectrogram of One M/EEG Data File. Using the Welch periodogram method, the spectrogram of one whole data set can be computed. The output is saved into a time-frequency data file, also in an SPM8-`meeg` compatible format. Before the computation itself, the data are bandpass filtered. Then, the spectrogram is computed over overlapping time windows.

Once calculated, the spectrogram can be displayed in two ways: "spectrogram" is the time-frequency representation of one channel and a pulldown menu is used to select the channel to display (see Figure 3(a)). In the "frequency band" display mode, the evolution of the power in a specified frequency band is displayed for one or two channels (see Figure 3(b)).

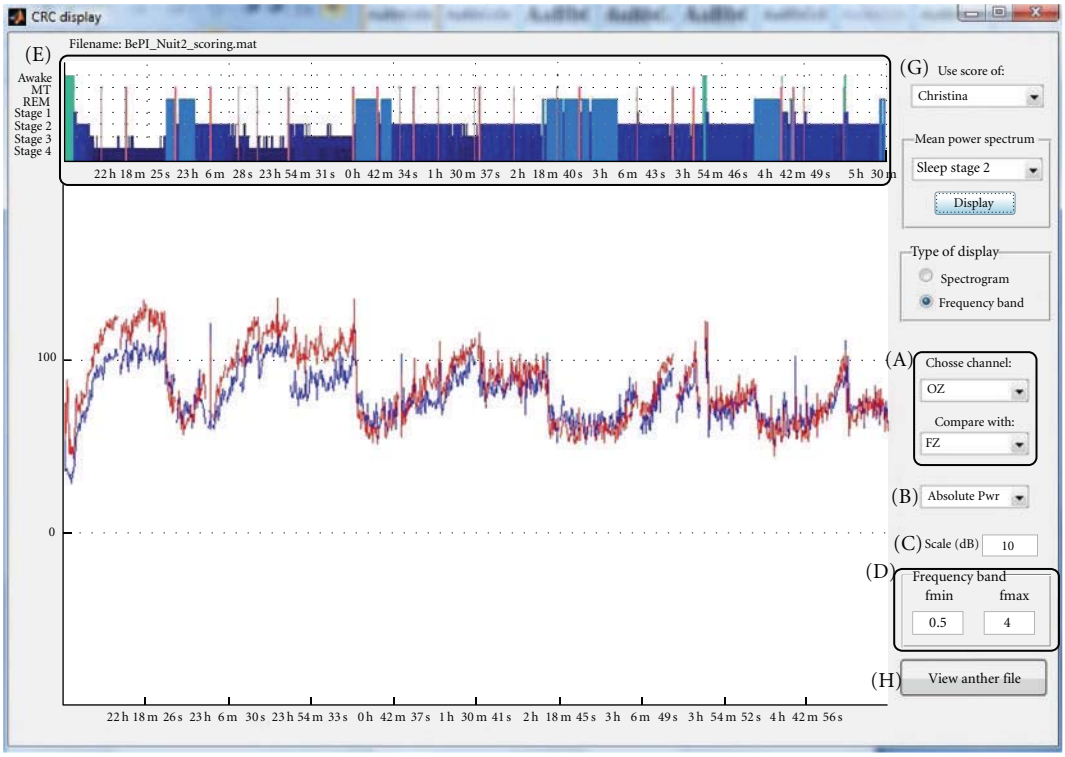
3.2. EEG-fMRI Artefact Rejection Tools. When EEG is recorded during fMRI acquisition, two types of artefacts are induced on top of the neural EEG signal.

(i) The "gradient artefact" (GA) is due to the gradient switching of the imaging sequence of the MR scanner [6].

(ii) The "pulse artefact" (PA) is due to the interaction between the static field of the MR scanner and the heartbeats [11]. This artefact is present even if no fMRI data are acquired.



(a)



(b)

Figure 3: Main display of spectrogram GUI. (a) "Spectrogram display": (A) display mode toggle and (B) channel selection and (b) "frequency band display": (A) channel(s) selection, (B) scaling type, (C) scale for display, (D) frequency band, (E) night hypnogram, (F) mean power spectrum for a specific sleep stage, (G) selection of the hypnogram scorer, and (H) selection of another data file.

One should always suppress the gradient artefact before the pulse artefact, as the amplitude of the former is several orders of magnitude larger than the latter.

3.2.1. Gradient Artefact Rejection. The GA is removed using the "average artefact subtraction" (AAS) method developed by Allen et al. 2000 [6]. AAS estimates the shape of the GA over a "repetition time" (TR, or the time elapsed between the acquisition of two fMRI volumes) by averaging the signal over several (30 by default) contiguous fMRI volume acquisitions. This "averaged artefact" is estimated for each TR and subtracted from the recorded EEG signal. The efficiency of the AAS approach relies on the stationarity of the GA picked in the EEG signal. This stationarity can be enforced by synchronizing the clocks of the EEG amplifier(s) with that of the MR scanner. This is a crucial point, and any user applying this algorithm to data acquired without clock synchronization may (and most certainly will) have improper GA rejection.

The beginning of each fMRI volume can be specified either by triggers sent from the scanner (the safest option), or by using the sequence TR and automatically detecting the scanning episode (less reliable). When triggers (one per fMRI volume or slice) are available, then the correction will be based exclusively on these. If they are not available, the user can manually specify the beginning and end (in seconds from the beginning of the of EEG file) of the EEG episode to correct. This interval can also be automatically detected using a simple amplitude criteria: a stretch of EEG data with the mean (over a specified time window) absolute signal amplitude of a specific channel above some threshold (by default, one second, the first channel and 350 μV), then this is considered as an artefacted episode to be corrected. The TR provided is then used for the AAS correction. Finally, the sampling frequency of the original file (typically 5 kHz) being usually higher than necessary for further processing, the data are downsampled during the process (to 500 Hz by default).

3.2.2. Pulse Artefact Rejection. The PA is induced by the interaction between the heartbeats of the subject, which induce small movements, and the static field of the MR scanner. The PA is more difficult to remove than the GA because of its nonstationarity: it varies from heartbeat to heartbeat! Moreover, its amplitude (a few 10s of μV in our 3T scanner) and power spectrum (main frequency around 1-2 Hz and higher harmonics) render it difficult to disambiguate and filter out from genuine EEG signal.

One key step for the PA rejection is the detection of the heartbeats on one ECG channel. The method developed by Niazy and available in "the FMRIB plugin for EEGLAB" [8] is included in FA$\mathbb{S}$T. We found it very robust even on relatively noisy ECG channels. FA$\mathbb{S}$T currently provides five methods to reject the PA: "PCA" (from the FMRIB plugin), "Gaussian mean" (AAS from the FMRIB plugin) [12, 13], "constrained ICA (automatic)", "constrained ICA (manual)" [7] and "AAS and PCA combined" (based on the FMRIB plugin). We would advise users with 30 channels or more to choose a "constrained ICA" (cICA) method. cICA was shown to be more efficient than AAS and PCA at rejecting the PA and to better preserve the spectrum of the "true" EEG signal [7]. This is particularly important when analyzing the time course of spontaneous activity (such as in sleep studies). See Figure 4 for an example of correction using AAS, PCA, and cICA on a stretch of EEG data acquired on a sleeping subject. It is difficult to pick the best correction just by looking at the corrected signal.

With fewer channels (<30) or lots of movement activity, single-channel methods are better suited. "Gaussian mean" and PCA method do a good job in general. PCA is usually regarded as more efficient than "Gaussian mean" but Leclercq et al. [7] showed that for sleep EEG this is not the case: PCA tends to remove too much sleep activity, such as the slow waves which were picked up among the first few "optimal basis functions" and, therefore, removed for the recording. The "AAS and PCA combined" method is experimental and has not been rigorously tested. During the preparation of [7], we noticed that AAS was more efficient for the lower part of the data spectrum and PCA for the higher part. "AAS and PCA combined" thus uses a combination of AAS and PCA: AAS is applied to the low-pass filtered (4 Hz) signal and PCA on the high-pass filtered (4 Hz) signal, then the 2 corrected parts are recombined afterwards.

Users are in effect advised to test different correction methods to find out the most suitable one for their own data, depending on their final application and the usefulness of the validation criteria: here is a nonexhausitve list of proposed methods [11, 12, 14–20] which have been validated and applied on different types of data. Note that users are welcome to add other correction methods within FA$\mathbb{S}$T and hopefully share it with the other users.

3.3. Sleep Specific Tools. Specifically developed for the manual scoring of sleep M/EEG, these tools could be adapted to "scoring" other types of data. The aim is, therefore, to provide a user-friendly GUI to enter, for each time window of activity, a score via a set of predefined key presses. Scores provided by another mean, such as the automatic sleep scoring system ASEEGA [21, 22], can easily be added into the data structure, then displayed and used in FA$\mathbb{S}$T. Afterwards some summary statistics can also be calculated from the encoded score(s) and, in the case of sleep data, sleep slow waves can be automatically detected.

3.3.1. Sleep Scoring and Statistics. This tool is similar to the simple visualization tool (Section 3.1.1) but lets the user manually attribute a "score" to any (fixed) time window of signal (see Figure 5). The same file may be scored by different users and their scorings reviewed later on.

The keypad is simply used to assign a score to the current window. Each number corresponds to a specific stage, by default: "wake state" (0), "sleep stage 1" (1), "sleep stage 2" (2), "sleep stage 3" (3), "sleep stage 4" (4), "REM sleep" (5), and "movement time" (6). Each time a score is assigned to the current window, the display moves on to the next time

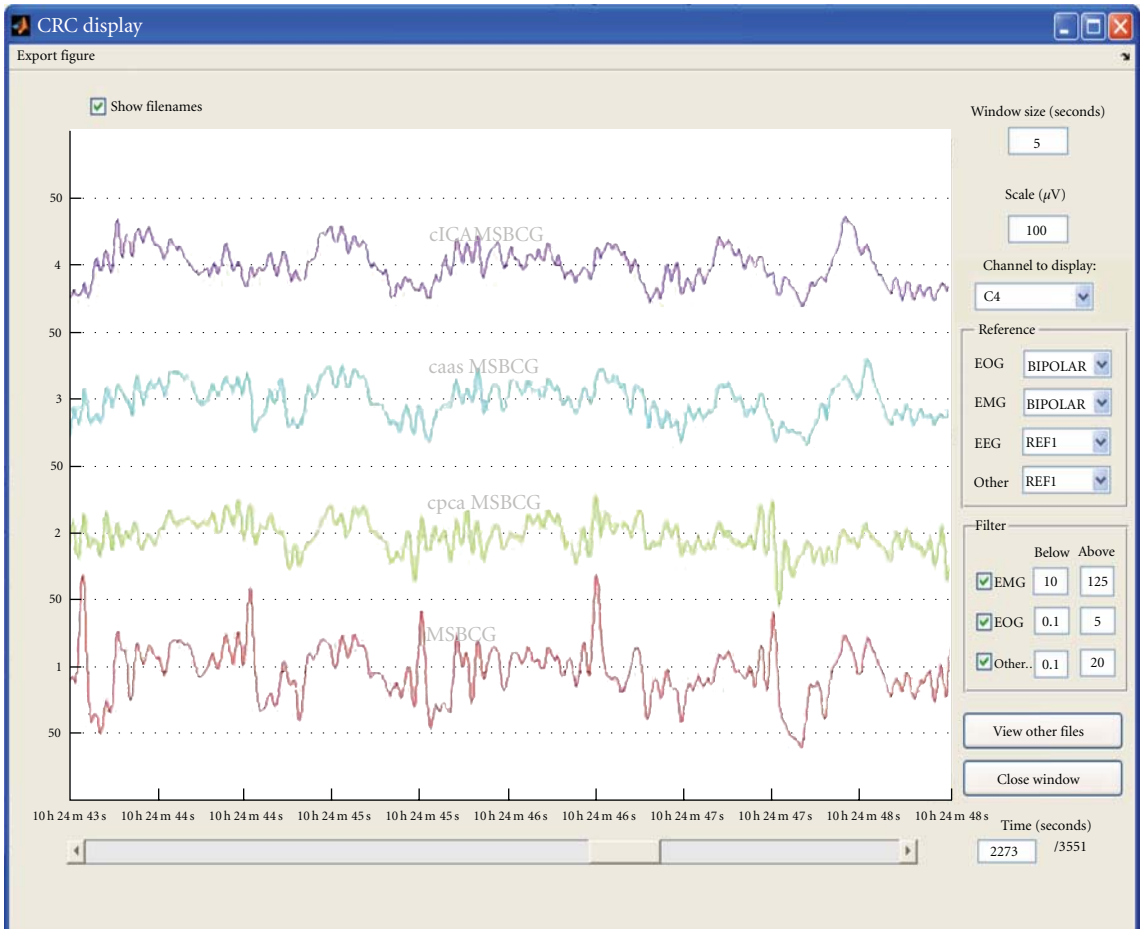


Figure 4: Comparison between the 3 correction methods, the signal displayed comes from electrode C4 and lasts 5 seconds: from top to bottom, EEG signal corrected by the cICA, AAS, and PCA method, and the original signal before correction.

window. An hypnogram is automatically constructed along the scoring.

Other types of markers can also be added at any time: "artefact and arousal" (which will mark the window as artefacted for power spectrum computation) and "event of interest" (e.g., spindles, epileptic spikes, etc.). The "FPL marker", that is, "closing door and light" marker (in French "fermer porte and lumière"), and the "OPL marker", that is, "opening door and light" marker (in French "ouvrir porte and lumière") specify the beginning and end of the "sleep recording" and are important to compute sleep statistics.

3.3.2. Spectral Power Calculation and Display. If the file was scored, then the spectral power is calculated as in Section 3.1.4, but sections scored as movement time or marked as artefacted are left out of the spectrogram calculation (power is set to zero). The hypnogram is also displayed (see Figure 3) alongside the spectrogram display. In the "frequency band" display mode, three scaling types are available: "absolute power", "relative power", that is, how much power is dissipated at time t in the considered band divided by the whole power at time t, and the "Mongrain view", which shows the power dissipated at time t in the selected frequency band divided by the mean power in deep sleep stage during the night. Moreover, the mean power spectrum of one specific channel during a specific sleep stage can also be computed and displayed (see Figure 6).

3.3.3. Slow Wave Detection. This tool, still in *BETA*-version, aims at automatically detecting slow waves (SWs) in sleep EEG recordings. It proceeds in successive steps: (1) extraction of the episode of interest, (2) bandpass filtering, (3) SW detection in four scalp "regions of interest" (ROIs), and (4) extraction of the SW trajectory on all electrodes.

The data episode to analyze can be the whole file, for example, if the data were previously chunked, or a part of the continuous file. This time window is either specified manually or relies on already defined sleep scores. To decrease the computational load, SWs detection is first performed on averaged signals from all the electrodes located in four scalp ROIs, by defaults: frontal, central left, central right, and parietal, that is, around the Fz, C3, C4, and Pz channels in the extended 10–20 system. SWs detection itself is performed in a spatiotemporal way following Massimini's criteria [23] which were adapted according to our observations on different data sets, see Figure 7(a).

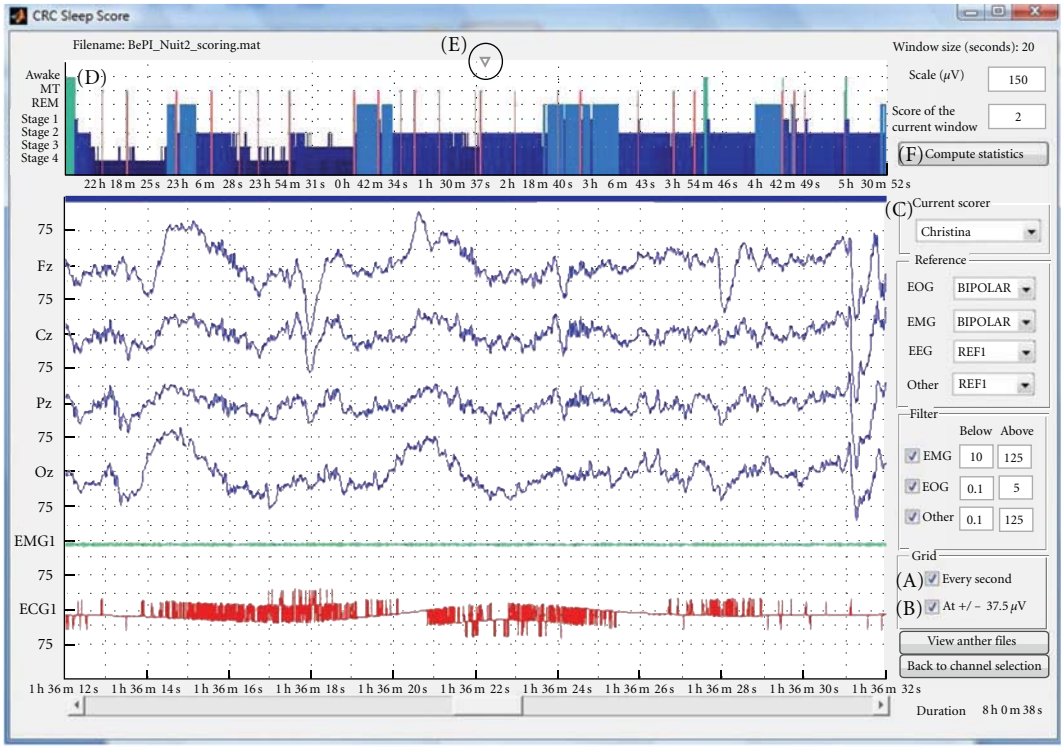


Figure 5: Sleep scoring GUI: (A) vertical grid toggle, (B) horizontal grid (± 37 μV toggle), (C) scorer selection, (D) hypnogram display, (E) current time position indicator, and (F) "compute sleep statistics" button.

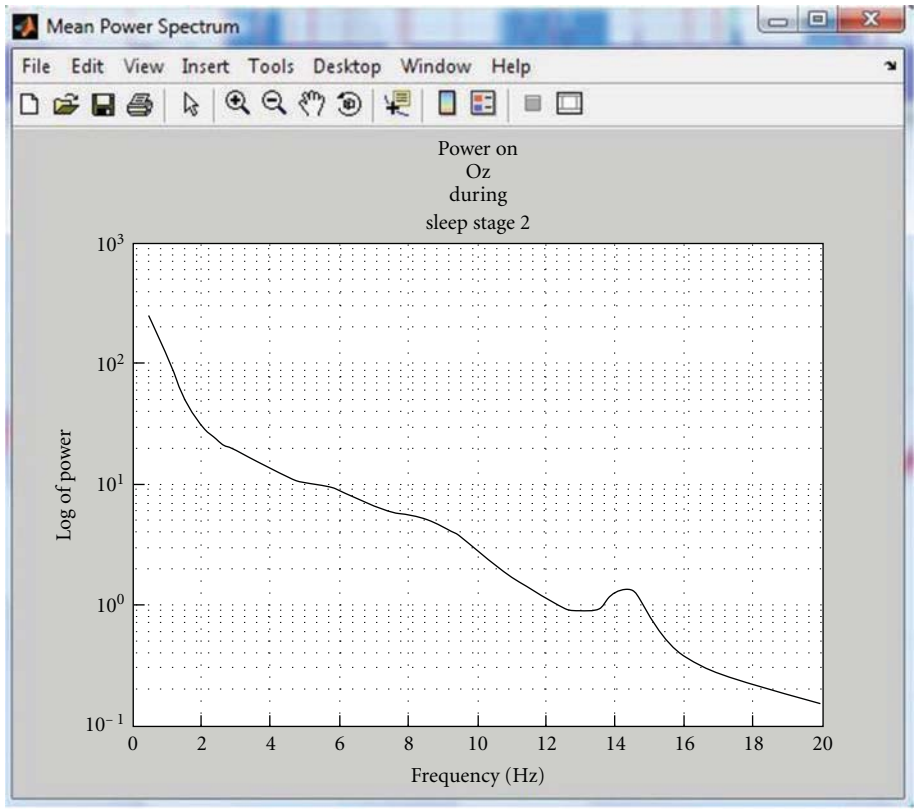


Figure 6: Mean power spectrum during a specific sleep stage (stage 2) for one channel (Oz here).

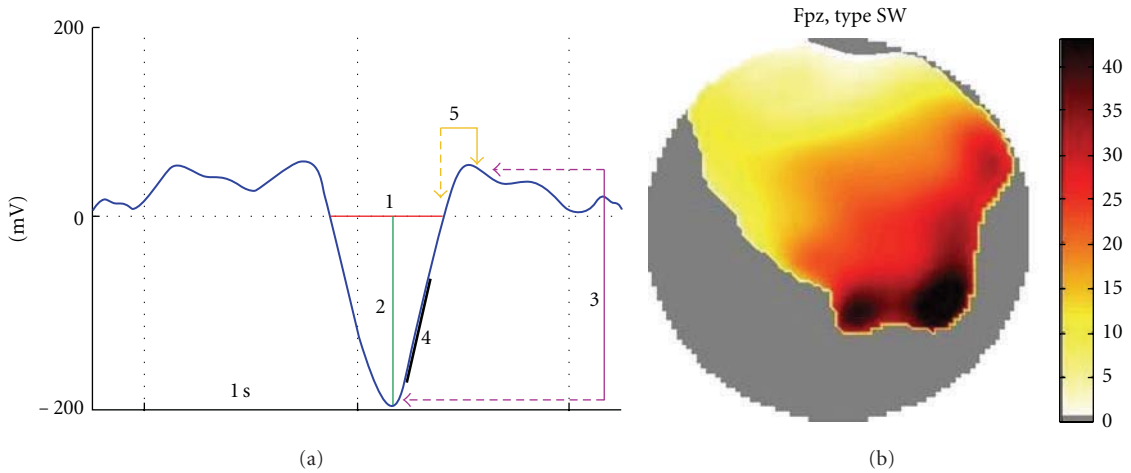


Figure 7: (a) SW detection criteria: (1) A negative zero crossing (downzero crossing) and a subsequent positive zero crossing (upzero crossing) separated by 0.25–1.25 sec, (2) A negative peak between the two zero crossings with voltage < – 80 μV, (3) A negative-to-positive peak-to-peak amplitude >140 μV, (4) A positive slope >90% of the maximum slope, and (5) A positive zero crossing and a subsequent positive peak separated by maximum 2 sec. Right, display of a SW trajectory as map of delays.

The SW trajectory over the scalp is based on the temporal occurrence of the negative peak at all the electrodes, where the SW was detected. The SW "time delay" of each electrode, where a SW is detected, is defined as the difference between the negative peak time at this electrode and the negative peak time at the first electrode detecting the SW (Figure 7(b)). The characteristics of each SW are saved in the data structure for further use, and their occurrences are saved as "events" for an easy epoching of the data.

4. Conclusions and Perspectives

As stated earlier, we started writing this toolbox to process sleep EEG-fMRI data and tackle three crucial issues typical of this kind of data: data manipulation, fMRI-artefact rejection, and manual sleep-scoring. As far as we know, FA$\mathbb{S}$T is currently the only free toolbox that can deal with these specific issues in an efficient and flexible way on a standard computer. Nevertheless, FA$\mathbb{S}$T can certainly be a useful tool for other researchers in the EEG/MEG community, as data reviewing, marking, and handling are very common tasks.

Since FA$\mathbb{S}$T is the result of ongoing research project, more features and improvements are expected in the future: we are currently working on adding more sleep tools, such as an automatic spindle detection, and a better integration with SPM8 batching system. With the batch, the exact parameters used for one operation on a data set can be saved and reapplied, with or without modification, on any other data set. We are also open to suggestions and personal additions to the code. FA$\mathbb{S}$T is available here: http://www.montefiore.ulg.ac.be/ phillips/FASST.html/.

Acknowledgments

This M/EEG toolbox is developed by researchers from the Cyclotron Research Centre, University of Liège, Belgium, with the financial support of the Fonds de la Recherche Scientifique-FNRS, the Queen Elizabeth's funding, and the University of Liège. Q. Noirhomme, C. Phillips, and P. Maquet are respectively Postdoctoral Researcher, Research Associate and Research Director at the FRS-FNRS. Y. Leclercq and J. Schrouff are both PhD students on a FRIA-FNRS grant. The authors would like to thank the CRC staff for their comments and suggestions during the development of FA$\mathbb{S}$T. A special thank goes to Virginie Sterpenich and Christina Schmidt whose suggestions were very helpful during the design and the very birth of FA$\mathbb{S}$T and also to Pierre Maquet who imagined FA$\mathbb{S}$T before they made it real. They are also grateful to all the users at the CRC (especially Luca Matarazzo) who helped them debug the code by boldly using the alpha version of the toolbox. Thanks also go to the "foreign labs" who provided some data sets to test the reading functions and the fMRI-artefact rejection tools.

References

[1] J. M. Kilner, J. Mattout, R. Henson, and K. J. Friston, "Hemodynamic correlates of EEG: a heuristic," *NeuroImage*, vol. 28, no. 1, pp. 280–286, 2005.

[2] M. Schabus, T. T. Dang-Vu, G. Albouy et al., "Hemodynamic cerebral correlates of sleep spindles during human non-rapid eye movement sleep," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 104, no. 32, pp. 13164–13169, 2007.

[3] T. T. Dang-Vu, M. Schabus, M. Desseilles et al., "Spontaneous neural activity during human slow wave sleep," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 105, no. 39, pp. 15160–15165, 2008.

[4] Members and collaborators of the Wellcome Trust Centre for Neuroimaging, "Statistical Parametric Mapping, SPM8," Wellcome Trust Centre for Neuroimaging, University College London, UK, 2008, http://www.fil.ion.ucl.ac.uk/spm.

[5] A. Delorme and S. Makeig, "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including

independent component analysis," *Journal of Neuroscience Methods*, vol. 134, no. 1, pp. 9–21, 2004.

[6] P. J. Allen, O. Josephs, and R. Turner, "A method for remov ing imaging artifact from continuous EEG recorded during functional MRI," *NeuroImage*, vol. 12, no. 2, pp. 230–239, 2000.

[7] Y. Leclercq, E. Balteau, T. Dang-Vu et al., "Rejection of pulse related artefact (PRA) from continuous electroencephalographic (EEG) time series recorded during functional magnetic resonance imaging (fMRI) using constraint independent component analysis (cICA)," *NeuroImage*, vol. 44, no. 3, pp. 679–691, 2009.

[8] R. K. Niazy, "FMRIB plug-in for EEGLAB," Centre for Functional MRI of the Brain (FMRIB), University of Oxford, UK, 2006, http://www.fmrib.ox.ac.uk/eeglab/fmribplugin/index.html.

[9] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005.

[10] R. Oostenveld, J.-M. Schoffelen, E. Maris, P. Fries, and O. Jensen, *Fieldtrip toolbox for EEG/MEG-analysis*, Donders Institute for Brain, Cognition and Behaviour, Radboud University Nijmegen, The Netherlands, 2009, http://www.ru.nl/neuroimaging/fieldtrip.

[11] S. Debener, K. J. Mullinger, R. K. Niazy, and R. W. Bowtell, "Properties of the ballistocardiogram artefact as revealed by EEG recordings at 1.5, 3 and 7 T static magnetic field strength," *International Journal of Psychophysiology*, vol. 67, no. 3, pp. 189–199, 2008.

[12] R. K. Niazy, C. F. Beckmann, G. D. Iannetti, J. M. Brady, and S. M. Smith, "Removal of FMRI environment artifacts from EEG data using optimal basis sets," *NeuroImage*, vol. 28, no. 3, pp. 720–737, 2005.

[13] G. D. Iannetti, R. K. Niazy, R. G. Wise et al., "Simultaneous recording of laser-evoked brain potentials and continuous, high-field functional magnetic resonance imaging in humans," *NeuroImage*, vol. 28, no. 3, pp. 708–719, 2005.

[14] T. Warbrick and A. P. Bagshaw, "Scanning strategies for simultaneous EEG-fMRI evoked potential studies at 3 T," *International Journal of Psychophysiology*, vol. 67, no. 3, pp. 169–177, 2008.

[15] D. Mantini, M. G. Perrucci, S. Cugini, A. Ferretti, G. L. Romani, and C. Del Gratta, "Complete artifact removal for EEG recorded during continuous fMRI using independent component analysis," *NeuroImage*, vol. 34, no. 2, pp. 598–607, 2007.

[16] J. L. Vincent, L. J. Larson-Prior, J. M. Zempel, and A. Z. Snyder, "Moving GLM ballistocardiogram artifact reduction for EEG acquired simultaneously with fMRI," *Clinical Neurophysiology*, vol. 118, no. 5, pp. 981–998, 2007.

[17] E. Briselli, G. Garreffa, L. Bianchi et al., "An independent component analysis-based approach on ballistocardiogram artifact removing," *Magnetic Resonance Imaging*, vol. 24, no. 4, pp. 393–400, 2006.

[18] W. Nakamura, K. Anami, T. Mori, O. Saitoh, A. Cichocki, and S. I. Amari, "Removal of ballistocardiogram artifacts from simultaneously recorded EEG and fMRI data using independent component analysis," *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 7, pp. 1294–1308, 2006.

[19] G. Srivastava, S. Crottaz-Herbette, K. M. Lau, G. H. Glover, and V. Menon, "ICA-based procedures for removing ballistocardiogram artifacts from EEG data acquired in the MRI scanner," *NeuroImage*, vol. 24, no. 1, pp. 50–60, 2005.

[20] K. H. Kim, H. W. Yoon, and H. W. Park, "Improved ballistocardiac artifact removal from the electroencephalogram recorded in fMRI," *Journal of Neuroscience Methods*, vol. 135, no. 1-2, pp. 193–203, 2004.

[21] C. Berthomier, X. Drouot, M. Herman-Stoïca et al., "Automatic analysis of single-channel sleep EEG: validation in healthy individuals," *Sleep*, vol. 30, no. 11, pp. 1587–1595, 2007.

[22] Physip SA, "ASEEGA," http://www.physip.fr/pub/en/research/aseega.html.

[23] M. Massimini, R. Huber, F. Ferrarelli, S. Hill, and G. Tononi, "The sleep slow oscillation as a traveling wave," *Journal of Neuroscience*, vol. 24, no. 31, pp. 6862–6870, 2004.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 982185, 11 pages
doi:10.1155/2011/982185

Research Article

Highly Automated Dipole EStimation (HADES)

C. Campi,[1] A. Pascarella,[2] A. Sorrentino,[1] and M. Piana[1]

[1] *Dipartimento di Matematica, Università di Genova, 16126 Genova, Italy*
[2] *Sezione di Fisiologia, Dipartimento di Neuroscienze, Università di Parma, 143121 Parma, Italy*

Correspondence should be addressed to C. Campi, campi@dima.unige.it

Received 29 July 2010; Revised 5 November 2010; Accepted 17 January 2011

Academic Editor: Sylvain Baillet

Automatic estimation of current dipoles from biomagnetic data is still a problematic task. This is due not only to the ill-posedness of the inverse problem but also to two intrinsic difficulties introduced by the dipolar model: the unknown number of sources and the nonlinear relationship between the source locations and the data. Recently, we have developed a new Bayesian approach, particle filtering, based on dynamical tracking of the dipole constellation. Contrary to many dipole-based methods, particle filtering does not assume stationarity of the source configuration: the number of dipoles and their positions are estimated and updated dynamically during the course of the MEG sequence. We have now developed a Matlab-based graphical user interface, which allows nonexpert users to do automatic dipole estimation from MEG data with particle filtering. In the present paper, we describe the main features of the software and show the analysis of both a synthetic data set and an experimental dataset.

1. Introduction

Traditional dipole fitting of MEG evoked fields is a time-consuming procedure providing subjective results and requiring expert users for reliable source estimation; however, it is still largely used even for evaluating MEG inverse methods based on the distributed current assumption [1, 2] and, in any case, proved to be notably effective in the reconstruction of focal sources [3]. Estimating current dipoles from MEG data is in fact a hard task, as it involves solving several interacting problems such as model order selection (for determining the number of sources), nonlinear optimization (for estimating the source locations), and linear least-squares fitting (for calculating the dipole strengths). Most automatic algorithms for dipole estimation presented so far, and in fact even traditional dipole fitting, work under a couple of important approximations: (1) the number of dipoles is assumed to be fixed during the whole sequence, presence or absence of a given source being coded in the strength of the source itself; (2) the source locations are fixed in time. The second assumption is justified by physiological arguments, because a neural population hardly moves within the head. Also the first assumption appears to be reasonable; however, methods based on these assumptions can hardly discriminate nearby sources, even if they are not overlapping in time, because two dipoles placed at close distance will interact and produce spurious activity. Furthermore, in some cases, particularly when the number of sources is estimated from the data covariance matrix exploiting algebraic results [4], temporal correlation can prevent automatic algorithms from correctly recovering the neural sources.

In [5], we have described a source estimation method exploiting Bayesian filtering and random finite sets and based on a completely dynamical model, rejecting the assumptions (1) and (2) previously mentioned: the number of sources can change during the sequence, as well as the dipole locations. The number of active dipoles and their locations are estimated dynamically and updated at each time sample from the data. The method works by approximating with a particle filter, that is, a sequential Monte Carlo algorithm, the posterior densities involved in the Bayesian filter. In a couple of publications, we have discussed possible advantages and limitations of particle filtering for MEG, showing direct [6] and indirect [5] comparisons with other available methods.

In the present paper, we describe the use of the graphical user interface (GUI) we have developed for the particle filter, HADES (highly automatic dipole estimation). HADES is an open-source, freely downloadable, Matlab-based software. The purpose of the GUI is at least twofold: on one hand, we aim at sharing methods and results with other researchers

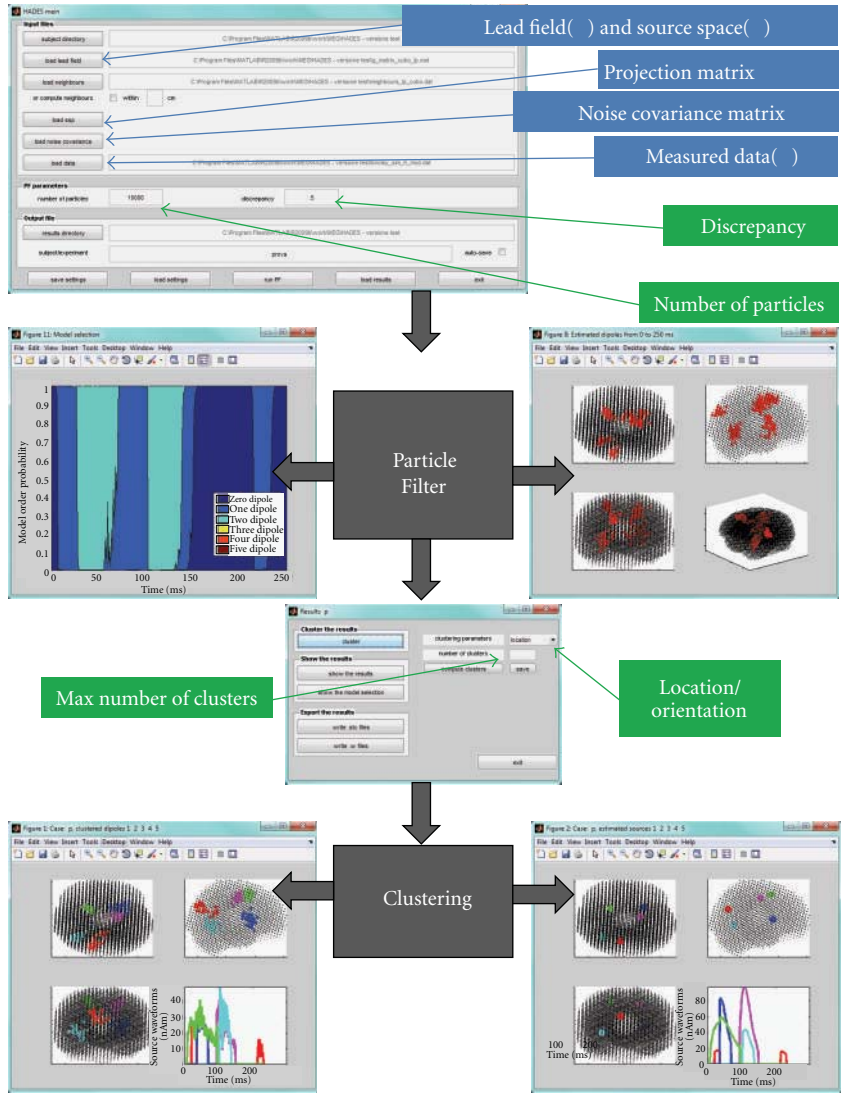


Figure 1: The input-output scheme of HADES: on top, the main window where the user can load input data (blue) and set the parameters (green); an asterisk indicates mandatory input data. The particle filter algorithm is presented as a black box giving two outputs (the model selection function and the estimated dipoles); the clustering algorithm assigns individual dipoles to clusters and computes the average location and the waveform of each cluster.

in the field, who may have the chance to investigate by themselves the potential and limits of particle filtering; on the other hand, we aim at reaching a larger audience of neuroscientists who may be less curious about the methodological aspects but more interested in the possible applications.

The paper is organized as follows. In Section 1, a nontechnical description of the methodological issues is presented. In Section 2, we provide details on the software, including supported data types, license details, and computational aspects. In Section 3, we follow step-by step the analysis of both a synthetic data set and an experimental dataset, so as to introduce the reader to the practical use of the interface. In Section 4, we briefly summarize the main features of the presented software.

2. Methods

The present section describes the computational algorithm at the basis of HADES and the way it has been implemented in the software. It contains three subsections: the first one describes the models adopted and the input data; the second one describes the particle filter and the run-time parameters necessary for the filter to run; the third one describes the estimation procedure and the output provided by HADES.

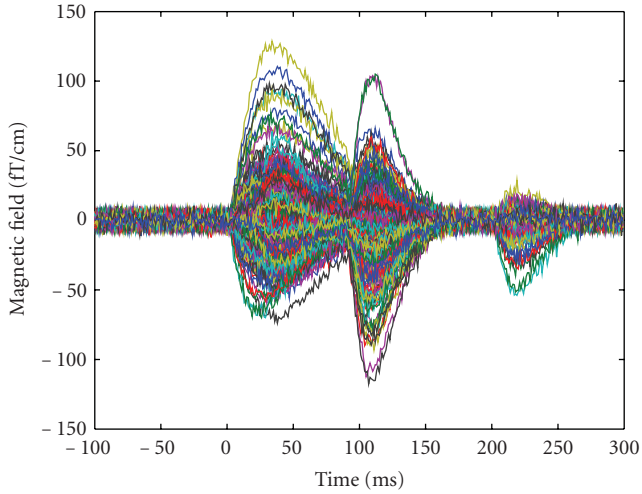


Figure 2: Synthetic data produced by the six sources described in Table 1 and Figure 3; only the signals from the gradiometers are shown.

2.1. Model Assumptions and Input Data. HADES is based on a dynamical dipolar model of neural activations: at each time point, each active area is represented as a single current dipole. There is no prior assumption on the number of active sources, and there is no limit on the total number of neural sources; however, for computational reasons, we impose an upper bound to the number of simultaneous active dipoles.

HADES is based on a discretized source space: dipoles can take only a finite set of predefined possible locations. The main advantage of this approximation is that lead fields can thus be used to save computational time. Furthermore, the source space can be either the whole brain volume or else the cortical surface when available; to further increase localization accuracy, also an orientation constraint can be optionally used (although cortical constraints should be managed carefully, since there are neurophysiological situations where, using dipole fitting, they may lead to biased or wrong results).

All the source parameters are assumed to be dynamical parameters. The number of sources is a dynamical variable, to be estimated from the data. Sources are also allowed to move during time, that is, to jump between neighboring points of the source space.

Noise is assumed to have a Gaussian distribution. An estimate of the noise spatial covariance matrix can be either loaded or calculated; using such estimate corresponds to a prewhitening of the data. Alternatively, one can assume that noise is white Gaussian and calculate an estimate of the noise power.

The input data needed to run HADES are therefore the source space and the corresponding lead field. The neighboring matrix, listing all the neighbours within a user-selected radius, is calculated by HADES. Optional inputs are the noise covariance matrix and a signal space projection matrix. See Figure 1 for a schematic representation.

2.2. Particle Filter and Run-Time Parameters. The core of HADES is a random finite sets (RFS) particle filter. Random

Table 1: Parameters of the six sources used to simulate the data: source location, peak latency, and measured signal at the peak. Colors refer to Figure 3.

Source n	x (cm)	y (cm)	z (cm)	t (ms)	fT/cm
1 (red)	– 1.37	– 5.43	7.34	20	51
2 (blue)	3.74	4.54	5.66	40	57
3 (green)	– 2.04	3.73	9.56	40	130
4 (magenta)	2.96	2.11	9.42	110	100
5 (cyan)	– 3.43	– 2.71	4.07	110	110
6 (yellow)	– 1.37	– 5.43	7.34	220	51

finite sets are a mathematical tool for dealing with an unknown and varying number of objects [7]. Particle filtering [8] refers to an algorithm which tries a large number of dipole configurations, also called *particles*, choosing these configurations based on probabilistic criteria. The algorithm is sequential: it begins by analyzing the data measured at the first time point, $t = 1$, and proceeds time sample per time sample. At each time sample t, assume that a set of N_p dipole configurations is available; then the algorithm performs the following operations:

(1) assign a weight to each dipole configuration, based on the difference between the measured data and the exact field produced by the dipole configuration,

(2) use the cloud of weighted dipole configurations to calculate estimates of the number of sources and their parameters,

(3) discard particles with low weights and multiply particles with high weights, in order to maintain only the most likely dipole configurations while preserving the total number of particles N_p,

(4) let each dipole configuration evolve randomly, thus producing the set of dipole configurations at time $t+1$ needed at step 1, and start again from step 1.

According to the RFS framework, the number of dipoles in each particle may vary from zero to a maximum; dipole configurations may undergo loss or birth of dipoles during the temporal evolution at the fourth step.

The number N_p of particles is the first parameter to set: using a large N_p guarantees in principle better results; the computational time is linearly increasing with this number, hence a good balance between stability and computational time has to be sought.

In the weighting procedure at step 1, the prior assumptions on the noise statistics play an important role, because the expected difference between the measured data and the exact field should be of the order of the noise. However, for several reasons the noise estimate can be unsatisfactory in many situations. In this case, one may want to have a weaker/stronger fit with the data, with respect to that provided by straightforward noise estimate. Therefore, we introduced the *discrepancy* parameter as a multiplicative factor for the noise estimate. Setting a small value (<1) for the discrepancy means that a stronger fit is required; the

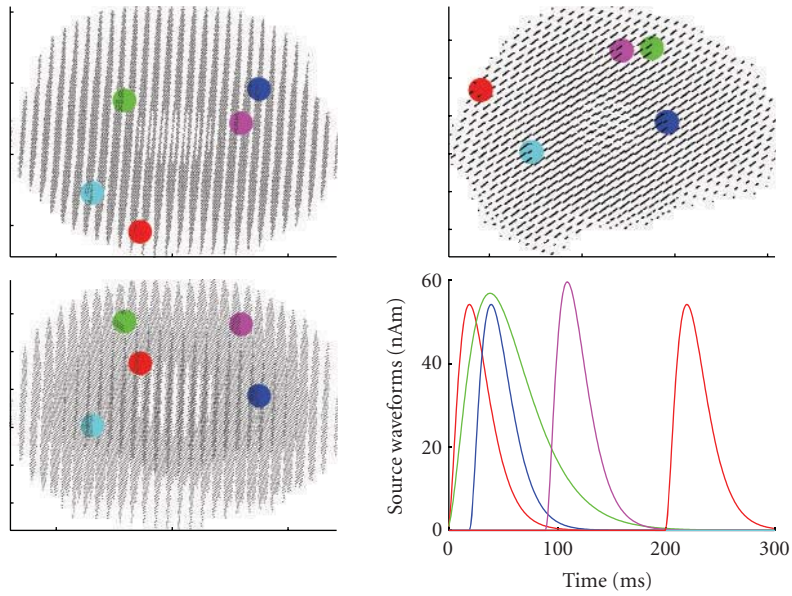


Figure 3: Location and dynamics of the 6 sources: the waveform of the cyan source is not visible as it is overridden by the time-correlated magenta source; the location of the yellow source is not visible as it is the same as the red source.

algorithm will then try to reproduce finer details in the data, possibly using a larger number of sources and possibly exhibiting a lower degree of stability and reliability. On the contrary, setting a large value means that a weaker fit is required, with the opposite consequence of ending up with a lower number of stable sources.

2.3. Source Estimates and Output. The estimation procedure in HADES goes through three main steps. The first two steps are performed at every sampled time point of an MEG sequence (step 2 in the previous subsection), and produce time-varying estimates of the dipole parameters: first, the algorithm obtains an estimate $\hat{N}_t$ of the number of dipoles and, then, calculates estimates of the actual parameters (location and dipole moment) for $\hat{N}_t$ dipoles. These dynamical estimates, however, do not identify individual neural sources in time, because there is no straightforward relationship between dipoles estimated and different time points. Given the collection of all dipoles estimated at all time points, a third step is then applied, which binds together dipoles estimated at different time points but possibly representing the same neural source. This clustering can be performed in two different configurations: either dipoles are grouped based only on their location, or else dipoles are grouped based both on location and orientation. The final number of clusters is estimated automatically with a recursive procedure, which starts from the user-defined maximum number of clusters and decreases this number until all the estimated clusters are significantly different. Once dipoles have been assigned to different clusters, likely corresponding to different neural sources, it makes sense to compute the average location of all dipoles belonging to each cluster; this average location can be considered as an estimate of the neural source location, and the corresponding source waveform can also be calculated.

The output of HADES consists in the dynamical estimates of the number of sources and of the source parameters, plus a global picture obtained from the clustering. Referring to Figure 4 as a typical result of a data analysis performed with HADES, the user can view the following.

(1) The dynamical model order estimate (Figure 4(a)), that is, the posterior probability that the data have been produced by 1, 2, 3, ... dipoles as a function of time; the cumulative distribution for the number of sources is visualized as an area, with different colors representing the probabilities of different models.

(2) The dynamical estimates of the source locations (Figure 4(b)); while in this figure all the dipoles estimated in the whole sequence are superimposed, the user can in fact choose to visualize only the dipoles estimated in a selected time window.

(3) The clustered dipole location estimates, with the corresponding amplitude waveforms (Figure 4(c)); since these waveforms are calculated for dynamical source locations, they exhibit a certain level of discontinuity in correspondence of jumps of the source location.

(4) The average source location of each cluster (Figure 4(d)), with the corresponding amplitude waveform which is now continuous.

While the localization panels show the three standard views of the brain, figures contain in fact 3-dimensional information and the user can rotate the view.

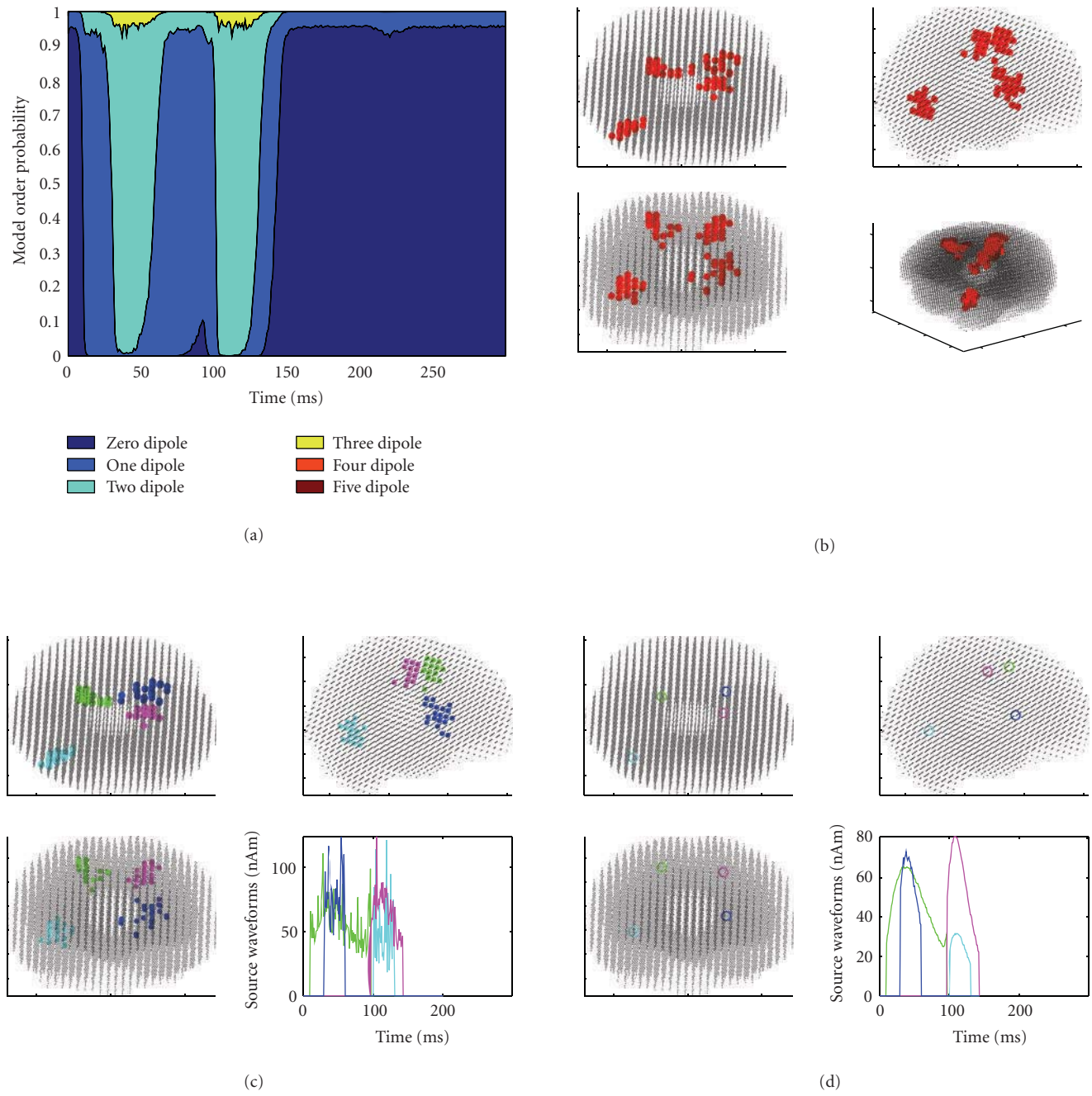


Figure 4: Results obtained with 10,000 particles and discrepancy 1. (a) Dynamical model selection function. (b) Superposition of all estimated dipoles at all time points. (c) Clustered dipoles and corresponding waveforms. (d) Average dipoles of the clusters in (c). Estimated dipoles show an expected spread around the true sources (cf Figure 3). From (b), it is evident that Sources 1 and 6, that is, the ones producing the weakest field, are not recovered. The model selection function indicates neural activity beginning at 10 ms (when the maximum probability switches from the zero-dipole model to the one-dipole model) and lasting until about 145 ms; in two time windows (30–55 ms and 105–130 ms), a two-dipole model is selected.

3. Software Details

HADES is a Matlab-based graphical user interface, which needs Matlab to run. It has been written and tested under Matlab version 7.9.0, hence full compatibility is not guaranteed under earlier versions.

Input data can be provided in standard Matlab *.mat* format and in plain *ASCII* format; the Neuromag *.fif* format is supported through the set of functions contained in the MNE [9] Matlab toolbox http://www.nmr.mgh.harvard.edu/martinos/userInfo/data/sofMNE. More details on

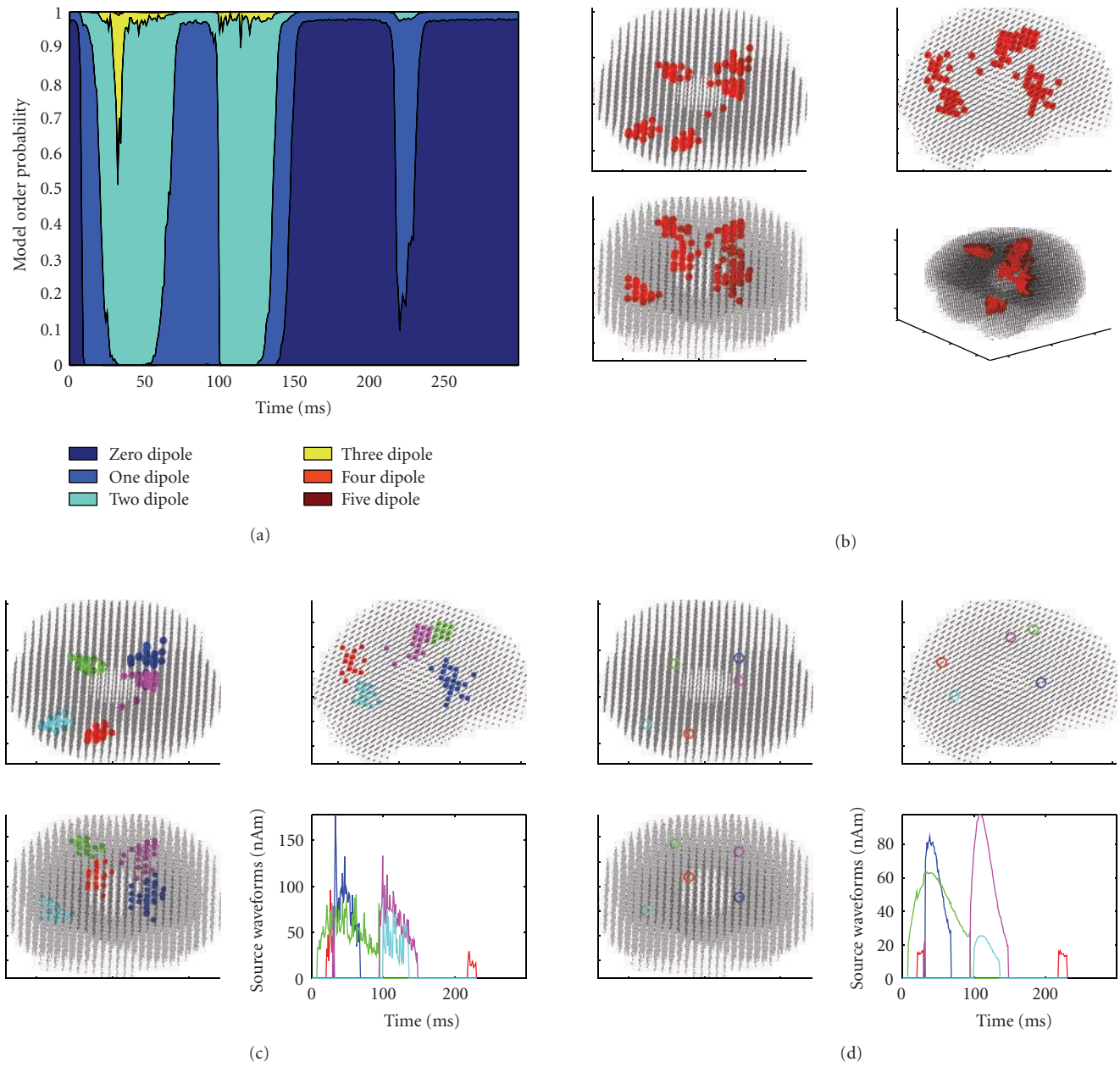


FIGURE 5: Results obtained with 10,000 particles and discrepancy 0.7. The model selection indicates now a one-dipole model in a short time window around 220 ms, corresponding to Source 6. In fact, both Sources 1 and 6 are now recovered correctly; the clustering procedure binds them in a single source, because they are exactly in the same location. The model selection also indicates that the two-dipole model is now selected for larger time windows with respect to the previous case with unit discrepancy; moreover, around 30 ms, the three-dipole model appears to have a nonnegligible probability, even though it does not exceed the 50%.

the format of input data can be found in the HADES manual, available at http://hades.dima.unige.it/.

Results can be exported in different formats, for visualization in other toolboxes. At the moment, HADES features the following export options:

(i) a *.stc* file which contains the sequence of estimated dipoles in time, and can be visualized as a movie in MNE; furthermore, the very first time sample of the exported file contains the superposition of all the estimated dipoles, to get the overall picture of the estimated neural activity;

(ii) a *.mat* file which contains the sequence of estimated dipoles in time and can be visualized as a movie in BrainStorm (http://neuroimage.usc.edu/brainstorm) again, the very first time sample of the exported file contains the superposition of all the estimated dipoles, to get the overall picture of the estimated neural activity;

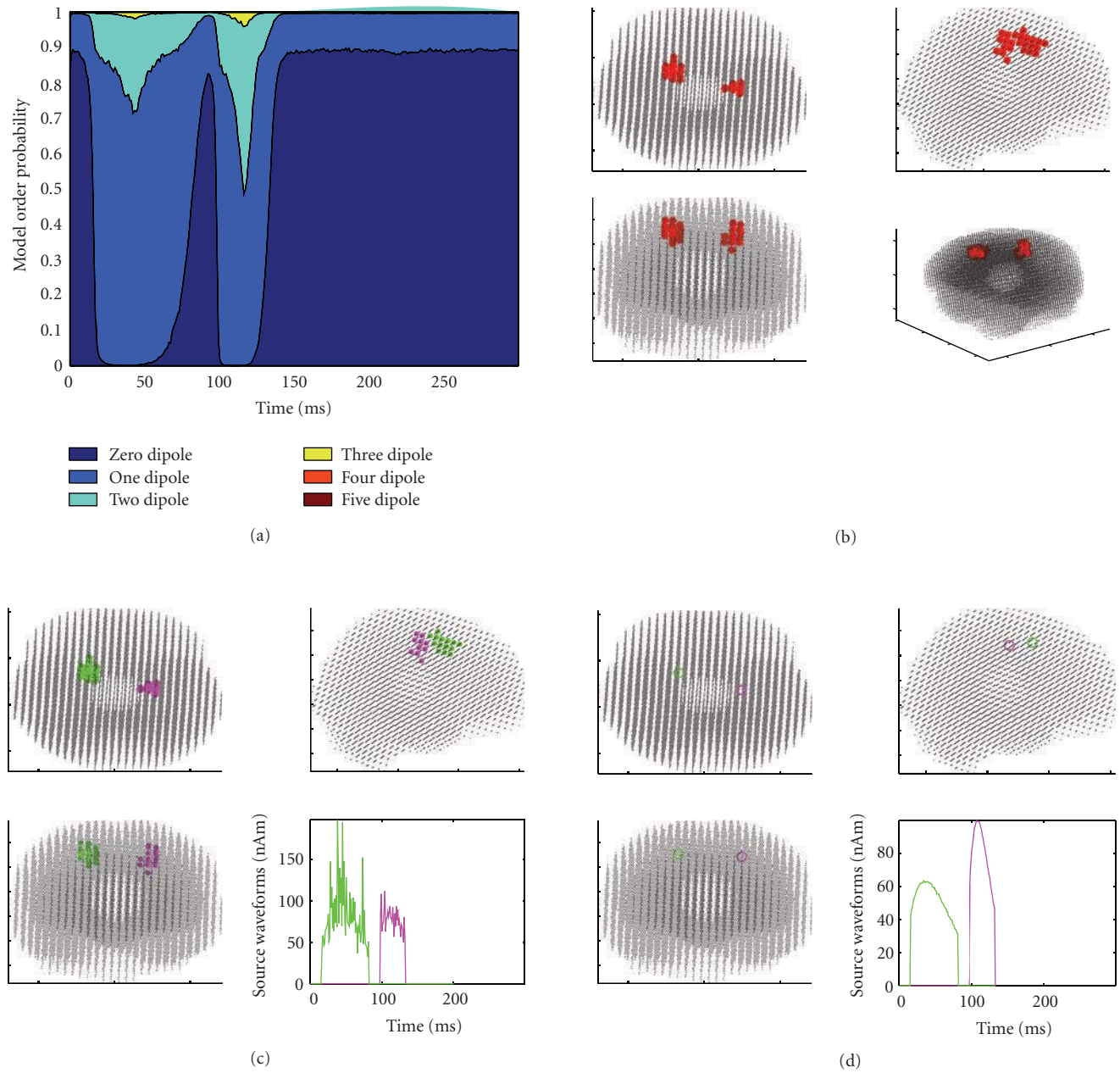


FIGURE 6: Results obtained with 10,000 particles and discrepancy 2. With this large value for the discrepancy parameter, the model selection in (b) exhibits lower probability for larger models, and the two-dipole model has nonnegligible posterior probability but is never the mode of the distribution. The set of estimated dipoles is now smaller and contains only two activations, corresponding to Source 3 and 4, that is, the two dipoles producing the strongest field.

(iii) a *.w* file which contains the location of all dipoles estimated at all time points and can be visualized in FreeSurfer (http://surfer.nmr.mgh.harvard.edu/) [10, 11].

HADES is not bound to a specific hardware for MEG: all the hardware-dependent components are in fact contained in the input data (lead field, source space, and measurements). In principle, HADES may be applied to EEG data as well; experimental validation with electroencephalographic measurements is in progress.

The computational cost of the algorithm increases linearly with (i) the number of analyzed time samples and (ii) the number of particles. For running with 10,000 particles on a standard PC (CPU Intel Core2 Quad 2.83 GHz, RAM 4 GB) the algorithm takes on average 0.8 seconds per time sample.

HADES (http://hades.dima.unige.it/) is a free but copyrighted software, distributed under the terms of the GNU

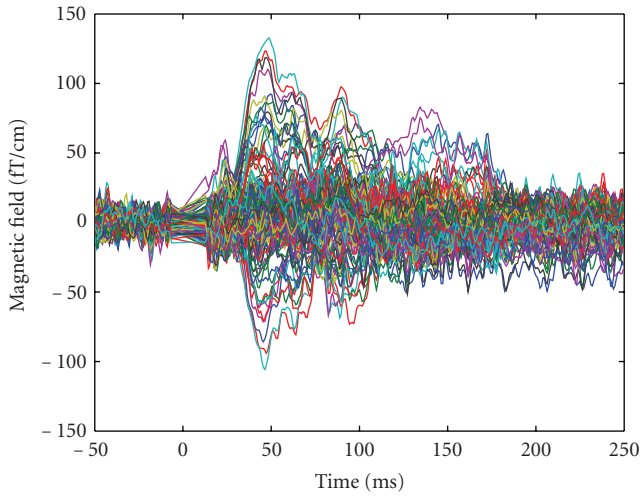


Figure 7: Averaged magnetic field for the stimulation of the left thumb.

General Public Licence as published by the Free Software Foundation (either version 2 or at your option any later version).

4. Results

In this section we present two examples of source modeling performed using HADES. First, we use synthetic data so that the ground truth is known; the sample data analyzed here are contained in the HADES package for further analysis and testing. Then, we analyze an experimental data set corresponding to stimulation of left and right thumb.

4.1. Synthetic Data. Data (see Figure 2) are produced by six sources: Table 1 summarizes locations and peak latencies of the sources, while Figure 3 shows both source locations and dynamics. Sources 2 and 3 have the same latency, but a different duration; Sources 4 and 5 have exactly the same waveform, that is, they are time correlated; Sources 1 and 6 are in the same location. The source points do not belong to the source space which is used by the inverse algorithm. MEG sensors correspond to the Neuromag Vectorview system which features 102 locations and 3 channels per location, one magnetometer and 2 planar gradiometers, for 306 channels. Here, we employ only the 204 planar gradiometers. White Gaussian noise is added: the noise standard deviation is $3\,fT/cm$; the SNR at the peak of the strongest source, calculated as $10\log_{10}|D|^2/|N|^2$, where D is the data matrix, N is the noise matrix, and $|\cdot|$ is the Frobenius norm, is about 10 dB. The superposition of all signals is shown in Figure 2.

We first load the source space and the lead field from the popup window. Then, we load the measurements: we set the starting time point (– 100 ms), the sampling frequency (1,000 Hz), and the length of the prestimulus interval (from – 100 to 0 ms) for estimation of the noise variance. The source space is formed by 13026 points with a regular spacing of 0.5 cm in the brain volume, and no cortical constraints are used.

4.1.1. Single Run. We set the number of particles to 10,000 and the discrepancy parameter to 1 and run the particle filter.

The results are shown in Figure 4. Two of the 6 sources producing the data are missing: in fact, they are the two Sources 1 and 6 in the same location, which are also the ones producing the smallest signal at the sensor level. The initial number of clusters was set to 4, due to both visual inspection of reconstructed dipoles (Figure 4(b)) and evidence from the model selection (Figure 4(a)), which indicates a two-dipole model in two separate temporal windows.

Considering all the reconstructed dipoles at all time points, the average distance between the dipoles and the corresponding sources is 1.1 cm, with a standard deviation of 0.8 cm, the maximum distance is 3.3 cm, and the minimum distance is 0.24 cm. Despite this large maximum error, the mean dipoles of the clusters (Figure 4(d)) appear to be good approximations of the true sources (cf Figure 3), featuring distances of 0.3 cm, 0.4 cm, 0.9 cm, and 1.35 cm from the true sources. This is explained as the estimated dipoles being quite symmetrically distributed around the true sources.

4.1.2. Tuning the Parameters. As described in the previous section, tuning the discrepancy parameter corresponds to requiring higher/lower fit with the data. We run again the particle filter with 10,000 particles, first setting the discrepancy to 0.7 (higher fit required) and then to 2 (lower fit). The results are shown in Figures 5 and 6, respectively. With the lower discrepancy, the algorithm recovers also the two weaker sources, Source 1 and 6. The figure has been obtained by clustering the dipoles in 5 groups. With the higher discrepancy, the algorithm looses track of Sources 1, 2, 5, and 6.

Average distances between reconstructed dipoles and true sources are in the same range as for the unitary discrepancy.

4.2. Experimental Data. MEG data were provided courtesy of Dr. Sabine Meunier (La Salpetriere Hospital, Paris), as made available for download on BrainStorm's website. The data were recorded on a CTF machine (151 axial gradiometers) at La Salpetriere Hospital, Paris. The protocol comprised shuffled electrical stimulation of the fingers from both hands; the analyzed data are averaged responses (400 trials) for the stimulation of the right thumb (R) and of the left thumb (L) (see Figure 7). The lead field matrix was exported using the BrainStorm software, as well as the source space; the source space consists of 15,010 source points distributed along the cortical surface. A distance of 1 centimeter was selected for calculation of the neighboring matrix. Both data sets were analyzed using 10,000 particles and the discrepancy parameter set to 1; the orientation constraint was not used, although available. Results for the left and right thumb stimulation experiment are described in Figures 8 and 9 respectively. With the left data, reasonable source localization is obtained with the first run, with the standard discrepancy value. With the right data, on the contrary, the standard parameter value provided reasonable localization in correspondence with the peak of activation, plus some other dipoles at later time

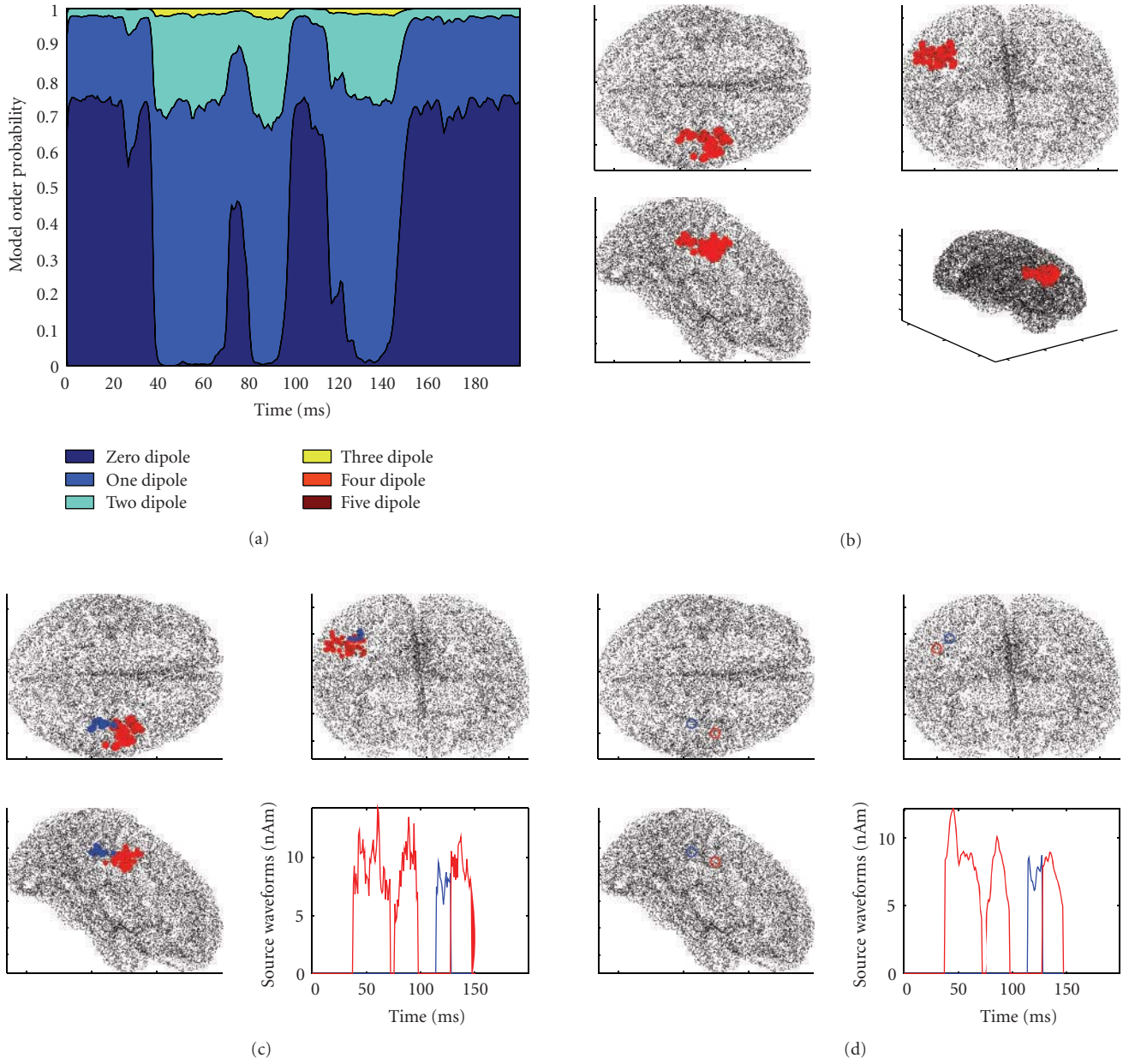


Figure 8: Left thumb stimulation; results obtained with 10,000 particles and discrepancy 1. The model selection in (a) indicates activity in a first time window beginning around 40 milliseconds after the stimulus and lasting until 100 ms and in a second time window between 115 and 140 ms. All estimated dipoles are in the right hemisphere, located around the somatosensory cortex. Clustering does not seem to add significant information to the estimated sources: the blue cluster is smaller and lasts few milliseconds.

points scattered in apparently less likely locations. Cleaner reconstructions can be obtained increasing the discrepancy parameter (see Figures 9(c) and 9(d)).

5. Discussion and Conclusions

HADES is a Matlab-based, freely downloadable software for dynamical estimation of current dipoles from MEG data. It is distributed under the GPL and has a simple graphical user interface, which allows nonexpert users to do dipole modeling automatically.

The particle filter HADES is based on [5] and tracks in time the posterior density for the dipole constellation; statistical estimators are used to provide dynamical estimates of the number of sources and of the source parameters. The main innovative feature of HADES, with respect to the available dipole estimation methods, is related to the underlying dynamical model: dipoles are not constrained to have a fixed position nor to be active for the whole time sequence. Instead, the number of sources and all source parameters are estimated at each sampled time point; in particular, HADES provides a dynamical model selection

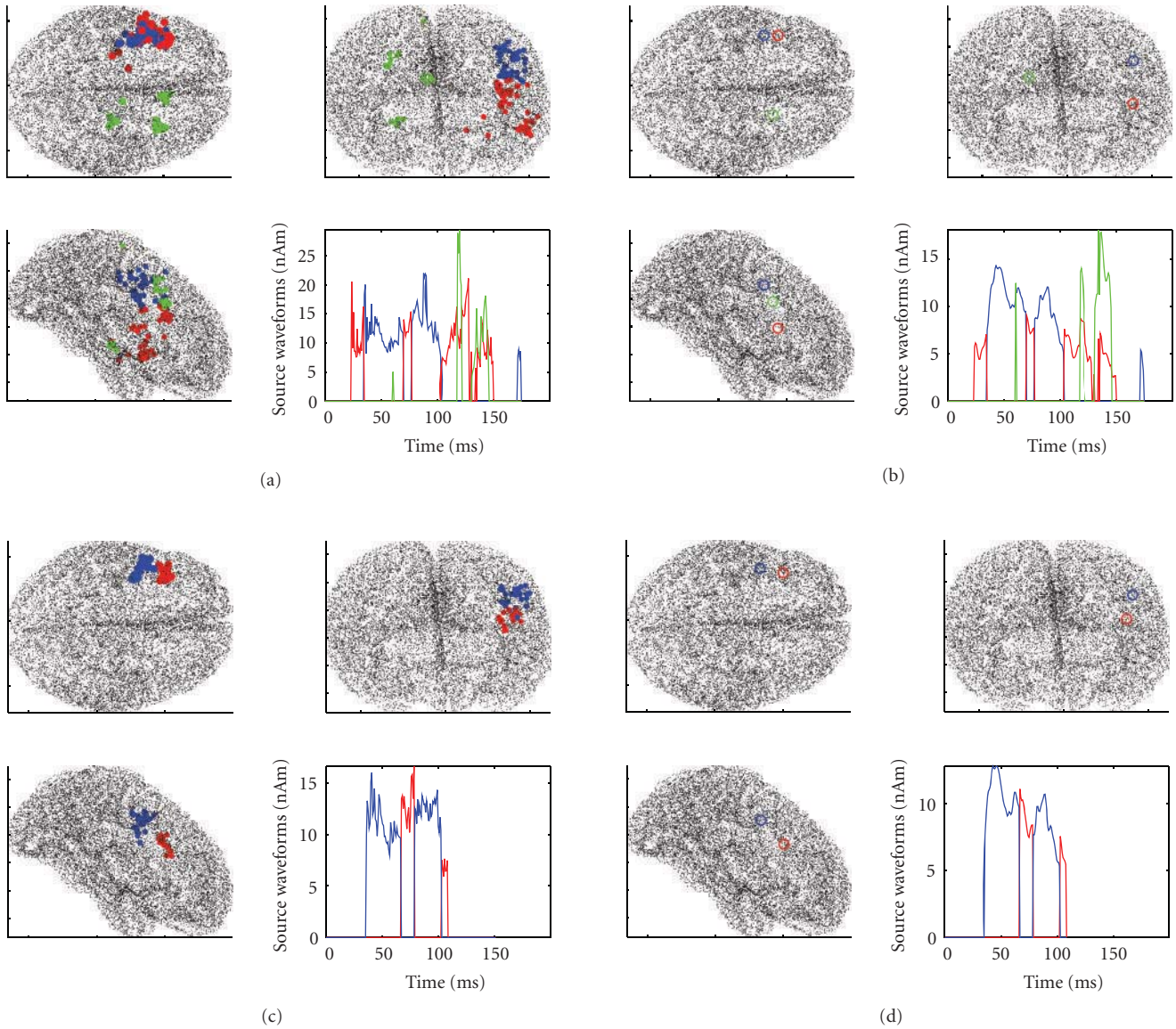


Figure 9: Right thumb stimulation. First row: results obtained with 10,000 particles and discrepancy 1. (a) shows the clustered reconstructions: most dipoles are located in the left hemisphere in proximity of the somatosensory cortex; however some reconstructions fall in the right hemisphere and are rather unstable. Arguing from such instability that the noise estimate was slightly too tight, we increased the discrepancy parameter to 1.5 to get the cleaner results of the second row.

function, which indicates at each time point the probability that the data have been produced by $1, 2, \ldots, N$ dipoles. To obtain stable source estimates and continuous source waveforms, clustering procedures are implemented which bind dipoles representing the same source at different time points. Due to the generality of the underlying model, HADES can recover correlated sources and discriminate nearby dipoles with different orientations. On the other hand, the particle filter is more computationally demanding with respect to other estimation methods, and semianalytic solutions [12] to Bayesian filtering feature better statistical properties but higher computational requirements.

The performances of HADES were illustrated with a set of synthetic data produced by a complicated source configuration, as well as with a set of experimental data. Synthetic data were particularly useful to illustrate how the discrepancy parameter plays an important role in selecting larger/smaller number of sources. The same conclusion can be drawn also from the experimental data set, with the further consideration that in real situations the peculiar structure of neural noise is more likely to produce spurious activity.

The visualization of the results is limited to a very simple 3d plot of the source space with the estimated sources superimposed. However, the results can be exported for visualization in other toolboxes where superimposition onto high resolution MRI slices or inflated surfaces are possible. Export options to MNE, Freesurfer, and BrainStorm are

supported at the moment. Forthcoming releases of the toolbox may feature better built-in visualization tools.

HADES has been thought as a highly specialized toolbox for dipole estimation. As such, it does not mean to replace other toolboxes but possibly to integrate with them to provide a different perspective on a data set. For this reason, no tools for multisubject analysis are under development at the moment, although HADES-reconstructed dipoles are saved in the.mat file of the results and can be utilized for statistical analysis by means of external toolboxes.

More in general, the toolbox is at its very first stage, and the development of the method will likely add more features to the toolbox. Possible future methodological developments include

(i) investigating strategies to remove spurious activations produced by neural noise,

(ii) providing an estimate of the localization accuracy for each source, based on the spread of the underlying posterior density,

(iii) modeling the neural sources as nondipolar currents, such as multipolar sources or cortical patches.

All future developments will head towards automation and reliability of source estimation from MEG/EEG data.

Acknowledgments

Professor Matti Hämäläinen is kindly acknowledged for his help in the use of the MNE software. The authors have been partly supported by a grant from the Fondazione della Cassa di Risparmio di Verona.

References

[1] L. Stenbacka, S. Vanni, K. Uutela, and R. Hari, "Comparison of minimum current estimate and dipole modeling in the analysis of simulated activity in the human visual cortices," *NeuroImage*, vol. 16, no. 4, pp. 936–943, 2002.

[2] M. Liljestrom, J. Kujala, O. Jensen, and R. Salmelin, "Neuromagnetic localization of rythimc activity in the human brain: a comparison of three methods," *NeuroImage*, vol. 25, pp. 734–745, 2005.

[3] E. F. Chang, S. S. Nagarajan, M. Mantle, N. M. Barbaro, and H. E. Kirsch, "Magnetic source imaging for the surgical evaluation of electroencephalography-confirmed secondary bilateral synchrony in intractable epilepsy: clinical article," *Journal of Neurosurgery*, vol. 111, no. 6, pp. 1248–1256, 2009.

[4] J. C. Mosher, P. S. Lewis, and R. M. Leahy, "Multiple dipole modeling and localization from spatio-temporal MEG data," *IEEE Transactions on Biomedical Engineering*, vol. 39, no. 6, pp. 541–557, 1992.

[5] A. Sorrentino, L. Parkkonen, A. Pascarella, C. Campi, and M. Piana, "Dynamical MEG source modeling with multi-target bayesian filtering," *Human Brain Mapping*, vol. 30, no. 6, pp. 1911–1921, 2009.

[6] A. Pascarella, A. Sorrentino, C. Campi, and M. Piana, "Particle filtering, beamforming and multiple signal classification for the analysis of magnetoencephalography time series: a comparison of algorithms," *Inverse Problems and Imaging*, vol. 4, no. 1, pp. 169–170, 2010.

[7] I. Molchanov, *Theory of Random Sets*, Springer, Berlin, Germany, 2005.

[8] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.

[9] M. S. Hämäläinen and R. J. Ilmoniemi, "Interpreting magnetic fields of the brain: minimum norm estimates," *Medical and Biological Engineering and Computing*, vol. 32, no. 1, pp. 35–42, 1994.

[10] A. M. Dale and M. I. Sereno, "Improved localization of cortical activity by combining EEG and MEG with MRI cortical surface reconstruction: a linear approach," *Journal of Cognitive Neuroscience*, vol. 5, no. 2, pp. 162–176, 1993.

[11] A. M. Dale, B. Fischl, and M. I. Sereno, "Cortical surface-based analysis: I. Segmentation and surface reconstruction," *NeuroImage*, vol. 9, no. 2, pp. 179–194, 1999.

[12] C. Campi, A. Pascarella, A. Sorrentino, and M. Piana, "A Rao-Blackwellized particle filter for magnetoencephalography," *Inverse Problems*, vol. 24, no. 2, Article ID 025023, 2008.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 923703, 13 pages
doi:10.1155/2011/923703

Research Article

Forward Field Computation with OpenMEEG

Alexandre Gramfort,[1] Théodore Papadopoulo,[2] Emmanuel Olivi,[2] and Maureen Clerc[2]

[1] *Parietal Project Team, INRIA Saclay Ile-de-France, Neurospin-CEA, Bât 145, Point Courrier 156, 91191 Gif/Yvette, France*
[2] *Athena Project Team, INRIA Sophia Antipolis-Méditerranée, 2004, Route des Lucioles, 06902 Sophia Antipolis, France*

Correspondence should be addressed to Alexandre Gramfort, alexandre.gramfort@inria.fr

Received 14 September 2010; Revised 14 December 2010; Accepted 17 January 2011

Academic Editor: Sylvain Baillet

To recover the sources giving rise to electro- and magnetoencephalography in individual measurements, realistic physiological modeling is required, and accurate numerical solutions must be computed. We present OpenMEEG, which solves the electromagnetic forward problem in the quasistatic regime, for head models with piecewise constant conductivity. The core of OpenMEEG consists of the symmetric Boundary Element Method, which is based on an extended Green Representation theorem. OpenMEEG is able to provide lead fields for four different electromagnetic forward problems: Electroencephalography (EEG), Magnetoencephalography (MEG), Electrical Impedance Tomography (EIT), and intracranial electric potentials (IPs). OpenMEEG is open source and multiplatform. It can be used from Python and Matlab in conjunction with toolboxes that solve the inverse problem; its integration within FieldTrip is operational since release 2.0.

1. Introduction

It is well recognized that conductivity models and forward solutions play an important role in accurate source localization from EEG [1, 2]. This is also true for MEG, though to a lesser degree [3].

Despite the simple mathematical nature of the equations giving rise to the electric potential and the magnetic field, these equations are not trivial to solve numerically, because of large conductivity ratios arising between neighboring tissues of the head. The field of forward modeling in EEG and MEG dates back to Barnard, who derived integral equations for electrocardiography [4, 5] and to Geselowitz [6]. After these seminal papers, several groups proposed Boundary Element solutions to these problems (as well as other types of solutions, notably Finite Elements, that are beyond the scope of this paper) [7–9].

The difficulty in the numerical resolution of the forward problem arises when electric sources are close to the boundary between two such tissues, in which case the solvers face accuracy problems [8]. Such source positions are not rare occurrences: indeed the gray matter, where the electric dipoles representing brain activity may be assumed to reside, is quite close to the cerebrospinal fluid (CSF) and to the skull. An accuracy correction method, called Isolated Skull Approach (ISA), was proposed to alleviate these accuracy issues [10]. Although the accuracy was improved in most cases, it was sometimes degraded [8]. Until recently, no acceptable solution was available that did not use the ISA.

For these reasons, a research program on forward modeling for EEG and MEG was conducted at INRIA, leading to the development of the symmetric BEM [11–13]. The OpenMEEG software package makes this new development available to the MEG/EEG community.

The thrust of OpenMEEG is to propose accurate forward problems, in several instances. The most classical instances are EEG and MEG, but OpenMEEG also allows to compute the electric potential due to boundary current injection (as occurs in Electrical Impedance Tomography or in Functional Electrical Stimulation) and to compute the electric potential measured within the brain (as occurs in stereographic EEG).

For each of these instances, the result of the forward problem is expressed as a lead field, that is, the matrix representing the linear relation between sources and measurements, a.k.a. "Gain matrix."

The modeling assumptions of OpenMEEG are explained in Section 2. Section 3 then details the four instances of forward computation: EEG, MEG, EIT, and Internal Potential (IP), from the physical model to the OpenMEEG commands. Section 4 provides practical information on OpenMEEG usage. The accuracy of OpenMEEG is assessed in a benchmark comparison test in Section 5, and the paper ends with a conclusion. The material presented here refers to releases 2.1 and later.

2. Modeling Assumptions

The quasistatic regime of Maxwell's equations is valid at the frequencies of interest in EEG and MEG, and also for EIT and functional electrical stimulation, at stimulation frequencies below 1 kHz. In this regime, the electrical potential V is governed by the law of electrostatics

$$\cdot(\sigma \quad V) = \quad \cdot \mathbf{J}^{\mathbf{p}}, \tag{1}$$

where σ is the conductivity field and $\mathbf{J}^{\mathbf{p}}$ is the source distribution. The brain sources are modeled as dipoles, representing average postsynaptic currents within pyramidal cortical neurons. A boundary condition fixes the value of the normal current on the domain boundary

$$\sigma \quad V \cdot \mathbf{n} = j. \tag{2}$$

In EEG and MEG, the value of the normal current on the scalp is $j = 0$, but in electrical impedance tomography, j takes the values of the current injected on the scalp.

2.1. Head Model. OpenMEEG is based on a Boundary Element representation of physical fields, implying that the conductivity model, describing the conductivity field σ, must be piecewise constant. The physical fields are thus represented on the boundaries of the regions of constant conductivity. More precisely, OpenMEEG is restricted to nested conductivity models, that is, in which there are successive layers of constant conductivity (see Figure 1(a)). This model is generally well suited to the head, as it can handle the brain, CSF, skull, and scalp conductivities. Extensions of the symmetric BEM have been proposed to handle nonnested regions as in Figure 1(b) but are not yet handled by OpenMEEG [13]. Regarding the conductivity field, the only theoretical restriction for using Boundary Element methods is that the conductivity field must be translation invariant in each domain. Thus, for complex 3D domains as the head, anisotropic conductivity cannot be handled with a BEM, and other solvers using volumic approaches must be used (e.g., Finite Element methods).

2.2. Source Models. The primary current within the brain $\mathbf{J}^{\mathbf{p}}$ in (1) is represented as a distribution of dipoles. This distribution may be either pointwise or surfacic. A pointwise source distribution is a collection of pointwise dipoles, defined by their positions and moments. A surfacic source distribution is defined over a surfacic mesh, as

$$\mathbf{J}^{\mathbf{p}}(r) = \sum_i \phi_i(r) J_i \mathbf{n}(r), \tag{3}$$

where the sum runs over all vertices, ϕ_i is the piecewise linear function equal to 1 on vertex i and 0 on all others, and $\mathbf{n}(r)$ is the normal to the surface at position r. The source intensity is linear on each triangle and equal to J_i on vertex i.

Note that the pointwise source distribution is the most commonly used, because it is difficult to define a surface supporting the sources—hence matching the gyri and sulci—on which the orientations are sufficiently smooth.

Another type of source that can be considered is the normal component of the boundary current: j in (2). This normal current is modeled as piecewise constant on the mesh, that is,

$$j(r) = \sum_k \psi_k(r) j_k, \tag{4}$$

where the sum here runs over all triangles and ψ_k is the piecewise constant function equal to 1 on triangle k and 0 on all others.

2.3. Sensor Models. Four types of modalities are considered: EEG electrodes, MEG sensors, current injection electrodes, and intracranial electrodes for measuring the potential. In each case, the sensor model considered by OpenMEEG is very basic, that is, it does not model capacitive effects, nor electrode extension.

EEG and intracranial electrodes are assumed punctual and defined by their 3D coordinates. In the case of EEG, the 3D electrode position is projected orthogonally onto the scalp surface. Each MEG sensor is defined by a collection of points and weights, thus modeling magneto- or gradiometers, possibly with the shape of the coil wiring. Current injection electrodes are defined by their 3D coordinates, and the current injection model is a uniform current over the closest triangle to the injection electrode.

3. Forward Field Computation

For the models explained above, OpenMEEG is equipped to compute four different types of lead fields. We now detail the computations for each of them. In addition, the reader can refer to a global flowchart in Figure 5, which explains the structure of the commands and of the input/output arguments. Information on input/output format is provided in Section 4.

3.1. EEG Lead Field. Computing an EEG lead field amounts to computing the potential V on electrodes, due to a set of dipolar sources at prescribed positions and orientations. (For simplicity, our description considers a pointwise source distribution, but the method also applies to a surfacic source distribution) The potential V is defined, up to an additive constant, as the solution of (1) with a boundary condition (2) in which no current flows across the scalp. Considering a nested conductivity model as in Figure 1(a), the symmetric Boundary Element expresses the solution of this problem,

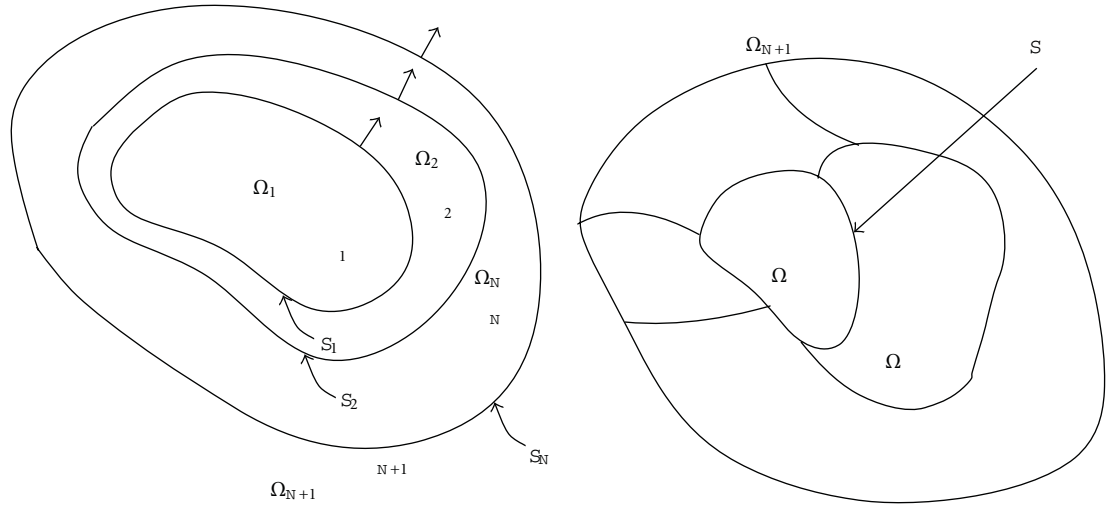


FIGURE 1: Boundary Elements are well suited for piecewise constant isotropic conductivity models. OpenMEEG handles nested regions (a) and could in principle be extended to more general, disjoint regions (b) [13].

restricted to the domain boundaries, as the solution of the set of equations

$$\texttt{HeadMatrix} \cdot \begin{bmatrix} V_{S_1} \\ (\sigma \partial_{\mathbf{n}} V)_{S_1} \\ V_{S_2} \\ (\sigma \partial_{\mathbf{n}} V)_{S_2} \\ \vdots \\ V_{S_N} \end{bmatrix} = \texttt{SourceMatrix} \cdot \begin{bmatrix} J_1 \\ J_2 \\ \vdots \\ J_p \end{bmatrix} \quad (5)$$

for a set of p source intensities corresponding to p prescribed dipoles [11]. Both the potential V_{S_i} and the normal current $(\sigma \partial_{\mathbf{n}} V)_{S_i}$ are discretized on each boundary S_i (except the scalp where only the potential needs to be discretized since the normal current vanishes). The potential is represented with piecewise linear boundary elements, while the normal current is represented with piecewise constant boundary elements.

The two matrices `HeadMatrix` and `SourceMatrix` involve Boundary Integral operators which OpenMEEG is equipped to compute. Computing the EEG lead field L_{EEG} amounts to solving the symmetric linear system (The denomination "Symmetric BEM" is due to the symmetric nature of the `HeadMatrix`):

$$\texttt{HeadMatrix} \cdot X = \texttt{SourceMatrix} \quad (6)$$

and applying to the result X an interpolation operator to infer the potential at the scalp electrode positions

$$L_{\text{EEG}} = \texttt{Head2EEGMatrix} \cdot X. \quad (7)$$

Matrices are assembled in OpenMEEG by invoking the command

```
om_assemble Option Parameters Matrix.
```  (8)

`HeadMatrix` is assembled with the `-HeadMat` option and Parameters containing the geometry and conductivity description.

`Head2EEGMatrix` is assembled with the `-Head2EEGMat` option, and the same `Parameters`.

`SourceMatrix` is assembled with the `-DipSourceMat` or `-SurfSourceMat` option, depending on the source model (Section 2.2), and `Parameters` containing the geometry, conductivity, and source description (positions and orientations, or surface supporting a surfacic source).

Finally, (6) and (7) are solved by successively

(i) inverting matrix `HeadMatrix`:

```
om_minverser HeadMatrix HeadMatrixInv;
```  (9)

(ii) applying the interpolation and the inverse matrix to the `SourceMatrix`:

```
om_gain -EEG HeadMatrixInv
SourceMatrix Head2EEGMatrix GainMatrix.
```  (10)

The EEG lead field is the output of the previous command.

3.2. MEG Lead Field. The magnetic field $\mathbf{B}$ depends both on the electric potential V and on the current source distribution $\mathbf{J}^{\mathbf{p}}$, through the Biot and Savart law

$$\mathbf{B}(\mathbf{r}) = \frac{\mu_0}{4\pi} \int (\mathbf{J}^{\mathbf{p}}(\mathbf{r}\,) - \sigma \quad V(\mathbf{r}\,)) \times \frac{\mathbf{r} - \mathbf{r}}{\mathbf{r} - \mathbf{r}^{\,3}} d\mathbf{r}\,, \quad (11)$$

when $j = 0$ on the boundary.

The magnetic field $\mathbf{B}$ can be split into two contributions, the primary field generated by the primary current and the ohmic field. The primary field is computed as a linear

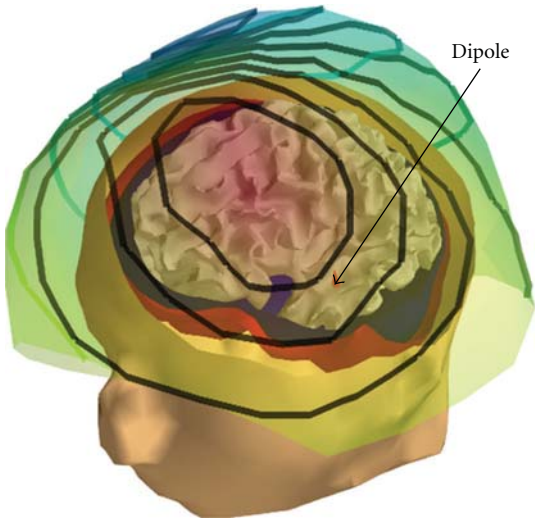


Figure 2: MEG simulation: visualization on a surface interpolating the sensors (radial magnetometers) of the field induced by a dipolar source on the left temporal cortex (in red). Computation is done with OpenMEEG and a 3-layer head model.

relation between sources and measurements, via a matrix `Source2MEGMatrix`. The Ohmic field is computed as a linear relation between electrical potential and measurements. Computing this Ohmic lead field amounts to solving (6) (as when computing L_{EEG}) and applying to the result X an operator `Head2MEGMatrix`. Finally the MEG lead field L_{MEG} is equal to:

$$L_{\text{MEG}} = \texttt{Head2MEGMatrix} \cdot X + \texttt{Source2MEGMatrix}. \tag{12}$$

`HeadMatrix` and `SourceMatrix` are identical to those of the EEG *lead field*, and their assembly has been explained in Section 3.1. Matrices `Head2MEGMatrix` and `Source2MEGMatrix` are obtained through the `om_assemble` command. `Head2MEGMatrix` is assembled with the option `-Head2MEGMat` and with `Parameters` describing the geometry, conductivity, and sensors; `Source2MEGMatrix` is assembled with the option `-DipSource2MEGMat` (pointwise source) or `-SurfSource2MEGMat` (surfacic source), and with the previously listed parameters, plus the source description (discrete points and orientations, or a surface). Finally, the MEG lead field L_{MEG} is computed by invoking `om_gain` with the option `-MEG`, and input matrices `HeadMatrixInv`, `SourceMatrix`, `Head2MEGMatrix`, `Source2MEGMatrix`.

Figure 2 displays a magnetic field corresponding to a single dipole and interpolated on a surface containing the magnetometer positions.

3.3. EIT Lead Field. OpenMEEG also allows to compute the electric potential due to an applied current on the boundary of the domain. This occurs in electrical impedance tomography, and also in functional electrical stimulation. We will denote this type of problem "EIT," bearing in mind that it may also concern other application fields. Electrical Impedance Tomography (EIT) seeks to estimate the conductivities of the model, by analyzing the potential resulting from the application of a current on the boundary. In EIT, the conductivities must then be adjusted to match the measured current potential correspondence [14–16]. OpenMEEG allows to compute this current potential correspondence, for fixed values of conductivity. This amounts to solving (1) and (2), for prescribed injected current j, and selecting the values of the potential on the electrodes as for EEG in (7).

It is interesting to note that only the right-hand side of (6) must be changed when EIT is being solved instead of EEG. The source matrix for EIT is computed by invoking `om_assemble` with the `-EITSourceMat` option and as parameters the geometry file, conductivity file, and the file describing the EIT electrodes.

After inverting the left-hand side matrix in (6) (yielding `HeadMatrixInv`) and computing the electrode interpolation matrix `Head2EEGMatrix`, the EIT lead field is computed using `om_gain` with the `-EEG` option

```
om_gain -EEG HeadMatrixInv EITSourceMatrix
Head2EEGMatrix GainMatrix.
```
(13)

Figure 3 displays the scalp potential corresponding to a current injection between two electrodes.

Note that, for a new set of conductivity values, the computation of the `HeadMatrix` is immediate, because of the form of the `HeadMatrix` (14) (refer to [11] for a proof). This makes the EIT inverse problem quite tractable using OpenMEEG [17]:

$$\texttt{HeadMatrix} = \sum_{i=1}^{N} (\sigma_i A_i + \sigma_i^{-1} B_i). \tag{14}$$

3.4. Internal Potential (IP) Lead Field. In certain clinical settings, the potential may be measured within the brain (intracranial EEG or stereotaxic EEG). Given a distribution of current generators within the head, OpenMEEG is able to compute the potential at any position within the head (brain, skull, scalp). This may appear surprising, because OpenMEEG is based on a Boundary Element Method that, by definition, only represents the potential on the interfaces between domains. But computing the potential within a domain from the knowledge of the potential and normal current on the surrounding interfaces is only a matter of applying a Green harmonic representation theorem.

In practise, this relation is provided by a matrix `Head2IPMatrix`. One must also take into account a contribution from the sources belonging to the same domain as the electrodes.

Computing the internal potential lead field L_{IP} proceeds by solving for X, as for the computation of L_{EEG},

$$\texttt{HeadMatrix}.\ X = \texttt{SourceMatrix}, \tag{15}$$

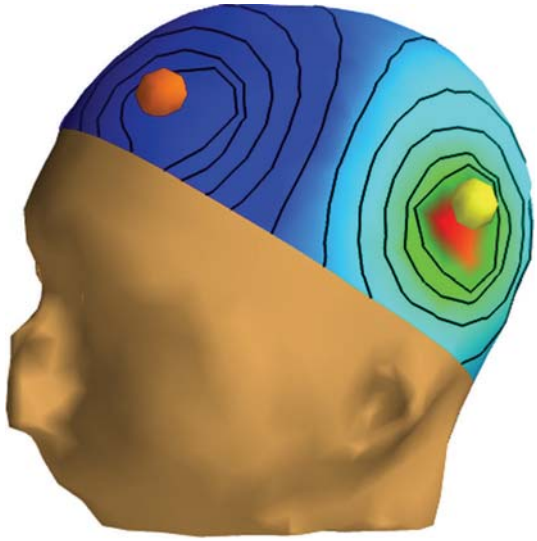


Figure 3: EIT simulation: visualization on a surface interpolating EEG electrodes of the electric potential when a current flows from one electrode (in yellow) to another (in orange). Computation is done with OpenMEEG and a 3-layer head model.

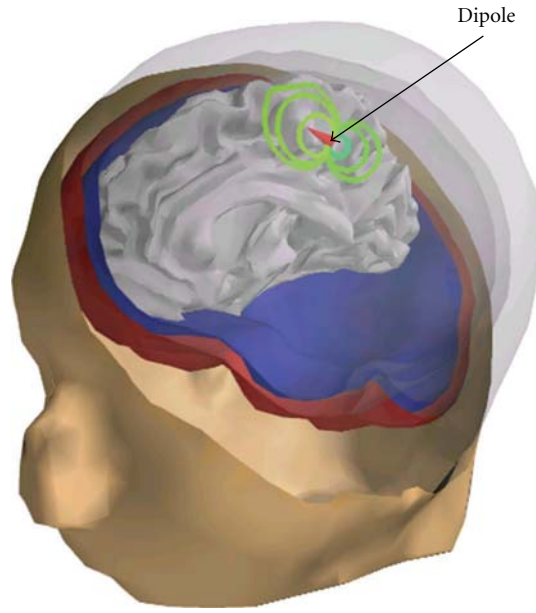


Figure 4: Internal Potential simulation: visualization of the internal potential computed using OpenMEEG in a 3-layer head model. A dipolar source (red cone) is located in the left hemisphere of the cortex (not represented), and the curved lines are isopotential lines. A bending of the isopotentials near the skull can be observed.

and then computing

$$L_{\mathrm{IP}} = \texttt{Head2IPMatrix} \cdot X + \texttt{Source2IPMatrix}. \tag{16}$$

`Head2IPMatrix` is assembled with the `om_assemble` command with option `-Head2InternalPotMat` and the usual parameters (geometry and conductivity description) along with the internal points. `Source2IPMatrix` is assembled with the same command with option `-DipSource2InternalPotMat` and, in addition to the previous parameters, the source description. Finally, L_{IP} is computed by invoking `om_gain` with the option `-InternalPotential` and input matrices `HeadMatrixInv`, `SourceMatrix`, `Head2IPMatrix`, and `Source2IPMatrix`.

Figure 4 displays the internal potential due to a single dipole.

4. Usage of OpenMEEG

4.1. I/O File Formats. OpenMEEG handles several file formats corresponding to several types of objects: vectors, matrices, head geometries, meshes, dipoles, conductivities, and sensors.

By default, matrices and vectors are stored on disk using a Matlab file format. Symmetric matrices, for which Matlab does not propose a format, are represented as a Matlab structure. Alternatively OpenMEEG handles plain ASCII files (usually used for sensors and dipole descriptions) and BrainVisa textures.

OpenMEEG geometrical models are described via several files. Note that OpenMEEG considers SI units (point coordinates should be expressed in meters (m), conductivities in S/m, etc.). The top level file (with the extension `.geom`) describes the nested structure of the different domains (see Figure 6). An associated conductivity file (with extension `.cond`) contains the conductivities of the domains (see Figure 7).

Mesh formats supported are BrainVisa `.tri` files (default) and ASA `.bnd` files.

4.2. Example Scripts and Demos. Much effort has been devoted to facilitating the use of OpenMEEG by the M/EEG community. OpenMEEG can be invoked either via a command line interface (see Figure 5) or via higher-level languages. OpenMEEG can be used from Python or from Matlab via the FieldTrip Toolbox, where it is fully integrated in the forward modeling routines. Within FieldTrip, OpenMEEG can compute lead fields for head models with 1, 2, 3, or 4 nested layers.

Algorithms 1 and 2 provide sample Python and FieldTrip scripts.

4.3. Technical Details. OpenMEEG is distributed under the French open source license CeCILL-B, intended to give users the freedom to modify and redistribute the software. It is therefore compatible with popular open source licenses such as the GPL and BSD licenses. Due to the CeCILL-B license, anybody distributing a software incorporating OpenMEEG has an obligation to give credits by citing the appropriate publications. (The references to be cited to comply with the license can be found on the OpenMEEG webpage http://openmeeg.gforge.inria.fr.)

OpenMEEG is implemented in C/C++ with limited external dependencies. It uses the Intel MKL libraries on Windows and ATLAS (BLAS/LAPACK) on Unix systems

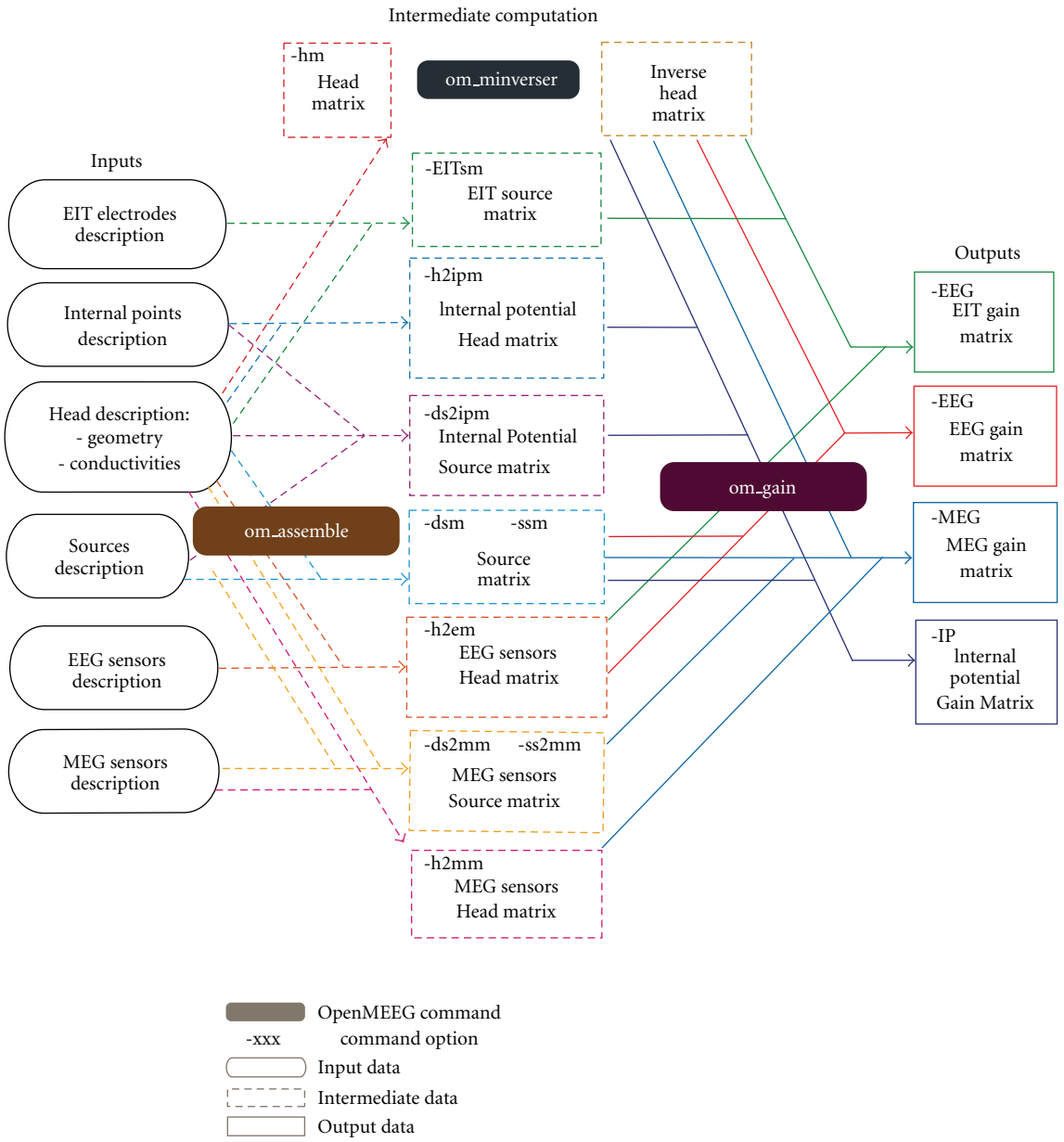


Figure 5: Flowchart describing the OpenMEEG procedure for computing EIT, EEG, MEG, and IP lead fields.

for fast and accurate linear algebra. A modified version of the MATIO library (http://sourceforge.net/projects/matio) has been integrated in OpenMEEG for Matlab compatibility. The source code of OpenMEEG is hosted on the INRIA GForge platform and is accessible from http://openmeeg.gforge.inria.fr.

OpenMEEG is available as precompiled binaries for GNU-Linux systems, Mac OS, and Windows (32 and 64 bits). OpenMEEG's build and packaging system is based on *CMake/CPack* (http://www.cmake.org) allowing easy development and deployment on all architectures.

To accelerate the computations, OpenMEEG can be compiled with OpenMP, a technology that enables parallel computation at limited cost in terms of software design. The numerical integration, on which most of the computation time is spent, can then be run in parallel. Figure 8 presents observed computation times for the computation of an EEG lead field with the head model of Figure 4 (approximately 700 vertices per layer, 3 layers and 15000 dipoles). It can be observed that with 4 threads, the computation is almost 3 times faster. The memory requirement for this example is 770 MB. The computation time of a forward field computation with OpenMEEG can be roughly broken down into three main components: the Head Matrix assembly, its inversion, and the Source Matrix assembly (identified as HM, HMINV, and DSM in Figure 8).

```
import openmeeg as om
## Load data
geom = om.Geometry()
geom.read('head_model.geom', 'head_model.cond')
dipoles = om.Matrix()
dipoles.load('cortex_dipoles.txt')
meg_sensors = om.Sensors()
meg_sensors.load('meg_channels.squids')
eeg_electrodes = om.Matrix()
eeg_electrodes.load('eeg_channels.txt')
int_electrodes = om.Matrix()
int_electrodes.load('internal_electrodes.txt')
## Assemble matrices
gauss_order = 3
use_adaptive_integration = True
hm = om.HeadMat(geom, gauss_order)
hminv = hm.inverse()
dsm = om.DipSourceMat(geom, dipoles, gauss_order, use_adaptive_integration)
# For EEG
h2em = om.Head2EEGMat(geom, eeg_electrodes)
# For MEG
ds2mm = om.DipSource2MEGMat(dipoles, meg_sensors)
h2mm = om.Head2MEGMat(geom, meg_sensors)
# For EIT (using the same electrodes as EEG)
eitsm = om.EITSourceMat(geom, eeg_electrodes, gauss_order)
# For Internal Potential
iphm = om.Surf2VolMat(geom, int_electrodes)
ipsm = om.DipSource2InternalPotMat(geom, dipoles, int_electrodes)
## Compute leadfiels
eeg_leadfield = om.GainEEG(hminv, dsm, h2em)
meg_leadfield = om.GainMEG(hminv, dsm, h2mm, ds2mm)
eit_leadfield = om.GainEEG(hminv, eitsm, h2em)
ip_leadfield = om.GainInternalPot(hminv, dsm, iphm, ipsm)
```

Algorithm 1: Demo script for computing the 4 types of lead fields with OpenMEEG in python.

```
%% The structure for the BEM volume conduction model
% vol.bnd(k).pnt: vertices of the mesh of layer "k"
% vol.bnd(k).tri: triangles of the mesh of layer "k"
% vol.cond      : conductivities of each layer
%% EEG electrodes
% sens.pnt        : locations of the EEG electrodes
%% Positions of the dipoles
% pos             : positions of the dipoles
%% Compute the BEM
% choose implementation (OpenMEEG, BEMCP or Dipoli)
cfg.method = 'openmeeg';
% Compute the BEM matrix
vol = ft_prepare_bemmodel(cfg, vol);
cfg.vol = vol;
cfg.grid.pos = pos;
cfg.elec = sens;
% Get the computed leadfield
lf_openmeeg = ft_prepare_leadfield(cfg);
```

Algorithm 2: Demo script for computing an EEG lead field with OpenMEEG in FieldTrip.

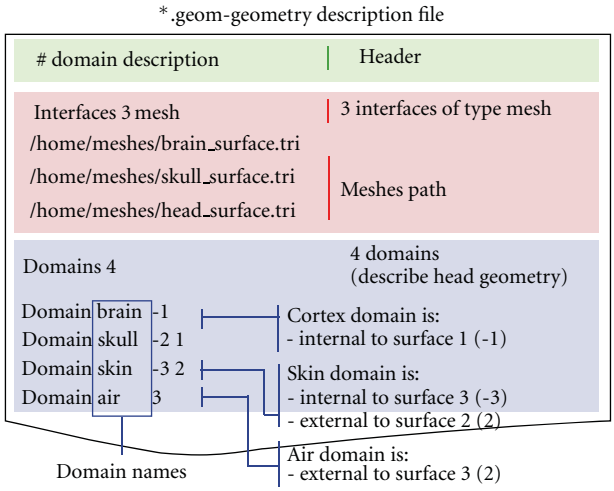


Figure 6: Sample geometry file. The `Interfaces` section provides the meshes, while the section `Domains` informs of the physical model.

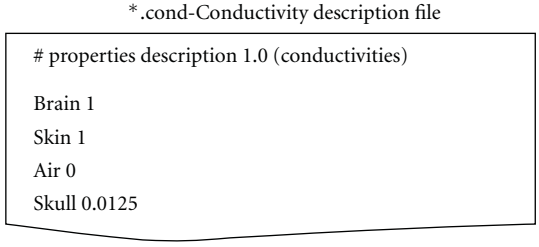


Figure 7: Sample conductivity file associated to the geometry file in Figure 6.

Deployment on multiple architectures with heterogenous hardware and software environments requires testing procedures to assess the stability of the solutions provided by compiled binaries. This testing procedure, based on *CMake/CTest*, guarantees the integrity of the results, in particular by comparison with analytical results on spherical models.

5. EEG and MEG Comparison Study

5.1. Benchmark Presentation. When the head model consists of nested concentric spheres, the accuracy of EEG and MEG forward computations can be assessed by comparing the computed solution with the analytical solution. We here present an excerpt of the benchmark study presented in [18].

The precision of a forward solution is tested with two measures: the Relative Difference Measure (RDM) and the magnitude ratio (MAG) [19].

The RDM between the forward field given by a numerical solver g_n and the analytical solution g_a is defined as

$$\mathrm{RDM}(g_n, g_a) = \left\| \frac{g_n}{||g_n||} - \frac{g_a}{||g_a||} \right\| \quad [0, 2], \tag{17}$$

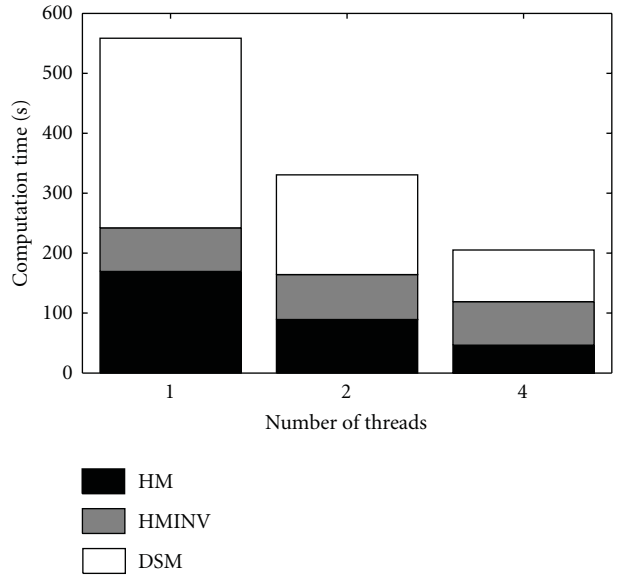


Figure 8: Computation time of an EEG lead field with the head model of Figure 4 (approximately 700 vertices per layer, 3 layers and 15000 dipoles). With 4 threads, the computation is almost 3 times faster. These results were obtained on a quad-core Intel Xeon CPU working at 3.20 GHz.

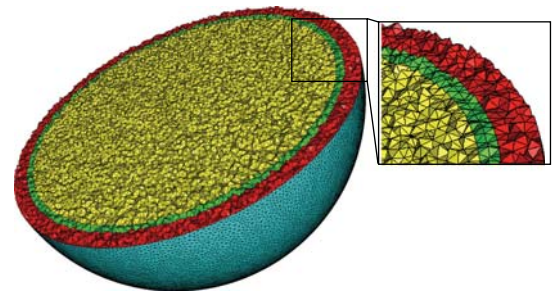


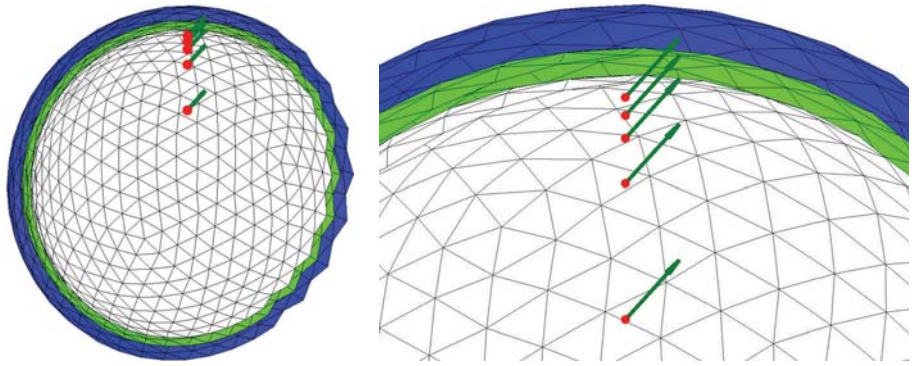
Figure 9: The 3D mesh used in FEM computations. The 3 layers are shown in red, green, and yellow, respectively.

while the MAG between the two forward fields is defined as

$$\mathrm{MAG}(g_n, g_a) = \frac{||g_n||}{||g_a||}. \tag{18}$$

In both of these expressions, the norm is the Euclidian ℓ^2 norm over the set of sensor measurements.

Geometrical Models. The comparisons were made both on classic regular sphere meshes as in Figure 10, and on random meshes [18]. The BEM solvers are tested with three-layer head models which model the inner and outer skull, and the skin. The radii of the 3 layers are set to 88, 92, and 100, while the conductivities of the 3 homogeneous volumes are set to standard values: 1, 1/80 (skull) and 1. For every head model, solvers are tested with the same 5 dipoles positioned on the z-axis with orientation (1,0,1) and various distances to the inner layer (cf. Figure 10).



(a) 3-layer spherical head model (b) Zoom

Figure 10: Head model made of 3 nested regularly meshed spheres with 5 dipoles close to the inner layer.

Results: Accuracy of the Electric Potential Simulations. Alternative BEM software available to the community are all based on the Geselowitz formulation [6]. From this formulation, different implementations may be derived. The potential may be modeled either with constant elements (i.e., the potential is piecewise constant over each mesh) or, for more precision, with linear elements (i.e., the potential is piecewise linear over each mesh). The computation may then be achieved with a Galerkin method involving numerical integration, as in OpenMEEG, or with a more simple collocation method (see [8] for a detailed study of Galerkin versus collocation methods). The linear collocation (LC) method is implemented by BEMCP [20] which is available from FieldTrip and is the default forward solver in SPM. In order to improve the accuracy of LC methods, the Isolated Skull Approach (ISA) has been proposed [10]. It is used by SimBio [21], Dipoli [22], and the Helsinki BEM [23], which implements both a simple LC and an LC with ISA (LCISA). Within SimBio, we only consider here its BEM solver, referred to as SimBio-BEM [24], as opposed to SimBio-FEM [25, 26] that focuses on inhomogeneous and anisotropic head volume conductor models.

The Helsinki BEM is the implementation used in this benchmark. However all the aforementioned solvers have been tested, and it has been confirmed that all LCISA solvers tested provide almost the same results, as do all the LC solvers tested [18]. One of the features of OpenMEEG is to use an adaptive numerical integration method. To demonstrate its influence on the results, we have also tested a nonadaptive version of OpenMEEG (OMNA).

For the sake of completeness, let us mention that the BEM solver implemented in the MNE (http://www.nmr.mgh.harvard.edu/martinos/userInfo/data/sofMNE.php) software package is also LCISA based.

Furthermore, as a crude comparison, a basic finite element method with P1 basis elements on a tetrahedral mesh (TFEM) has also been run. It is a classical FEM, with a preconditioned conjugated gradient as solver (Jacobi preconditioner), and the dipole source is modeled through partial integration. Such a model approximates the dipole with a continuous distribution of sources supported over a small region, which introduces a source approximation error which does not exist for BEM models. Note that there are solutions to better model dipole sources with FEM such as the subtraction method or the Venant direct approach [27]. Such methods are beyond the scope of this contribution. The mesh (427,000 vertices, with 43,768 vertices on the outer surface) was generated with CGAL (CGAL, Computational Geometry Algorithms Library, http://www.cgal.org.) A view of this mesh is shown in Figure 9.

Results: Accuracy of the Magnetic Field Simulations. Magnetic fields are commonly computed, in the MEG community, using analytical solutions on spheres. While (Ohmic) volume currents do not contribute to the radial component of the magnetic field on a nested spherical model; they do on a realistic geometry and must then be computed. OpenMEEG and SimBio-BEM are two freely available software projects that provide a computation of the magnetic field depending on the electrical potential.

5.2. Results

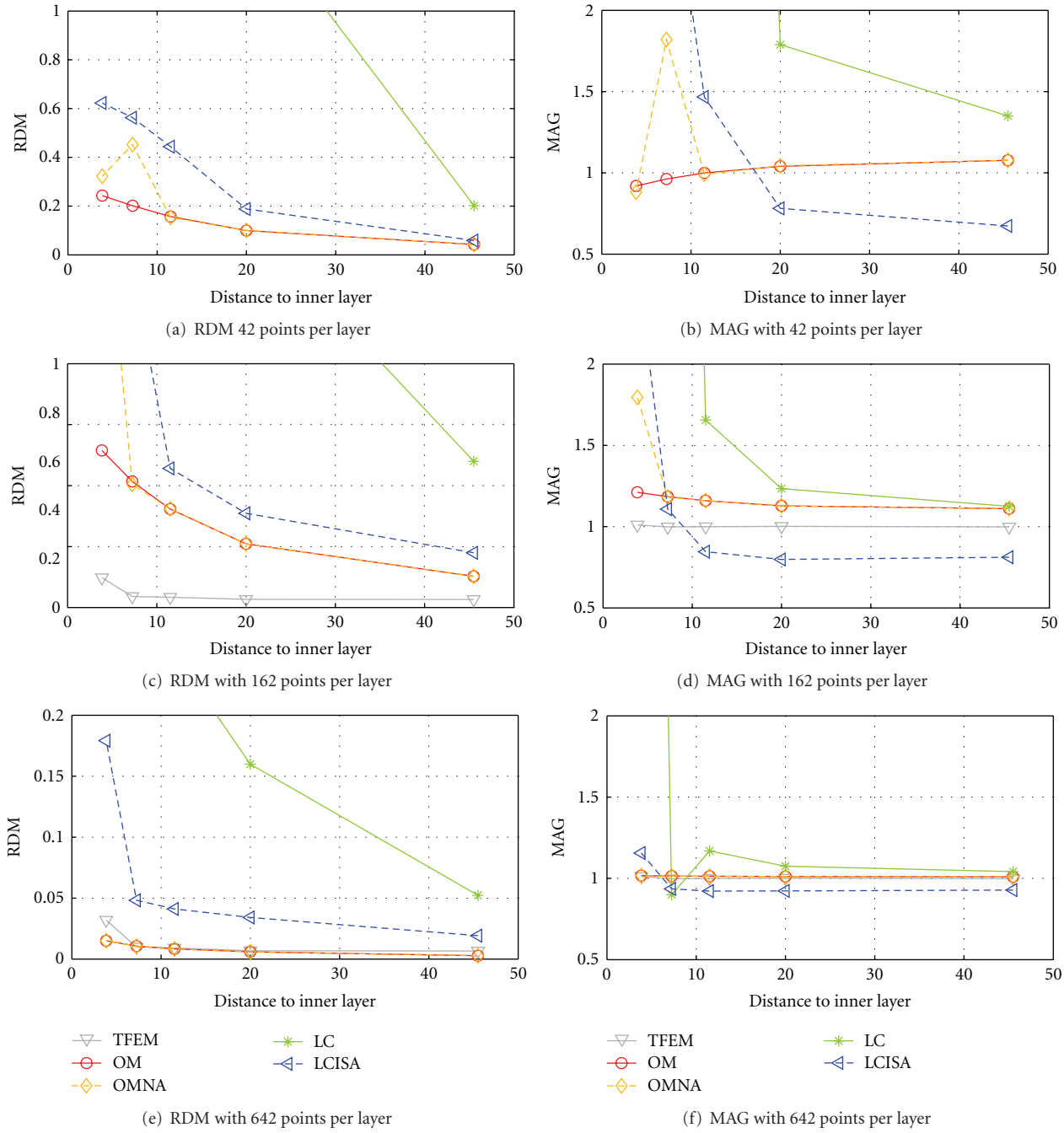
5.2.1. Regularly Meshed Spheres. Results on regularly meshed spheres are presented in Figure 11, for 3 different point samplings on each interface. The coarsest sampling has only 42 vertices per layer and 42 EEG electrodes, the intermediate one has 162 points per layer and 162 EEG electrodes, and the finest sampling has 642 points per layer and 642 EEG electrodes.

From these simulations the following can be observed.

(i) The simple linear collocation method is clearly the least accurate.

(ii) The linear collocation method with ISA correction is more accurate.

(iii) OpenMEEG provides the most accurate solutions even when no adaptive integration is used. The adaptive integration further improves the results, particularly when the meshes are coarsely sampled (42 and 162 vertices per layer).



(a) RDM 42 points per layer

(b) MAG with 42 points per layer

(c) RDM with 162 points per layer

(d) MAG with 162 points per layer

(e) RDM with 642 points per layer

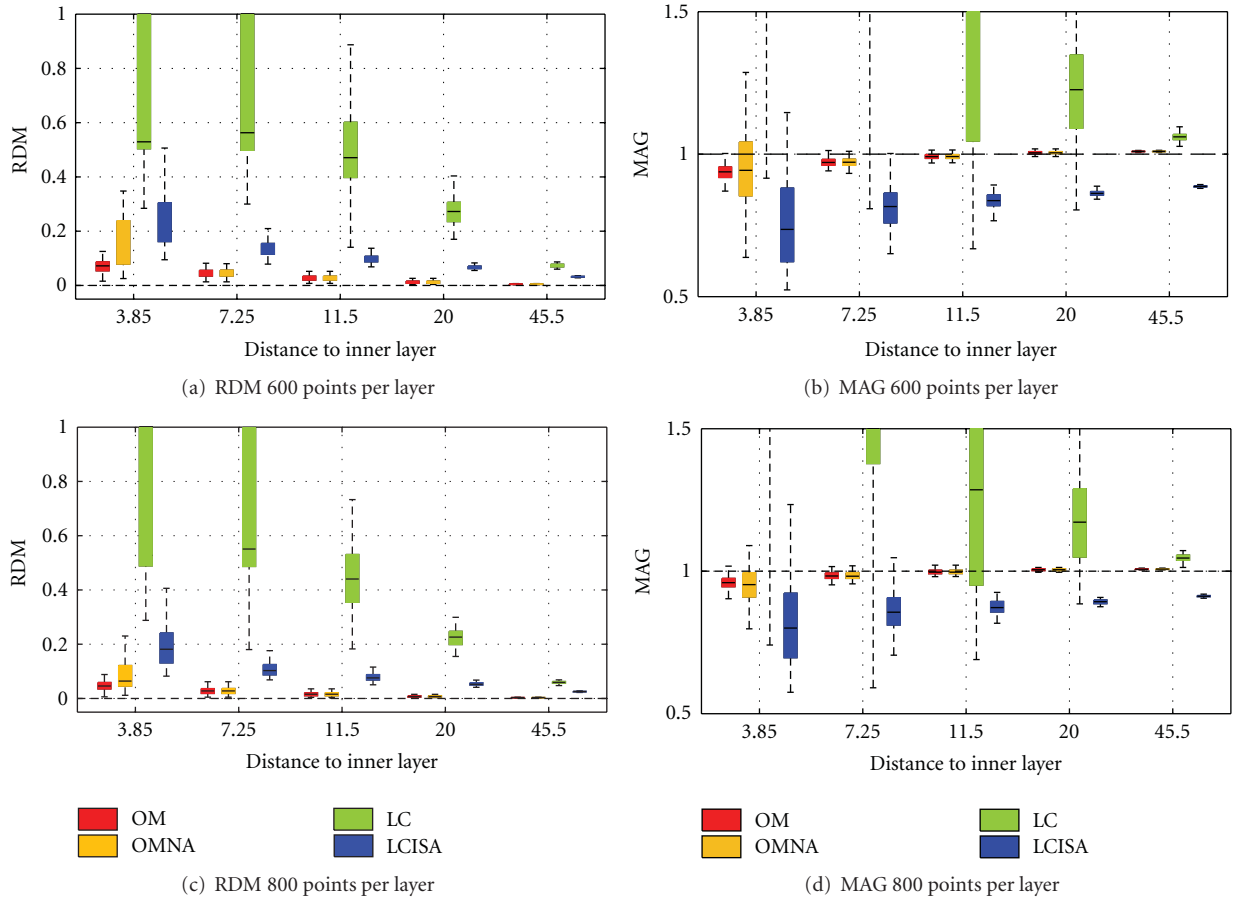
(f) MAG with 642 points per layer

Figure 11: Forward EEG: accuracy comparison of different BEM solvers with three-layer sphere head models. We observe that the symmetric BEM outperforms the other BEM methods in term of precision. The TFEM was run on a mesh with 427,000 vertices, and the results were interpolated to 162 points in (c) and (d) and 642 points in (e) and (f).

(iv) Despite the high resolution of the mesh used with the FEM, OpenMEEG is more accurate for the model with 642 vertices per layer.

5.2.2. Randomly Meshed Spheres. Simulations have also been run on a large number of randomly meshed spherical meshes, in order to study the robustness of the solvers. Please refer to [18] for the meshing procedure. The results are obtained by testing each solver on 100 random head models. The mean accuracy measures (RDM and MAG) are represented using box plots.

EEG Results. Figure 12 presents the box plots obtained by running the solvers on random head models with either 600 or 800 vertices per layer. The mean results follow the ranking of Figure 11. However the variances



(a) RDM 600 points per layer

(b) MAG 600 points per layer

(c) RDM 800 points per layer

(d) MAG 800 points per layer

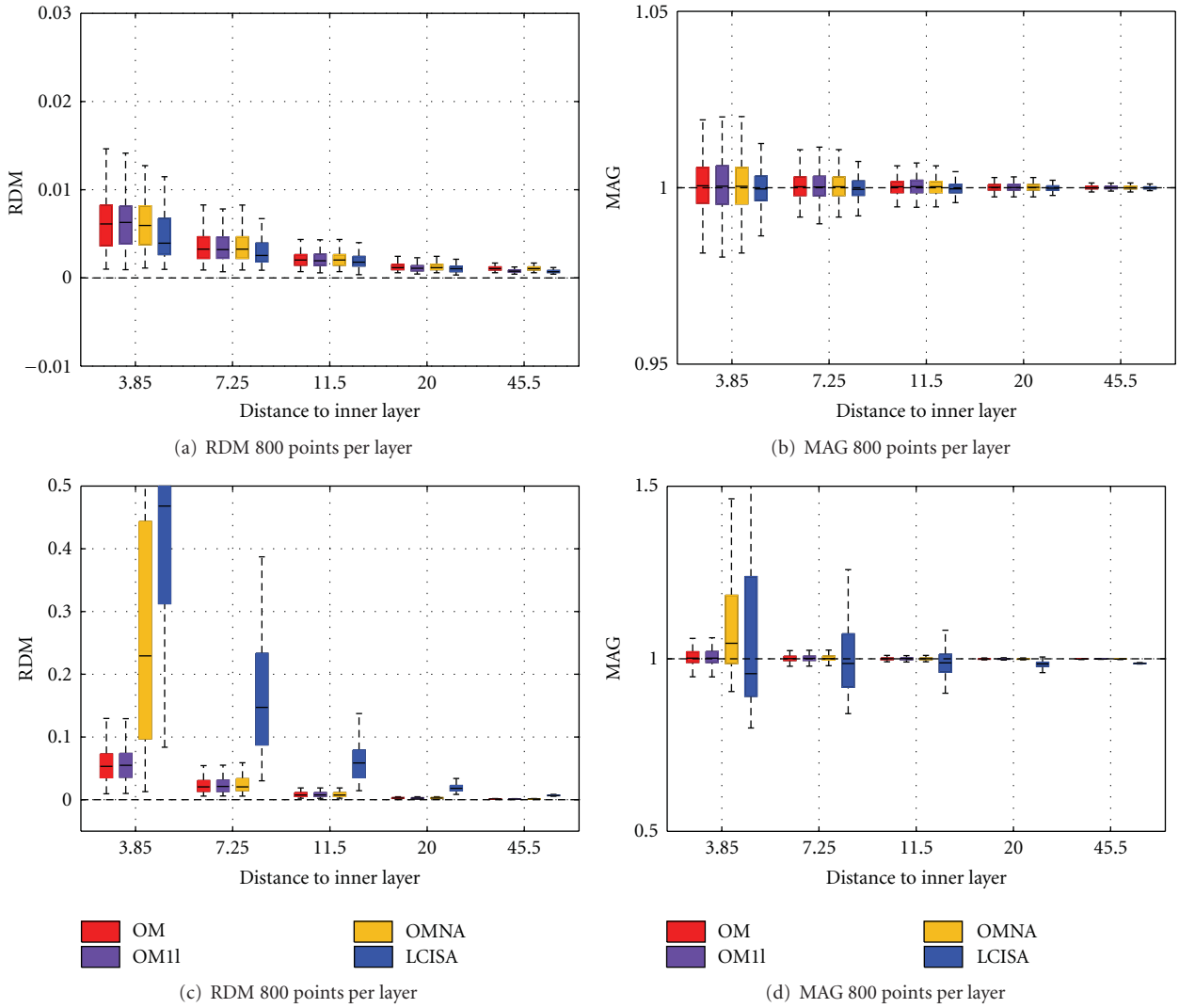
Figure 12: Forward EEG: RDM and MAG box plots obtained on 100 random 3-layer sphere models. Each layer contains 600 or 800 random vertices.

tell us that OM is not only very accurate, but also very precise because of its very small variance. (For the distinction between accuracy and precision, refer to http://en.wikipedia.org/wiki/Accuracy_and_precision.). The OMNA solver is also accurate but less precise: it has a larger variance, demonstrating that the adaptive integration improves robustness to the meshing. Linear collocation with ISA gives intermediate results. Also observe that linear collocation without ISA has significantly larger variance than the other solvers, meaning that it is very sensitive to the meshing.

MEG Results. As explained in Section 3.2, the magnetic field depends on the electric potential, thus its accuracy follows from that of EEG. Although MEG machines generally only provide radially oriented sensors with respect to the helmet (except for some reference channels), we have, in the following experiments, computed the nonradial magnetic field in order to validate the Ohmic field contribution. Indeed, in spherical geometry, for radial sensors, the magnetic field does not depend on the Ohmic contribution—which is no longer true for more realistic head models. Two types of sensors were thus considered: a set of magnetometers oriented in the Cartesian direction (1,0,1) and located at a distance 120 from the center of the model and one set of radially oriented sensors at the same locations. Figure 13 presents, for both types of sensors, the results of OpenMEEG on a 3-layer model, with and without adaptive integration (OM and OMNA), OpenMEEG on a one-layer model (OM1l), and LCISA (SimBio-BEM implementation) on a one-layer model. The use of a 3-layer model with OpenMEEG slightly improves the results obtained with only one layer. For radial magnetometers, one notices a slight advantage to LCISA both for accuracy and precision, but for nonradially oriented sensors, OpenMEEG outperforms both OMNA and LCISA. Performances of LCISA can however be significantly improved by increasing the number of vertices in each layer. In our investigations with SimBio-BEM, a number of 3400 nodes led to an RDM of maximally 0.047 and a MAG above 0.97.

6. Conclusion

OpenMEEG is a comprehensive, open source software package for solving many different instances of forward problems in quasistatic electromagnetism. It can compute lead fields for EEG and MEG, as well as EIT (or Functional Electrical Stimulation) and intracerebral EEG. Regarding



(a) RDM 800 points per layer

(b) MAG 800 points per layer

(c) RDM 800 points per layer

(d) MAG 800 points per layer

Figure 13: Forward MEG: RDM and MAG box plots obtained on 100 randomly meshed sphere models. OM and OMNA use a 3-layer model while OM1l and LCISA use a one-layer model. Each layer contains 800 random vertices. In (a) and (b), *radial magnetometers* are considered, whereas in (c) and (d) use *nonradial magnetometers*.

accuracy, OpenMEEG represents the state-of-the-art. Besides its excellent accuracy and its versatility, several other features of this software make it unique:

(i) the number of nested layers is unrestricted;

(ii) dipolar sources may be positioned within any of the domains;

(iii) EEG, EIT, MEG, and IP lead fields can be jointly computed on the same head model;

(iv) it is interfaced with Python and Matlab via FieldTrip for a maximal ease of use.

The progress brought forth by this new software however only represents a limited contribution in modeling brain functional activity. Head model generation is a crucial problem in practice, and the need for automatized procedures in this domain is crying. When more complex head models (involving inhomogeneous and anisotropic conductivity) are needed, Boundary Element Methods are no longer applicable, and one must resort to Finite Element Methods, of which few open source solvers are yet available (SimBio-FEM [21]). Nevertheless, for the head models commonly used in practice, OpenMEEG represents the state of the art for forward computation.

Acknowledgments

The authors acknowledge support from the ANR Grant ViMAGINE ANR-08-BLAN-0250-02 and for E. Olivi from a Ph.D. grant from the Regional Council of Provence Alpes Cote d'Azur. Help is gratefully acknowledged from C. Micheli and R. Oostenveld regarding the integration of OpenMEEG within Fieldtrip. Figures 2, 3, and 4 were generated with Mayavi [28].

References

[1] C. Ramon, P. H. Schimpf, and J. Haueisen, "Influence of head models on EEG simulations and inverse source localizations," *BioMedical Engineering Online*, vol. 5, no. 5, 2006.

[2] N. Von Ellenrieder, C. H. Muravchik, and A. Nehorai, "Effects of geometric head model perturbations on the EEG forward and inverse problems," *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 3, pp. 421–429, 2006.

[3] R. Van Uitert, C. Johnson, and L. Zhukov, "Influence of head tissue conductivity in forward and inverse magnetoencephalographic simulations using realistic head models," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 12, pp. 2129–2137, 2004.

[4] A. C. Barnard, I. M. Duck, and M. S. Lynn, "The application of electromagnetic theory to electrocardiology. I. Derivation of the integral equations," *Biophysical Journal*, vol. 7, no. 5, pp. 443–462, 1967.

[5] A. C. Barnard, I. M. Duck, M. S. Lynn, and W. P. Timlake, "The application of electromagnetic theory to electrocardiology. II. Numerical solution of the integral equations," *Biophysical Journal*, vol. 7, no. 5, pp. 463–491, 1967.

[6] D. B. Geselowitz, "On bioelectric potentials in an inhomogeneous volume conductor," *Biophysics Journal*, vol. 7, pp. 1–11, 1967.

[7] J. C. De Munck, "A linear discretization of the volume conductor boundary integral equation using analytically integrated elements," *IEEE Transactions on Biomedical Engineering*, vol. 39, no. 9, pp. 986–990, 1992.

[8] J. C. Mosher, R. M. Leahy, and P. S. Lewis, "EEG and MEG: forward solutions for inverse methods," *IEEE Transactions on Biomedical Engineering*, vol. 46, no. 3, pp. 245–259, 1999.

[9] N. G. Gençer and I. O. Tanzer, "Forward problem solution of electromagnetic source imaging using a new BEM formulation with high-order elements," *Physics in Medicine and Biology*, vol. 44, no. 9, pp. 2275–2287, 1999.

[10] M. S. Hamalainen and J. Sarvas, "Realistic conductivity geometry model of the human head for interpretation of neuromagnetic data," *IEEE Transactions on Biomedical Engineering*, vol. 36, no. 2, pp. 165–171, 1989.

[11] J. Kybic, M. Clerc, T. Abboud, O. Faugeras, R. Keriven, and T. Papadopoulo, "A common formalism for the integral formulations of the forward EEG problem," *IEEE Transactions on Medical Imaging*, vol. 24, no. 1, pp. 12–28, 2005.

[12] J. Kybic, M. Clerc, O. Faugeras, R. Keriven, and T. Papadopoulo, "Fast multipole acceleration of the MEG/EEG boundary element method," *Physics in Medicine and Biology*, vol. 50, no. 19, pp. 4695–4710, 2005.

[13] J. Kybic, M. Clerc, O. Faugeras, R. Keriven, and T. Papadopoulo, "Generalized head models for MEG/EEG: boundary element method beyond nested volumes," *Physics in Medicine and Biology*, vol. 51, no. 5, pp. 1333–1346, 2006.

[14] S. Gonçalves, J. C. De Munck, J. P.A. Verbunt, R. M. Heethaar, and F. H. Lopes da Silva, "In vivo measurement of the brain and skull resistivities using an EIT-based method and the combined analysis of SEF/SEP data," *IEEE Transactions on Biomedical Engineering*, vol. 50, no. 9, pp. 1124–1128, 2003.

[15] C. H. Wolters, S. Lew, R. S. MacLeod, and M. S. Hämäläinen, "Combined EEG/MEG source analysis using calibrated finite elementhead models," in *Proceedings of the 4th Annual Conference of the German Society for Biomedical Engineering (DGBMT '10)*, Rostock-Warnemünde, Germany, October 2010.

[16] S. Vallaghé and M. Clerc, "A global sensitivity analysis of three- and four-layer EEG conductivity models," *IEEE Transactions on Biomedical Engineering*, vol. 56, no. 4, pp. 988–995, 2009.

[17] M. Clerc, J.-M. Badier, G. Adde, J. Kybic, and T. Papadopoulo, "Boundary element formulation for electrical impedance tomography," in *ESAIM: Proceedings*, vol. 14, pp. 63–71, EDP Sciences, 2005.

[18] A. Gramfort, T. Papadopoulo, E. Olivi, and M. Clerc, "OpenMEEG: opensource software for quasistatic bioelectromagnetics," *BioMedical Engineering Online*, vol. 9, article 45, 2010.

[19] J. W. H. Meijs, O. W. Weier, M. J. Peters, and A. Van Oosterom, "On the numerical accuracy of the boundary element method," *IEEE Transactions on Biomedical Engineering*, vol. 36, no. 10, pp. 1038–1049, 1989.

[20] C. Phillips, *Source estimation in eeg*, Ph.D. dissertation, Université de Liège, 2000.

[21] SimBio Development Group, "SimBio: a generic environment forbio-numerical simulations," November 2010, https://www.mrt.uni-jena.de/simbio.

[22] T. F. Oostendorp and A. Van Oosterom, "Source parameter estimation in inhomogeneous volume conductors of arbitrary shape," *IEEE Transactions on Biomedical Engineering*, vol. 36, no. 3, pp. 382–391, 1989.

[23] M. Stenroos, V. Mäntynen, and J. Nenonen, "A Matlab library for solving quasi-static volume conduction problems using the boundary element method," *Computer Methods and Programs in Biomedicine*, vol. 88, no. 3, pp. 256–263, 2007.

[24] F. Zanow and T. R. Knösche, "ASA—advanced source analysis of continuous and event-related EEG/MEG signals," *Brain Topography*, vol. 16, no. 4, pp. 287–290, 2004.

[25] C. H. Wolters, H. Köstler, C. Möller, J. Härdtlein, L. Grasedyck, and W. Hackbusch, "Numerical Mathematics of the subtraction method for the modeling of a current dipole in EEG source reconstruction using finite element head models," *SIAM Journal on Scientific Computing*, vol. 30, no. 1, pp. 24–45, 2007.

[26] M. Dannhauer, B. Lanfer, C. H. Wolters, and T. R. Knösche, "Modeling of the humanskull in EEG source analysis," *Human Brain Mapping*. In press.

[27] S. Lew, C. H. Wolters, T. Dierkes, C. Röer, and R. S. MacLeod, "Accuracy and run-time comparison for different potential approaches and iterative solvers in finite element method based EEG source analysis," *Applied Numerical Mathematics*, vol. 59, no. 8, pp. 1970–1988, 2009.

[28] Prabhu Ramachandran and Gael Varoquaux, "Mayavi: 3D Visualization of Scientific Data," *Computing in Science & Engineering*, vol. 13, no. 2, pp. 40–51, 2011.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 406391, 7 pages
doi:10.1155/2011/406391

Research Article

PyEEG: An Open Source Python Module for EEG/MEG Feature Extraction

Forrest Sheng Bao,[1] Xin Liu,[2] and Christina Zhang[3]

[1] *Department of Computer Science, Department of Electrical Engineering, Texas Tech University, Lubbock TX 79409-3104, USA*
[2] *ECHO Labs, Nanjing University of Posts and Telecommunications, Nanjing 210003, China*
[3] *Department of Physiology, McGill University, Montreal, QC, Canada H3G 1Y6*

Correspondence should be addressed to Forrest Sheng Bao, forrest.bao@gmail.com

Received 31 August 2010; Revised 26 October 2010; Accepted 31 December 2010

Academic Editor: Sylvain Baillet

Computer-aided diagnosis of neural diseases from EEG signals (or other physiological signals that can be treated as time series, e.g., MEG) is an emerging field that has gained much attention in past years. Extracting features is a key component in the analysis of EEG signals. In our previous works, we have implemented many EEG feature extraction functions in the Python programming language. As Python is gaining more ground in scientific computing, an open source Python module for extracting EEG features has the potential to save much time for computational neuroscientists. In this paper, we introduce PyEEG, an open source Python module for EEG feature extraction.

1. Introduction

Computer-aided diagnosis based on EEG has become possible in the last decade for several neurological diseases such as Alzheimer's disease [1, 2] and epilepsy [3, 4]. Implemented systems can be very useful in the early diagnosis of those diseases. For example, traditional epilepsy diagnosis may require trained physicians to visually screen lengthy EEG records whereas computer-aided systems can shorten this time-consuming procedure by detecting and picking out EEG segments of interest to physicians [5, 6]. On top of that, computers can extend our ability to analyze signals. Recently, researchers have developed systems [3, 4, 7, 8] that can hopefully use (any) random interictal (i.e., non-seizure) EEG records for epilepsy diagnosis in instances that are difficult for physicians to make diagnostic decisions with their naked eyes. In addition to analyzing existing signals, this computer-based approach can help us model the brain and predict future signals, for example, seizure prediction [9, 10].

All the above systems rely on characterizing the EEG signal into certain features, a step known as feature extraction. EEG features can come from different fields that study time series: power spectral density from signal processing, fractal dimensions from computational geometry, entropies from information theory, and so forth. An open source tool that can extract EEG features would benefit the computational neuroscience community since feature extraction is repeatedly invoked in the analysis of EEG signals. Because of Python's increasing popularity in scientific computing, and especially in computational neuroscience, a Python module for EEG feature extraction would be highly useful. In response, we have developed PyEEG, a Python module for EEG feature extraction, and have tested it in our previous epileptic EEG research [3, 8, 11].

Compared to other popular programming languages in scientific computing such as C++ or MATLAB, Python is an open source scripting language of simple syntax and various high-level libraries (for detailed advantages of Python, read http://www.python.org/about/), such as Scipy (http://www.scipy.org/) which allows users to run MATLAB codes after slight modification. There have been several popular open source Python projects in the neuroimaging community already, such as NIPY (http://nipy.sourceforge.net/). However, in neural physiology community, Python is not yet quite popular. As we are not aware of any open source tools in Python (or other programming languages) that can extract

EEG features as mentioned above, we introduce and release PyEEG in this paper.

Though originally designed for EEG, PyEEG can also be used to analyze other physiological signals that can be treated as time series, especially MEG signals that represent the magnetic fields induced by currents of neural electrical activities.

The rest of the paper is organized as follows. In Section 2, we introduce the framework of PyEEG. Section 3 gives the definitions to compute EEG features. A tutorial of applying PyEEG onto a public real EEG dataset is given in Section 4. Section 5 concludes the paper.

2. Main Framework

PyEEG's target users are programmers (anyone who writes programs) working on computational neuroscience. Figure 1 shows its framework. PyEEG is a Python module that focuses only on extracting features from EEG/MEG segments. Therefore, it does not contain functions to import data of various formats or export features to a classifier. This is due to the modularity and composition principles of building open source software which indicate that small programs that can work well together via simple interfaces are better than big monolithic programs. Since open source tools like EEG/MEG data importers (e.g., EEGLab, Biosig, etc.) and classifier front-ends are already available, there is no need for us to reinvent the wheel. Users can easily hook PyEEG up with various existing open source software to build toolchains for their EEG/MEG research.

PyEEG consists of two sets of functions.

(1) *Preprocessing functions*, which do not return any feature values. Only two such functions have been implemented so far. `embed_seq()` builds embedding sequence (from given lag and embedding dimension) and `first_order_diff()` computes first-order differential sequence. One can build differential sequences of higher orders by repeatedly applying first-order differential computing.

(2) *Feature extraction functions*, that return feature values. These are listed in Table 1.

PyEEG only uses functions in standard Python library and SciPy, the *de facto* Python module for scientific computing. PyEEG does not define any new data structure, but instead uses only standard Python and NumPy data structures. The reason is that we want to simplify the use of PyEEG, especially for users without much programming background. The inputs of all functions are a time sequence as a list of floating-point numbers and a set of optional feature extraction parameters. Parameters have default values. The output of a feature extraction function is a floating-point number if the feature is a scalar or a list of floating-point numbers (a vector) otherwise. Details about functions are available in the PyEEG reference guide at http://PyEEG.SourceForge.net/.

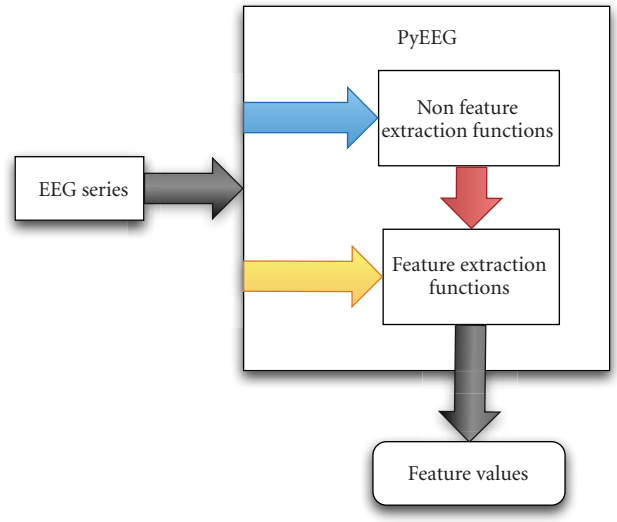


Figure 1: PyEEG framework.

3. Supported Feature Extraction

In this section, we detail the definitions and computation procedures to extract EEG features (as shown in Table 1) in PyEEG. Since there are many parameters and various algorithms for one feature, the numerical value of a feature extracted by PyEEG may be different from that extracted by other toolboxes. Users may need to adjust our code or use non-default values for the parameters in order to meet their needs. Please note that the index of an array or a vector starts from 1 rather than 0 in this section.

3.1. Power Spectral Intensity and Relative Intensity Ratio. To a time series $[x_1, x_2, \ldots, x_N]$, denote its Fast Fourier Transform (FFT) result as $[X_1, X_2, \ldots, X_N]$. A continuous frequency band from f_{low} to f_{up} is sliced into K bins, which can be of equal width or not. Boundaries of bins are specified by a vector $band = [f_1, f_2, \ldots, f_K]$, such that the lower and upper frequencies of the ith bin are f_i and f_{i+1}, respectively. Commonly used unequal bins are EEG/MEG rhythms, which are, δ (0.5–4 Hz), θ (4–7 Hz), α (8–12 Hz), β (12–30 Hz), and γ (30–100 Hz). For these bins, we have $band = [0.5, 4, 7, 12, 30, 100]$.

The Power Spectral Intensity (PSI) [12] of the kth bin is evaluated as

$$\text{PSI}_k = \sum_{i=N(f_k/f_s)}^{N(f_{k+1}/f_s)} |X_i|, \quad k = 1, 2, \ldots, K-1, \tag{1}$$

where f_s is the sampling rate, and N is the series length.

Relative Intensity Ratio (RIR) [12] is defined on top of PSI

$$\text{RIR}_j = \frac{\text{PSI}_j}{\sum_{k=1}^{K-1} \text{PSI}_k}, \quad j = 1, 2, \ldots, K-1. \tag{2}$$

PSI and RIR are both vector features.

Table 1: PyEEG-supported features and extraction functions with their return types.

Feature name	Function name	Return type
Power Spectral Intensity (PSI) and Relative Intensity Ratio (RIR)	`bin_power()`	Two 1-D vectors
Petrosian Fractal Dimension (PFD)	`pdf()`	A scalar
Higuchi Fractal Dimension (HFD)	`hfd()`	A scalar
Hjorth mobility and complexity	`hjorth()`	Two scalars
Spectral Entropy (Shannon's entropy of RIRs)	`spectral_entropy()`	A scalar
SVD Entropy	`svd_entropy()`	A scalar
Fisher Information	`fisher_info()`	A scalar
Approximate Entropy (ApEn)	`ap_entropy()`	A scalar
Detrended Fluctuation Analysis (DFA)	`dfa()`	A scalar
Hurst Exponent (Hurst)	`hurst()`	A scalar

3.2. Petrosian Fractal Dimension (PFD). To a time series, PFD is defined as

$$\mathrm{PFD} = \frac{\log_{10} N}{\log_{10} N + \log_{10}(N/(N + 0.4N_\delta))}, \tag{3}$$

where N is the series length, and N_δ is the number of sign changes in the signal derivative [13]. PFD is a scalar feature.

3.3. Higuchi Fractal Dimension (HFD). Higuchi's algorithm [14] constructs k new series from the original series $[x_1, x_2, \ldots, x_N]$ by

$$x_m, x_{m+k}, x_{m+2k}, \ldots, x_{m+\ (N-m)/k\ k}, \tag{4}$$

where $m = 1, 2, \ldots, k$.

For each time series constructed from (4), the length $L(m, k)$ is computed by

$$L(m,k) = \frac{\sum_{i=2}^{(N-m)/k} \left| x_{m+ik} - x_{m+(i-1)k} \right| (N-1)}{(N-m)/k\ k}. \tag{5}$$

The average length is computed as $L(k) = [\sum_{i=1}^{k} L(i,k)]/k$.

This procedure repeats $k_{\max}$ times for each k from 1 to $k_{\max}$, and then uses a least-square method to determine the slope of the line that best fits the curve of $\ln(L(k))$ versus $\ln(1/k)$. The slope is the Higuchi Fractal Dimension. HFD is a scalar feature.

3.4. Hjorth Parameters. To a time series $[x_1, x_2, \ldots, x_N]$, the Hjorth mobility and complexity [15] are, respectively, defined as $\sqrt{M2/\mathrm{TP}}$ and $\sqrt{(M4 \cdot \mathrm{TP})/(M2 \cdot M2)}$, where $\mathrm{TP} = \sum x_i/N$, $M2 = \sum d_i/N$, $M4 = \sum (d_i - d_{i-1})^2/N$, and $d_i = x_i - x_{i-1}$. Hjorth mobility and complexity are both scalar features.

3.5. Spectral Entropy. The spectral entropy [16] is defined as follows

$$H = -\frac{1}{\log(K)} \sum_{i=1}^{K} \mathrm{RIR}_i \log \mathrm{RIR}_i, \tag{6}$$

where RIR_i and K are defined in (2). Spectral entropy is a scalar feature.

3.6. SVD Entropy. Reference [17] defines an entropy measure using Singular Value Decomposition (SVD). Let the input signal be $[x_1, x_2, \ldots, x_N]$. We construct delay vectors as

$$\mathbf{y}(i) = [x_i, x_{i+\tau}, \ldots, x_{i+(d_E-1)\tau}], \tag{7}$$

where τ is the delay and d_E is the embedding dimension. In this paper, $d_E = 20$ and $\tau = 2$. The embedding space is then constructed by

$$Y = [\mathbf{y}(1), \mathbf{y}(2), \ldots, \mathbf{y}(N - (d_E - 1)\tau)]^T. \tag{8}$$

The SVD is then performed on matrix Y to produce M singular values, $\sigma_1, \ldots, \sigma_M$, known as the singular spectrum.

The SVD entropy is then defined as

$$H_{\mathrm{SVD}} = -\sum_{i=1}^{M} \overline{\sigma}_i \log_2 \overline{\sigma}_i, \tag{9}$$

where M is the number of singular values and $\overline{\sigma}_1, \ldots, \overline{\sigma}_M$ are normalized singular values such that $\overline{\sigma}_i = \sigma_i / \sum_{j=1}^{M} \sigma_j$. SVD entropy is a scalar feature.

3.7. Fisher Information. The Fisher information [18] can be defined in normalized singular spectrum used in (9)

$$I = \sum_{i=1}^{M-1} \frac{(\overline{\sigma}_{i+1} - \overline{\sigma}_i)^2}{\overline{\sigma}_i}. \tag{10}$$

Fisher information is a scalar feature.

3.8. Approximate Entropy. Approximate entropy (ApEn) is a statistical parameter to quantify the regularity of a time series [19].

ApEn is computed by the following steps.

(1) Let the input signal be $[x_1, x_2, \ldots, x_N]$.

(2) Build subsequence $x(i, m) = [x_i, x_{i+1}, \ldots, x_{i+m-1}]$ for $1 \quad i \quad N - m$, where m is the length of the subsequence. In [7], $m = 1, 2$, or 3.

(3) Let r represent the noise filter level, defined as $r = k \times \mathrm{SD}$ for $k = 0, 0.1, 0.2, \ldots, 0.9$.

(4) Build a set of subsequences $\{x(j,m)\} = \{x(j,m) \mid j \in [1..N-m]\}$, where $x(j,m)$ is defined in step 2.

(5) For each $x(i,m) \in \{x(j,m)\}$, compute

$$C(i,m) = \frac{\sum_{j=1}^{N-m} k_j}{N-m}, \tag{11}$$

where

$$k_j = \begin{cases} 1 & \text{if } |x(i,m) - x(j,m)| < r, \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

(6)

$$\text{ApEn}(m,r,N) = \frac{1}{N-M}\left[\sum_{i=1}^{N-m} \ln \frac{C(i,m)}{C(i,m+1)}\right]. \tag{13}$$

ApEn is a scalar feature.

3.9. Detrended Fluctuation Analysis. Detrended Fluctuation Analysis (DFA) is proposed in [20].

The procedures to compute DFA of a time series $[x_1, x_2, \ldots, x_N]$ are as follows.

(1) First integrate x into a new series $y = [y(1), \ldots, y(N)]$, where $y(k) = \sum_{i=1}^{k}(x_i - \overline{x})$ and $\overline{x}$ is the average of $x_1, x_2, \ldots, x_N$.

(2) The integrated series is then sliced into boxes of equal length n. In each box of length n, a least-squares line is fit to the data, representing the *trend* in that box. The y coordinate of the straight line segments is denoted by $y_n(k)$.

(3) The root-mean-square fluctuation of the integrated series is calculated by $F(n) = \sqrt{(1/N)\sum_{k=1}^{N}[y(k) - y_n(k)]^2}$, where the part $y(k) - y_n(k)$ is called detrending.

(4) The fluctuation can be defined as the slope of the line relating $\log F(n)$ to $\log n$.

DFA is a scalar feature.

3.10. Hurst Exponent. The hurst exponent (HURST) [21] is also called Rescaled Range statistics (R/S). To calculate the hurst exponent for time series $X = [x_1, x_2, \ldots, x_N]$, the first step is to calculate the accumulated deviation from the mean of time series within range T

$$X(t,T) = \sum_{i=1}^{t}(x_i - \overline{x}), \quad \text{where } \overline{x} = \frac{1}{T}\sum_{i=1}^{T} x_i, \ t \in [1..N]. \tag{14}$$

Then, $\text{R}(T)/\text{S}(T)$ is calculated as

$$\frac{\text{R}(T)}{\text{S}(T)} = \frac{\max(X(t,T)) - \min(X(t,T))}{\sqrt{(1/T)\sum_{t=1}^{T}[x(t) - \overline{x}]^2}}. \tag{15}$$

The Hurst Exponent is obtained by calculating the slope of the line produced by $\ln(\text{R}(n)/\text{S}(n))$ versus $\ln(n)$ for $n \in [2..N]$. Hurst Exponent is a scalar feature.

4. Using PyEEG on Real Data

In this section, we use PyEEG on a real EEG dataset to demonstrate its use in everyday research.

The dataset (http://epileptologie-bonn.de/cms/front_content.php?idcat=193&lang=3), from Klinik für Epileptologie, Universität Bonn, Germany [22], has been widely used in previous epilepsy research. In total, there are five sets, each containing 100 single-channel EEG segments. Each segment has 4096 samples. Data in sets A and B are extracranial EEGs from 5 healthy volunteers with eyes open and eyes closed, respectively. Sets C and D are intracranial data over interictal periods while Set E over ictal periods. Segments in D are from within the epileptogenic zone, while those in C are from the hippocampal formation of the opposite hemisphere of the brain. Sets C, D, and E are composed from EEGs of 5 patients. The data had a spectral bandwidth of 0.5–85 Hz. Please refer to [22] for more details.

Using PyEEG is like using any other Python module. Users simply need to import PyEEG and then call its functions as needed. PyEEG is provided as a single Python file. Therefore, it only needs to be downloaded and placed under a directory on Python module search paths, such as the working directory. Alternatively, `PYTHONPATH` environment variable can be set to point to the location of PyEEG.

On Python interpreter, we first import PyEEG and load the data

```
>>> import pyeeg
>>> fid = open('Z001.txt', 'r')
>>> tmp = fid.readlines()
>>> data = [float(k) for k in tmp]
```

where `Z001.txt` is the first segment in set A. The data type of `data` is `list`. After loading EEG data, we can use PyEEG to extract features as follows (using all default parameters):

```
>>> DFA = pyeeg.dfa(data)
>>> DFA
0.81450526948129354
>>> Hurst_Exponent = pyeeg.hurst(data)
>>> Hurst_Exponent
0.68053321812240675
>>> PFD = pyeeg.pfd(data)
>>> PFD
0.58651018327048932
```

Due to space limitations, we are not able to print all feature values of all EEG segments. Instead, we visualize the averages of the features (except RIR and PSI) within each of the five sets in Figure 2. Error bars represent the variances of features in each set. PSIs for five sets are plotted in Figure 3. Users can replot these pictures and get averages of features on Python interpreter by a testing script (http://code.google.com/p/pyeeg/wiki/TestScript) from our project website.

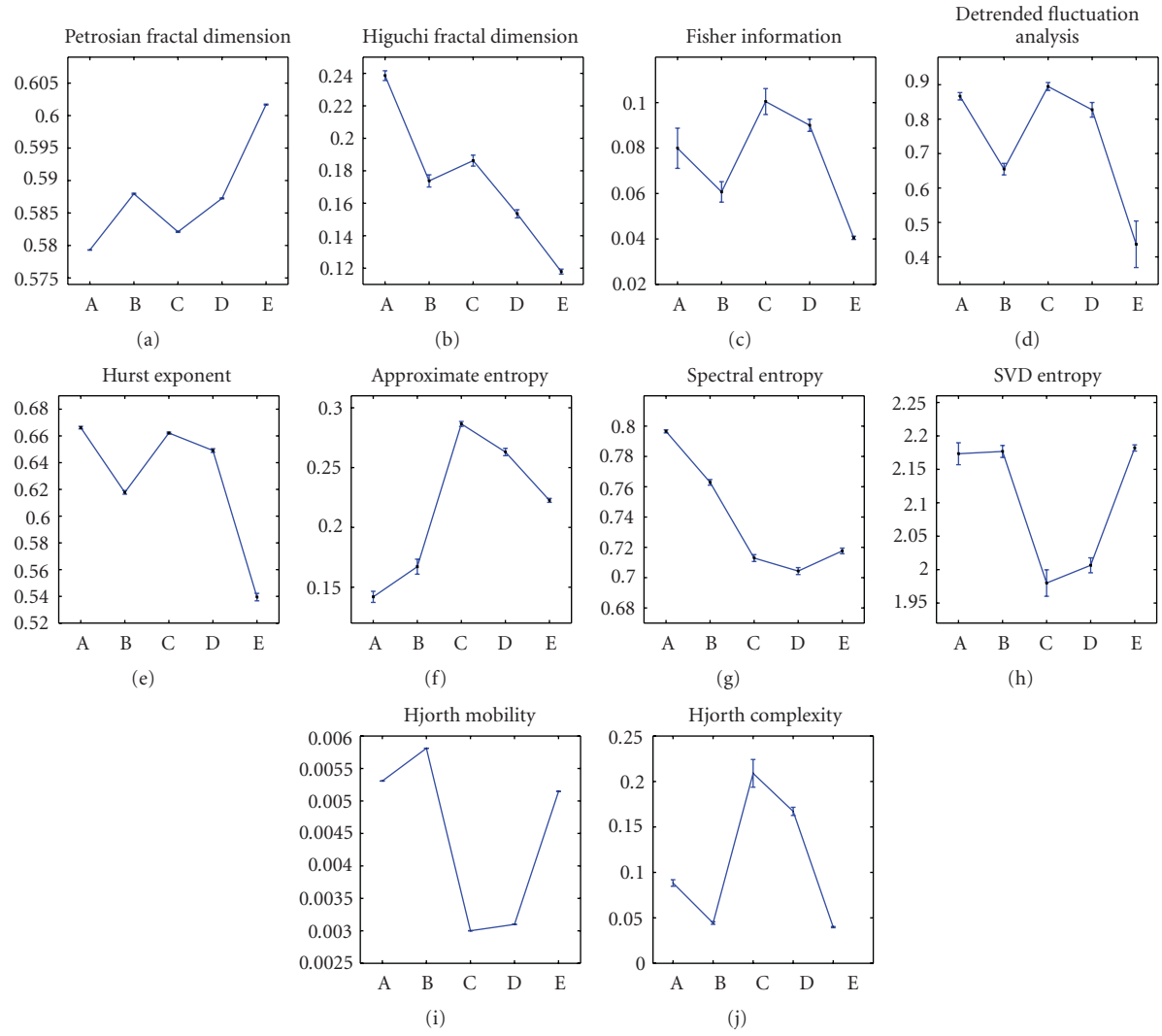


Figure 2: Distributions of ten features extracted by PyEEG in each set.

Table 2: Values of parameters used in our example.

Parameter name	Value	In feature(s)
$k_{\max}$	5	HFD
τ	4	SVD Entropy
d_E	10	Fisher Information
r	0.3σ[1]	ApEn
m	10	
f_s	173	Spectral Entropy
band	$[1, 3, 5, \ldots, 85]$	PSI and RIR

[1] σ is the standard deviation of the EEG segment.

From Figures 2 and 3, we can see that healthy, interictal, and ictal EEG signals have different distributions for most features. Table 2 lists parameters used in this experiment.

5. Discussion and Future Development

So far, we have listed features that can be extracted by PyEEG and their definitions. Our implementation sticks on their definitions precisely even though faster algorithms may exist. There are many other EEG features, such as Lyapunov Exponents, that have not been yet implemented in PyEEG. More EEG features will be added into PyEEG in the future while we finish unit testing and documentation for each function. In personal emails, some open source projects, such as ConnectomeViewer (http://www.connectomeviewer.org/viewer) and NIPY/PBrain (http://nipy.sourceforge.net/pbrain/), have expressed the interest in including PyEEG into their code. Therefore, we will keep maintaining PyEEG as long as it can benefit the entire computational neuroscience community.

Availability

The software is released under GNU GPL v.3 at Google Code: http://code.google.com/p/pyeeg/. No commercial software is required to run PyEEG. Because Python is cross-platform, PyEEG can run on Linux, Mac OS, and Windows.

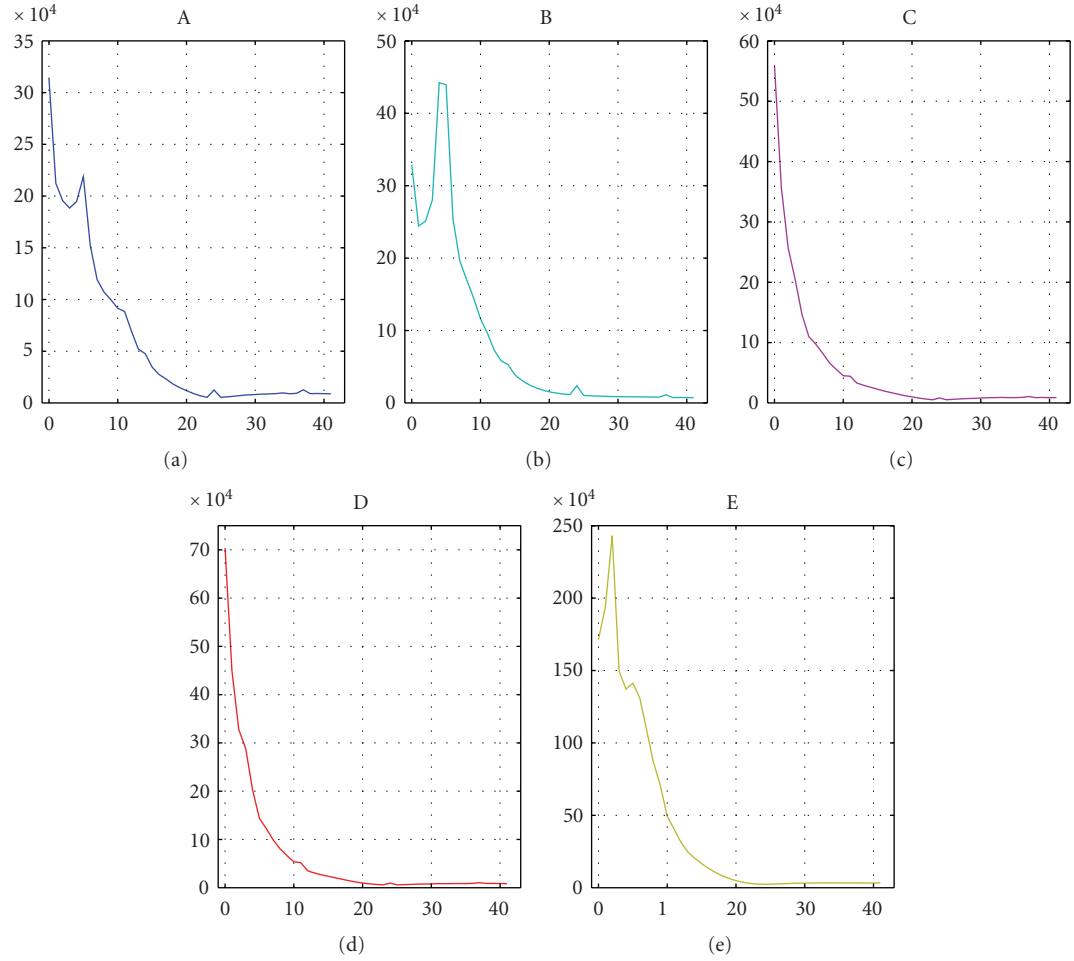


Figure 3: Average PSI of each set. Note that the scale in y-axis of set E is much larger than that of other sets.

References

[1] J. Dauwels, F. Vialatte, and A. Cichocki, "A comparative study of synchrony measures for the early detection of Alzheimer's disease based on EEG," in *Proceedings of the 14th International Conference on Neural Information Processing (ICONIP '07)*, vol. 4984 of *Lecture Notes in Computer Science*, pp. 112–125, 2008.

[2] A. A. Petrosian, D. V. Prokhorov, W. Lajara-Nanson, and R. B. Schiffer, "Recurrent neural network-based approach for early recognition of Alzheimer's disease in EEG," *Clinical Neurophysiology*, vol. 112, no. 8, pp. 1378–1387, 2001.

[3] F. S. Bao, D. Y. C. Lie, and Y. Zhang, "A new approach to automated epileptic diagnosis using EEG and probabilistic neural network," in *Proceedings of the 20th IEEE International Conference on Tools with Artificial Intelligence (ICTAI '08)*, vol. 2, pp. 482–486, November 2008.

[4] J. Dauwels, E. Eskandar, and S. Cash, "Localization of seizure onset area from intracranial non-seizure EEG by exploiting locally enhanced synchrony," in *Proceedings of the 31st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC '09)*, pp. 2180–2183, September 2009.

[5] A. B. Gardner, A. M. Krieger, G. Vachtsevanos, and B. Litt, "One-class novelty detection for seizure analysis from intracranial EEG," *Journal of Machine Learning Research*, vol. 7, pp. 1025–1044, 2006.

[6] C. W. Ko and H. W. Chung, "Automatic spike detection via an artificial neural network using raw EEG data: effects of data preparation and implications in the limitations of online recognition," *Clinical Neurophysiology*, vol. 111, no. 3, pp. 477–481, 2000.

[7] V. Srinivasan, C. Eswaran, and N. Sriraam, "Approximate entropy-based epileptic EEG detection using artificial neural networks," *IEEE Transactions on Information Technology in Biomedicine*, vol. 11, no. 3, pp. 288–295, 2007.

[8] F. S. Bao, J. M. Gao, J. Hu, D. Y. C. Lie, Y. Zhang, and K. J. Oommen, "Automated epilepsy diagnosis using interictal scalp EEG," in *Proceedings of the 31st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC '09)*, pp. 6603–6607, September 2009.

[9] K. Lehnertz, F. Mormann, T. Kreuz et al., "Seizure prediction by nonlinear EEG analysis," *IEEE Engineering in Medicine and Biology Magazine*, vol. 22, no. 1, pp. 57–63, 2003.

[10] E. O'Sullivan-Greene, I. Mareels, D. Freestone, L. Kulhmann, and A. Burkitt, "A paradigm for epileptic seizure prediction using a coupled oscillator model of the brain," in *Proceedings of the 31st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC '09)*, pp. 6428–6431, September 2009.

[11] F. S. Bao, Y.-L. Li, J.-M. Gao, and J. Hu, "Performance of dynamic features in classifying scalp epileptic interictal and normal EEG," in *Proceedings of 32nd International Conference of IEEE Engineering in Medicine and Biology Society (EMBC '10)*, 2010.

[12] R. Q. Quiroga, S. Blanco, O. A. Rosso, H. Garcia, and A. Rabinowicz, "Searching for hidden information with gabor transform in generalized tonic-clonic seizures," *Electroencephalography and Clinical Neurophysiology*, vol. 103, no. 4, pp. 434–439, 1997.

[13] A. Petrosian, "Kolmogorov complexity of finite sequences and recognition of different preictal EEG patterns," in *Proceedings of the 8th IEEE Symposium on Computer-Based Medical Systems*, pp. 212–217, June 1995.

[14] T. Higuchi, "Approach to an irregular time series on the basis of the fractal theory," *Physica D*, vol. 31, no. 2, pp. 277–283, 1988.

[15] B. Hjorth, "EEG analysis based on time domain properties," *Electroencephalography and Clinical Neurophysiology*, vol. 29, no. 3, pp. 306–310, 1970.

[16] T. Inouye, K. Shinosaki, H. Sakamoto et al., "Quantification of EEG irregularity by use of the entropy of the power spectrum," *Electroencephalography and Clinical Neurophysiology*, vol. 79, no. 3, pp. 204–210, 1991.

[17] S. J. Roberts, W. Penny, and I. Rezek, "Temporal and spatial complexity measures for electroencephalogram based brain-computer interfacing," *Medical and Biological Engineering and Computing*, vol. 37, no. 1, pp. 93–98, 1999.

[18] C. J. James and D. Lowe, "Extracting multisource brain activity from a single electromagnetic channel," *Artificial Intelligence in Medicine*, vol. 28, no. 1, pp. 89–104, 2003.

[19] S. M. Pincus, I. M. Gladstone, and R. A. Ehrenkranz, "A regularity statistic for medical data analysis," *Journal of Clinical Monitoring and Computing*, vol. 7, no. 4, pp. 335–345, 1991.

[20] C.-K. Peng, S. Havlin, H. E. Stanley, and A. L. Goldberger, "Quantification of scaling exponents and crossover phenomena in nonstationary heartbeat time series," *Chaos*, vol. 5, no. 1, pp. 82–87, 1995.

[21] T. Balli and R. Palaniappan, "A combined linear & nonlinear approach for classification of epileptic EEG signals," in *Proceedings of the 4th International IEEE/EMBS Conference on Neural Engineering (NER '09)*, pp. 714–717, May 2009.

[22] R. G. Andrzejak, K. Lehnertz, F. Mormann, C. Rieke, P. David, and C. E. Elger, "Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: dependence on recording region and brain state," *Physical Review E*, vol. 64, no. 6, Article ID 061907, 8 pages, 2001.

Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2011, Article ID 674605, 8 pages
doi:10.1155/2011/674605

Research Article

TopoToolbox: Using Sensor Topography to Calculate Psychologically Meaningful Measures from Event-Related EEG/MEG

Xing Tian,[1] David Poeppel,[1] and David E. Huber[2]

[1] *Department of Psychology, New York University, 6 Washington Place Suite 275, New York, NY 10003, USA*
[2] *Department of Psychology, University of California, San Diego, La Jolla, CA 92093, USA*

Correspondence should be addressed to Xing Tian, xing.tian@nyu.edu

Received 2 September 2010; Revised 7 December 2010; Accepted 3 February 2011

Academic Editor: Sylvain Baillet

The open-source toolbox "TopoToolbox" is a suite of functions that use sensor topography to calculate psychologically meaningful measures (similarity, magnitude, and timing) from multisensor event-related EEG and MEG data. Using a GUI and data visualization, TopoToolbox can be used to calculate and test the topographic similarity between different conditions (Tian and Huber, 2008). This topographic similarity indicates whether different conditions involve a different distribution of underlying neural sources. Furthermore, this similarity calculation can be applied at different time points to discover when a response pattern emerges (Tian and Poeppel, 2010). Because the topographic patterns are obtained separately for each individual, these patterns are used to produce reliable measures of response magnitude that can be compared across individuals using conventional statistics (Davelaar et al. Submitted and Huber et al., 2008). TopoToolbox can be freely downloaded. It runs under MATLAB (The MathWorks, Inc.) and supports user-defined data structure as well as standard EEG/MEG data import using EEGLAB (Delorme and Makeig, 2004).

1. Introduction

This tutorial introduces a free open-source toolbox that includes functions for topographic analyses of event-related electrophysiological data (EEG/MEG). These analyses do not anatomically locate neural sources. Instead, by providing robust measures of response similarity between conditions and response magnitude for each condition, multivariate analyses are used to test psychological theories. These techniques are not new and were previously proposed and validated [1, 2]. However, their implementation within a user friendly toolbox is new. The core routines of TopoToolbox calculate a measure of angle between EEG/MEG topographies in n-dimensional sensor space, where n is the number of sensors. This toolbox is called TopoToolbox and it uses MATLAB (The MathWorks, Inc.) to analyze either user-defined or EEGLAB [3] standardized data sets. It can be downloaded from https://files.nyu.edu/xt235/public/ where a detailed tutorial, manual, and example data can be found.

Multivariate methods are frequently used to analyze fMRI experiments [4–8], and similar multivariate methods are beginning to appear in EEG/MEG studies. However, unlike fMRI studies in which multivariate analyses involve multiple anatomically defined voxels, multivariate analyses in EEG/MEG involve multiple sensors (e.g., electrodes or SQUID magnetometers) that each reflect a mixture of underlying neural sources. Thus, for EEG/MEG, these analyses are often in sensor space rather than source space.

We briefly review several previously proposed EEG/MEG multivariate analysis methods. Several of these are closely related to the analyses contained in TopoToolbox, and we further consider these relations in the Discussion. Global field power (GFP; [9]) was one of the first measures to use multiple sensors in EEG data. GFP is the standard deviation of all sensors from the global mean. Lehmann and Skrandies [9] also proposed a topographic measure termed global dissimilarity (DISS), which is the square root of the mean of the squared differences between the

sensors after first scaling the sensor values in each condition by dividing by the GFP of that condition (i.e., Euclidean distance between the two sensor vectors after normalizing them to have length 1.0). A nonparametric method called TANOVA (topographic ANOVA) has been proposed to statistically test the significance of the DISS value between two grand average topographies by calculating a null hypothesis distribution from repeated random permutations of the data [10–12]. Similar to the analyses contained in TopoToolbox, DISS is a measure based on the sensor space. In contrast, some recent multivariate analyses have been developed that transform the data of multi-sensor event-related EEG/MEG experiments using a basis set, such as with independent components analysis (e.g., [13] for a review see [14]). Thus, these techniques operate in component space, where each component is a derived topographic pattern, rather than performing tests based on the raw topographic sensor space.

Compared with traditional waveform-based analyses, topographic analyses have the following advantages. First, topographic analyses use all of the data in a single test and do not suffer from problems related to "double dipping" that can occur with multiple comparisons [15, 16]. Second, EEG waveform analyses are highly dependent on reference channel selection (see the review by Murray et al. [17]) and MEG waveform analyses are difficult to combine across individuals in sensor space due to large differences in the response of the same sensor for different individuals [18]. Third, waveform analyses cannot determine whether a change between conditions is more likely due to a change in neural response magnitude or a change in the distribution of underlying neural sources that gave rise to the response [1]. Even if the goal is anatomical localization, analyses based on sensor topography can provide an important intermediate step and validity check prior to source analyses. Furthermore, multivariate analyses can be used to test psychological theories (e.g., "how") in the absence of anatomical localization (e.g., "where").

There are a range of techniques that use multiple sensors to anatomically locate neural responses [19]. However, these techniques often make strong assumptions such as temporal and anatomical independence between the underlying neural sources. Working within a component space based on sensor topography, rather than source space, independent components analysis [20] has proven useful for extracting independent noise components such as the beating heart or eye blinks [21]. Pascual-Marqui and colleagues [22] proposed a data-driven, hypothesis-free topographic analysis of electrophysiology that blindly separates the grand average into different response components by using multiple spatial templates as applied to each individual data set.

Aside from the choice of analyzing a select few sensors versus the entire multivariate sensor topography, another choice in electrophysiological experiments is whether to analyze each individual separately versus the entire data set across all participants. Because individuals differ both in terms of anatomical structure and in terms of task related neural responses [23, 24], averaging across individuals can produce unreliable results, particularly with MEG data (see [1] for a reliability comparison between a sensor selection analysis and the projection test contained in the TopoToolbox). However, if the goal is to infer something about the adult population in general, then it is necessary to use a statistical test with subject as a random factor. If spatial and temporal individual differences are not considered when averaging across individuals, at best this will reduce the signal-to-noise ratio and at worst it might bias the results. The aforementioned multivariate methods do not provide measures that can be compared across individuals in a reliable manner in light of these individual differences. One of the primary advantages of the TopoToolbox is its ability to normalize against individual differences and derive a single magnitude measure that is psychologically meaningful [1]. To date, this method has been successfully applied to MEG data across a variety of experimental paradigms [1, 2, 25, 26], demonstrating its ability to produce reliable measures that can be compared across individuals.

2. Methods

This section describes the equations and algorithms implemented in the TopoToolbox. Some details are omitted, such as navigation of the menus in the toolbox and particular parameter selections. Descriptions of these details and example data can be downloaded from https://files.nyu.edu/xt235/public/. The core of the toolbox is a two-stage analysis that first quantifies the topographic similarity and second quantifies response magnitude through topographic projection. In this tutorial, we also describe a new addition to the toolbox that assesses dynamic variations in the observed topography.

For the first stage (*angle test*), similarity measures are calculated between the results of different conditions. Significant dissimilarity indicates that the observed pattern across the sensors qualitatively changed, such as what might occur with differing mixtures of underlying neural sources. For instance, if one condition evokes a response in auditory cortex while another condition evokes a response in visual cortex, then this analysis would conclude that the patterns were dissimilar even when measured at the same latency. However, if the patterns are not found to be dissimilar, then the second stage calculates geometric projections between patterns, which are used to indicate whether there has been a change in response magnitude (i.e., more or less of the pattern). This is done separately for each individual based on that individual's "template" response. Because these projections are normalized for each individual, the conclusion of this second stage is a statistical test across individuals.

Because these methods use traditional null hypothesis testing (in future work, we plan to supplement the TopoToolbox with Bayesian statistics), a failure to find a significant difference with the angle test does not necessarily indicate that the conditions of interest did not differ (i.e., there is an unknown type II error rate). Furthermore, if exceedingly unlucky, two different distributions of neural sources can in theory produce exactly the same topographic pattern (e.g., an inverse problem). Putting aside this remote possibility, the two tests can be used in combination to determine whether the best interpretation of a change between conditions is a

change in the distribution of neural sources versus a change in response magnitude. More specifically, because both tests operate on the same data, they have equivalent statistical power, and a result in which the angle test fails to find a difference but the projection test produces a significant difference supports the conclusion that there was a change in magnitude.

2.1. Angle Test: Topographic Similarity. The topographic analyses in TopoToolbox assume that each of the n sensors provides a unique dimension of variation. Thus, the 2D or 3D spatial arrangement of the sensors is irrelevant. Instead, all sensors are equally important regardless of their position. The n-dimensional spatial patterns across sensors for different experimental conditions (e.g., the pattern for condition X_1 versus the pattern for condition X_2) are first assessed with an *angle test*. The n-dimensional sensor space angle (θ) is calculated to measure similarity between these patterns (see Figure 1 for an example with 2 sensors, which is the largest number of sensors that can be accurately portrayed on the written page). If the two conditions produce a similar distribution of neural sources, then the angle in sensor space will be 0 degrees even if one condition produces a larger response magnitude than the other condition. However, if the two patterns are completely opposite (i.e., sign flip), then the angle is π. The angle is measured by calculating the cosine of the angle, which is a normalized dot product between the two sensor vectors (1). If the sensor data are zero centered (e.g., using an average reference channel), this is formally the same as the Pearson correlation coefficient. We term this cosine angle the *angle measure*. The *angle measure* ranges from −1 to 1, where −1 is observed for completely opposite patterns and 1 is observed for the perfectly similar patterns (regardless of magnitude). Because this *angle measure* is calculated between conditions, we term it the *between* angle measure:

$$\cos\theta = \frac{\vec{X}_1^{\,T}\,\vec{X}_2}{\left|\vec{X}_1\right|\left|\vec{X}_2\right|}. \tag{1}$$

A null hypothesis is needed to statistically assess the *between* angle measure (i.e., is the angle between conditions greater than expected based on chance). There may be other methods for constructing a null hypothesis, but a simple solution is to separate the experiment into two halves and then calculate *between* versus *within* angle measures based on the average patterns found for each half, condition, and individual. The *angle measure* is calculated separately at each evoked time point, and then these separate *angle measures* are averaged over a temporal window to increase reliability. Both the beginning and the end point of the averaging window can be set. In particular, the middle of the averaging window can be adjusted separately for each individual considering that different people tend to produce waveforms that achieve peak values at different times (see [1] for evidence of individual differences in the duration to reach a peak response). If the separation of the experiment into two halves is done according to trial number (first versus second half of the experimental session), this produces a null hypothesis that includes variance due to changes over time, such as what might occur with head position shifts. However, the toolbox also allows that the separation into halves can be done in an interleaved manner (odd number trials versus even number trials) or through a random split of trials. The null hypothesis is based on the *within* angle measure that compares the responses between the two halves for the same condition for each individual whereas the *between* angle measure compares the responses between the two halves for different conditions for each individual.

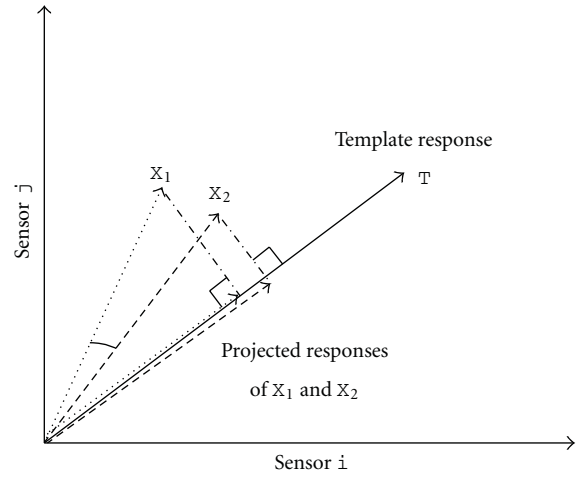


Figure 1: Illustration of *angle test* and *projection test* with 2 sensors, although the technique is typically applied to an n-dimensional sensor space where n is the number of sensors. Each experimental condition (X_1 and X_2) produces a magnitude of response for each sensor. The angle (θ) between these conditions is used as a measure of pattern similarity. Some other condition defines a template pattern (T), and projections onto this template provide numbers for "how much" of the template each condition produced. These response magnitudes can then be compared across individuals.

To understand the nature of these calculations, consider a comparison between two conditions (X_1 and X_2) across the experimental halves (a and b) with 10 individuals in the experiment. The null hypothesis *within* angle measure for the first individual is found by averaging the X_{1a}/X_{1b} angle measure with the X_{2a}/X_{2b} angle measure, and the experimental *between* angle measure is found by averaging the X_{1a}/X_{2b} angle measure with the X_{2a}/X_{1b} angle measure. These same values are calculated for the other 9 individuals, and then differences between the 10 *within* and 10 *between* angle measures are statistically assessed. It is important to note that although a direct measure of angle would require a statistical test designed for circular data, the angle measure in TopoToolbox is the cosine angle, which is a noncircular interval scale for the null hypothesis of no difference. Furthermore, if the sensor data are zero-centered, then the angle measure is the same as the Pearson correlation coefficient, which is traditionally tested using a t distribution. Therefore, TopoToolbox uses a paired t-test to determine if the response patterns were significantly dissimilar across the population for the two experimental conditions.

If the *between* angle measure is significantly smaller than the *within* angle measure, then the two experimental conditions are significantly dissimilar, leading to an unambiguous conclusion that a different mix of neural sources was responsible for the change between conditions. Furthermore, such a result suggests that the projection test, described next, should not be run and would produce ambiguous results because it confounds response magnitude with response similarity. Alternatively, the failure to conclude that the two conditions are significantly dissimilar implies that (a) a similar distribution of neural sources produced the response pattern in both conditions (b) that two different distributions of neural sources happened to produce the same topographic pattern (a remote possibility), or (c) that the *t*-test was not sufficiently powerful to detect dissimilarity. The question of statistical power can be addressed with the projection test. More specifically, if the projection test concludes that the response magnitude is significantly different between the two conditions, this suggests that there was sufficient power to have detected a difference in response similarity.

2.2. The Projection Test: Normalizing against a Template to Measure Response Magnitude. Most event-related electrophysiological studies analyze response magnitude of a select few sensors in different conditions. For these analyses, it is tempting to conclude that increases in response magnitude (either greater positivity or greater negativity) correspond to increases in the underlying neural response. However, when considering just a few sensors, it is unclear whether an increase reflects an increase in the magnitude of the neural response or whether an increase might instead reflect a shift in the distribution of neural sources, with some new source producing the apparent increase. Simply put, the question is whether the brain did the same thing to a greater extent in one condition, or whether the brain did two different things in the two different conditions. The answer to this question can be used to distinguish between competing psychological theories. As described above, the *angle test* can be used to determine if the distribution of neural sources changed between conditions. If the conditions appear to be sufficiently similar (not significantly dissimilar), then *the projection test* can be used to determine if the magnitude of the underlying neural sources has increased or decreased.

Besides providing a conclusion based on neural response magnitude (upon failure of the *angle test*), another advantage of the *projection test* is its ability to normalize against individual differences, thus providing a more reliable measure. This is achieved by projecting (2) the sensor pattern in each condition (X_i) onto a template pattern (T) for that participant (Figure 1). As with the *angle measure*, this is done separately at each time point and window averaging is used to further increase reliability. The template is typically a response pattern across the sensors in some other condition using the same response window for averaging. Critical to the success of this projection is choice of the template. The template should include the same psychological processes as the conditions of interest (see an example below). Because a different template pattern is used for every individual, the projection values should lie on the same scale (i.e., more or less of that individual's template response). Therefore, individual topographic differences are eliminated through normalization. Furthermore, provided that the template is a "clean" pattern that is relatively devoid of overlapping waveform responses (in contrast to experimental conditions), the projection can serve to decontaminate response magnitude by eliminating overlapping waveform responses that are orthogonal to the template:

$$\left|\vec{X}_i\right| \cos\theta = \frac{\vec{T}^{T}\,\vec{X}_i}{\left|\vec{T}\right|}. \tag{2}$$

Like the *angle test*, the *projection test* is statistically tested across individuals using a paired *t*-test. However, in the case of the *projection test*, the comparison is not a *between* angle measure versus a *within* angle measure but rather the projection value in one condition versus the projection value in the other condition for each individual.

An immediate priming experiment [26] provides an example of an appropriate template response and use of the angle and projection tests. In this experiment, every trial presented a prime word for 1,850 ms followed by the appearance of a second prime word for 150 ms (both prime words remained on the screen for the final 150 ms). Next, both primes disappeared, and a target word was briefly flashed and then masked. There were three conditions depending on whether this target word repeated the long duration prime word (the *long* condition), the short duration prime word (the *short* condition), or neither of the prime words (the *novel* condition). MEG was recorded in this experiment, and the measure of interest was the M170 to the target word. However, because the short duration prime appeared 150 ms prior to target word, and because a mask followed the target word, there was substantial contamination of the M170 pattern to the target word (due to the M400 to the short duration prime word and also the M100 to the mask). In contrast, the M170 to the first prime word provided a "clean" M170 that was used as a template pattern to normalize each individual's target word M170. Because there were individual differences in the timing of the M170, each individual was given a different 22 ms template time window according to that individual's peak M170 (as determined by the root mean square across all 157 sensors).

A priming effect is defined as the difference between a primed condition (e.g., the *long* condition) and an unprimed condition (e.g., the *novel* condition). Before concluding whether priming caused the M170 to decrease or increase, the angle test was used to assess whether each priming condition was dissimilar from the unprimed condition. These *within* and *between* angle measures were calculated for each individual using the same individually specified 22 ms time window as determined by that individual's template response (except that this window was placed in relation to the onset of the target word rather than the onset of the first prime word). Because the resultant angle tests failed to find any similarity differences, the projection test was used for each priming effect. As predicted by a neural

habituation model of priming [27], these tests revealed that there was a significant neural response reduction in the target word's M170 when it repeated a long duration prime word but not when it repeated a short duration prime word [26]. Without these topographical analyses, this theoretical conclusion would not have been possible because (1) topographic differences made statistical test across individuals unreliable, (2) overlapping waveforms produced a target word M170 that was contaminated and thus unreliable, and (3) a statistical conclusion based on the magnitude of a few sensors might have confounded a change in the neural source distribution with a change in the neural response magnitude.

2.3. Angle Dynamics Test: Assessing Pattern Similarity over Time. For classic well-defined responses such as the M170 to a visual stimulus, the *angle test* and *projection test* can be used to measure similarity and response magnitude. However, in other circumstances, the waveform peaks are less well-defined and it can be difficult to determine when a certain response pattern reaches its peak and how long that pattern lasts. The dynamics of response patterns can be assessed by using the same *angle test* for similarity except that the test is applied at every time point rather than only at a well-defined peak. That is, the *angle measure* between a template and a condition of interest can be calculated at each sample time point for that condition (3). Just as with a well-defined peak, the *within* and *between* angle measures can be calculated at every time point to determine when the pattern defined by template is maximally exhibited in the condition of interest and during what time periods the template pattern does not exist:

$$\cos\theta(t) = \frac{\vec{T}^{T}\,\vec{X}_i(t)}{\left|\vec{T}\right|\left|\vec{X}_i(t)\right|}. \tag{3}$$

A simple motor experiment [2] provides an example demonstrating the usefulness of this dynamic pattern analysis. This MEG experiment investigated the temporal characteristics of the neural sources involved in motor execution and imagery although only the motor execution results are summarized here. In this experiment, participants were asked to press a button at a comfortable pace upon hearing an auditory cue. They were encouraged to respond at a similar speed throughout the entire experiment. The MEG motor response was measured both by using an average that was time locked to the auditory cue (cue locked) and by using an average that was time locked to the button press (response-locked). The *angle dynamics test* was implemented by using the response-locked motor response as a template pattern (i.e., a classically defined motor response template) that was compared to every time point in the cue-locked epoch. An important validation of this angle dynamics test was whether it could be used to recover the same peak time in the cue-locked epoch as defined using classical methodology. The classically defined peak was identified using the root mean square (RMS) across the sensors to find a peak response. However, in the cue-locked epoch it is not always clear when to look for this peak, and so an RMS peak was chosen for each individual that was near the average reaction time of that individual. The important question was whether the angle dynamics test could find these RMS-defined motor response times in the cue-locked epoch without knowing the average reaction time of each individual.

As seen in Figure 2, *between* and *within* angle measures were found at each sample point within the cue-locked epoch (using the response-locked template). This was done separately for each individual, and then these values were averaged and graphed with 95% confidence levels to produce the plots. The zero point of the x-axis is the time at which the motor response reached its peak value as classically defined by RMS. This was done separately for each individual, and time is shown relative to these individually determined peak times. As seen in Figure 2, the *between* angle measure approaches the *within* angle measure 50 ms before the RMS-defined peak latency (i.e., the zero point on the x-axis) and falls below the *within* angle measure 50 ms after the peak latency. Furthermore, beyond validating the timing of the peak time using the *angle dynamics test*, the *angle test* at the peak latency was not significantly dissimilar from the response-locked template, whereas they were significantly dissimilar 100 ms before and after the peak latency. The grand average topographies in Figure 2 further confirmed the results of the *angle dynamics test*: the cue-locked response at 0 ms shared the same distribution as the template, whereas the responses at −100 ms and 100 ms were apparently different from the template. This suggests that distribution of neural sources responsible for the motor response was different from the distribution of neural sources just before and just after the response. In contrast, during the peak time as defined by the *angle dynamics test* applied to the cue-locked epoch, the pattern across the sensors was similar to the template as defined by the response-locked epoch [2].

This is an important validation of the *angle dynamics test,* and it may prove to be of use in experiments where there is a need to find the timing of peaks that are not strictly locked to stimulus onset. For instance, consider an experiment that involves left key presses versus right key presses in a difficult task that produces many errors. Response-locked epochs could be used to define the template pattern for a left or a right key press, and then the *angle dynamics test* could be calculated for each of these templates to assess the online decision process as individuals gain more information favoring one response or the other (see [28] for a related method for assessing decision evidence accumulation in EEG data).

3. Discussion

There have been recent and exciting developments in the use of EEG and MEG analyses based on the topographic pattern across the entire sensor array [17]. Many of these techniques are highly complex and attempt to extract the responses of specific anatomically located neural sources. The techniques in TopoToolbox also use the topographic pattern across the entire sensor array to extract more information from MEG/EEG data. However, rather than attempting to measure particular neural sources, the goal of these analyses is

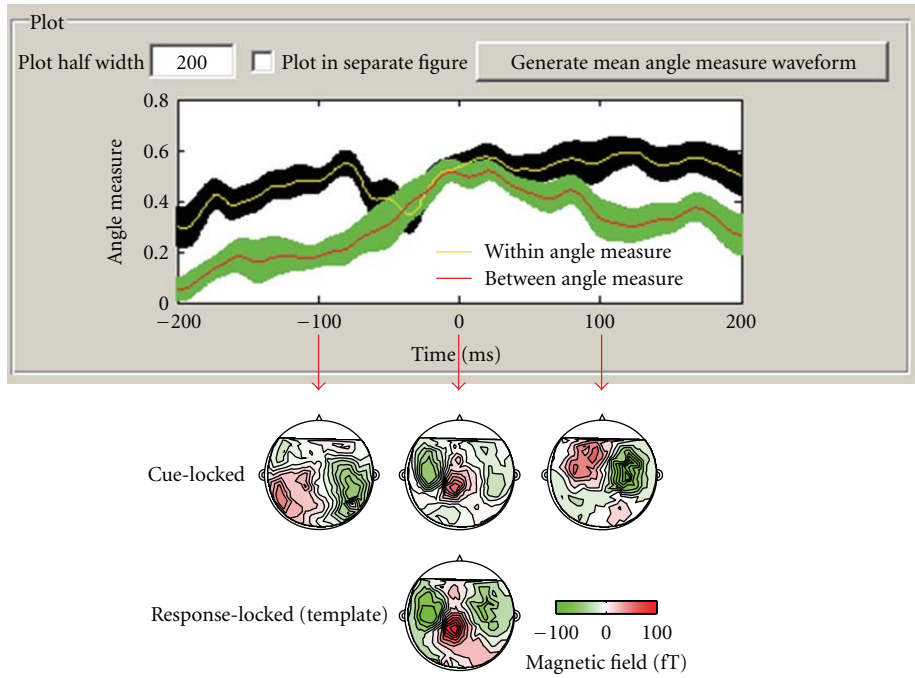


Figure 2: Results comparing the timing of motor response peaks as determined classically through the root mean square (RMS) across MEG sensors versus motor response peaks as determined by the *angle dynamics test*. The *angle dynamics test* used a template defined by the response-locked epoch, which was compared (*angle test*) at each time point along the cue-locked epoch. Because the cue-locked epoch has been adjusted according to each individual's RMS defined motor response peak, the zero point on the *x*-axis is the classically defined motor peak. Validating the *angle dynamics test*, the difference between the *within* and *between* angle measures becomes nonsignificant during a 101 ms window around the zero point. Shaded regions show the 95% confidence intervals for the *between* and *within* angle measures. The grand average of template and cue-locked responses at 3 different times are depicted on the bottom. As seen in these grand average responses, the cue-locked topography is similar to the response-locked template at the zero point but dissimilar 100 ms before and 100 ms after the zero point.

to more simply ask whether the distribution of neural sources changed between conditions and, if not, whether that distribution was more or less active. The resultant techniques are relatively simple and can be used to ask functional questions such as how (same or different from the *angle test*), how much (*projection test*), and when (*angle dynamics test*).

There are several analysis methods and associated software that are closely related to the TopoToolbox, such as TANOVA in LORETA [29, 30] and Cartool (http://brainmapping.unige.ch/cartool). Amongst the three core tests contained in the TopoToolbox, the *angle test* is the component most similar to the measures contained in these alternative software packages. In particular, although the equation for the angle measure is not identical to the equation for the DISS measure used in TANOVA, it has been proven that there is a linear relation between these two measures [31]. However, unlike application of DISS in TANOVA, the TopoToolbox calculates the angle measure separately for each participant and uses a statistical test with subject as a random factor whereas the statistical test of DISS in TANOVA tests for between-condition similarity differences in the topographies after averaging across subjects and uses nonparametric bootstrap sampling to test reliability across individuals. Beyond accounting for individual differences, another advantage of the *angle test* in TopoToolbox is that by splitting the experimental session into two halves, the null hypothesis distribution properly includes nuisance factors such as fatigue and head movements.

Beyond similarities between the angle measure of TopoToolbox and the DISS measure used in TANOVA, the TopoToolbox also contains *the projection test* and *angle dynamics test*, which are not found in other software packages. Thus, although there are other methods for assessing similarity of topographic patterns, only the TopoToolbox has a technique for determining whether the topographic pattern has increased or decreased in its response magnitude, as determined with a measure that decontaminates by using a clean template pattern, and also a technique for determining when the topographic pattern becomes most similar to a template pattern. The combination of the angle test and the projection test is particularly useful because in combination they can determine whether the best explanation for a change between conditions is that the topographic pattern changed (suggesting a different distribution of neural sources) or whether the topographic pattern magnitude changed (suggesting an increase or decrease in the neural response).

Due to the "inverse problem," it is difficult, if not impossible, to infer underlying neural sources from scalp measurements; for any topographic pattern there are infinitely many combinations of neural sources that can give rise to exactly that pattern. It is for this reason that the techniques of the TopoToolbox do not attempt to identify the underlying

neural sources. Instead, the goal of TopoToolbox is qualitative comparisons that can assess whether the distribution of underlying neural sources is likely to have changed, which would produce a different topographic pattern, and whether the distribution of underlying neural sources is likely to have increased or decreased, which would produce the same topographic pattern but a change in response magnitude for that pattern. However, there is still an inverse problem with these techniques; it is conceivable that the same topographic pattern is observed in two conditions (i.e., a failure to find dissimilar patterns with the *angle test*) even though the distributions of underlying neural sources are different. Nevertheless, the chance of this occurring would seem to be low considering that the conditions being compared are typically within the same task that involves the same cognitive processes. To partly address this question, Tian and Poeppel [2] compared the results from the *angle test* to the results of a source analysis, revealing that the source analysis suggested differently located sources when the *angle test* suggested that distribution of neural sources was different. However, source analysis also suffers from an inverse problem, and so the ideal method for validating this limitation, as well as limitations due to the use of null hypothesis testing, would be to use a "ground truth" comparison such as with intracranial EEG.

In the absence of further validation of these techniques with a comparison to intracranial EEG, we have demonstrated that *projection test* reduces variability by normalizing against individual topographic differences, individual timing of peak response differences, and contamination from overlapping waveforms [1] and we have also demonstrated that the *angle dynamics test* can recover the timing of a motor response (as reported here and also by Tian and Poeppel [2]). Most importantly, a growing number of studies have found these techniques to be reliable and useful (e.g., [2, 25, 26]).

4. Conclusions

This paper introduced a new topographically based analysis toolbox for electrophysiological studies (EEG/MEG). We demonstrated that these within-participant analyses can normalize individual differences and derive psychologically meaningful metrics (similarity, magnitude, and timing) from high-density sensor arrays in a manner that overcomes several limitations of traditional waveform analyses.

Acknowledgment

This research was supported by the National Science Foundation (NSF) under Grant BCS-0843773 and by the Department of Defense (DOD) under MURI ARO no. 54228-LS-MUR.

References

[1] X. Tian and D. E. Huber, "Measures of spatial similarity and response magnitude in MEG and scalp EEG," *Brain Topography*, vol. 20, no. 3, pp. 131–141, 2008.

[2] X. Tian and D. Poeppel, "Mental imagery of speech and movement implicates the dynamics of internal forward models," *Frontiers in Psychology*, vol. 1, p. 66, 2010.

[3] A. Delorme and S. Makeig, "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis," *Journal of Neuroscience Methods*, vol. 134, no. 1, pp. 9–21, 2004.

[4] D. Braze, W. E. Mencl, W. Tabor et al., "Unification of sentence processing via ear and eye: an fMRI study," *Cortex*, vol. 47, no. 4, pp. 416–431, 2011.

[5] N. Kriegeskorte, R. Goebel, and P. Bandettini, "Information-based functional brain mapping," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 103, no. 10, pp. 3863–3868, 2006.

[6] T. M. Mitchell, S. V. Shinkareva, A. Carlson et al., "Predicting human brain activity associated with the meanings of nouns," *Science*, vol. 320, no. 5880, pp. 1191–1195, 2008.

[7] K. A. Norman, S. M. Polyn, G. J. Detre, and J. V. Haxby, "Beyond mind-reading: multi-voxel pattern analysis of fMRI data," *Trends in Cognitive Sciences*, vol. 10, no. 9, pp. 424–430, 2006.

[8] S. M. Polyn, V. S. Natu, J. D. Cohen, and K. A. Norman, "Neuroscience: category-specific cortical activity precedes retrieval during memory search," *Science*, vol. 310, no. 5756, pp. 1963–1966, 2005.

[9] D. Lehmann and W. Skrandies, "Reference-free identification of components of checkerboard-evoked multichannel potential fields," *Electroencephalography and Clinical Neurophysiology*, vol. 48, no. 6, pp. 609–621, 1980.

[10] I. Kondakor, R. D. Pascual-Marqui, C. M. Michel, and D. Lehmann, "Event-related potential map differences depend on the prestimulus microstates," *Journal of Medical Engineering & Technology*, vol. 19, no. 2-3, pp. 66–69, 1995.

[11] C. Michel, T. Koenig, D. Brandeis, L. Gianotti, and J. Wackermann, *Electrical Neuroimaging*, Cambridge University Press, Cambridge, UK, 2009.

[12] W. K. Strik, A. J. Fallgatter, D. Brandeis, and R. D. Pascual-Marqui, "Three-dimensional tomography of event-related potentials during response inhibition: evidence for phasic frontal lobe activation," *Electroencephalography and Clinical Neurophysiology*, vol. 108, no. 4, pp. 406–413, 1998.

[13] S. Makeig, T. P. Jung, A. J. Bell, D. Ghahremani, and T. J. Sejnowski, "Blind separation of auditory event-related brain responses into independent components," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 94, no. 20, pp. 10979–10984, 1997.

[14] J. Onton, M. Westerfield, J. Townsend, and S. Makeig, "Imaging human EEG dynamics using independent component analysis," *Neuroscience & Biobehavioral Reviews*, vol. 30, no. 6, pp. 808–822, 2006.

[15] N. Kriegeskorte, W. K. Simmons, P. S. Bellgowan, and C. I. Baker, "Circular analysis in systems neuroscience: the dangers of double dipping," *Nature Neuroscience*, vol. 12, no. 5, pp. 535–540, 2009.

[16] E. Vul, C. Harris, P. Winkielman, and H. Pashler, "Puzzlingly high correlations in fMRI studies of emotion, personality, and social cognition," *Perspectives on Psychological Science*, vol. 4, no. 3, p. 274, 2009.

[17] M. M. Murray, D. Brunet, and C. M. Michel, "Topographic ERP analyses: a step-by-step tutorial review," *Brain Topography*, vol. 20, no. 4, pp. 249–264, 2008.

[18] L. Liu and A. A. Ioannides, "A correlation study of averaged and single trial MEG signals: the average describes multiple

histories each in a different set of single trials," *Brain Topography*, vol. 8, no. 4, pp. 385–396, 1996.

[19] P. Hansen, M. Kringelbach, and R. Salmelin, Eds., *Meg: An Introduction to Methods*, Oxford University Press, Oxford, UK, 2010.

[20] S. Makeig, A. Bell, T. Jung, and T. Sejnowski, "Independent component analysis of electroencephalographic data," *Advances in Neural Information Processing Systems*, vol. 8, pp. 145–151, 1996.

[21] T. P. Jung, S. Makeig, C. Humphries et al., "Removing electroencephalographic artifacts by blind source separation," *Psychophysiology*, vol. 37, no. 2, pp. 163–178, 2000.

[22] R. D. Pascual-Marqui, C. M. Michel, and D. Lehmann, "Segmentation of brain electrical activity into microstates: model estimation and validation," *IEEE Transactions on Biomedical Engineering*, vol. 42, no. 7, pp. 658–665, 1995.

[23] L. F. H. Basile, E. P. Brunetti, J. F. Pereira et al., "Complex slow potential generators in a simplified attention paradigm," *International Journal of Psychophysiology*, vol. 61, no. 2, pp. 149–157, 2006.

[24] P. L. Nunez and R. Srinivasan, *Electric Fields of the Brain: The Neurophysics of EEG*, Oxford University Press, New York, NY, USA, 2006.

[25] E. J. Davelaar, X. Tian, D. E. Huber, and C. T. Weidemann, "A habituation account of change detection in same/different judgments," submitted.

[26] D. E. Huber, X. Tian, T. Curran, R. C. O'Reilly, and B. Woroch, "The dynamics of integration and separation: ERP, MEG, and neural network studies of immediate repetition effects," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 34, no. 6, pp. 1389–1416, 2008.

[27] D. E. Huber and R. C. O'Reilly, "Persistence and accomodation in short-term priming and other perceptual paradigms: temporal segregation through synaptic depression," *Cognitive Science*, vol. 27, no. 3, pp. 403–430, 2003.

[28] R. Ratcliff, M. G. Philiastides, and P. Sajda, "Quality of evidence for perceptual decision making is indexed by trial-to-trial variability of the EEG," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 106, no. 16, pp. 6539–6544, 2009.

[29] R. Pascual-Marqui, "Review of methods for solving the EEG inverse problem," *International Journal of Bioelectromagnetism*, vol. 1, no. 1, pp. 75–86, 1999.

[30] R. D. Pascual-Marqui, C. M. Michel, and D. Lehmann, "Low resolution electromagnetic tomography: a new method for localizing electrical activity in the brain," *International Journal of Psychophysiology*, vol. 18, no. 1, pp. 49–65, 1994.

[31] D. Brandeis, H. Naylor, R. Halliday, E. Callaway, and L. Yano, "Scopolamine effects on visual information processing, attention, and event-related potential map latencies," *Psychophysiology*, vol. 29, no. 3, pp. 315–336, 1992.