Journal of Electrical and Computer Engineering

Iterative Signal Processing in Communications

Guest Editors: Christian Schlegel, Peter Hoeher, Owe Axelsson, and Lance Pérez

Iterative Signal Processing in Communications

Journal of Electrical and Computer Engineering

Iterative Signal Processing in Communications

Guest Editors: Christian Schlegel, Peter Hoeher,
Owe Axelsson, and Lance Peréz

Editorial Board

The editorial board of the journal is organized into sections that correspond to the subject areas covered by the journal.

Circuits and Systems

Communications

Signal Processing

Contents

Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2010, Article ID 862392, 2 pages
doi:10.1155/2010/862392

Editorial

Iterative Signal Processing in Communications

Christian Schlegel,[1] Peter Hoeher,[2] Owe Axelsson,[3] and Lance Pérez[4]

[1] *Department of Computing Science, University of Alberta, Edmonton, AB, Canada T6G 2E8*
[2] *Faculty of Engineering, Christian-Albrechts-University of Kiel, 24143 Kiel, Germany*
[3] *Institute of Geonics, AS CR, Ostrava, 60200 Brno, Czech Republic*
[4] *Department of Electrical Engineering, University of Nebraska-Lincoln, Lincoln, NE 68588, USA*

Correspondence should be addressed to Christian Schlegel, schlegel@ualberta.ca

Received 22 September 2010; Accepted 22 September 2010

Iterative signal processing in communications has experienced an explosive rise in popularity in recent years. The catalytic origins of this paradigm-shifting new philosophy among communications experts can be traced to the invention of turbo coding, and the subsequent rediscovery of low-density parity check (LDPC) coding, both in the field of error control coding. Both systems rely on iterative decoding algorithms to achieve their astounding performance. However, iterative signal processing is not confined to the decoding of error control codes and rather quickly spread to many other possible applications. The purpose of this special issue is to examine the concept of iterative signal processing, highlight its potential, and draw the attention of communications engineers to this fascinating topic.

Of course, as the mathematicians know, but many communications experts may not realize, iterative processing has a long and productive history in the theory of linear matrix solution methods, that is, in finding approximative low-complexity numerical equation solving techniques which avoid the cubic complexity of direct solution methods. We start off this special issue with Axelsson's key paper giving an in-depth look at the methods and milestones in the development of linear iterative solution methods, to illustrate that iterative processing has long been a major tool in the areas of signal processing dealing with large systems of equations. The second paper by Huckle and Sedlack takes another detailed look and shows that it can be computationally efficient to replace classical smoothers used in iterative solution methods with sparse approximate inverses combined with subspace approximations. This technique can be applied to ill-posed problems such as recovering underlying information from blurred signals. The third paper by Donatelli and Serra-Capizzano discusses the use of antireflective boundary conditions for such deblurring problems. The precision of reconstruction and both iterative and noniterative regularization solvers are discussed.

The next three papers deal with the issue of separating signals in channels with large-scale crossinterference, where optimal, but exponentially complex, detection methods fail due to an infeasibility of implementation. These papers deal with situations where the number of mutually interfering signals is so large, that the traditional methods have to be replaced with something new, and this new methodology is based on iterative processing. Depending on the preferences of the authors, the iterative processing can be introduced as a variant of turbo decoding or a variant of iterative equation solution methods. Also popular is the view of the algorithm as a message-passing process on a connectivity graph—this approach was popularized by the LDPC coding community.

The first paper in this group by Schlegel et al., considers the problem of decoding signals that are modulated onto random signal waveforms, such as one may observe in multiple antenna fading channels. The authors show that an iterative cancellation decoder can achieve the capacity of this multiple access channel by exploiting signal redundancy introduced into the waveforms, and that this decoder has in fact the same order complexity as that of an iterative matrix solver. The second paper by Wo et al. addresses a very similar situation, called superposition modulation, where the individual waveforms are chosen deterministically, but correlated, and with signal attributes that shape the amplitude distribution of the signals. This is done to

harness the shaping gain of 1.53 dB that theory promises with respect to standard one-dimensional pulse-amplitude modulation. The paper introduces and analyses an iterative receiver that overcomes the decoding complexity challenge. The third paper by Wo and Hoeher looks at the random waveform channel created by multiple antenna transmission systems and models the random mutual interference as correlated Gaussian noise. Rather than utilizing cancellation, the authors derive an iterative log-likelihood message passing algorithm based on this correlated Gaussian approximation and study examples where this decoder achieves the ideal performance of an interference-free system.

The next two papers consider specific complex signal processing challenges that are addressed by utilizing an iterative algorithm to obtain a solution with a low effort in complexity. The first paper by Kerr and Lodge treats the acquisition of an unknown phase of a spreading sequence. This problem is cast into the standard constraint form of a linear error control code. The received signal is then decoded iteratively by using a connectivity graph for that code model. The novelty lies in the fact that for each iteration a new code connectivity graph is formed using the outcomes of the previous iteration. The second paper by Ma et al., considers transmission of self-encoded signals over multiple antenna channels. The consequence of this self-encoding is that the received signal follows a finite-state model, and that the maximum-likelihood detector amounts to a sequence search with exponential complexity. It is precisely this complexity which is avoided by the use of an iterative algorithm.

The last paper by Psota et al. brings us back to error control coding, where iterative processing has made its first postclassical impact. In this paper, the authors discuss problems that can occur with the iterative decoding algorithm. They discuss the code structures that can cause the most common iterative decoders to get trapped in local incorrect solutions. A discussion of the issues and an illustration of the complexity of the problem are given using the example of LDPC codes, and an algorithmic enumeration procedure is presented.

Christian Schlegel
Peter Hoeher
Owe Axelsson
Lance Pérez

Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2010, Article ID 972794, 33 pages
doi:10.1155/2010/972794

Review Article

Milestones in the Development of Iterative Solution Methods

Owe Axelsson

Institute of Geonics AS CR, 70800 Ostrava, Czech Republic

Correspondence should be addressed to Owe Axelsson, owe.axelsson@it.uu.se

Received 10 June 2010; Accepted 27 August 2010

Academic Editor: Christian Schlegel

Iterative solution methods to solve linear systems of equations were originally formulated as basic iteration methods of defect-correction type, commonly referred to as Richardson's iteration method. These methods developed further into various versions of splitting methods, including the successive overrelaxation (SOR) method. Later, immensely important developments included convergence acceleration methods, such as the Chebyshev and conjugate gradient iteration methods and preconditioning methods of various forms. A major strive has been to find methods with a total computational complexity of optimal order, that is, proportional to the degrees of freedom involved in the equation. Methods that have turned out to have been particularly important for the further developments of linear equation solvers are surveyed. Some of them are presented in greater detail.

1. Introduction

In many applications of quite different types appearing in various sciences, engineering, and finance, large-scale linear algebraic systems of equations arise. A particular type of problems appear in signal processing. This also includes nonlinear systems of equation, which are normally solved by linearization at each outer nonlinear iteration step, but they will not be further discussed in this paper.

Due to their high demand of computer memory and computer time, which can grow rapidly with increasing problem size, direct solution methods, such as Gaussian elimination, are in general not feasible unless the size of the problem is relatively small. In the early computer age, when available size of computer central memories was very small and the speed of arithmetic operations slow, this was found to be the case even for quite modest-sized problems.

Even for modern computers with exceedingly large memories and very fast arithmetics it is still an issue because nowadays one wants to solve much more involved problems of much larger sizes, for instance to enable a sufficient resolution of partial differential equation problems with highly varying (material) coefficients, such as is found in heterogeneous media. Presently problems with up to billions of degrees of freedom (d.o.f.) are solved. For instance, if an elliptic equation of elasticity type is discretized and solved on a 3D mesh with 512 meshpoints in each coordinate direction, then an equation of that size arises.

A basic iteration method to solve a linear system

$$A\mathbf{x} = \mathbf{b}, \tag{1}$$

where A is nonsingular, has the following form.

Given an initial approximation $\mathbf{x}^0$, for $k = 0, 1, \ldots$ until convergence, let $\mathbf{r}^k = A\mathbf{x}^k - \mathbf{b}$,$\mathbf{e}^k = -\tau\mathbf{r}^k$, $\mathbf{x}^{k+1} = \mathbf{x}^k + \mathbf{e}^k$. Here, $\tau > 0$ is a parameter to be chosen.

This method can be described either as a defect ($\mathbf{r}^k$)—correction ($\mathbf{e}^k$) method or, alternatively, as a method to compute the stationary solution of the evolution equation

$$\frac{d\mathbf{x}(t)}{dt} = A\mathbf{x}(t) - \mathbf{b}, \quad t > 0, \ \mathbf{x}(0) = \mathbf{x}^0, \tag{2}$$

by timestepping with time-step τ, that is,

$$\mathbf{x}(t+\tau) = \mathbf{x}(t) - \tau(A\mathbf{x}(t) - \mathbf{b}), \quad t = 0, \tau, \ldots. \tag{3}$$

Such methods are commonly referred to as Richardson iteration methods (e.g., see [1–4]). However, already in 1823 Gauss [5] wrote, "Fast jeden Abend mache ich eine neue Auflage des Tableau, wo immer leicht nachzuhelfen ist. Bei der Einförmigkeit des Messungsgeschäfts gibt dies immer eine angenehme Unterhaltung; man sieht daran auch

immer gleich, ob etwas Zweifelhaftes eingeschlichen ist, was noch wünschenswert bleibt usw. Ich empfehle Ihnen diesen Modus zur Nachahmung. Schwerlich werden Sie je wieder direct eliminieren, wenigstens nicht, wenn Sie mehr als zwei Unbekannte haben. Das indirecte Verfahren läßt sich halb im Schlafe ausführen oder man kann während desselben an andere Dingen denken." (Freely translated, "I recommend this modus operandi. You will hardly eliminate directly anymore, at least not when you have more than two unknowns. The indirect method can be pursued while half asleep or while thinking about other things.")

It holds that

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \tau\left(A\mathbf{x}^k - \mathbf{b}\right), \tag{4}$$

or

$$\mathbf{e}^{k+1} = (I - \tau A)\mathbf{e}^k, \tag{5}$$

where $\mathbf{e}^k = \mathbf{x} - \mathbf{x}^k$ is the iteration error and $\mathbf{x}$ is the solution of (1).

Hence,

$$\mathbf{e}^k = (I - \tau A)^k \mathbf{e}^0, \quad k = 0, 1, \ldots. \tag{6}$$

For convergence of the method, that is $\mathbf{e}^k \to 0$, the parameter τ must in general be chosen such that $\rho := \|I - \tau A\| < 1$, where $\|\cdot\|$ is a matrix norm, subordinate to the chosen vector norm. (We remark here that this is not possible if A is indefinite.)

Let $\rho(\cdot)$ denote the spectral radius of a matrix, that is, the maximal absolute value of the eigenvalues of the matrix.

If A is self–adjoint, then it can be shown that $\rho(A) = \|A\|_2 = \sqrt{\rho(A^*A)}$, where $\|\cdot\|_2$ denotes the matrix norm subordinate to the Euclidian vector norm. For general, nonsymmetric matrices it has been shown (e.g., see [6, page 162]) that there exist matrix norms that are arbitrarily close to the spectral radius. These can, however, correspond to an unnatural scaling of the matrix.

The rate of convergence is determined by the convergence factor ρ. For symmetric positive definite matrices, the optimal value of τ to minimize ρ, is $\tau = 2/(\lambda_1 + \lambda_n)$, where λ_1, λ_n are the extreme eigenvalues of A. Normally, however, the eigenvalues are not available.

As an example, for second order elliptic diffusion type of problems in $\Omega^d (d = 2, 3)$ using a standard central difference or a finite element method, the spectral condition number $\lambda_n/\lambda_1 = O(h^{-2})$, where h is the (constant) meshsize parameter. Hence, the number of iterations to reach a relative accuracy ε is of order $O(h^{-2})|\log\varepsilon|)$, $h \to 0$.

Since each iteration uses $O(h^{-d})$ elementary arithmetic operations, this shows that the total number of operations needed to reduce the error to a given tolerance is of order $O(h^{-d-2})$. This is in general smaller than for a direct solution method when $d \geq 2$, but still far from the optimal order, $O(h^{-d})$, that we aim at.

To improve on this, often a splitting of the matrix A is used. It is readily shown that for any initial vector, the number of iterations required to get a relative residual, $\|\mathbf{r}^k\|/\|\mathbf{r}^0\| < \varepsilon$, for some $\varepsilon, 0 < \varepsilon < 1$, is at most $k_{it} = \lceil \ln(1/\varepsilon)/\ln(1/\rho) + 1 \rceil$, where $\lceil\ \rceil$ denotes the integer part. Frequently, $\rho = 1 - c\delta^r$, where c is a constant, r is a positive integer, often $r = 2$ and δ is a small number, typically $\delta = 1/n$, which decreases with increasing problems size n. This implies, that the number of iterations is propotional to $(1/\delta)^r$, which number increases rapidly when $\delta \to 0$.

For $\tau = 1$, the splitting $A = C - R$ of A in two terms is used, where C is nonsingular. The iterative method (4) then takes the form

$$C\mathbf{x}^{k+1} = R\mathbf{x}^k + \mathbf{b}, \quad k = 0, 1, \ldots. \tag{7}$$

Method (7) is convergent if $\rho(C^{-1}R) < 1$. Splitting methods will be discussed in Section 2.

Let $B = C^{-1}R$. If $\|B\|$ is known and $\|B\| < 1$, we can use the following theorem to get a test when the iteration error is small enough, that is, when to stop the iterations.

Theorem 1. *Let $\|B\| < 1$, $B = C^{-1}R$, and $\mathbf{x}^m$ be defined by (7). Then,*

$$\|\mathbf{x} - \mathbf{x}^m\| \leq \frac{\|B\|}{1 - \|B\|}\left\|\mathbf{x}^m - \mathbf{x}^{m-1}\right\|, \quad m = 1, 2, \ldots. \tag{8}$$

Proof. From (7) follows $\mathbf{x}^{m+1} - \mathbf{x}^m = B(\mathbf{x}^m - \mathbf{x}^{m-1})$ and, by recursion,

$$\mathbf{x}^{m+k+1} - \mathbf{x}^{m+k} = B^{k+1}\left(\mathbf{x}^m - \mathbf{x}^{m-1}\right), \quad k = 0, 1, \ldots. \tag{9}$$

Note now that $\mathbf{x}^{m+p} - \mathbf{x}^m = \sum_{k=0}^{p-1}(\mathbf{x}^{m+k+1} - \mathbf{x}^{m+k})$. Hence, by the triangle inequality and (9)

$$\begin{aligned}\left\|\mathbf{x}^{m+p} - \mathbf{x}^m\right\| &\leq \sum_{k=0}^{p-1}\left\|B^{k+1}\right\|\left\|\mathbf{x}^m - \mathbf{x}^{m-1}\right\| \\ &\leq \frac{\|B\| - \|B\|^{p+1}}{1 - \|B\|}\left\|\mathbf{x}^m - \mathbf{x}^{m-1}\right\|.\end{aligned} \tag{10}$$

Letting $p \to \infty$ and noting that $\mathbf{x}^{m+p} \to \hat{\mathbf{x}}$, (8) follows. □

The basic iteration method (4) or the splitting methods in Section 2, can be improved in various ways. This will be the major topic of this paper.

Note first that application of the splitting in (7) requires in general that the matrix R is given in explicit form, which can make the method less viable.

The most natural way to improve (4) is to introduce an approximation C of A, to be used when the correction $\mathbf{e}^k$ in (4) is computed. The relation $\mathbf{e}^k = -\tau\mathbf{r}^k$ is then replaced by $C\mathbf{e}^k = -\tau\mathbf{r}^k$.

Such a matrix is mostly called preconditioner since, by a proper choice, it can significantly improve the condition number $\mathcal{K}$ of A, that is,

$$\mathcal{K}(C^{-1}A) \ll \mathcal{K}(A), \tag{11}$$

where $\mathcal{K}(B) = \|B\|\ \|B^{-1}\|$.

Clearly, in practice, the matrix C must be chosen such that the linear systems with C can be solved with relatively

little expense compared to a solution method for A. In particular, the expense for C is much smaller than that for A using a direct solution method.

For badly scaled matrices A a simple, but often practically useful, choice of C is the (block) diagonal part D of A. Much more efficient choices will be discussed later in the paper.

Early suggestions to use such a matrix C can be found in papers by D'Yakonov [7] and Gunn [8]. For an automatic scaling procedure, see [9] and references therein.

In the present paper, we will survey various choices of C which have proven to be useful in practice. The paper attempts to give a more personal account of the development of iterative solution methods. It is also not our ambition to present the present state- of- the- art but rather to describe the unfolding of the field.

In the remainder of the paper, we discuss, in order, methods based on splitting of the given matrix, the accelerated iterative methods of the Chebyshev and (generalized) conjugate gradient types, pointwise and block incomplete factorization preconditioning methods, symmetrized preconditioners of SSOR and ADI type, approximate inverses, and augmented subspace preconditioning methods. If space had allowed it, it would have been followed by presentation of geometric and algebraic multigrid methods, two-level and multilevel methods, elementwise preconditioning methods, and domain decomposition methods. Also, iteration error estimates and influence of rounding errors, and preconditioners for matrices of saddle point type would have been included. The paper ends with some concluding remarks.

The following relations will be used; if $A = [a_{ij}]$, then $A^T = [a_{ji}]$ denotes the transpose of A, while $A^* = [\overline{a}_{ji}]$, denotes the Hermitian transpose.

2. Splitting Methods

A comprehensive, early presentation of splitting methods, and much more on iterative solution methods, is found in Varga [10]. Often there is a natural splitting of the given matrix as

$$A = C - R, \tag{12}$$

where C is nonsingular. This can be used to formulate an iterative solution method in the form

$$C\mathbf{x}^{k+1} = R\mathbf{x}^k + \mathbf{b}, \quad k = 0, 1, \ldots. \tag{13}$$

This method converges if $\rho(C^{-1}R) < 1$.

Definition 1. (a) A matrix C is said to be *monotone* if C is nonsingular and $C^{-1} \geq 0$ (componentwise).

(b) $A = C - R$ is called a *regular splitting* [10], if C is monotone and $R \geq 0$.

(c) a *weak regular splitting* [11], if C is monotone and $C^{-1}R \geq 0$.

(d) a *nonnegative splitting* [12], if C is nonsingular and $C^{-1}R \geq 0$.

The following holds, see, for example, [6].

Theorem 2. *Let $A = C - R$ be a nonnegative splitting of A. Then, the following properties are equaivalent:*

(a) $\rho(B) < 1$, *that is, $A = C - R$ is a convergent splitting,*

(b) $I - B$ *is monotone,*

(c) A *is nonsingular and $G = A^{-1}R \geq 0$.*

(d) A *is nonsingular and $\rho(B) = \rho(G)/[1 + \rho(G)]$, where $G = A^{-1}R$.*

Corollary 1. *If $A = C - R$ is a weak regular splitting, then the splitting is convergent if and only if A is monotone.*

Proof. (see, e.g., [6]). □

A splitting method that became popular in the fifties is the SOR method. Here, $A = D - L - U$, where D is the (block) diagonal and L, U are the (block) lower and upper triangular parts of A, respectively. The successive relaxation method takes the form

$$\left(\frac{1}{\omega}D - L\right)\mathbf{x}^{k+1} = \left[\left(\frac{1}{\omega} - 1\right)D + U\right]\mathbf{x}^k + \mathbf{b}, \quad k = 0, 1, \ldots, \tag{14}$$

where $\omega \neq 0$ is a parameter, called the relaxation parameter. For $\omega = 1$ one gets the familiar Gauss-Seidel method (Gauss 1823 [5] and Seidel 1814 [13]) and for $\omega > 1$ the successive overrelaxation (SOR) method (Frankel 1950 [14] and Young 1950 [15]).

For the iteration matrix in (14),

$$\mathcal{L}_\omega = \left(\frac{1}{\omega}D - L\right)^{-1}\left(\left(\frac{1}{\omega} - 1\right)D + U\right), \tag{15}$$

it holds that $\rho(\mathcal{L}_\omega) \leq |\omega - 1|$, where the upper bound is sharp. Therefore, the relaxation method is divergent for $\omega \leq 0$ and $\omega \geq 2$ (see, e.g., [6, 16]).

An optimal value of ω can be determined as follows. Assume then that A has property (A^π), that is, there exists a permutation matrix P such that PAP^T is a block tridiagonal matrix. The following Lemma holds.

Lemma 1 (see [15]). *Assume that A has property (A^π) and let $\omega \neq 0$. Let $B := D^{-1}(L + U)$. Then,*

(a) *if $\lambda \neq 0$ is an eigenvalue of $\mathcal{L}_\omega$ and μ satisfies*

$$\mu^2 = \frac{(\lambda + \omega - 1)^2}{(\omega^2\lambda)}, \tag{16}$$

then, μ is an eigenvalue of B,

(b) *if μ is an eigenvalue of B and λ satisfies*

$$\lambda + \omega - 1 = \omega\mu\lambda^{1/2}, \tag{17}$$

then, λ is an eigenvalue of $\mathcal{L}_\omega$.

Proof. For a short proof, see [6]. □

Theorem 3. *Assume that*

(a) A *has property* (A^π), *and*

(b) *the block matrix* $B = I - D^{-1}A$ *has only real eigenvalues.*

Then, the SOR method converges for any initial vector if and only if $\rho(B) < 1$ *and* $0 < \omega < 2$. *Further, we have*

$$\omega_{opt} = \frac{2}{1+\sqrt{1-\rho(B)^2}}, \tag{18}$$

for which the asymptotic convergence factor is given as

$$\min_{\omega}\rho(\mathcal{L}_\omega) = \rho\left(\mathcal{L}_{\omega_{opt}}\right) = \omega_{opt} - 1 = \frac{\left(1-\sqrt{1-\rho(B)^2}\right)}{\left(1+\sqrt{1-\rho(B)^2}\right)}. \tag{19}$$

Proof. Fort a short proof, see [6]. □

The eigenvalues of $C^{-1}A$ are in general complex, and for $\omega = \omega_{\text{opt}}$ it can be shown that they are distributed around a circle in the complex plane. This implies that the method can not be polynomially accelerated. (See Section 3 for a presentation of polynomial acceleration methods.) Further, the efficiency of the SOR method turns out to be critically dependent on the choice of ω.

A similar result as in Theorem 3 has been shown in [6], see also [17], that holds even if A does not have property (A^π), but is Hermitian.

Theorem 4. *Let* A *be Hermitian and positive definite and let*

$$\widetilde{\mathcal{L}}_\omega = D^{1/2}L_\omega D^{-1/2} = \left(\frac{1}{\omega}I - \widetilde{L}\right)^{-1}\left(\left(\frac{1}{\omega}-1\right)I - \widetilde{L}^*\right), \tag{20}$$

where $\widetilde{L} = D^{-1/2}LD^{1/2}$, *and let* $0 < \omega < 2$. *Then,*

$$\rho(L_\omega)^2 = \rho\left(\widetilde{L}_\omega\right)^2 \le 1 - \frac{2/\omega - 1}{(1/\omega - 1/2)^2\delta^{-1} + \gamma + 1/\omega}, \tag{21}$$

where

$$\gamma = \sup_{x\neq 0}\left\{\frac{\left[\left(\left|\mathbf{x},\widetilde{L}\mathbf{x}\right|^2/(\mathbf{x},\mathbf{x})\right) - 1/4(\mathbf{x},\mathbf{x})\right]}{\left(\widetilde{A}\mathbf{x},\mathbf{x}\right)}\right\},$$
$$\delta = \lambda_{\min}\left(\widetilde{A}\right) = \frac{\min_{x\neq 0}\left(\widetilde{A}\mathbf{x},\mathbf{x}\right)}{(\mathbf{x},\mathbf{x})}. \tag{22}$$

Further, if $|(\mathbf{x},\widetilde{L}\mathbf{x})| \le 1/2(\mathbf{x},\mathbf{x})$, *then*

$$\omega^* = \frac{2}{1+\sqrt{2\delta}}, \tag{23}$$

minimizes the upper bound of $\rho(L_\omega)$ *and we have*

$$\rho(L_{\omega^*})^2 = \frac{1-\sqrt{\delta/2}}{1+\sqrt{\delta/2}}. \tag{24}$$

For a proof, see [6].

In Section 4, we present a symmetric version of the SOR method where acceleration is possible.

3. Accelerated Iterative Methods

In this section, the important Chebyshev and conjugate gradient iteration methods are presented.

Consider first the iterative method (4) with variable time-steps τ_k,

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \tau_k C^{-1}\mathbf{r}^k, \qquad \mathbf{r}^k = A\mathbf{x}^k - \mathbf{b}, \quad k = 0,1,\ldots. \tag{25}$$

Here, $\{\tau_k\}$ is a sequence of iteration (acceleration) parameters. If $\tau_k = \tau$, $k = 0,1,\ldots$, we talk about a stationary iterative method, otherwise about a nonstationary or semiiterative method.

Let $\mathbf{e}^k = \mathbf{x} - \mathbf{x}^k$, the iteration error. Then, it follows from (25) that $\mathbf{e}^{k+1} = (I - \tau_k C^{-1}A)\mathbf{e}^k$, $k = 0,1,\ldots$, so $\mathbf{e}^m = P_m(C^{-1}A)\mathbf{e}^0$ (and $\mathbf{r}^m = AP_m(C^{-1}A)A^{-1}\mathbf{r}^0 = P_m(AC^{-1})\mathbf{r}^0$). Here, $P_m(\lambda) = \Pi_{k=0}^m(1-\tau_k\lambda)$ a polynomial of degree m having zeros at $1/\tau_k$ and satisfying $P_m(0) = 1$.

We want to choose the parameters $\{\tau_k\}$ such that $\|\mathbf{e}^m\|$ is minimized. However, this would mean that in general the parameters would depend on $\mathbf{e}^0$, which is not known. Also the eigenvalues of $C^{-1}A$ are not known. We then take the approach of minimizing $\|\mathbf{e}^m\|/\|\mathbf{e}^0\|$ for all $\mathbf{e}^0$; that is, we want to minimize $\|P_m(C^{-1}A)\mathbf{r}^0\|$.

3.1. The Chebyshev Iterative Method. In case the eigenvalues of $C^{-1}A$ are real and positive and if a positive lower (a) and (b) an upper bound are known of the spectrum, then we see that $\{\tau_k\}$ should be chosen such that $\max_{a\le\lambda\le b}|P_m(\lambda)|$ is minimized over all $P_m \in \Pi_m^0$, that is, over the set of polynomials of degree m satisfying $P_m(0) = 1$.

The solution to this min-max problem is well known,

$$P_m(\lambda) = \frac{\mathrm{T}_m((b+a-2\lambda)/(b-a))}{T_m((b+a)/(b-a))}, \tag{26}$$

where $T_m(z) = (1/2)[(z+(z^2-1)^{1/2})^m + (z-(z^2-1)^{1/2})^m] = \cos(m\arccos z)$ are the Chebyshev polynomials of the first kind. The corresponding values of τ_k satisfy

$$\frac{1}{\tau_k} = \frac{b-a}{2}\cos\Theta_k + \frac{b+a}{2}, \quad \Theta_k = \frac{2k-1}{2m}\pi, \ k = 1,2,\ldots,m, \tag{27}$$

which are the zeros of the polynomial. The corresponding method is referred to as the Chebyshev (one-step) acceleration method, see, for example, [10, 18]. It is an easy matter to show that

$$\frac{1}{T_m}\left(\frac{b+a}{b-a}\right) \le 2\varrho^m, \quad \text{where } \varrho = \frac{(b^{1/2}-a^{1/2})}{(b^{1/2}+a^{1/2})}. \tag{28}$$

This implies that if the number of iterations satisfies $m \ge \ln\varrho^{-1}\ln(2/\varepsilon)$, that is, in particular if

$$m \ge \frac{1}{2}\left(\frac{b}{a}\right)^{1/2}\ln(2/\varepsilon), \quad \varepsilon > 0, \tag{29}$$

then $\|\mathbf{e}^m\|/\|\mathbf{e}^0\| \le \varepsilon$.

The disadvantage with this method is that to make it effective one needs accurate estimates of a and b, and we need to determine m beforehand (which, however, can be done by (29)). The method cannot utilize any special distribution of the eigenvalues in the spectrum (as opposed to the conjugate gradient method, see below). More important, however, is that this method is actually numerically unstable (similarly to an explicit time stepping method for initial value problems when the time steps are too large). This is due to the fact that $\|I - \tau_k C^{-1}A\|$ is much larger than unity for several of the values τ_k. However, one may prove that with some particular permutation of the parameters, their instability effect can be avoided.

There is an alternative to the choice (25). Namely, there exists a three term form of the Chebyshev acceleration method

$$\mathbf{x}^{k+1} = \alpha_k \mathbf{x}^k + (1-\alpha_k)\mathbf{x}^{k-1} - \beta_k C^{-1}\mathbf{r}^k, \quad k = 1, 2, \ldots, \tag{30}$$

where $\mathbf{x}^1 = \mathbf{x}^0 - (1/2)\beta_0 C^{-1}\mathbf{r}^0$.

Here, the parameters are chosen as $\beta_0 = 4/(a+b)$,

$$\alpha_k = \frac{a+b}{2}\beta_k, \qquad \beta_k^{-1} = \frac{a+b}{2} - \left(\frac{b-a}{4}\right)^2 \beta_{k-1} \qquad k = 1, 2, \ldots. \tag{31}$$

Hence, we do not have to determine the number of steps beforehand. More importantly, it has been shown in [18] that this method is numerically stable. (For some related remarks, see [6]). A similar form of the method was proposed a long time ago, see Golub and Varga [19] and the references cited therein.

It is interesting to note that the parameters approach stationary values. If $C^{-1}A = I - B$ and B has eigenvalues in $[-\varrho, \varrho]$, $\varrho = \varrho(B) < 1$ (the spectral radius of B), then

$$a = 1 - \varrho, \qquad b = 1 + \varrho,$$
$$\alpha_k = \frac{a+b}{2}\beta_k \longrightarrow \frac{2}{\left[1 + \left(1 - \varrho^2\right)^{1/2}\right]}, \tag{32}$$

which is recognized as the parameter ω_{opt} of the optimal SOR method (see Section 2). Young [20] has proven that the asymptotic rate of convergence is retained even if one uses the stationary values throughout the iterations.

For the case of complex eigenvalues of $C^{-1}A$ with positive real parts and contained in an ellipse one may choose parameters similarly. See [6, 21, 22] for details. For comments on the optimaly of the method, see [23]. For application of the method for nonsymmetric problems, see [6, 24].

Perhaps the main thrust during the 1970 has been in using the conjugate gradient method as an acceleration method. Already much has been written on the subject; we refer to [25–27] for a historical account, to [18, 28–33] for an exposition of the preconditioned conjugate gradient PCG method and to [18, 34] for a survey of generalized and truncated gradient methods for nonsymmetric and indefinite matrix problems.

Given	$\mathbf{x}^{(0)}$, ε	initial guess and absolute or relative stopping tolerance
Set	$\mathbf{x}^{(0)}$, $\mathbf{g} = A\mathbf{x} - \mathbf{b}$, $\delta_0 = \mathbf{g}^T\mathbf{g}$	
	$\mathbf{d} = -\mathbf{g}$	initial search direction
Repeat	until convergence	
	$\mathbf{h} = A\mathbf{d}$	
	$\tau = \delta_0/(\mathbf{d}^T\mathbf{h})$	
	$\mathbf{x} = \mathbf{x} + \tau\mathbf{d}$	new approximation
	$\mathbf{g} = \mathbf{g} + \tau\mathbf{h}$	new (iterative) residual
	$\delta_1 = \mathbf{g}^T\mathbf{g}$	
	if $\delta_1 \le \varepsilon$ then stop,	otherwise
	$\beta = \delta_1/\delta_0$, $\delta_0 = \delta_1$	
	$\mathbf{d} = -\mathbf{g} + \beta\mathbf{d}$	new search direction

Algorithm 1: Standard conjugate gradient algorithm.

The advantage with conjugate gradient methods is that they are self adaptive; the optimal parameters are calculated by the algorithm so that the error in energy norm $\|e^l\|_{A^{1/2}} = \{(e^l)^T A e^l\}^{1/2}$ is minimized. This applies to a problem where C and A are symmetric and positive definite (SPD) or, more generally, if $C^{-1}A$ is similarly equivalent to an SPD matrix. Hence, there is no need to know any bounds for the spectrum. Since the method converges at least as fast as the Chebyshev method it follows that $\|x - x^m\|_{A^{1/2}} \le \varepsilon\|x - x^0\|_{A^{1/2}}$, if

$$m = \text{int}\left\{\frac{1}{2}\mathcal{K}^{1/2}\ln\left(\frac{2}{\varepsilon}\right) + 1\right\}. \tag{33}$$

We describe now the conjugate gradient method. Thereby we follow the presentations in [29, 35].

3.2. The Preconditioned Conjugate Gradient Method. During the past 40 years or so, the preconditioned conjugate gradient method has become the major iterative solution method for linear systems of algebraic equations, in particular those arising in science and engineering. The author of these notes became interested in the method by the beginning of 1970 (cf. [18]).

The conjugate gradient algorithm to solve a system of linear equations the $A\mathbf{x} = \mathbf{b}$, where $A(n \times n)$ is symmetric and positive definite, was originally introduced by Hestenes and Stiefel [25] in 1950. Before we discuss the properties of the CG method, we describe its implementation. Namely, the algorithm consists of the steps in Algorithm 1.

What one sees from a first glance is that the CG algorithm is quite simple. Each iteration consists of one matrix-vector multiplication, two vector updates and two scalar products. Apart from the initial guess $\mathbf{x}^{(0)}$ (which can be taken to be the zero vector) and stopping tolerance, there are no other method parameters to be determined or tuned by the user. Thus, the method is easily programmable, cheap in terms of arithmetic operations and performs as a black box.

For some problems the standard (unpreconditioned) CG method performs impressively well and this can be explained by some particular properties of this powerful algorithm.

The CG method is best described as a method to minimize the quadratic functional

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} - \mathbf{b}^T\mathbf{x} + \mathbf{c}, \tag{34}$$

over a set of vectors. If A is nonsingular, then we can rewrite f in the form

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T(A\mathbf{x} - \mathbf{b})^T A^{-1}(A\mathbf{x} - \mathbf{b}) - \frac{1}{2}\mathbf{b}^T A^{-1}\mathbf{b} + \mathbf{c}, \tag{35}$$

so, minimizing the quadratic functional is equivalent to solving the system $A\mathbf{x} = \mathbf{b}$. If A is singular and A^{-1} in (35) is replaced by a generalized inverse of A, then the above equivalence still holds if the minimization takes place on a subspace in the orthogonal complement to the null-space of A.

Given an initial approximation $\mathbf{x}^{(0)}$ and the corresponding residual $\mathbf{r}^{(0)} = A\mathbf{x}^{(0)} - \mathbf{b}$, the minimization in the conjugate gradient method takes place successively on a subspace

$$\mathcal{K}_k = \left\{\mathbf{r}^{(0)},\ A\mathbf{r}^{(0)},\ A^2\mathbf{r}^{(0)}, \ldots, A^{k-1}\mathbf{r}^{(0)}\right\}, \tag{36}$$

of growing dimension. This subspace is referred to as the *Krylov set*.

In the derivation of the algorithm, the next approximate solution is constructed as

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \tau_k \mathbf{d}^{(k)}, \tag{37}$$

where τ_k is chosen

$$\tau_k = \frac{-\mathbf{d}^{(k)^T}\mathbf{g}^{(k)}}{\mathbf{d}^{(k)^T}A\mathbf{d}^{(k)}} = \frac{-\mathbf{d}^{(k)^T}\left(A\mathbf{x}^{(k)} - \mathbf{b}\right)}{\mathbf{d}^{(k)^T}A\mathbf{d}^{(k)}}, \tag{38}$$

which minimizes the function $f(\mathbf{x}^{(k)} + \tau\mathbf{d}^{(k)})$, $-\infty < \tau < \infty$. Also, the gradient of f at $\mathbf{x}^{(k+1)}$ is made orthogonal to the search direction $\mathbf{d}^{(k)}$. This is seen from the following relations:

$$\begin{aligned}
\mathbf{x}^{(k+1)} &= \mathbf{x}^{(k)} + \tau_k\mathbf{d}^{(k)} \Longrightarrow A\mathbf{x}^{(k+1)} - \mathbf{b} \\
&= A\mathbf{x}^{(k)} - \mathbf{b} + \tau_k A\mathbf{d}^{(k)} \Longrightarrow \mathbf{g}^{(k+1)} = \mathbf{g}^{(k)} + \tau_k A\mathbf{d}^{(k)} \\
&\Longrightarrow \mathbf{d}^{(k)^T}\mathbf{g}^{(k+1)} = \mathbf{d}^{(k)^T}\mathbf{g}^{(k)} + \tau_k \mathbf{d}^{(k)^T}A\mathbf{d}^{(k)} = 0.
\end{aligned} \tag{39}$$

As in Fourier type minimization methods, it turns out to be efficient to work with orthogonal (A-orthogonal) search directions $\mathbf{d}^{(k)}$ which, since A is symmetric, can be determined from a three-term recursion

$$\mathbf{d}^{(0)} = \mathbf{r}^{(0)}, \qquad \mathbf{d}^{(k+1)} = -A\mathbf{d}^{(k)} + \widetilde{\beta}_k\mathbf{d}^{(k)}, \quad k = 1, 2, \ldots, \tag{40}$$

or equivalently, from

$$\mathbf{d}^{(k+1)} = -\mathbf{r}^{(k+1)} + \beta_k\mathbf{d}^{(k)}. \tag{41}$$

This recursive choice of search directions is done so that at each step the solution has smallest error in the A-norm, $\|\mathbf{x} - \mathbf{x}^{(k)}\|_A = \{\mathbf{e}^{(k)^T}A\mathbf{e}^{(k)}\}^{1/2}$, where $\mathbf{e}^{(k)} = \mathbf{x} - \mathbf{x}^{(k)}$ is the iteration error. As mentioned, the minimization takes place over the set of (Krylov) vectors $\mathcal{K}_k$ and, as is readily seen

$$\begin{aligned}
\mathcal{K}_k &= \left\{\mathbf{x}^{(1)} - \mathbf{x}^{(0)},\ \mathbf{x}^{(2)} - \mathbf{x}^{(0)}, \ldots, \mathbf{x}^{(k)} - \mathbf{x}^{(0)}\right\} \\
&= \left\{\mathbf{g}^{(0)},\ \mathbf{g}^{(1)}, \ldots, \mathbf{g}^{(k-1)}\right\} \\
&= \left\{\mathbf{d}^{(0)},\ \mathbf{d}^{(1)}, \ldots, \mathbf{d}^{(k-1)}\right\}.
\end{aligned} \tag{42}$$

To summarize, the CG method possesses the following remarkable properties.

Theorem 5. *Let the CG Algorithm 1 be applied to a symmetric positive definite matrix A. Then, in exact arithmetic the following properties hold:*

(1) *the iteratively constructed residuals* $\mathbf{g}$ *are mutually orthogonal, that is,* $\mathbf{g}^{(k)^T}\mathbf{g}^{(j)} = 0$, $j < k$;

(2) *the search directions* $\mathbf{d}$ *are A-orthogonal (or conjugate), that is,* $\mathbf{d}^{(k)^T}A\mathbf{d}^{(j)} = 0$, $j < k$:

(3) *as long as the method has not converged, that is,* $\mathbf{g}^{(k)} \neq 0$, *the algorithm proceeds with no breakdown and* (42) *holds;*

(4) *as long as the method has not converged, the newly constructed approximation* $\mathbf{x}^{(k)}$ *is the unique point in* $\mathbf{x}^{(0)} \oplus \mathcal{K}_k$ *that minimizes* $\|\mathbf{e}^{(k)}\|_A = \|\mathbf{x} - \mathbf{x}^{(k)}\|_A$,

(5) *the convergence is monotone in A-norm, that is,* $\|\mathbf{e}^{(k)}\|_A < \|\mathbf{e}^{(k-1)}\|_A$ *and* $\mathbf{e}^{(m)} = 0$ *will be achieved for some* $m \leq n$.

For a proof of the above theorem consult, for instance, [29] or [6].

Since the method is optimal, that is it gives the smallest error on a subspace of growing dimension, it *terminates* with the exact solution (ignoring round-off errors) in at most n steps (the dimension of the whole vector space $\mathbf{x} \in R^n$). In fact, it can be readily seen that the CG algorithm terminates after m steps, where m is the degree of the minimal polynomial Q_m to A with respect to the initial residual vector, in other words, Q_m has the smallest degree of all polynomials Q for which $Q(A)\mathbf{r}^{(0)} = 0$. Therefore, the CG method can be viewed also as a direct solution method. However, in practice we want convergence to occur to an acceptable accuracy in much fewer steps than n or m. Thus, we use CG as an iterative method.

For further discussions of the CG methods, see [6, 34, 35].

When one experiments with CG to solve systems with various matrices one observes some phenomena which need special attention. This can be illustrated by a simple example.

Consider the solution of $A\mathbf{x} = \mathbf{b}$ by the standard conjugate gradient, where

$$A = \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & & \\ & & -1 & 2 & -1 \\ & & & -1 & 1 \end{bmatrix}, \qquad b = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{43}$$

The exact solution is $\hat{\mathbf{x}} = [1, 1, \ldots, 1]^T$. Starting with $\mathbf{x}^{(0)} = [0, 0, \ldots, 0]^T$ one finds that after k iterations

$$\mathbf{x}^{(k)} = \left[\frac{k}{k+1}, \frac{k-1}{k+1}, \ldots, \frac{1}{k+1}, 0, \ldots, 0\right]^T, \tag{44}$$

for $1 \leq k \leq n-1$ and $\mathbf{x}^{(n)} = \hat{\mathbf{x}}$. Hence, the information travels one step at a time from left to right and it takes n steps before the last component has changed at all. The algorithm converges exactly in n steps and terminates due to the final recurrence property of the method.

Another detail one observes is that the norm of the error, $\|\mathbf{x} - \mathbf{x}^{(k)}\|$, can be much larger than the norm of the iteratively computed residual.

These examples illustrate the fact that although the method has an optimal order of convergence rate in the *energy norm*, its actual convergence rate in spectral norm can be different and depends both on the distribution of the eigenvalues of the (preconditioned) system matrix and on the initial approximation (or residual). (For comparison, we note that the rate of convergence for steepest descent depends only on the ratio of the extremal eigenvalues of A.) Faster convergence for the CG method is expected when the eigenvalues are clustered.

One way to get a better eigenvalue distribution is to precondition A by a proper preconditioner B. Hence, in order to achieve a better eigenvalue distribution it is crucial in practice to use some form of preconditioning, that is, a matrix B which approximates A in some sense, which is relatively cheap to solve systems with and for which the spectrum of $B^{-1}A$ (equivalently $B^{-1/2}AB^{-1/2}$ if B is s.p.d.) is more favorable for the convergence of the CG method. As it turns out, if B is symmetric and positive definite, the corresponding preconditioned version, the PCG method, is best derived by replacing the inner product with $(\mathbf{u}, \mathbf{v}) = \mathbf{u}^T B\mathbf{v}$. It takes the following form, see Algorithm 2.

Here, $[B]^{-1}$ denotes the action of B^{-1}, that is, one does not multiply with the inverse matrix B^{-1}, but normally solves a linear system with matrix B.

In order to understand what is wanted of a *good* preconditioning matrix, we discuss first some issues of major importance related to the rate of convergence of the CG method. Thereby it becomes clear that the standard spectral condition number is often too simple to explain the detailed convergence behaviour. In particular we discuss the sub- and superlinear convergence phases frequently observed in the convergence history of the conjugate gradient method.

Given	$\mathbf{x}^{(0)}$, ε	initial guess and stopping tolerance
Set	$\mathbf{x}^{(0)}$, $\mathbf{g} = A\mathbf{x} - \mathbf{b}$, $\mathbf{h} = [B]^{-1}\mathbf{g}$	
	$\delta_0 = \mathbf{g}^T\mathbf{h}$	
	$\mathbf{d} = -\mathbf{h}$	initial search direction
Repeat	until convergence	
	$\mathbf{h} = A\mathbf{d}$	
	$\tau = \delta_0/(\mathbf{d}^T\mathbf{h})$	
	$\mathbf{x} = \mathbf{x} + \tau\mathbf{d}$	new approximation
	$\mathbf{g} = \mathbf{g} + \tau\mathbf{h}$	new (iterative) residual
	$\delta_1 = \mathbf{g}^T\mathbf{g}$	
	$\mathbf{h} = [B]^{-1}\mathbf{g}$	new pseudoresidual
	$\delta_1 = \mathbf{g}^T\mathbf{h}$	
	if $\delta_1 \leq \varepsilon$ then stop	
	$\beta = \delta_1/\delta_0$, $\delta_0 = \delta_1$	
	$\mathbf{d} = -\mathbf{h} + \beta\mathbf{d}$	new search direction

Algorithm 2: Preconditioned conjugate gradient algorithm.

A preconditioner can be applied in two different manners, namely, as $B^{-1}A$ or BA. The first form implies the necessity to solve a system with B at each iteration step while the second form implies a matrix-vector multiplication with B (a *multiplicative preconditioner*). In the latter, case B can be seen as an approximate inverse of A. One can also use a hybrid form $\alpha B_1^{-1} + \beta B_2$.

The presentation here is limited to symmetric positive semidefinite matrices. It is based mainly on the articles [29, 31].

3.3. On the Rate of Convergence Estimates of the Conjugate Gradient Method. Let A be symmetric, positive semidefinite and consider the solution of $A\mathbf{x} = \mathbf{b}$ by a preconditioned conjugate gradient method. In order to understand how an efficient preconditioner to A should be chosen we must first understand some general properties of the rate of convergence of conjugate gradient methods.

3.3.1. Rate of Convergence Estimates Based on Minimax Approximation. As is well known (see e.g., [6, 30]), the conjugate gradient method is a norm minimizing method. For the preconditioned standard CG method, we have

$$\left\|\mathbf{e}^k\right\|_A = \min_{P_k \in \pi_k} \left\|P_k(B)\mathbf{e}^0\right\|_A, \tag{45}$$

where $\|\mathbf{u}\|_A = \{\mathbf{u}^T A\mathbf{u}\}^{1/2}$, $\mathbf{e}^k = \mathbf{x} - \mathbf{x}^k$ is the iteration error and π_k denotes the set of polynomials of degree k which are normalized at the origin, that is, $P_k(0) = 1$. This is a norm on the subspace orthogonal to the mullspace of A, that is, on the whole space, if A is nonsingular.

Consider the C-innerproduct $(\mathbf{u}, \mathbf{v}) = \mathbf{u}^T C\mathbf{v}$, and note that $B = C^{-1}A$ is symmetric with respect to this innerproduct, let $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n$ be orthonormal eigenvectors

and let λ_i, $i = 1, \ldots, n$ be the corresponding eigenvalues of B. Let

$$\mathbf{e}^0 = \sum_{j=1}^{n} \alpha_j \mathbf{v}_j, \tag{46}$$

be the eigenvector expansion of the initial vector where $\alpha_j = (\mathbf{e}^0, \mathbf{v}_i)$, $i = 1, \ldots, n$. Note further that the eigenvectors are both A- and C-orthogonal. Then, by the construction of the CG method, it follows

$$\mathbf{e}^k = \sum_{j=1}^{n} \alpha_j P_k\left(\lambda_j\right) \mathbf{v}_j, \tag{47}$$

and, using the nonnegativity of the eigenvalues, we find

$$\begin{aligned} \left\|\mathbf{e}^k\right\|_A &= \left\| \sum_{j=1}^{n} \alpha_j P_k(\lambda_j) \mathbf{v}_j \right\|_A = \left\{ \sum_{j=1}^{n} \alpha_j^2 \lambda_j P_k^2\left(\lambda_j\right) \right\}^{1/2} \\ &\leq \left\{ \sum_{1, \lambda_j > 0}^{n} \alpha_j^2 \lambda_j \right\}^{1/2} \max_{\substack{1 \leq i \leq n \\ \lambda_i > 0}} |P_k(\lambda_i)| \\ &= \max_{\substack{1 \leq i \leq n \\ \lambda_i > 0}} |P_k(\lambda_i)| \left\|\mathbf{e}^0\right\|_A. \end{aligned} \tag{48}$$

Here we have used the A-orthogonality of the eigenvectors. Similarly, using the C-orthogonality, we find

$$\left\|\mathbf{e}^k\right\|_C = \left\{ \sum_{j=1}^{n} \alpha_j^2 P_k^2\left(\lambda_j\right) \right\}^{1/2} \leq \max_{1 \leq i \leq n} \left| P_k\left(\lambda_j\right) \right| \left\|\mathbf{e}^0\right\|_C. \tag{49}$$

Due to the minimization property (45) there follows from (48) the familiar bound

$$\left\|\mathbf{e}^k\right\|_A \leq \min_{P_k \in \pi_k^1} \max_{\substack{1 \leq i \leq n \\ \lambda_i > 0}} |P_k(\lambda_i)| \left\|\mathbf{e}^0\right\|_A. \tag{50}$$

Estimate (50) is sharp in the respect that for every k there exists an initial vector for which equality is attained. In fact, for such a vector we necessarily have that $\alpha_j \neq 0$ if and only if α_j belongs to a set of $k + 1$ points (the so-called Haar condition) where $\max_i |P_k(\lambda_i)|$ is taken. For such an initial vector (49) shows that, if the eigenvalues are positive, we have also

$$\left\|\mathbf{e}^k\right\|_C = \min_{P_k \in \pi_k^1} \max_{1 \leq i \leq n} |P_k(\lambda_i)| \left\|\mathbf{e}^0\right\|_C. \tag{51}$$

The rate of convergence of the iteration error $\|\mathbf{e}^k\|_A$ is measured by the average convergence factor

$$\left\{ \frac{\left\|\mathbf{e}^k\right\|_A}{\|\mathbf{e}^0\|_A} \right\}^{1/k}. \tag{52}$$

Inequality (50) shows that this can be majorized with an estimate of the rate of convergence of a best polynomial approximation problem (namely the best approximation of the function $\equiv 0$, of polynomials in π_k^1) in maximum norm on the discrete set formed by the spectrum of B. Clearly, multiple eigenvalues are treated as single so the actual approximation problem is

$$\min_{P_k \in \pi_k^1} \max_{1 \leq i \leq m} \left| P_k\left(\widetilde{\lambda}_i\right) \right|, \tag{53}$$

where the disjoint positive eigenvalues $\widetilde{\lambda}_j$ have been ordered in increasing value, $0 < \widetilde{\lambda}_i < \cdots < \widetilde{\lambda}_m$, and m is the number of such eigenvalues. However, the solution of this problem requires knowledge of the spectrum, which is not available in general. Even if it is known, the estimate (53) can be troublesome in practice, since it involves approximation on a general discrete set of points.

Besides being costly to apply, such estimates do not give any qualitative insight in the behaviour of the conjugate gradient method for various typical eigenvalue distributions.

That is why we make some further assumptions on the spectrum in order to simplify the approximation problem and at the same time, present estimates which can be used both to estimate the number of iterations and to give some insight in the qualitative behaviour of the iteration method.

3.3.2. Standard Condition Number: Linear Convergence. If the eigenvalues are (densely) located in an interval $[a, b]$ where $a > 0$, we can majorize the best approximation problem on the discrete set with the best approximation problem on the interval and frequently still get a good estimate. We have

$$\min_{P_k \in \pi_k^1} \max_{1 \leq i \leq m} \left| P_k\left(\widetilde{\lambda}_i\right) \right| \leq \min_{P_k \in \pi_k^1} \max_{a \leq x \leq b} |P_k(x)|. \tag{54}$$

The solution to this min max problem is well known and uses Chebyshev polynomials. One finds that

$$\min_{P_k \in \pi_k^1} \max_{a \leq x \leq b} |P_k(x)| = \frac{1}{T_k((b+a)/(b-a))} = \frac{2\sigma^k}{1 + \sigma^{2k}}, \tag{55}$$

where $\sigma = (1 - \sqrt{a/b})/(1 + \sqrt{a/b})$, and $T_k(x) = (1/2)[(x + \sqrt{x^2 - 1}) + (x - \sqrt{x^2 - 1})^k]$, $a = \widetilde{\lambda}_1$, $b = \widetilde{\lambda}_m$. Hence, the average rate of convergence of the upper bound approaches σ as $k \to \infty$. Also, it is readily found (see [35]) that the relative iteration error $\|e^k\|_A / \|e^0\|_A \leq \varepsilon$ if

$$k = k^*(a, b, \varepsilon) = \left\lceil \frac{\ln\left((1/\varepsilon) + \sqrt{(1/\varepsilon^2) - 1}\right)}{\ln \sigma^{-1}} \right\rceil. \tag{56}$$

Here $\lceil \xi \rceil$ denotes the smallest integer not less than ξ.

It turns out that the above holds more generally if A is nonsymmetric but the eigenvalues are contained in an ellipse with foci a, b, where $b \geq a > 0$, if one replaces σ with $\hat{\sigma} = \sigma\sqrt{(1+\delta)/(1-\delta)}$, where δ is the eccentricity of the ellipse, (i.e., the ratio of the semiaxes) and $\delta < 2\sqrt{a/b}/(1 + a/b)$.

Also, in a similar way, the case of eigenvalues contained in two separate intervals or ellipses can be analysed, see, for example, [35] for further details.

When $b/a \to \infty$, $\delta = 0$, and $\varepsilon \to 0$ the following upper bound becomes an increasingly accurate replacement of (56),

$$k^* \le \left\lceil \frac{1}{2}\sqrt{\frac{b}{a}} \ln \frac{2}{\varepsilon} \right\rceil. \tag{57}$$

The above estimate of the rate of convergence and of the number of iterations show that they depend only on the condition number b/a and on the eccentricity of the ellipse, containing the eigenvalues. Therefore, except in special cases, this estimate is not very accurate. When we use a more detailed information of the spectrum and the initial error vector, sometimes substantially better estimates can be derived. This holds for instance when there are well separated small and/or large eigenvalues. Before we consider this important case, we mention briefly another similar minimax result which holds when we use *different norms* for the iteration error vector and for the initial vector.

By (48), we have

$$\begin{aligned} \left\|\mathbf{e}^k\right\|_A &= \left\{\sum \alpha_j^2 \lambda_j P_k^2(\lambda_j)\right\}^{1/2} \\ &= \left\{\sum \alpha_j^2 \lambda_j^{1-2s} \lambda_j^{2s} P_k^2(\lambda_j)\right\}^{1/2} \\ &\le \min_{P_k \in \pi_k^1} \max_{1 \le \lambda_j \le m} \left|\lambda_j^s P_k(\lambda_j)\right| \left\{\sum \alpha_j^2 \lambda_j^{(1-2s)}\right\}^{1/2} \\ &= \min_{P_k \in \pi_k^1} \max_{1 \le \lambda_j \le m} \left|\lambda_j^s P_k(\lambda_j)\right| \left\|\mathbf{e}^0\right\|_{A^{1-2s}}. \end{aligned} \tag{58}$$

If the initial vector is such that Fourier coefficients for the highest eigenvalue modes are dominating, then $\|\mathbf{e}^0\|_{A^{1-2s}}$ may exist and take not too large values even for some $s \ge 1/2$. We consider the interesting case where $s \ge 1/2$, for which the following theorem holds (see [6, 36]).

Theorem 6. *Let π_k^1 denote the set of polynomials of degree k such that $P_k(0) = 1$. Then for $k = 1, 2 \ldots$ and for any $s \ge 1/2$ such that $2s$ is an integer, it holds*

$$\frac{\left\|\mathbf{e}^k\right\|_A}{\left\|\mathbf{e}^0\right\|_A^{1-2s}} \le \min_{P_k \in \pi_k^1} \max_{0 \le x \le 1} |x^s P_k(x)| \le \left(\frac{s}{k+s}\right)^{2s}. \tag{59}$$

Remark 1. For $s = 1/2$ it holds

$$\max_{0 \le x \le 1} \left|x^{1/2} P_k(x)\right| = \frac{1}{2k+1}, \tag{60}$$

for $P_k(x) = U_{2k}(\sqrt{1-x})$ and for $s = 1$, it holds

$$\max_{0 \le x \le 1} |x P_k(x)| = \frac{1}{k+1} \tan\frac{\pi}{4k+4} < \frac{1}{(k+1)^2}, \tag{61}$$

for $P_k(x) = (x^{-1}(-1)^k)/(k+1)\tan(\pi/(4k+4))\, T_{k+1}((1+\cos(\pi/(2k+2)))x - \cos(\pi/(2k+2)))$ where $T_k(x)$ and $U_k(x)$ are the Chebyshev polynomials of kth degree of the first and second kind, respectively.

For other values (59) is an upper bound only, that is, not sharp. At any rate, it shows that the error $\|\mathbf{e}^k\|_A$ converges (initially) at least as fast as $(s/(k+s))^{2s}$, that is, as $1/(2k+1)$ for $s = 1/2$ and as $(1/(k+1))^2$ for $s = 1$.

Note that this convergence rate does not depend on the eigenvalues, in particular not on the spectral condition number.

Conclusion 1. By computing the initial approximation vector from a coarse mesh, the components for $\mathbf{e}^0$ for the first Fourier modes will be small and $\|\mathbf{e}^0\|_{A^{1-2s}}$ may take on values that are not very large even when $s = 1$ or bigger. Therefore, there is an initial decay of the residual as $O(k^{2s})$, independent of the condition number. Note, however, that the actual errors may not have decayed sufficiently even if the residual has.

We consider now upper bound estimates which show how the convergence history may enter a superlinear rate.

An Estimate to Show a Superlinear Convergence Rate Based on the K-Condition Number. A somewhat rough, but simple and illustrative superlinear convergence estimate can be obtained in terms of the so called K-condition number, (see [37, 38])

$$K = K(B) = \frac{((1/n)\,\mathrm{tr}(B))^n}{\det(B)} = \left(\frac{1}{n}\sum_{i=1}^n \lambda_i\right)^n \left(\Pi_{i=1}^n \lambda_i\right)^{-1}, \tag{62}$$

where we assume that B is s.p.d.

Note that $K^{1/n}$ equals the quotient between the arithmetic and geometric averages of the eigenvalues. This quantity is similar to the spectral condition number $\kappa(B)$ in that it is never smaller than 1, and is equal to 1 if and only if $B = \alpha I$, $\alpha > 0$ (recall that B is symmetrizable).

Based on the K-condition number, a superlinear convergence result can be obtained as follows.

Theorem 7. *Let $k < n$ be even and $k \ge 3 \ln K$. Then,*

$$\frac{\|\mathbf{e}^k\|_A}{\|\mathbf{e}^0\|_A} \le \left(\frac{3 \ln K}{k}\right)^{k/2}. \tag{63}$$

Proof. Let $k = 2m$ and the polynomial P_k be of a simplest possible form, that is, let it vanish at the m smallest and m largest eigenvalues of B. As follows from (48), we have then

$$\begin{aligned} \frac{\left\|\mathbf{e}^k\right\|_A}{\|\mathbf{e}^0\|_A} &\le \max_{\lambda_1 \le \lambda \le \lambda_n} \left|\Pi_{i=1}^m \left(1 - \frac{\lambda}{\lambda_i}\right)\left(1 - \frac{\lambda}{\lambda_{n+1-i}}\right)\right| \\ &= \Pi_{i=1}^m \max_{\lambda_i \le \lambda \le \lambda_{n+1-i}} \left(\frac{\lambda}{\lambda_i} - 1\right)\left(1 - \frac{\lambda}{\lambda_{n+1-i}}\right) \\ &= \Pi_{i=1}^m \left(\frac{(\lambda_i + \lambda_{n+1-i})^2}{4\lambda_i \lambda_{n+1-i}} - 1\right) \\ &\le \left(\left(\Pi_{i=1}^m \frac{(\lambda_i + \lambda_{n+1-i})^2}{4\lambda_i \lambda_{n+1-i}}\right)^{1/m} - 1\right)^m. \end{aligned} \tag{64}$$

The latter follows from $(\Pi_{i=1}^{m}(1-\Theta_i))^{1/m} + (\Pi_{i=1}^{m}\Theta_i)^{1/m} \le 1$ with $\Theta_i = 4\lambda_i\lambda_{n+1-i}(\lambda_i + \lambda_{n+1-i})^2$. Using now twice the inequality between the arithmetic and geometric mean values, one has

$$\begin{aligned}\Pi_{i=1}^{m}\frac{(\lambda_i+\lambda_{n+1-i})^2}{4\lambda_i\lambda_{n+1-i}} &\le \frac{\left((1/n)-2m\sum_{i=m+1}^{n-m}\lambda_i\right)^{n-2m}}{\Pi_{i=m+1}^{n-m}\lambda_i}\\ &\quad\times\frac{\Pi_{i=1}^{m}(\lambda_i+\lambda_{n+1-i}/2)^2}{\Pi_{i=1}^{m}\lambda_i\lambda_{n+1-i}}\\ &\le \frac{\left(1/n\left(\sum_{i=m+1}^{n-m}\lambda_i+\sum_{i=1}^{m}(\lambda_i+\lambda_{n+1-i})\right)\right)^{n}}{\Pi_{i=m+1}^{n-m}\lambda_i\ \Pi_{i=1}^{m}\lambda_i\lambda_{n+1-i}}\\ &= K.\end{aligned} \tag{65}$$

Using $\exp(2x) - 1 \le 2x/(1-x)$, $x < 1$, we get the required estimate,

$$\begin{aligned}\frac{\left\|\mathbf{e}^k\right\|_A}{\|\mathbf{e}^0\|_A} &\le \left(K^{2/k}-1\right)^{k/2} \le \left(\frac{2\ln K}{k-\ln K}\right)^{k/2}\\ &\le \left(\frac{2\ln K}{2k/3(k/3-\ln K)}\right)^{k/2} \le \left(\frac{3\ln K}{k}\right)^{k/2}.\end{aligned} \tag{66}$$

□

A somewhat better result of the same type exists. The estimate is of similar type, that is,

$$\frac{\|\mathbf{r}^k\|_{C^{-1}}}{\|\mathbf{r}^0\|_{C^{-1}}} \le \left(K^{1/k}-1\right)^{k/2}, \tag{67}$$

where $\mathbf{r}^k = A\mathbf{e}^k$ was obtained using more complicated techniques, see [6, 37], and the references quoted therein. Note that here as $k/\ln K \to \infty$, we have

$$\begin{aligned}\frac{\|\mathbf{e}^k\|_{C^{-1}}}{\|\mathbf{e}^0\|_{C^{-1}}} &\le \left(\mathbf{e}^{1/k\ln K}-1\right)^{k/2} \approx \left(1+\frac{\ln K}{2k}\right)^{k/2}\left(\frac{\ln K}{k}\right)^{k/2}\\ &\approx K^{1/4}\left(\frac{\ln K}{k}\right)^{k/2},\end{aligned} \tag{68}$$

that is, the simpler upper bound in (63) is asymptotically worse than this bound (albeit in a different norm) by the factor $3^{k/2}/\ln K^{1/4}$.

The upper bounds in the above estimates involve a convergence factor which decreases with increasing iteration number and show hence a superlinear rate of convergence. Note, however, that $K^{1/n} \le \kappa(B) \lesssim 4K$ (see [6]) where $\kappa(B) = \lambda_n/\lambda_1$ is the spectral condition number, so the K-condition number may take very large values when $\kappa(B)$ is large.

The estimates based on the K-condition number involve only "integral" characteristics of the preconditioned matrix (the trace and the determinant). Sometimes, it is possible to obtain a practical estimate of $K(B)$ which can be useful for the a priori construction of good preconditioners and for the a posteriori assessment of their quality, see Section 5 for further details.

The estimate

$$\frac{\left\|\mathbf{r}^k\right\|_{C^{-1}}}{\|\mathbf{r}^0\|_{C^{-1}}} \le \left(K^{1/k}-1\right)^{k/2} \le \varepsilon \tag{69}$$

shows that

$$K^{1/2}\left(1-K^{-1/k}\right)^{k/2} \le \varepsilon \tag{70}$$

or

$$\frac{k}{2}\log_2\frac{1}{1-K^{-1/k}} \ge \log_2\frac{K^{1/2}}{\varepsilon}, \tag{71}$$

which holds if

$$k > \log_2 K + 2\log_2\frac{1}{\varepsilon}. \tag{72}$$

Hence, when $K \gg \varepsilon^{-2}$ the estimated number of iterations depends essentially only on $\log_2 K$, that is, depends little on the relative accuracy ε, which indicates a fast superlinear convergence, when $k > \log_2 K$.

When actually estimating the number of iterations, Theorem 6 shows a useful result only when $k > O(\ln K) = n(\ln K^{1/n})$, that is, the quotient between the arithmetic and geometric averages of the eigenvalues, which equals $K^{1/n}$, must be close to unity and the eigenvalues must be very well clustured so $K^{1/n} = 1 + O(n^{-\varepsilon})$ for some $\varepsilon > 0$; otherwise the estimated number of iterations will be $O(n)$, which is normally a useless result. The next example illustrates this further.

Example 1. Consider a geometric distribution of eigenvalues of A, $\lambda_j = j^s$, $j = 1, 2, \ldots, n$ for some positive s. Here, asymptotically

$$\operatorname{tr}(A) = \sum_{1}^{n} j^s \sim \frac{1}{s+1}\,n^{s+1}, \quad n \longrightarrow \infty. \tag{73}$$

Using Stirling's formula, we find

$$\det(A) = \prod_{1}^{n}\lambda_j = \left(\prod_{1}^{n} j\right)^s \sim (2\pi n)^{s/2}\left(\frac{n}{e}\right)^{ns}, \quad n \longrightarrow \infty, \tag{74}$$

so,

$$\kappa(A)^{1/n} \sim \frac{e^s}{s+1}, \quad n \longrightarrow \infty. \tag{75}$$

Hence, s must be sufficiently small for the estimate in Theorem 6. to be useful. On the other hand, the spectral condition number (i) $\kappa(A) = n^s$, and the simple estimate based on $\kappa(A)$ leads to $k \sim O(n^{s/2})$ and gives hence, asymptotically, a smaller upper bound when $s < 2$. For

further discussions on superlinear rate of convergence, see [39].

3.4. Generalized Conjugate Gradient Methods. The rate of convergence estimates as given above, holds for a restricted class of matrices, symmetric or, more generally, for normal matrices.

To handle more general classes of problems for which such optimal rate of convergence results as in (45) holds, one needs more involved methods. Much work has been devoted to this problem. This includes methods like generalized minimum residual (GMRES), see Saad and Schultz [40], generalized conjugate residual (GCR), and generalized conjugate gradient (GCG), see [6] and for further details [41]. As opposed to the standard conjugate and gradient method, they require a long version of updates for the search directions, as the newest search direction at each stage is not in general, automatically (in exact precision) orthogonal to the previous search directions, but must be orthogonalized at each step. This makes the computational expense per step grow linearly and the total expense grows quadratically with the iteration index. In addition, due to finite precision, there is a tendency of loss of orthogonality, even for symmetric problems when many iterations are required. One remedy which has been suggested is to use the method only for a few steps, say 10, and restart the method with the current approximation as initial approximation.

Clearly, however, in this way, the optimal convergence property of the whole Krylov set of vector is lost. For this and other possibilities, see, for example, [42].

Another important version of the generalized conjugate gradient methods occurs when one uses variable preconditioners. Variable preconditioners, that is, a preconditioner changed from one iteration to the next iteration step, are used in many contexts.

For instance, one can use variable drop tolerance, computed adaptively, in an incomplete factorization method (see Section 4). When the given matrix is partitioned in two by two blocks, it can be efficient to use inner iterations when solving arising systems for one, or both, of the diagonal block matrices, see, for example, [43], and the flexible conjugate gradient method in Saad, [44, 45].

Due to space limitations, the above topics will not be further discussed in this paper.

4. Incomplete Factorization Methods

There exist two classes of preconditioning methods that are closely related to direct solution methods. In this paper, we make a survey only of their main ingredients, but delete many of the particular aspects.

The first method is based on incomplete factorization were some entries arising during a triangular factorization are neglected to save in memory. The deletion can be based on some drop tolerance criterion or on a normally a priori, chosen sparsity pattern. The factorization based on a drop tolerance takes the following form. During the elimination (or equivalently, triangular factorization), the off-diagonal entries are accepted only if they are not too small. For instance,

$$a_{ij} := \begin{cases} a_{ij} - a_{ir}a_{rr}^{-1}a_{rj} & \text{if } \left|a_{ij}\right| \geq \varepsilon \sqrt{a_{ii}a_{jj}}, \\ 0, & \text{otherwise.} \end{cases} \tag{76}$$

Here, ε, $0 < \varepsilon \ll 1$ is the drop-tolerance parameter. Such methods may lead to too much fill-in (i.e., $a_{ij} \neq 0$ in positions where the original entry was occupied by a zero), because to be robust, they may require near machine-precision drop tolerances. Furthermore, as direct solution methods, they are difficult to parallelize efficiently.

The incomplete factorization method can readily be extended to matrices partitioned in block form. Often, instead of a drop tolerance, one prescribes the sparsity pattern of the triangular factors in the computed preconditioner, that is, entries arising outside the chosen pattern are ignored. An early presentation of such incomplete factorization methods was given by Meijerink and van der Vorst [46]. One can make a diagonal compensation of the neglected entries, that is add them to the diagonal entries in the same row, possibly first multiplied by some scalar Θ,$0 < \Theta \leq 1$. For discussions of such approaches, see [29, 30, 47, 48]. This frequently moves small eigenvalues, corresponding to the smoother harmonics, to cluster near the origin, in this way sometimes improving the spectral condition number by an order of magnitude (see [6, 47]).

The other class of methods are based on approximate inverses G, for instance such that minimizes a Frobenius norm of the error matrix $I - GA$, see Section 5 for further details. To be sufficiently accurate these methods lead frequently to nearly full matrices. This can be understood as the matrices we want to approximate are often sparse discretizations of diffusion problems. The inverse of such an operator is a discrete Green's function which, as wellknown, often has a significantly sized support on nearly the whole domain of definition.

However, we can use an additive approximation of the inverse involving two, or more, terms which is approximate on different vector subspaces. By defining in this way the preconditioner recursively on a sequence of lower dimensional subspaces, it may preserve the accurate approximation property of the full, inverse method while still needing only actions of sparse operators.

Frequently, the given matrices are partitioned in a natural way in a two by two block form. For such matrices, it can be seen that the two approaches are similar. Consider namely

$$A = \begin{bmatrix} A_1 & A_{12} \\ A_{21} & A_2 \end{bmatrix}, \tag{77}$$

where we assume that A_1 and the Schur complement matrix $S = A_2 - A_{21}A_1^{-1}A_{12}$ are nonsingular. (This holds, in particular, if A is symmetric and positive definite.) We can construct either a block approximate factorization of A or approximate the inverse of A on additive form. As the

following shows, the approaches are related. First, a block matrix factorization of A is

$$A = \begin{bmatrix} A_1 & 0 \\ A_{21} & S \end{bmatrix} \begin{bmatrix} I_1 & A_1^{-1}A_{12} \\ 0 & I_2 \end{bmatrix}, \tag{78}$$

where I_1, I_2 denote the unit matrices of proper order. For its inverse, it holds

$$\begin{aligned} A^{-1} &= \begin{bmatrix} I_1 & -A_1^{-1}A_{12} \\ 0 & I_2 \end{bmatrix} \begin{bmatrix} A_1^{-1} & 0 \\ -S^{-1}A_{21}A_1^{-1} & S^{-1} \end{bmatrix} \\ &= \begin{bmatrix} A_1^{-1} + A_1^{-1}A_{12}S^{-1}A_{21}A_1^{-1} & -A_1^{-1}A_{12}S^{-1} \\ -S^{-1}A_{21}A_1^{-1} & S^{-1} \end{bmatrix}, \end{aligned} \tag{79}$$

or

$$A^{-1} = \begin{bmatrix} A_1^{-1} & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} -A_1^{-1}A_{12} \\ I_2 \end{bmatrix} S^{-1}\left[-A_{21}A_1^{-1},\ I_2\right]. \tag{80}$$

A straightforward computation reveals that $A_{\widetilde{V}} \equiv \widetilde{V}^T A \widetilde{V} = S$ and, hence,

$$\begin{aligned} A^{-1} &= \begin{bmatrix} A_1^{-1} & 0 \\ 0 & 0 \end{bmatrix} + \widetilde{V}\left(\widetilde{V}^T A \widetilde{V}\right)^{-1} \widetilde{V}^T \\ &= \begin{bmatrix} A_1^{-1} & 0 \\ 0 & 0 \end{bmatrix} + \widetilde{V} A_{\widetilde{V}}^{-1} \widetilde{V}^T, \end{aligned} \tag{81}$$

where

$$\widetilde{V} = \begin{bmatrix} -A_1^{-1}A_{12} \\ I_2 \end{bmatrix}. \tag{82}$$

Let $M_1 \simeq A_1$ be an approximation of A_1 (for which linear systems are simpler to solve than for A_1) and let $G_1 \simeq A_1^{-1}$ be a sparse approximate inverse. Possibly $G_1 = M_1^{-1}$. Then,

$$\begin{aligned} M &= \begin{bmatrix} M_1 & 0 \\ A_{21} & B_2 \end{bmatrix} \begin{bmatrix} I_1 & M_1^{-1}A_{12} \\ 0 & I_2 \end{bmatrix} \\ &= \begin{bmatrix} M_1 & A_{12} \\ A_{21} & A_2 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & B_2 + A_{21}M_1^{-1}A_{12} - A_2 \end{bmatrix} \end{aligned} \tag{83}$$

is a preconditioner to A and

$$B = \begin{bmatrix} G_1 & 0 \\ 0 & 0 \end{bmatrix} + V B_2^{-1} V^T \tag{84}$$

is an approximate inverse, where $V = \begin{bmatrix} -G_1A_{12} \\ I_2 \end{bmatrix}$ and B_2 is an approximation of S. If $B_2 = V^T \widetilde{A} V$, where $\widetilde{A} = \begin{bmatrix} G_1^{-1} & A_{12} \\ A_{21} & A_2 \end{bmatrix}$, then

$$\begin{aligned} B &= \begin{bmatrix} G_1 & 0 \\ 0 & 0 \end{bmatrix} + V\left(V^T \widetilde{A} V\right)^{-1} V^T \\ &= \begin{bmatrix} G_1 & 0 \\ 0 & 0 \end{bmatrix} + V S\left(\widetilde{A}\right) V^T, \end{aligned} \tag{85}$$

where $S(\widetilde{A}) = A_2 - A_{21}G_1A_{12}$. If $M_1 = G_1^{-1}$, then in this case

$$B = M^{-1}. \tag{86}$$

Hence, a convergence estimate for one method can be directly applied for the other method as well. For further discussions of block matrix preconditioners, see, for example, [49–52]. As can be seen from the above, Schur complement matrices play a major role in both matrix factorizations. For sparse approximations of Schur complement matrix, in particular element, e.g., element type approximations, see, for example [53–55].

We consider now multilevel extensions of the additive approximate inverse subspace correction method. It is illustrative to consider first the exact inverse and its relation to Gaussian (block matrix) elimination.

4.1. The Exact Inverse on Additive Form. Let then $A^{(0)} = A$ and consider a matrix

$$A^{(k)} = \begin{bmatrix} A_1^{(k)} & A_{12}^{(k)} \\ A_{21}^{(k)} & A_2^{(k)} \end{bmatrix}, \tag{87}$$

in a sequence defined by

$$A^{(k+1)} \equiv S_2^{(k)} = A_2^{(k)} - A_{21}^{(k)} A_1^{(k)^{-1}} A_{21}^{(k)}, \quad k = 0,\ 1, \ldots,\ k_0, \tag{88}$$

where each $A_1^{(k)}$ in nonsingular, being a block diagonal of a symmetric positive definite matrix. Hence, the following recursion holds

$$\begin{aligned} A^{(k)^{-1}} &= \begin{bmatrix} A_1^{(k)^{-1}} & 0 \\ 0 & 0 \end{bmatrix} \\ &\quad + \begin{bmatrix} -A_1^{(k)^{-1}} A_{12}^{(k)} \\ I_2^{(k+1)} \end{bmatrix} A^{(k+1)^{-1}} \left[A_{21}^{(k)} A_1^{(k)^{-1}},\ I_2^{(k+1)}\right], \end{aligned} \tag{89}$$

$k = 0,\ 1, \ldots, k_0$. Here, $I_2^{(k+1)}$ is the identity matrix on level $k+1$. Note that in this example the dimensions decrease with increasing level number and the final matrix (i.e., Schur complement) in the sequence is $A^{(k_0)} = S_2^{(k_0+1)}$. The above recursion can be rewritten in compact form

$$A^{-1} = \begin{bmatrix} A_1^{(0)^{-1}} & 0 \\ 0 & 0 \end{bmatrix} + U D^{-1} L, \tag{90}$$

where the kth column of the block upper triangular matrix U equals $\begin{bmatrix} -A_1^{(k)^{-1}} A_{12}^{(k)} \\ I_2^{(k+1)} \end{bmatrix}$ and $L = U^T$. Further,

$$D = \begin{bmatrix} A_1^{(k)} & & 0 \\ & A^{(2)} & \\ & & \ddots \\ 0 & & A^{(k_0)} \end{bmatrix}. \tag{91}$$

Hence, this is the (block matrix) Gaussian elimination method applied directly to form the inverse matrix. In this way, there is no need to first form the factorization $A = \tilde{L}D\tilde{U}$ and then $A^{-1} = \tilde{U}^{-1}D^{-1}\tilde{L}^{-1}$. As is wellknown and readily seen, the columns of $\tilde{U}^{-1}$ and $\tilde{L}^{-1}$ are formed directly with no additional computation, from those of $\tilde{U}$ and $\tilde{L}$, respectively. Note that $\tilde{U}^{-1}$ is upper (block) triangular and $\tilde{L}^{-1}$ is lower (block) triangular).

The matrix D contains the (block) pivot matrices which arise during the factorization. Permutations to increase the stability by finding dominating pivots can be done by replacing $A^{(k+1)}$ with $P^{(k)^T}\tilde{A}^{(k+1)}P^{(k)}$ where $\tilde{A}^{(k+1)} = P^{(k)}A^{(k+1)}P^{(k)^T}$ is the permuted matrix on which the next elimination step takes place.

An incomplete factorization method for approximate inverses can be defined by approximating each arising Schur complement matrix with some sparse matrix $\tilde{B}^{(k+1)}$ and possibly also approximating $A_1^{(k)^{-1}}$ with some matrix $B_1^{(k)}$, to yield the approximate inverse

$$B^{(k)} = \begin{bmatrix} B_1^{(k)} & 0 \\ 0 & 0 \end{bmatrix} + V^{(k)}\tilde{B}^{(k+1)}V^{(k)^T}, \tag{92}$$

where $V^{(k)} = \begin{bmatrix} -B_1^{(k)}A_{12}^{(k)} \\ I_2^{(k+1)} \end{bmatrix}$.

In forming the approximate Schur complement one can use a simpler matrix $D_1^{(k)}$than $B_1^{(k)}$, often a diagonal matrix suffices. The intermediate Schur complement matrix $\tilde{S}_2^{(k)} = A_2^{(k)} - A_{21}^{(k)}D_1^{(k)}A_{12}^{(k)}$ can be possibly further approximated by deleting certain off-diagonal entries to preserve sparsity. These entries can be compensated for by modifying the diagonal of $\tilde{S}_2^{(k)}$ to form the final approximation $\tilde{B}^{(k+1)}$. Thereby, it can be important to make the approximate Schur complement exact on some particular vector or vector space. (We do not go into these aspects further here, see [43, 56] for details.)

The eigenvectors for the smallest eigenvalues of A provide efficient column vectors for the matrix V to reduce significantly the condition number of $B\,A$ as compared to that of A. However, in general the eigenvectors are not known, and even if they are known it would be costly to apply the corresponding preconditioner as V would be a full matrix. A more viable choice is to let V be defined by the basis functions $\{\varphi_i^{(H)}\}$ of a coarse mesh (or coarsened matrix graph) so that

$$\operatorname{Im} V = \operatorname{span}\left\{\varphi_i^{(H)}\right\}. \tag{93}$$

V, V^T acts then, respectively, as prolongation and restriction operators and

$$V = \begin{bmatrix} J_{12} \\ I_2 \end{bmatrix}, \tag{94}$$

where J_{12} is the interpolation matrix from the coarse mesh (Ω_H) to the fine mesh (Ω_h), and we assume $\Omega_H \subset \Omega_h$. Further, letting the matrices be variationally defined, as in a finite element method, we have

$$A_H = V^T A V, \tag{95}$$

where A is the finite element matrix on the fine mesh.

Now, the eigenvectors for the smaller eigenvalues of A_H are normally accurate approximations of the corresponding eigenvectors for A. Furthermore, the eigenvectors of A_H are members of Im V. Therefore, the matrix V in (94) acts nearly as well as the eigenvector matrix but, in addition, is sparse. Hence the approximate inverse takes the form

$$B = G + \sigma V A_H^{-1} V^T, \tag{96}$$

where $\sigma = \lambda_{\max}(GA)$, $A_H = V^T A V$.

Here, the projection matrix

$$P = V A_H^{-1} V^T A, \tag{97}$$

projects vectors on the subspace Im V, containing normally good approximations of the eigenvectors for the smallest eigenvalues of A, that is, those who may cause severe ill-conditioning. Clearly, the approximation is more accurate as closer Ω_H is to Ω_h. However, the cost of the action of P (mainly the coarse mesh solver for the action of A_H^{-1}) increases when Ω_H expands. One can balance Ω_H to Ω_h in order to let the action of P involve the same order of computational complexity as an action of A, that is, $O(h^{-2})$ for a sparse matrix A. Assuming that the cost of an action of A_H^{-1} is $O(H^{-2.5})$ in a 2D diffusion problem (e.g., using a modified incomplete factorization method as preconditioner for the conjugate gradient method), we find $H = h^{4/5}$. The number of outer iterations with preconditioner B depends also on the choice of G. We refer the discussion of how G can be chosen properly to [56].

As an example, for a model diffusion problem with constant coefficients on a regular mesh, say for the Laplacian operator on unit square, the eigenvectors for the Laplacian $(-\Delta)$ on Ω_h with Dirichlet boundary conditions are

$$v_{k,l}^{(h)} = \sin k\pi \sin l\pi y, \quad x, y \in \Omega_h, \tag{98}$$

where $k, l = 1, 2, \ldots, h^{-1} - 1$, for the eigenvalues

$$\lambda_{k,l}^{(h)} = \left(2\sin\frac{k\pi h}{2}\right)^2 + \left(2\sin\frac{l\pi h}{2}\right)^2. \tag{99}$$

For the coarse mesh, it holds

$$v_{k,l}^{(H)} = \sin k\pi x \sin l\pi y, \quad x, y \in \Omega_H, \tag{100}$$

where $k, l, = 1, 2, \ldots, H^{-1}$, and here $V v_{k,l}^{(H)}$ are good approximations (interpolants) of the eigenvectors $v_{k,l}^{(h)}$ on Ω_h for the smallest eigenvalues.

An alternative choice of matrix V is to take eigenvectors from a nearby problem, normally defined by taking limit values of some problem parameter, see [56].

Multigrid, algebraic multilevel and algebraic multigrid methods have been presented thoroughly in, for example [29, 43, 57, 58]. Because of space limitations, they can not be presented here.

4.2. Symmetrization of Preconditioners; the SSOR and ADI Methods. As we have seen, the incomplete factorization methods require first a factorization step. There exists simpler preconditioning methods that require no factorization but have a form similar to the incomplete factorization methods. We will present two methods of this type. As an introduction, consider first an iterative method of the form

$$M\left(\mathbf{x}^{l+1}-\mathbf{x}^{l}\right)=\mathbf{b}-A\mathbf{x}^{l},\quad l=0,\ 1,\ldots, \tag{101}$$

to solve $A\mathbf{x}=\mathbf{b}$, where A and M are nonsingular. As we saw in Section 2, the asymptotic rate of convergence is determined by the spectral radius of the iteration matrix

$$B=I-M^{-1}A. \tag{102}$$

For a method such as the SOR method (which also requires no factorization), with optimal overrelaxation parameter ω (assuming that A has property A^{π} or A is s.p.d., see Section 2), the eigenvalues of the corresponding iteration matrix B are situated on a circle. No further acceleration is then possible.

There is, however, a simple remedy to this, based on taking a step in the forward direction of the chosen ordering, followed by a backward step, that is, a step in the opposite order to the vector components. This method is said to have its origin in the early days of computers when programs were stored on tapes that had to be rewound before a new forward SOR step could begin. It was found that this otherwise useless computer time for the rewinding could be better used for a backward SOR sweep!

As we will see, for symmetric and positive definite matrices the combined forward and backward sweeps correspond to a s.p.d. matrix which, contrary to the SOR method, has the advantage that it can be used as a preconditioning matrix in an iterative acceleration method. This method, called the SSOR method, will be defined later.

For an early discussion of the SSOR method, used as a preconditioner, see [59]. For discussions about symmetrization of preconditioners, see [6, 60, 61]. More generally, if A is s.p.d, we consider the symmetrization of an iterative method in the form

$$\mathbf{x}^{l+1}=\mathbf{x}^{l}+M^{-1}\left(\mathbf{b}-A\mathbf{x}^{l}\right). \tag{103}$$

For the analysis only, we consider the transformed form of (103),

$$\mathbf{y}^{l+1}=\left(I-A^{1/2}\,M^{-1}A^{1/2}\right)\mathbf{y}^{l}+\widetilde{\mathbf{b}}, \tag{104}$$

where

$$\mathbf{y}^{l}=A^{1/2}\mathbf{x}^{l},\qquad \widetilde{\mathbf{b}}=A^{1/2}M^{-1}\widetilde{\mathbf{b}}. \tag{105}$$

If M is unsymmetric, the iteration matrix $I-A^{1/2}M^{-1}A^{1/2}$ is also unsymmetric. We will now consider a method using M and another preconditioner chosen so that the iteration matrix for the combined method becomes symmetric. We call this the *symmetrization* of the method.

Let M_1, M_2 be two such preconditioning matrices. Let

$$B_i=I-\widetilde{M}_i^{-1},\qquad \widetilde{M}_i=A^{-1/2}M_iA^{-1/2}, \tag{106}$$

and consider the combined iteration matrix B_2B_1. As we will now see, it arises as an iteration matrix for the combined method

$$\begin{aligned} M_1\left(\mathbf{x}^{l+1/2}-\mathbf{x}^{l}\right)&=\mathbf{b}-Ax^{l},\\ M_2\left(\mathbf{x}^{l+1}-\mathbf{x}^{l+1/2}\right)&=\mathbf{b}-A\mathbf{x}^{l+1/2},\quad l=0,\ 1,\ \ldots. \end{aligned} \tag{107}$$

For the analysis only, we transform this to the form

$$\begin{aligned} \mathbf{y}^{l+1/2}-\mathbf{y}^{l}&=\widetilde{\mathbf{b}}^{(1)}-\widetilde{M}_1^{-1}\mathbf{y}^{l},\\ \mathbf{y}^{l+1}-\mathbf{y}^{l+1/2}&=\widetilde{\mathbf{b}}^{(2)}-\widetilde{M}_2^{-1}\mathbf{y}^{l+1/2}, \end{aligned} \tag{108}$$

where

$$\widetilde{\mathbf{b}}^{(i)}=A^{(1/2)}M_i^{-1}\mathbf{b}. \tag{109}$$

This iteration takes the form

$$\mathbf{y}^{l+1}=\widetilde{\mathbf{b}}^{(2)}+\left(I-\widetilde{M}_2^{-1}\right)\widetilde{\mathbf{b}}^{(1)}+\left(I-\widetilde{M}_1^{-1}\mathbf{y}^{l}\right), \tag{110}$$

that is,

$$\mathbf{y}^{l+1}=\widetilde{\mathbf{b}}^{(2)}+\left(I-\widetilde{M}_2^{-1}\right)\widetilde{\mathbf{b}}^{(1)}+\left(I-\widetilde{M}_2^{-1}\right)\left(I-\widetilde{M}_1^{-1}\right)\mathbf{y}^{l}, \tag{111}$$

or

$$\mathbf{y}^{l+1}=\widehat{\mathbf{b}}+B_2B_1\mathbf{y}^{l},\quad l=0,\ 1,\ldots, \tag{112}$$

where

$$\widehat{\mathbf{b}}=\widetilde{\mathbf{b}}^{(2)}+\left(I-\widetilde{M}_2^{-1}\right)\mathbf{b}^{(1)}. \tag{113}$$

For the following we need a lemma.

Lemma 2. *If A, B, and C are Hermitian positive definite and each pair of them commute, then ABC is Hermitian positive definite.*

Proof. We have $(ABC)^{*}=CBA$ and use commutativity to find

$$CBA=BCA+BAC=ABC. \tag{114}$$

Hence, ABC is Hermitian. Next, we show that the product of two s.p.d matrices that commute is positive definite. We have

$$A^{-1/2}ABA^{1/2}ABA^{1/2}=A^{1/2}BA^{1/2}, \tag{115}$$

which is Hermitian positive definite. Hence, by similarity, the eigenvalues of AB are positive and, since

$$(AB)^{*}=AB, \tag{116}$$

AB is Hermitian positive definite. In the same way, $(AB)C$ is Hermitian positive definite. □

Lemma 3. *Let A be s.p.d. and assume either of the following additional conditions:*

(a) $M_2^* = M_1$.

(b) M_1, M_2 *are s.p.d.* $\rho(A^{1/2}M_i^{-1}A^{1/2}) < 1$, $i = 1, 2$, *and the pair of matrices* M_1, M_2, *commutes.*

Then, the combined iteration method (107) *converges if and only if* $M_1 + M_2 - A$ *is s.p.d.*

Proof. It is readily seen that $\mathbf{y} = \hat{\mathbf{b}} + B_2B_1\mathbf{y}$ (i.e., the iteration method is consistent with $A\mathbf{x} = \mathbf{b}$), where $\mathbf{y} = A^{1/2}\mathbf{x}$ and $\mathbf{x} = A^{-1}\mathbf{b}$. Hence,

$$\mathbf{y} - \mathbf{y}^{l+1} = B_2B_1\left(\mathbf{y} - \mathbf{y}^l\right), \tag{117}$$

and the iteration method (112), and hence (107) converges for any initial vector if and only if $\rho(B_2B_1) < 1$, where $\rho(\cdot)$ denotes the spectral radius. But

$$\begin{aligned} B_2B_1 &= I - \widetilde{M}_1^{-1} - \widetilde{M}_2^{-1} + \widetilde{M}_1^{-1}\widetilde{M}_2^{-1} \\ &= I - A^{1/2}M_1^{-1}(M_1 + M_2 - A)M_2^{-1}A^{1/2}. \end{aligned} \tag{118}$$

It is readily seen that under either of the given conditions (a) or (b),

$$M_1^{-1}(M_1 + M_2 - A)M_2^{-1} = M_1^{-1} + M_2^{-1} - M_1^{-1}AM_2^{-1} \tag{119}$$

is symmetric. Further, it is positive definite if and only if $M_1 + M_2 - A$ is positive definite. Hence, $I - B_2B_1$ is s.p.d. Further, $B_2B_1 = (I - \widetilde{M}_2^{-1})(I - \widetilde{M}_1^{-1})$ is symmetric, and a similarity transformation shows that B_2B_1 is similar to $(I - \widetilde{M}_2^{-1})^{1/2}\ (I - \widetilde{M}_1^{-1})(I - \widetilde{M}_2^{-1})^{1/2}$, which is a congruence transformation of $I - \widetilde{M}_1^{-1}$, whose eigenvalues are positive. Hence, B_2B_1 has positive eigenvalues, so the eigenvalues of B_2B_1 are contained in the interval $(0, 1)$ and, in particular, $\rho(B_2B_1) < 1$. □

The proof of Lemma 3 shows that B_2B_1 is symmetric, so the combined iteration method is a *symmetrized version* of either of the simple methods.

Let us now consider a special class of symmetrized methods. We let A be split as $A = D + L + U$, where we assume that D is s.p.d., and let

$$V = \left(1 - \frac{1}{\omega}\right)D + L, \qquad H = \left(1 - \frac{1}{\omega}\right)D + U, \tag{120}$$

$\hat{D} = (2/\omega - 1)D$, where ω is a parameter, $0 < \omega < 2$. (Here, L and U are not necessarily the lower and upper triangular parts of A.) Note that

$$\hat{D} + V + H = A, \tag{121}$$

so this is also a splitting of A. As an example of a combined, or symmetrized, iteration method, we consider the preconditioning matrix

$$C = \left(\hat{D} + V\right)\hat{D}^{-1}\left(\hat{D} + H\right), \tag{122}$$

and show that this leads to a convergent iteration method

$$C\left(\mathbf{x}^{l+1} - \mathbf{x}^l\right) = \mathbf{b} - A\mathbf{x}^l, \quad l = 0,\ 1, \ldots. \tag{123}$$

This corresponds to choosing $M_1 = \hat{D}^{-1/2}(\hat{D} + H)$ and $M_2 = (\hat{D} + V)\hat{D}^{-1/2}$, and it can be seen that the conditions of Lemma 3 hold if the conditions in the next theorem hold.

Theorem 8. *Let* $A = D + L + U$, *where* D *is s.p.d. Let* $V, H, \hat{D}$ *be defined by* (120), *and assume that either (a) or (b) holds, where*

(a) $U = L^*$

(b) L, U *are s.p.d. and each pair of matrices* L, U, D *commute. Then the eigenvalues* λ *of the matrix* $C^{-1}A$, *where* C *is defined in* (122), *are contained in the interval* $0 < \lambda \leq 1$.

Proof. This can be shown either by verifying the conditions in Lemma 3 or more directly as follows. As in the proof of Lemma 3, it follows that C is s.p.d. Hence, the eigenvalues of $C^{-1}A$ are positive. Further,

$$C = \hat{D} + V + H + V\hat{D}^{-1}H, \tag{124}$$

so, by (121)

$$C = A + V\hat{D}^{-1}H. \tag{125}$$

Under either condition (a) or (b), $C = V\hat{D}^{-1}H$ is symmetric and positive semidefinite.

This shows that $\mathbf{x}^*C\mathbf{x} \geq \mathbf{x}^*A\mathbf{x}$ for all $\mathbf{x}$, so the eigenvalues of $C^{-1}A$ are bounded above by 1. □

We will now show that the matrix C can also efficiently be used as a preconditioning matrix, which for a proper value of the parameter ω, and under an additional condition, can even reduce the order of magnitude of the condition number. In this respect, note that when C is used as a preconditioning matrix for the Chebyshev iterative method, it is not necessary to have C scaled so that $\lambda(C^{-1}A) \leq 1$, because it is suffices then that $0 < m \leq \lambda(C^{-1}A) \leq M$, for some numbers m, M. Hence, the factor $2/\omega - 1$ in $\hat{D}^{-1}$ can be neglected.

Theorem 9. *Let* $A = D + L + U$ *be a splitting of* A, *where* A *and* D *are s.p.d. and either (a)* $U = L^*$ *or (b)* L, U *are s.p.d. and each pair of* D, L, U *commute. Then, the eigenvalues of matrix* $C^{-1}A$, *where*

$$C = \left(\frac{1}{\omega}D + L\right)\hat{D}^{-1}\left(\frac{1}{\omega}D + U\right) \tag{126}$$

and $0 < \omega < 2$, $\hat{D} = (2/\omega - 1)D$, *are contained in the interval*

$$\left[\frac{(2 - \omega)}{\left\{1 + \omega(1/\omega - 1/2)^2\delta^{-1} + \omega\gamma\right\}},\ 1\right], \tag{127}$$

where

$$\begin{aligned} \delta &= \min_{\mathbf{x} \neq 0} \frac{\mathbf{x}^TA\mathbf{x}}{\mathbf{x}^TD\mathbf{x}} \\ \gamma &= \max_{\mathbf{x} \neq 0} \frac{\mathbf{x}^T(LD^{-1}U - 1/4D)\mathbf{x}}{\mathbf{x}^TA\mathbf{x}}. \end{aligned} \tag{128}$$

Further, if there exists a vector for which $\mathbf{x}^T(L+U)\mathbf{x}^T(L+U)\mathbf{x} \le 0$, *then* $\gamma \ge -1/4$, *and if*

$$\rho\left(\widetilde{L}\widetilde{U}\right) \le \frac{1}{4}, \tag{129}$$

then $\gamma \le 0$, *and if*

$$\rho\left(\widetilde{L}\widetilde{U}\right) \le \frac{1}{4} + O(\delta), \quad \text{then } \gamma \le O(1),\ \delta \longrightarrow 0. \tag{130}$$

Here, $\widetilde{L} = D^{-1/2}LD^{-1/2}$.

Proof. It is readily seen that

$$\begin{aligned} C &= \frac{1}{2-\omega}\left(\frac{1}{\omega}D + L\right)\left(\frac{1}{\omega}D\right)^{-1}\left(\frac{1}{\omega}D + U\right) \\ &= \frac{1}{2-\omega}\left[A + \left(\frac{1}{\omega} - 1\right)D + \omega LD^{-1}U\right] \\ &= \frac{1}{2-\omega}\left[A + \omega\left(\frac{1}{\omega} - \frac{1}{2}\right)^2 D + \omega\left(LD^{-1}U - \frac{1}{4}D\right)\right]. \end{aligned} \tag{131}$$

This shows the lower bound in (127); the upper bound follows by Theorem 8. By choosing a vector for which $\mathbf{x}^T(L+U)\mathbf{x} \le 0$, it follows that

$$\frac{\mathbf{x}^T(LD^{-1}U - (1/4)D)\mathbf{x}}{\mathbf{x}^T A\mathbf{x}} \ge \frac{\mathbf{x}^T(LD^{-1}\mathbf{x} - (1/4)D\mathbf{x})}{\mathbf{x}^T D\mathbf{x}} \ge -\frac{1}{4}, \tag{132}$$

which shows $\gamma \ge -1/4$. The remainder of the theorem is immediate. □

4.2.1. The Condition Number. Theorem 9 shows that the optimal value of ω to minimize the upper bound of the condition number of $C^{-1}A$ is the value that minimizes the real-valued function

$$f(\omega) = \frac{1 + \omega(1/\omega - 1/2)^2\delta^{-1} + \omega\gamma}{2-\omega}. \tag{133}$$

It is readily seen (see Axelsson and Barker, 1984 [30]), that $f(\omega)$ is minimized for

$$\begin{aligned} \omega^* &= \frac{2}{1 + 2\sqrt{(1/2+\gamma)\delta}}, \\ \min_{\omega} f(\omega) &= f(\omega^*) = \sqrt{\left(\frac{1}{2} + \gamma\right)\delta^{-1}} + \frac{1}{2}. \end{aligned} \tag{134}$$

In general, δ is not known, but we may know that $\delta = O(h^2)$, for some problem parameter, $h \to 0$ (such as for the step length in second-order elliptic problems). Then, if $\gamma = O(1)$, $h \to 0$, we let $\omega = 2/(1 + \xi h)$ for some $\xi > 0$, in which case

$$f(\omega) = O(h^{-1}) = O\left(\sqrt{\delta^{-1}}\right), \quad h \longrightarrow 0. \tag{135}$$

This means that $C^{-1}A$ has an order of magnitude smaller condition number than A itself, which latter is $O(\delta^{-1})$.

We consider now two applications of Theorem 9.

4.2.2. The SSOR Method. In the first case, L is the lower triangular part of A or the lower block triangular part, if A is partitioned in block matrix form and $U = L^*$. Then,

$$C = \frac{1}{2-\omega}\left(\frac{1}{\omega}D + L\right)\left(\frac{1}{\omega}D\right)^{-1}\left(\frac{1}{\omega}D + L^*\right), \tag{136}$$

is a symmetrized version of the SOR method and is called the SSOR (*symmetric successive overrelaxation*) method.

As an example, for an elliptic differential equation of second order it can be seen that the condition $\rho(\widetilde{L}\widetilde{L}^T) \le 1/4$ holds for problems with Dirichlet boundary conditions and constant coefficients. For extensions of this, see Axelsson and Barker [30]. For the model difference equation on a square domain with side π, we have

$$\delta = 2\left(\sin\frac{h}{2}\right)^2, \quad \gamma \le 0, \tag{137}$$

and we find

$$\begin{aligned} \omega^* &= \frac{2}{1 + 2\sin h/2} \sim \frac{2}{1+h}, \\ f(\omega^*) &= \sqrt{\frac{1}{2\delta}} + \frac{1}{2} \sim h^{-1} + \frac{1}{2}, \quad h \longrightarrow 0. \end{aligned} \tag{138}$$

4.3. The ADI Method. In the second case of methods of (101), we let L denote the off-diagonal part of the difference operator working in the x-direction and U off-diagonal part of the difference operator in the y-direction. D is its diagonal part. Then, the matrix

$$\widehat{C} = \left(\frac{1}{\omega}D + L\right)\left(\frac{1}{\omega}D\right)^{-1}\left(\frac{1}{\omega}D + U\right), \tag{139}$$

is called an *alternating direction preconditioning matrix* and the corresponding iteration method is called the ADI (alternating direction iteration) method. In this method, we solve alternately one-dimensional difference equations in x- and y-directions. Much has been written on the ADI-method which was originally presented in Peaceman and Rachford [62]; see Varga [10], for an early influential presentation and Birkhoff et al. [63] and Wachspress [64], for instance.

As we will see, for the model difference equations we get the same optimal value of ω as in (138). The condition $\gamma = O(1)$ may be less restrictive, for the ADI-method, but the condition of commutativity is much more restrictive, as the following lemma shows.

Lemma 4. *Let A, B be two Hermitian matrices of order n. Then AB = BA if and only if A and B have a common set of orthonormal eigenvectors.*

Proof. If such a common set of eigenvectors $\{\mathbf{v}_i\}$ exists, then $A\mathbf{v}_i = \sigma_i\mathbf{v}_i$, $B\mathbf{v}_i = \tau_i\mathbf{v}_i$ and

$$AB\mathbf{v}_i = \sigma_i\tau_i\mathbf{v}_i \qquad i = 1,\ 2, \ldots, n. \tag{140}$$

Since the eigenvector space of an Hermitian matrix is complete, we therefore have

$$AB\mathbf{x} = BA\mathbf{x}, \quad \forall \mathbf{x} \in \mathcal{C}^n, \tag{141}$$

which shows that $AB = BA$. Conversely, suppose that $AB = BA$. As A is Hermitian, take U to be a unitary matrix that diagonalizes A, that is

$$\widetilde{A} = UAU^* = \begin{bmatrix} \gamma^1 I_1 & & & 0 \\ & \gamma^2 I_2 & & \\ & & \ddots & \\ 0 & & & \gamma_r I_r \end{bmatrix}, \tag{142}$$

where $\gamma^1 < \gamma^2 < \cdots < \gamma_r$ are the distinct eigenvalues of A and I_j is the identity matrix of order n_j, the multiplicity of γ_j. (Here A is posssibly permuted accordingly.) Let $\widetilde{B} = U\,BU^*$ and partition $\widetilde{B}$ corresponding to the partitioning of A, that is,

$$\widetilde{B} = \begin{bmatrix} B_{11}B_{12} & \cdots B_{1r} \\ \vdots & \\ B_{r1}B_{r2} & \cdots B_{rr} \end{bmatrix}. \tag{143}$$

Since $AB = BA$, we have

$$\widetilde{A}\widetilde{B} = U\,ABU^* = U\,BAU^* = \widetilde{B}\widetilde{A}. \tag{144}$$

Carying out the block multiplication $\widetilde{A}\widetilde{B} = \widetilde{B}\widetilde{A}$, we find that this, in turn, implies $B_{ij} = 0$, $i \neq j$, since $\gamma_i \neq \gamma_j$, $i \neq j$. Simply stated, a (block) matrix commutes with a (block) diagonal matrix if and only if it is itself (block) diagonal. Hence, $\widetilde{B}$ is block diagonal and each Hermitian submatrix $B_{i,i}$ has n_i orthonormal eigenvectors that are also eigenvectors of the submatrix $\gamma_i I_i$ of $\widetilde{A}$. Since $\sum_{i=1}^{r} n_i = n$ and all eigenvectors are orthonormal, A and B must have the same set of eigenvectors. □

For the second-order elliptic difference equation in two space dimensions, it turns out that the commutativity of L and U essentially corresponds to the property that the original problem is separable, that is, that solutions of $\mathcal{L}u = f$ can be written in the form $u = \varphi(x)\psi(y)$. This means that the coefficients $a(x, y)$ and $b(x, y)$ in the differential operator $\partial/\partial x[a(x, y)\partial u/\partial x] + \partial/\partial y[b(x, y)\partial u/\partial x] + c(x, y)u$ must satisfy $a(x, y) = a(x)$,$b(x, y) = b(y)$, and $c(x, y) = c$, a constant. Hence, if $a(x, y) = b(x, y)$, then $a(x, y) = b(x, y) = a$, a constant. Furthermore, the convex closure of the meshpoints must be a rectangle with sides parallel to the coordinate axes (Varga, [10] 1962). If $A = A_1 + A_2$, the ADI-method can be written in the form

$$\begin{aligned} (I + \tau_1 A_1)\mathbf{x}^{l+1/2} &= (I - \tau_1 A_2)\mathbf{x}^l + \tau_1 \mathbf{b}, \\ (I + \tau_2 A_2)\mathbf{x}^{l+1} &= (I - \tau_2 A_1)\mathbf{x}^{l+1/2} + \tau_2 \mathbf{b}, \quad l = 0,\ 1, \ldots. \end{aligned} \tag{145}$$

This is the *Peaceman-Rachford* [62] *iteration method*. The iteration matrix M is similar to

$$(I - \tau_2 A_1)(I + \tau_1 A_1)^{-1}(I - \tau_1 A_2)(I + \tau_2 A_2)^{-1}. \tag{146}$$

When A_1, A_2 are Hermitian positive definite, their eigenvalues $\lambda_i^{(1)}$, $\lambda_i^{(2)}$ are positive, and

$$\begin{aligned} \left\|(I - \tau_2 A_1)(I + \tau_1 A_1)^{-1}\right\|_2 &= \rho\left((I - \tau_2 A_1)(I + \tau_1 A_1)^{-1}\right) \\ &= \max_i \left| \frac{1 - \tau_2 \lambda_i^{(1)}}{1 + \tau_1 \lambda_i^{(1)}} \right|. \end{aligned} \tag{147}$$

Thus,

$$\begin{aligned} \rho(M) &= \rho\left((I - \tau_2 A_1)(I + \tau_1 A_1)^{-1}(I - \tau_1 A_2)^{-1}(I + \tau_2 A_2)^{-1}\right) \\ &\leq \left\|(I - \tau_2 A_1)(I + \tau_1 A_1)^{-1}(I - \tau_1 A_2)(I + \tau_2 A_2)^{-1}\right\|_2 \\ &\leq \left\|(I - \tau_2 A_1)(I + \tau_1 A_1)^{-1}\right\|_2 \\ &\quad \times \left\|(I - \tau_1 A_2)(I + \tau_2 A_2)^{-1}\right\|_2 \\ &= \mu(\tau_1, \tau_2), \\ \rho(M) &\leq \mu(\tau_1, \tau_2) = \max_i \left| \frac{1 - \tau_2 \lambda_i^{(1)}}{1 + \tau_1 \lambda_i^{(1)}} \right| \max_i \left| \frac{1 - \tau_1 \lambda_i^{(2)}}{1 + \tau_2 \lambda_i^{(2)}} \right|. \end{aligned} \tag{148}$$

Note that for $\tau_1 = \tau_2 = \tau > 0$, $\mu(\tau_1, \tau_2) = \mu(\tau, \tau) < 1$, so we have $\rho(M) < 1$, that is convergence for any $\tau > 0$. This holds even if A_1, A_2 do not commute. Note also that when A_1 and A_2 commute, we have

$$\rho(M) = \mu(\tau_1,\ \tau_2). \tag{149}$$

Let us continue the analyses for the general case where A_1, A_2 do not necessarily commute. We want to compute the optimal values of τ_1 and τ_2 such that $\mu(\tau_1, \tau_2)$ is minimized. For simplicity, we assume that α, β are the same lower and upper bounds of the eigenvalues of A_1 and A_2, that is, $0 < \alpha \leq \lambda_i^{(j)} \leq \beta$, $j = 1, 2$. We have

$$\begin{aligned} \mu(\tau_1, \tau_2) \leq \max &\left\{ \left| \frac{1 - \tau_2 \alpha}{1 + \tau_1 \alpha} \right|,\ \left| \frac{1 - \tau_2 \beta}{1 + \tau_1 \beta} \right| \right\} \\ &\times \max \left\{ \left| \frac{1 - \tau_1 \alpha}{1 + \tau_2 \alpha} \right|,\ \left| \frac{1 - \tau_1 \beta}{1 + \tau_2 \beta} \right| \right\}. \end{aligned} \tag{150}$$

We want to choose $\tau_1, \tau_2 > 0$ such that this bound is as small as possible. Note, then, that for such values of τ_1, τ_2 we must have $1 - \tau_i \alpha > 0$ and $1 - \tau_i \beta < 0$. Next note that each factor in the bound (150) is minimized when

$$\frac{1 - \tau_2 \alpha}{1 + \tau_1 \alpha} = \frac{\tau_2 \beta - 1}{\tau_1 \beta + 1}, \qquad \frac{1 - \tau_1 \alpha}{1 + \tau_2 \alpha} = \frac{\tau_1 \beta - 1}{\tau_2 \beta + 1}, \tag{151}$$

respectively, that is, when

$$\tau_1\tau_2 - \frac{\alpha+\beta}{2\alpha\beta}(\tau_1-\tau_2) - \frac{1}{\alpha\beta} = 0,$$
$$\tau_1\tau_2 + \frac{\alpha+\beta}{2\alpha\beta}(\tau_1-\tau_2) - \frac{1}{\alpha\beta} = 0, \tag{152}$$

respectively. Thus, both factors are simultaneously minimized when $\tau_1 = \tau_2$, and then $\tau_1 = \tau_2 = 1/\sqrt{\alpha\beta}$.

Theorem 10. *Let $A = A_1 + A_2$, where A_1, A_2, are s.p.d., and consider the Peaceman-Rachford ADI method* (145) *to solve $A\mathbf{x} = \mathbf{b}$ with $\tau_1 = \tau_2 = 1/\sqrt{\alpha\beta}$. The spectral radius of the corresponding iteration matrix M satisfies*

$$\rho(M) \le \min_{\tau_1,\tau_2} \mu(\tau_1,\tau_2) \le \left(\frac{1-\sqrt{\alpha/\beta}}{1+\sqrt{\alpha/\beta}}\right)^2 \sim 1 - 4\sqrt{\frac{\alpha}{\beta}}, \tag{153}$$

if $\alpha/\beta \to 0$.

Proof. For $\tau_1 = \tau_2 = 1/\sqrt{\alpha\beta}$, we have

$$\mu(\tau_1,\tau_2) = \left(\frac{1-\sqrt{\alpha/\beta}}{1+\sqrt{\alpha/\beta}}\right)^2. \tag{154}$$

□

Remark 2. For a model difference equation for a second-order elliptic differential equation problem on a square with side π, we have with stepsize h,

$$\alpha = \left(\frac{\sin(h/2)}{h/2}\right)^2 \sim 1,$$
$$\beta = \left(\frac{\cos(h/2)}{h/2}\right)^2 \sim \frac{4}{h^2}, \quad h \longrightarrow 0. \tag{155}$$

Then,

$$\mu(\tau_1,\tau_2) = \left(\frac{1-\tan(h/2)}{1+\tan(h/2)}\right)^2 = \frac{1-\sin(h)}{1+\sin(h)} \sim 1-2h, \quad h \longrightarrow 0. \tag{156}$$

Note that this is just the convergence factor we get for the SOR method with an optimal overrelaxation parameter.

Since $\rho(M) \le \mu(\tau_1,\tau_2)$, this means that the ADI method with parameters (chosen as above) converges at least as fast as the SOR method. Note, however, that in the ADI-method we must solve two systems of equations with tridiagonal coefficient matrices $(I - \tau A_i)$ on each step, while the pointwise SOR method requires no solution of such systems.

4.4. The Commutative Case. Assume now that A_1, A_2 commute. Then, as we have seen, M is symmetric and has real eigenvalues, and we can apply the Chebyshev acceleration method. The eigenvalues of the corresponding preconditioned matrix $\widetilde{C}$ are related to the eigenvalues of M by

$$\lambda\left(\widetilde{C}\right) = 1 - \lambda(M). \tag{157}$$

Since $-\rho(M) \le \lambda(M) \le \rho(M)$, where

$$\rho(M) = \left(\frac{1-\sqrt{\alpha/\beta}}{1+\sqrt{\alpha/\beta}}\right)^2 \sim 1 - 4\sqrt{\frac{\alpha}{\beta}}, \tag{158}$$

we have

$$4\sqrt{\frac{\alpha}{\beta}} \sim 1 - \rho(M) \le \lambda\left(\widetilde{C}\right) = 1 + \rho(M) \sim 2 - 4\sqrt{\frac{\alpha}{\beta}} \sim 2, \quad \frac{\alpha}{\beta} \longrightarrow 0. \tag{159}$$

The asymptotic rate of convergence of the Chebyshev accelerated method therefore is

$$2\sqrt{\frac{1-\rho(M)}{1+\rho(M)}} \sim 2\sqrt{2}\left(\frac{\alpha}{\beta}\right)^{1/4}, \quad \alpha/\beta \longrightarrow 0. \tag{160}$$

For the model difference equation, we have the asymptotic rate of convergence

$$\sim 2h^{1/2}, \quad h \longrightarrow 0. \tag{161}$$

4.5. The Cyclically Repeated ADI Method. The real power of the ADI method is brought forth when we use a sequence of parametrs τ_l. Assume that A_1, A_2 commute, then we choose the parameters τ_l cyclically. With a cycle of q parameters and with the assumption

$$0 < \alpha \le \lambda_i^{(j)} \le \beta, \quad j = 1,\ 2, \tag{162}$$

we get the iteration matrix

$$M^{(q)} = \prod_{p=1}^{q} \left(I + \tau_p A_2\right)^{-1}\left(I - \tau_p A_1\right)\left(I + \tau_p A_1\right)^{-1}\left(I - \tau_p A_2\right). \tag{163}$$

The eigenvalues of $M^{(q)}$ are

$$\prod_{p=1}^{q} \frac{1-\tau_p\lambda_i^{(1)}}{1+\tau_p\lambda_i^{(1)}} \cdot \frac{1-\tau_p\lambda_i^{(2)}}{1+\tau_p\lambda_i^{(2)}}. \tag{164}$$

In the same way as above, $\rho(M^{(q)})$ is minimized when

$$d(\alpha,\beta,q) = \max_{\alpha\le x\le\beta} \prod_{p=1}^{q} \left|\frac{1-\tau_p x}{1+\tau_p x}\right|. \tag{165}$$

4.6. A Preconditioning Method for Complex Valued Matrices. Complex valued systems of equations arise in many applications. A commonly occuring case is the solution of a matrix polynomial equation

$$Q_m(A)x = b, \tag{166}$$

where A is a real square matrix and Q_m is a polynomial of degree m that has no zeroes at the eigenvalues of A. Here Q_m can be factored in the product of second degree, and possibly some factors of first degree polynomials with real coefficients.

The second degree polynomials can be factored in products of first degree polynomials with complex coefficients.

Consider then a linear system

$$Az = b. \tag{167}$$

in the form

$$(\mathcal{R} + iS)(x + iy) = c + id, \tag{168}$$

where $\mathcal{R}, S$ are real matrices of order n and $x, y, c, d \in \mathcal{R}^n$.

The system can be solved in complex arithmetic. However, complex arithmetic leads to heavier computational complexity and it is in general difficult to precondition complex valued matrices, as the eigenvalues of the given matrix or the preconditioned matrix can be spread in the whole complex plane and the iterative solution method may then converge too slowly.

One can alternatively apply a preconditioned conjugate gradient method to the Hermitian positive definite normal matrix system $A^H Au = A^H b$ for which the eigenvalues are real. At any rate, this involves complex arithmetic that costs typically three to four times as much as corresponding real arithmetic.

Complex arithmetic can be avoided by rewriting (168) in real valued form, such as

$$A^{(1)}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} R & -S \\ S & R \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} c \\ d \end{bmatrix}, \tag{169}$$

or

$$A^{(2)}\begin{bmatrix} x \\ -y \end{bmatrix} = \begin{bmatrix} R & S \\ S & -R \end{bmatrix}\begin{bmatrix} x \\ -y \end{bmatrix} = \begin{bmatrix} c \\ d \end{bmatrix}. \tag{170}$$

The block matrices are here real but, in general, nonsymmetric and/or indefinite. For the solution, one can use a generalized conjugate gradient method such as GMRES [40] or GCG [65].

For $A^{(1)}$ it holds that any eigenvalue λ appears in complex conjugate pairs $\lambda, \overline{\lambda}$. For $A^{(2)}$, which is real symmetric, for any eigenvalue $\lambda \neq 0, -\lambda$ is also an eigenvalue. Thus, the spectrum $\sigma(A^{(1)})$ is symmetric with respect to the real axes and the spectrum $\sigma(A^{(2)})$ is symmetric with respect to the imaginary axes, that is, in both cases the spectrum embraces the origin. From best polynomial approximation properties it is known that such point distributions leads to polynomials of essentially a square degree as for the same approximation accuracy compared to the case with a one-sided spectrum.

In [21], one finds further explanations why Krylov subspace methods can be inefficient for solving complex valued systems, represented in the above real forms. Several iterative solution methods, such as the QMR [22] have been developped and proven to be efficient for these types of problems. However, it is difficult to precondition complex valued matrices and unpreconditioned methods converge in general very slowly.

Following [66], we consider here instead an approach based on rewriting the equation in the form (170).

Instead of solving the full block matrix system we apply a Schur complement approach by the elimination of one component, which results in the following reduced system

$$Cx = f, \tag{171}$$

where

$$\begin{aligned} C &= R + SR^{-1}S, \\ f &= c + SR^{-1}d. \end{aligned} \tag{172}$$

As an introduction, assume first that R is symmetric and positive definite and S is symmetric and positive semidefinite. As a preconditioner to the matrix C in (171) we take $R + S$.

For the generalized eigen value problem,

$$\mu(R + S)z = (R + SR^{-1}S)z, \tag{173}$$

it holds then

$$\mu(I + H)y = (I + H^2)y, \tag{174}$$

where $\mu = R^{1/2}z$ and $H = R^{-1/2}SR^{-1/2}$.

If $\lambda Rz = Sz$, $z \neq 0$, or, equivalently, $Hy = \lambda y$, $y \neq 0$, it follows from (174) that

$$\mu = \frac{1 + \lambda^2}{(1 + \lambda)^2}, \tag{175}$$

that is,

$$\mu = \frac{1}{1 + (2\lambda/(1 + \lambda^2))}. \tag{176}$$

Since, by assumption $\lambda \geq 0$, it follows

$$\frac{1}{2} \leq \mu \leq 1, \tag{177}$$

and the condition number satisfies the bound $\mathcal{K}((R + S)^{-1}C) \leq 2$.

The correspondingly preconditioned conjugate gradient method to solve (174) converges therefore rapidly

There exists an even more efficient form of the iteration matrix that also shows that we can weaken the assumptions made on R and S. Hence, consider

$$\begin{aligned} Rx - Sy &= c, \\ Sx + Ry - d \end{aligned} \tag{178}$$

and assume that α is a real parameter such that $R + \alpha S$ is nonsingular. Such a parameter exists if $\ker(R) \cap \ker(S) = \emptyset$.

Multiplying the first equation by $-\alpha(R+\alpha S)^{-1}$, the second by $(R+\alpha S)^{-1}$ and adding yields

$$(R+\alpha S)^{-1}(S-\alpha R)x + y = (R+\alpha S)^{-1}(d-\alpha c). \tag{179}$$

Now multiplying this equation by S, using $Sy = Rx - c$, and rewriting the equation properly, we find

$$r \equiv Rx - c + S(R+\alpha S)^{-1}((S-\alpha R)x + \alpha c - d) = 0. \tag{180}$$

When solving the system by iteration, such as by Chebyshev iterations, r will be the residual and we observe that r can be written in the form (see (179))

$$\begin{aligned} y &= (R+\alpha S)^{-1}((S-\alpha R)x + \alpha c - d), \\ r &= Rx - Sy - c. \end{aligned} \tag{181}$$

In this form there is no need to compute the right hand side vector f initially as if (168) is used and the vector y is found during the iteration process. This saves two solutions with the matrix $R+\alpha S$. Since we need few iterations, such a saving can be important to decrease the total expense of the method.

To solve (179) we use $R+\alpha S$ as a preconditioner. The resulting preconditioned matrix takes the form

$$\begin{aligned} M_\alpha &= (R+\alpha S)^{-1}\left[R + S(R+\alpha S)^{-1}(S-\alpha R)\right] \\ &= (R+\alpha S)^{-1}[(R+\alpha S) - \alpha S](R+\alpha S)^{-1}R \\ &\quad + (R+\alpha S)^{-1}S(R+\alpha S)^{-1}S \\ &= \left((R+\alpha S)^{-1}R\right)^2 + \left((R+\alpha S)^{-1}S\right)^2. \end{aligned} \tag{182}$$

This form can also be used to derive eigenvalue estimates in more general cases than was done above. If R is nonsingular, we find

$$M_\alpha = \left(I + \alpha R^{-1}S\right)^{-2}\left(I + R^{-1}S\right)^2. \tag{183}$$

Therefore, the preconditioned matrix is a rational function in the matrix $R^{-1}S$. It follows that the eigenvalues μ of E_α satisfy

$$\mu = \frac{1+\lambda^2}{(1+\alpha\lambda)^2}, \tag{184}$$

were λ is an eigenvalue of $R^{-1}S$.

We want to choose α to minimize the spectral condition number

$$\mathcal{K}(M_\alpha) = \frac{\mu_{\max}}{\mu_{\min}}. \tag{185}$$

Theorem 11. *Assume that R is s.p.d and S is s.p.s-d. Then, the extreme eigenvalues of the preconditioned matrix M_α, defined in* (182) *satisfy*

$$\mu_{\min} = \begin{cases} \dfrac{1}{1+\alpha^2} & \text{if } 0 \le \alpha \le \hat{\lambda}, \\ \dfrac{1+\hat{\lambda}^2}{\left(1+\alpha\hat{\lambda}\right)^2} & \text{if } \alpha \ge \hat{\lambda}, \end{cases}$$
$$\mu_{\max} = \begin{cases} 1 & \text{if } \hat{\alpha} \le \alpha, \\ \dfrac{1+\hat{\lambda}^2}{\left(1+\alpha\hat{\lambda}\right)^2} & \text{if } 0 \le \alpha \le \hat{\alpha}, \end{cases} \tag{186}$$

where $\hat{\lambda}$ is the maximal eigenvalue of $R^{-1}S$,

$$\begin{aligned} R^{-1}S &\le \hat{\lambda} I, \\ \hat{\alpha} &= \frac{\hat{\lambda}}{1+\sqrt{1+\hat{\lambda}^2}}. \end{aligned} \tag{187}$$

The spectral condition number is minimized when $\alpha = \hat{\alpha}$, in which case

$$\begin{aligned} \mu_{\min} &= \frac{1}{1+\hat{\alpha}^2}, \quad \mu_{\max} = 1, \\ \mathcal{K}(M_\alpha) &= 1 + \hat{\alpha}^2 = 2\frac{\sqrt{1+\hat{\lambda}^2}}{1+\sqrt{1+\hat{\lambda}^2}}. \end{aligned} \tag{188}$$

Proof. The bounds of the extreme eigenvalues follow by elementary computations of $\mu = (1+\lambda^2)/(1+\alpha\lambda)^2$, $0 \le \lambda \le \hat{\lambda}$. Similarly, it is readily seen that $\mu_{\max}/\mu_{\max}$ is minimized for some α in the interval $\hat{\alpha} \le \alpha \le \hat{\lambda}$, where $\mu_{\max} = 1$. Hence, it is minimized for $\alpha = \arg\max_{\hat{\alpha}\le\alpha}(1+\alpha^2)^{-1}$, that is, for $\alpha = \hat{\alpha}$. □

For applications, see [66]. An important application arises when one uses Padé type approximations, and related implicit Runge-Kutta methods (see [67]), to solve initial value problems.

4.7. Historical Remarks. Because incomplete factorization methods has had a strong influence on the development of preconditioning methods we give here some historical remarks.

The idea of an incomplete factorization method goes back to early papers by Buleev [68], Varga [10], Oliphant [69], Dupont et al. [70], Dupont [71], and Woźnicki [72], where it was presented for matrices of a type arising from difference approximations of elliptic problems. The first more general form (unmodified methods for pointwise matrices) was studied for M-matrices by Meijerink and van der Vorst [46]. For a review and general formalism for describing such methods, see Axelsson [18], Birkhoff et al. [63], Beauwens [12], and Il'in [60]. For a similar but more involved type of methods for difference matrices, which

allowed for variable parameters from one iteration to the next, see Stone [73].

A modified form of the method, where a certain row sum criterion was imposed, was studied by Gustafsson [47]. Actually, as is readily seen, the method of Dupont et al. [70] and as further discussed in Axelsson [59], using a perturbation technique, can be seen as a modified version of the general incomplete factorization method when applied to the five-point elliptic difference matrices, assuming that no fill-in is accepted outside the sparsity structure of A itself and assuming a natural ordering of the grid points. The advantage of modified versions is that they can give condition numbers of the iteration matrices that are of an order of magnitude smaller than for the original matrix.

The incomplete factorization method can be readily generalized to matrices partitioned in block matrix form. This was done first for matrices partitioned in block tridiagonal form in Axelsson et al. [49] and Concus et al. [50], the latter being based on earlier work by Underwood [74]. A general form was presented in Axelsson [67] and Beauwens and Ben Bouzid [52], where existence of the method was proven for M-matrices.

The existence, that is, the existence of nonzero pivot entries of pointwise incomplete factorization methods for M-matrices was first shown by Meijerink and van der Vorst [46] and, for pointwise H-matrices, by Varga et al. [75]. The existence of incomplete factorization methods for M-matrices in block form was shown in Axelsson et al. [49] and Concus et al. [50] for block tridiagonal matrices; in Axelsson [76] and Beauwens and Ben Bouzid [52], for general block matrices; and in Axelsson and Polman [51] for relaxted versions of such methods.

Kolotilina [77] shows the existence of convergent splittings for block H-matrices, and Axelsson [78] shows the existence of general incomplete factorizations for block H-matrices.

5. Approximate Inverses Methods

In many applications, it is of interest to compute approximations of the inverse (A^{-1}) of a given matrix A, such that these approximations can be readily used in various iterative methods.

Let G denote an approximation of A^{-1}.

Following [6], first we present an example of an explicit and an implicit method, which is followed by a general framework for computing approximate inverses. At the end, we present an efficient way to construct symmetric and positive definite approximate inverses.

An approximate inverse to a given operator may be constructed in several ways. The simplest way is to use a Neumann expansion, that is let $D^{-1}A = I - E$, where D is the diagonal of A, for instance.

Assuming that $\|E\| < 1$, then the expansion

$$A^{-1} = (I - E)^{-1}D^{-1} = (I + E + E^2 + \cdots)D^{-1} \tag{189}$$

is convergent and any truncated part of this series provides an approximate inverse. However, this will normally give poore approximations. As we will see, more accurate approximate inverses can be constructed as best, possibly weighted, Frobenius norm approximations.

In many applications the matrix A is sparse, but the exact inverse will be just a full matrix. A natural condition on G then arises: we can impose that G has some a priori chosen sparsity pattern (the same as A or different) which will make the calculations with G easy and cheap, and also will provide a sufficient accuracy.

Let A have order n and $\underline{S} = \{(i, j), 1 \le i \le n;\ 1 \le i \le j \le n\}$. Any proper subset S of $\underline{S}$ will be referred to as a sparsity pattern $S \subset \underline{S}$.S_L denotes the corresponding sparsity pattern for the lower triangular matrix and $S_{\tilde{L}}$ denotes the corresponding sparsity pattern for the strictly lower triangular matrix.

For simplicity, we use the same notation S for matrices having sparsity pattern S. Thus, $A \in S$ if $a_{ij} \neq 0 \Leftrightarrow (i, j) \in S$.

5.1. Explicit Methods . In these methods, an approximation of the inverse A^{-1} of a given nonsingular matrix A is computed explicitly, that is, without solving a linear globally coupled system of equations.

Let S be a sparsity pattern. We want to compute $G \in S$, such that

$$(GA)_{ij} = \delta_{ij}, \quad (i, j) \in S, \tag{190}$$

that is

$$\sum_{k:(i,k)\in S} g_{ik}a_{kj} = \delta_{ij}, \quad (i, j) \in S. \tag{191}$$

Some observations can be made from (191):

(i) the elements in each row of G can be computed independently;

(ii) even if A is symmetric, G in not necessarily symmetric, because $g_{i,j}$, $j \neq i$, and $g_{j,i}$ are, in general, not equal.

5.2. Implicit Methods . These methods require that A is factored first. In practice, they are used mainly for band or "envelope" matrices. The algorithm was presented in [79]. It is based on an idea in [80]; see also [81].

Suppose that $A = LD^{-1}U$ is a triangular matrix factorization of A. If A is a band matrix then L and U are also band matrices.

Let

$$L = I - \tilde{L}, \quad U = I - \tilde{U}, \tag{192}$$

where $\tilde{L}$ and $\tilde{U}$ are strictly lower and upper triangular matrices correspondingly.

The following lemma holds.

Lemma 5. *Using the above notations it holds that*

(i) $A^{-1} = DL^{-1} + \tilde{U}A^{-1}$,

(ii) $A^{-1} = U^{-1}D + A^{-1}\ \tilde{L}$.

Proof. Consider the following

$$A = LD^{-1}U \Longrightarrow A^{-1} = U^{-1}DL^{-1} \Longrightarrow \left(I - \widetilde{U}\right)A^{-1} = DL^{-1} \Longrightarrow A^{-1} = DL^{-1} + \widetilde{U}A^{-1}. \tag{193}$$

Also,

$$A^{-1}\left(I - \widetilde{L}\right) = U^{-1}D \Longrightarrow A^{-1} = U^{-1}D + A^{-1}\widetilde{L}. \tag{194}$$

□

Since DL^{-1} is lower triangular and $\widetilde{U}$ is upper triangular, using (i) we can compute entries in the upper triangular part of A^{-1} with no need to use entries of L^{-1}. Similarly, using (ii) we can compute entries of the lower triangular part A^{-1} without computing U^{-1}.

Suppose now that A is a block banded matrix with a semibandwidth p, and we want to form A^{-1} also as block banded with a semibandwidth $q : q \geq p$. The identities (i) and (ii) can be used then for the computation of the upper and lower parts of A^{-1}.

Remark 3. (i) The algorithm involves only matrix × matrix operations.

(ii) There is no need to compute any entries outside the bands.

(iii) If A is symmetric then it suffices executing only (i) or (ii).

(iv) It can be seen that $(A^{-1})_{nn} = D_{nn}^{-1}$.

There are two drawbacks with the above algorithm. It requires first the factorization $A = LD^{-1}U$ and even if A is s.p.d, the band matrix part of A^{-1}, which is computed, need not be s.p.d. The next example illustrates this.

Example 2. Consider an s.p.d. matrix

$$G = \begin{bmatrix} 1 & -2 & 1 \\ -2 & 5 & -3 \\ 1 & -3 & 4 \end{bmatrix}, \qquad G_{\text{band}} = \begin{bmatrix} 1 & -2 & 0 \\ -2 & 5 & -3 \\ 0 & -3 & 4 \end{bmatrix}, \tag{195}$$

is indefinite.

5.3. A General Framework for Computing Approximate Inverses. It turns out that both the explicit and implicit method can be characterized as methods to compute best approximations of A^{-1} of all matrices having a given sparsity pattern, in some norm. The basic idea is due to Kolotilina and Yeremin [38, 79], see [6]. Recall that the trace function is defined by $\text{tr}(A) = \sum_{i=1}^{n} a_{ii}$, which also equals $\sum_{i=1}^{n} \lambda_i(A)$. Let a sparsity pattern S be given. Consider the functional

$$F_W(G) \equiv \|I - GA\|_W^2 = \text{tr}\left((I - GA)W(I - GA)^T\right), \tag{196}$$

where the weight matrix W is s.p.d. If $W \equiv I$ then $\|I - GA\|_I$ is the Frobenius norm of $I - GA$.

Clearly $F_W(G) \geq 0$. If $G = A^{-1}$ then $F_W(G) = 0$. We want to compute the entries of G in order to minimize $F_W(G)$, that is, to find $\hat{G} \in S$, such that

$$\left\|I - \hat{G}A\right\|_W \leq \|I - GA\|_W, \quad \forall G \in S. \tag{197}$$

The following properties of the trace function will be used

$$\text{tr}\, A = \text{tr}\, A^T, \ \text{tr}(A + B) = \text{tr}\, A + \text{tr}\, B. \tag{198}$$

Then,

$$\begin{aligned} F_W(G) &= \text{tr}(I - GA)W(I - GA)^T \\ &= \text{tr}\left(W - GAW - W(GA)^T + GAW(GA)^T\right) \\ &= \text{tr}\, W - \text{tr}\, GAW - \text{tr}\,(GAW)^T + \text{tr}\, GAWA^TG^T. \end{aligned} \tag{199}$$

Further, as we are interested in minimizing F_W with respect to $G \in S$, we consider the entries $g_{i,j}$ as variables. The necessary condition for a minimizing point are then

$$\frac{\partial F_W(G)}{\partial g_{ij}} = 0, \quad (i, j) \in S. \tag{200}$$

From (199) and (200), we get

$$-2\left(WA^T\right)_{ij} + 2\left(GAWA^T\right)_{ij} = 0, \tag{201}$$

or

$$\left(GAWA^T\right)_{ij} = \left(WA^T\right)_{ij}, \quad (i, j) \in S. \tag{202}$$

Depending on the particular matrix A and the choice of S and W, (202) may or may not have a solution. We give some examples where a solution exists.

Example 3. Let A be s.p.d. Choose $W = A^{-1}$ which is also s.p.d. Then, (202) implies

$$(GA)_{ij} = \delta_{ij}, \quad (i, j) \in S, \tag{203}$$

which is the formula for the previously presented explicit method which, hence, is a special case of the more general framework for computing approximate inverses using weighted Frobenius norm.

Example 4. Let $W = (A^TA)^{-1}$. Then (202) implies

$$(G)_{ij} = (A^{-1})_{ij}, \quad (i, j) \in S, \tag{204}$$

which is the relation for the previously presented implicit method. In this case the entries of G are the corresponding entries of the exact inverse.

Example 5. Let $W = I$. Then,

$$F_W(G) = n - \operatorname{tr}(GA),$$
$$\left(GAA^T\right)_{ij} = \left(A^T\right)_{ij}, \quad (i,j) \in S. \tag{205}$$

This method is also explicit.

We can expect that such methods will be accurate only if all elements of A which are not used in the computations are zero or are relatively small. In some cases the quality of the computed approximation G to A^{-1} can be significantly improved using diagonal compensation of the entries of A which are outside S. The best approximation G to A^{-1} in a (weighted) Frobenius norm is in general not symmetric and, as we have seen, not always positive definite. For this reason, the next, alternate method, is considered.

5.4. Constructing a Symmetric and Positive Definite Approximate Inverse. For some methods (as in the preconditioned Chebyshev and the conjugate gradient iteration methods) it is of importance to use s.p.d. preconditioners. As we have seen, the methods described till now do not guarantee that G will be such a matrix.

In order to compute an s.p.d. approximate inverse of an s.p.d. matrix, we can proceed as follows. It will be shown that this approximation gives a best approximation to minimize the K-condition number of the correspondingly preconditioned matrix.

A Symmetric and Positive Definite Approximate Factorized Inverse. Seek an approximate inverse in the form $G = LL^T$, where $L \in S_L$,

$$S_L = \{(i,j) \in S,\ i \geq j\}, \tag{206}$$

and let

$$S_{\widetilde{L}} = \{(i,j) \in S_L,\ i > j\}, \tag{207}$$

that is, denote by S_L and $S_{\widetilde{L}}$ the lower and strictly lower triangular part of the sparsity set S.

Theorem 12. *Let A be s.p.d. and consider matrices L with sparsity pattern S_l. Let the matrix $\widehat{L}$ be computed by the following steps.*

(i) *Compute first $\widetilde{L}$ such that*

$$\left(\widetilde{L}A\right)_{ij} = A_{ij}, \quad (i,j) \in S_{\widetilde{L}}, \tag{208}$$

and $\widetilde{L}_{ij} = 0$, $(i,j) \in S_L^c$ (the complement set).

(ii) *Let $\widehat{L} = D(I - \widetilde{L})$, where*

$$D = \operatorname{diag}(d_1, d_2, \ldots, d_n),$$
$$d_i = \frac{1}{\left[\left(I - \widetilde{L}\right)A\left(I - \widetilde{L}^T\right)\right]_{i,i}^{1/2}}. \tag{209}$$

Then, $\widehat{L} \in S_L$ and minimizes the K-condition number of $\widehat{L}A\widehat{L}^T$, that is,

$$\frac{\left((1/n)\operatorname{tr}\left(\widehat{L}A\widehat{L}^T\right)\right)^n}{\det\left(\widehat{L}A\widehat{L}^T\right)} = \inf_{L \in S_L} \frac{\left((1/n)\operatorname{tr}(LAL^T)\right)^n}{\det(LAL^T)}. \tag{210}$$

Proof. Let $D = \operatorname{diag}(d_1, \ldots, d_n)$ denote the diagonal part of a matrix $X \in S_L$ and let $\widetilde{X} = I - D^{-1}X$, that is, $\widetilde{X} \in S_{\widetilde{L}}$. Then,

$$\begin{aligned}
&\frac{\left((1/n)\operatorname{tr}(XAX^T)\right)^n}{\det(XAX^T)} \\
&\quad = \frac{\left((1/n)\sum_i (XAX^T)_{ii}\right)^n}{(\det(X))^2 \det(A)} \\
&\quad = \frac{\left((1/n)\sum_i \left[D\left(I - \widetilde{X}\right)A\left(I - \widetilde{X}^T\right)D\right]_{i,i}\right)^n}{(\det(X))^2 \det(A)} \\
&\quad = \frac{\left((1/n)\sum_i d_i^2\left[\left(I - \widetilde{X}\right)A\left(I - \widetilde{X}^T\right)\right]_{i,i}\right)^n}{(\Pi_i d_i^2)\det(A)} \\
&\quad = \frac{\left((1/n)\ \sum_i \alpha_i^2\right)^n}{\Pi_i \alpha_i^2} \cdot \frac{\Pi_i\left[\left(I - \widetilde{X}\right)A\left(I - \widetilde{X}^T\right)\right]_{i,i}}{\det(A)},
\end{aligned} \tag{211}$$

where $\alpha_i^2 = d_i^2[(I - \widetilde{X})A(I - \widetilde{X}^T)]_{i,i}$.

Note now that

$$\Pi_i\left[\left(I - \widetilde{X}\right)A\left(I - \widetilde{X}^T\right)\right]_{i,i}, \tag{212}$$

does not depend on d_i, so we can minimize this factor independently of d_i.

Consider then the general weighted Frobenius norm minimization problem

$$\min_{G \in S} \operatorname{tr}(I - GB)W(I - GB)^T. \tag{213}$$

As we have seen, its solution G satisfies the relation

$$\left(GBWB^T\right)_{i,j} = \left(WB^T\right)_{i,j}, \quad \forall (i,j) \in S. \tag{214}$$

Let now $G = \widetilde{X},\ W = A,\ B = I,\ S = S_{\widetilde{L}}$. Then,

$$\left(GBWB^T\right)_{ij} = \left(WB^T\right)_{ij} \tag{215}$$

takes the form

$$\left(\widetilde{X}A\right)_{ij} = A_{ij} \quad \forall i,j \in S_{\widetilde{L}}. \tag{216}$$

This is an explicit method and since the minimization is done rowwise it follows from (213), with the chosen matrices G, B and W, that each of

$$\left[\left(I - \widetilde{X}\right)A\left(I - \widetilde{X}\right)^T\right]_{i,i} \quad i = 1, \ldots, n \tag{217}$$

is minimized separately. By construction $\widetilde{L}$ satisfies (216), so the minimization problem is has the solution $\widetilde{X} = \widetilde{L}$. Hence,

$$\min_{\widetilde{X}} \Pi_i \left[\left(I - \widetilde{X}\right) A \left(I - \widetilde{X}\right)^T \right]_{i,i} = \Pi_i \left[\left(I - \widetilde{L}\right) A \left(I - \widetilde{L}\right)^T \right]_{i,i}. \tag{218}$$

Consider next the first factor in (211). Here,

$$\frac{\left((1/n)\sum_j \alpha_j^2\right)^n}{\Pi_j \alpha_j^2} \geq 1, \tag{219}$$

since a geometric average is less or equal to an arithmetic average. Equality is taken if and only if all α_j are equal and with no limitation we can take $\alpha_j = 1$, $j = 1, \ldots, n$. Hence,

$$d_i^2 = \frac{1}{\left[\left(I - \widetilde{L}\right) A \left(I - \widetilde{L}\right)^T \right]_{i,i}} \tag{220}$$

which completes the proof. □

The method above provides a simple and cheap method to compute approximate inverses on factorized form. The proof of the theorem shows that the K-condition number is reduced in a way as follows from the next corollary.

Corollary 2. *Let $\hat{L}$, D be defined as in Theorem 12. Then,*

$$\begin{aligned} K\left(\hat{L}A\hat{L}^T\right) &\equiv \frac{\left((1/n)\operatorname{tr}\left(\hat{L}A\hat{L}^T\right)\right)^n}{\det\left(\hat{L}A\hat{L}^T\right)} \\ &= \frac{\Pi_1^n \left[\left(I - \widetilde{L}\right) A \left(I - \widetilde{L}^T\right) \right]_{ii}}{\det(A)}, \end{aligned} \tag{221}$$

where $\widetilde{L} = I - D^{-1}L$.

Hence, the trace is replaced by a product, that is the n'th power of the arithmetic average is replaced the n'th power of a geometric average. This is illustrated in the next example.

Example 6. Let $S_L = \{(1,1),(2,2),\ldots,(n,n)\}$, that is, let L be a diagonal matrix. Then, we find $d_i^2 = a_{ii}$ and Corollary 2 shows that

$$K\left(LAL^T\right) = \min_{L \in S_L} \frac{\left((1/n)\operatorname{tr}\left(LAL^T\right)\right)^n}{\det\left(LAL^T\right)} = \frac{\Pi_1^n a_{ii}}{\det(A)}, \tag{222}$$

which is to be compared with

$$K(A) = \frac{\left((1/n)\operatorname{tr}(A)\right)^n}{\det(A)} = \frac{\left((1/n)\sum_1^n a_{ii}\right)^n}{\det(A)}, \tag{223}$$

that is, we have

$$K\left(LAL^T\right) = \left(\frac{g}{\underline{a}}\right)^n K(A). \tag{224}$$

Hence, the K-condition number $K(LAL^T)$ of the diagonally scaled matrix LAL^T is substantially smaller than $K(A)$ if the geometric average $\underline{g}$ of the diagonal entries a_{ii} of A are much smaller than their arithmetic average $\overline{a}$. This holds when the entries a_{ii} vary significantly. Note that it always holds that $\underline{g} \leq \overline{a}$.

We conclude this section by mentioning that the K-condition number can be take large values even for seemingly harmless eigenvalue distributions.

Example 7 (Arithmetic distribution). Let $0 < \lambda_1 < \lambda_2 < \cdots < \lambda_n$, the eigenvalues of B be distributed uniformly as an arithmetic sequence in the interval $[a,b]$, $a = \lambda_1$, $b = \lambda_n$. For simplicity, assume that $n/2$ is even. Then,

$$K(B) = \frac{(b+a)/2)^n}{\prod_1^n \lambda_i}. \tag{225}$$

On the other hand, (57) shows $k \leq (1/2)\sqrt{b/a}\ln 2/\varepsilon$, which is asymptotically smaller than n if $(b/a)n^{-2} = o(1)$. In particular, if b/a does not depend on n then we have $k \leq O(\ln 1/\varepsilon)$. Therefore, the estimate in Theorem 6 inferior even to the simple estimate in (56). For other distributions, however, Theorem 6 can give a smaller upper bound.

6. Augmented Subspace Preconditioning Method

6.1. Introduction; Preconditioners for Very Ill-Conditioned Problems. In this section, we consider the solution of systems $A\mathbf{x} = \mathbf{b}$, where A is an $n \times n$ matrix which is symmetric and positive definite (s.p.d) and can have a very large condition number, that is, be *ill-conditioned*. Such systems arise typically for near-limit values of some problem parameter. (Ratio of material coefficients, aspect ratio of the domain, nearly incompressible materials in elasticity theory, etc.) The condition number can be additionally very large due to the size of the matrix A (a small value of the discretization parameter) and also due to an irregular mesh and/or large aspect ratios of the mesh in partial differential equation (PDE) problems.

If the size of the system is not too large one can use direct solution methods, possibly coupled with an iterative refinement method.

Let $B = LDL^T$ (or $B = \widetilde{L}\widetilde{L}^T$) be a triangular matrix or the Cholesky factorization of A. Due to finite precision computations (say, in single precision) in general B is only an approximation of A. The iterative refinement method takes the following form.

Algorithm 1 (Iterative refinement method). Given $\mathbf{x}^{(0)} = 0$ for $k = 0, 1, 2, \ldots$, until convergence

(i) compute $\mathbf{r}^{(k)} = \mathbf{b} - A\mathbf{x}^{(k)}$,

(ii) solve $B\mathbf{d}^{(k)} = \mathbf{r}^k$,

(iii) let $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mathbf{d}^{(k)}$, and repeat (i)–(iii).

Frequently, it suffices with one iterative refinement step. The ability of iterative refinement to produce a more accurate solution vector depends crucially on how the computation of the residual vector $\mathbf{r}^{(k)}$ in (i) is implemented. A safe way is to

use double precision for this computation but possibly single precision in (ii) and (iii). However, as described in [30], if one rewrites the computation of $A\mathbf{x}^{(k)}$ as a sum of differences, in some cases it suffices to use single precision in (i) also.

The computational labor is normally dominated by the initial factorization of A. For large systems this cost can become too big as it grows in general fast with problem size. (For an elliptic difference problem on a 3D $N \times N \times N$ mesh it grows as $O(N^7)$ for certain band-matrix orderings. Furthermore, the demand of memory to store the factor L grows as $O(N^5)$. For certain nested dissection and other orderings, the complexity is somewhat reduced, however.)

Therefore, iterative solution methods become the ultimate methods of choice. As we have seen in Section 1, the basic idea behind the iterative solution technique is to use a cheaper (incomplete) factorization or other approximation B of A and to compensate for this approximation by repeating the steps in the iterative refinement method until the residual is sufficiently small. In addition, to speed up the convergence of the method, one or more acceleration parameters are introduced, for instance, (iii) becomes $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{d}^{(k)}$ for certain parameters α_k.

By a proper choice of B the number of required iterations may not be too big while the expense in solving systems with matrix B may not be larger than the order of some "work units", for instance, can correspond to a few actions (matrix-vector multiplications) of A on a vector. In this way, one can gain significantly in computational labor and less demand of memory resources as compared with a direct solver. Actually, the direct solver can be viewed as an approximate factorization with the full amount of fill-in allowed, while as we have seen, in a incomplete factorization method one controls the amount of fill-in either by using a predetermined sparsity pattern in L or by allowing a variable pattern, which depends on some relative drop tolerance. (Such a drop tolerance is to delete a fill-in entry a_{ij} if there holds $|a_{ij}| \leq \varepsilon\sqrt{a_{ii}a_{jj}}$, $j \neq i$ for some ε, $0 < \varepsilon < 1$. More details can be found in [82]).

A problem with iterative solution methods for ill-conditioned systems is that they may stagnate, that is, there is no further improvement as the method proceeds. This occurs typically for minimum residual or minimum A-norm methods. For other type of methods even divergence may be observed. Another problematic issue is the fact that if the residual norm has taken a small value, this does not necessarily mean that the error norm is sufficiently small, since

$$\left\|\mathbf{x} - \mathbf{x}^{(k)}\right\|_2 = \left\|A^{-1}A\left(\mathbf{x} - \mathbf{x}^{(k)}\right)\right\|_2 \leq \|A^{-1}\|_2 \left\|\mathbf{r}^{(k)}\right\|_2 = \frac{1}{\lambda_{\min}(A)}\left\|\mathbf{r}^{(k)}\right\|_2, \tag{226}$$

and here $\lambda_{\min}(A)$ takes very small values for ill-conditioned systems. Hence, even if $\|\mathbf{r}^{(k)}\|$ is small, $\|\mathbf{x} - \mathbf{x}^{(k)}\|$ may still be large. For ill-conditioned systems one sees then typically a reduction of the residual to some limit value while the errors hardly decay at all. This was illustrated in Section 3. For studies on the influence of inexact arithmetics, see for example [83–85].

This situation can be significantly improved by using a proper preconditioner. Then,

$$\left\|\mathbf{x} - \mathbf{x}^{(k)}\right\|_2 = \left\|\left(B^{-1}A\right)^{-1}B^{-1}A\left(\mathbf{x} - \mathbf{x}^{(k)}\right)\right\|_2 \leq \frac{1}{\lambda_{\min}(B^{-1}A)}\left\|\widetilde{\mathbf{r}}^{(k)}\right\|_2, \tag{227}$$

where $\widetilde{\mathbf{r}}^{(k)} = B^{-1}A(\mathbf{x} - \mathbf{x}^{(k)}) = B^{-1}\mathbf{r}^{(k)}$ is the so called *preconditioned* or *pseudo-residual*. Here $\lambda_{\min}(B^{-1}A) \gg \lambda_{\min}(A)$ with a proper preconditioner. Therefore, the importance of choosing a proper preconditioner is twofold:

(1) to increase the rate of convergence while keeping the expense in solving systems with B low, and

(2) to enable a small error norm when the pseudoresidual is small.

Preconditioning methods, such as the modified incomplete factoriztion method, multigrid and multilevel methods, aim at reducing arror components correponding both to the large eigenvalues with rapidly oscillating components and the smaller eigenvalues for smoother eigen functions. In the modified method, this is partly achieved by letting the preconditioner by exact for a particular smooth component of the solution, such as for the constant component vector. It has been shown, see [6, 47] when applied for elliptic difference problems, that under certain conditions the spectral condition number is reduced from $O(h^{-2})$ to $O(h^{-1})$. In multigrid methods, one works on two or more levels of meshes where the finer grid component should smooth out the fast, oscillating components in the iteration error, while the coarser mesh should handle the smooth components. Under certain conditions, such methods way reduce the above condition number to optimal order, $O(1)$, as $h \to 0$.

The multigrid method was first introduced for finite difference methods in the 1960s by Fedorenko [86], and Bakhvalov [87], and further developed and advocated by Brandt in the 1970s, see, for example, Brandt [88]. For finite elements it has been pursued by, for example, Braess [89], Hackbusch [90], Bramble et al. [91], Mandel et al. [92], McCormick [57], Bramble et al. [93] and Bank et al. [94], among others.

As it turns out, such standard preconditioning methods, namely (modified) incomplete factorization ((M)ILU), [46, 47], Multigrid (MG) [90], or Algebraic Multilevel Iteration (AMLI), [95–97], methods may not be efficient in both and in particular, in the second of the above mentioned requirements. This might be due to the fact that the smallest eigenvalue (in the preconditioned system) is caused by some problem parameter which these methods leave unaffected. Therefore there is a demand for new types of preconditioners (or new combinations of already known preconditioners). To satisfy the above need, two types of preconditioners have been constructed:

(a) deflation methods,

(b) augmented matrix methods,

which we now describe.

6.2. Deflation Methods. The deflation technique is based on a projection matrix. Assume that A has a number of (very) small eigenvalues, say $\widetilde{m}$, $0 < \lambda_1 \leq \lambda_2 \ldots \leq \lambda_{\widetilde{m}}$, and let $\mathcal{W} = \{\mathbf{w}^{(i)}\}, i = 1, \ldots, \widetilde{m}$ be their corresponding eigenvectors ($A\mathbf{w}_i = \lambda_i \mathbf{w}_i$). Let V be a rectangular matrix of order $n \times m$, where $m < n$ (in practice $m \ll n$) of full rank, where the m columns of V span a subspace γ, such that Im γ contains the eigenvectors corresponding to the "bad" subspace $\mathcal{W}$. Hence, $m \geq \widetilde{m}$.

Lemma 6. *Let $P = AVA_V^{-1}V^T$, where $A_V = V^TAV$. Then, the following holds:*

(a) $P^2 = P$, *that is a projector;*

(b) $P(AV) = AV$;

(c) $(I - P)\mathbf{b} = 0$ *if* $\mathbf{b} \in \mathit{Im}(AV)$;

(d) $P^TV = V$;

(e) $(I - P)A$ *is symmetric and positive definite and has a nullspace of dimension m.*

Proof. Note first that A_V is nonsingular since V has a full rank ($= m$). The statements follow now by straightforward computations. □

Lemma 6 shows that P is projection matrix which maps any vector onto AV. Similarly, P^T is a projection matrix which maps Vonto itself. We will use the matrix P in three slightly different ways to solve ill-conditioned systems

We split first the right-side vector $\mathbf{b}$ in two components:

$$\mathbf{b} = P\mathbf{b} + (I - P)\mathbf{b}. \tag{228}$$

(These components are A^{-1} orthogonal, i.e., $(Pb)^TA^{-1}(I - P)\mathbf{b} = 0$.) The first splits the computation of thesolution vector corresponding.

Method 1 (Splitting of the solution vector).

Let

$$\mathbf{x}^{(0)} = VA_V^{-1}V^T\mathbf{b}. \tag{229}$$

Then,

$$A\mathbf{x}^{(0)} = P\mathbf{b}. \tag{230}$$

Solve

$$A\mathbf{z} = (I - B)\mathbf{b}. \tag{231}$$

The solution $\mathbf{x}$ of $A\mathbf{x} = \mathbf{b}$ is then

$$\mathbf{x} = \mathbf{x}^{(0)} + \mathbf{z}. \tag{232}$$

Here, $\mathbf{x}^{(0)}$ and $\mathbf{z}$ are A-orthogonal.

Note that $A\mathbf{z} = \mathbf{b} - A\mathbf{x}^{(0)}$. The matrix A_V is normally of small order and the arising system in (229) can be solved with relatively little expense using a direct solution method. Furthermore, the system (231) is well-conditioned on the solution subspace, because, as follows from part (c) of Lemma 6, $(I - P)\mathbf{b}$, and hence $\mathbf{z}$ do not contain components of any of the first m "small" eigenvectors w_i, $i = 1, 2, \ldots, m$. Hence, (231) can be solved by the CG method with a rate of convergence determined by the *effective condition number* λ_n/λ_{m+1}, which is expected to be substantially smaller than λ_n/λ_1.

However, the method requires exact solution of systems with A_V and for some problems m is not that small. Also, it is assumed that the projection $P\mathbf{b}$ is computed exactly (or to a sufficient accuracy), which may be unfeasible in many applications.

Method 2 (Defect-correction with projectors). In the presence of round-off errors, $\mathbf{x}^{(0)}$ may not be sufficiently accurate and $\mathbf{b} - A\mathbf{x}^{(0)}$ may still contain components in the "bad" subspace. A defect-correction (iterative refinement) procedure may then help. Let $\mathbf{x}^{(0)} = 0$ for $k = 0, 1, 2, \ldots$, until convergence. Compute $\mathbf{r}^{(k)} = \mathbf{b}$–$A\mathbf{x}^{(k)}$. Solve $A\mathbf{d}^{(k)} = \mathbf{b}$–$A\mathbf{x}^{(k)}$ as follows:

(i) $\mathbf{d}^{(k,0)} = VA_V^{-1}V^T\mathbf{r}^{(k)}$,

(ii) $A\mathbf{z}^{(k)} = (I - P)\mathbf{r}^{(k)}$, or $A\mathbf{y}^{(k)} = \mathbf{r}^{(k)} - A\mathbf{d}^{(k,0)}$,

(iii) $\mathbf{d}^{(k)} = \mathbf{d}^{(k,0)} + \mathbf{y}^{(k)}$.

Let $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mathbf{d}^{(k)}$.

In this method, it normally suffices with few defect-correction steps.

For some extremely ill-conditioned systems, the implementation of the defect-correction method as a preconditioning method may be necessary. Note then that as follows from Lemma 6, $(I - P)A$ is symmetric so the standard conjugate gradient method can be used.

Method 3 (Preconditioning by a projection matrix). Let $\mathbf{x}^{(0)} = VA_V^{-1}V^T\mathbf{b}$. Solve $(I - P)A\mathbf{z} = (I - P)\mathbf{b}$, or $(I - P)A\mathbf{z} = \mathbf{b} - A\mathbf{x}^{(0)}$ by CG iteration. Let $\mathbf{x} = \mathbf{z} + \mathbf{x}^{(0)}$.

Note that $\mathbf{x}^{(0)}$ is contained in the null-space of $(I - P)A$, since $(I - P)A\mathbf{x}^{(0)} = (I - P)P\mathbf{b} = (P^2 - P)\mathbf{b} = 0$. Here, the system $(I - P)A$ is well-conditioned on the orthogonal complement to the null-space and, in addition, the right-hand-side has no or only small components in the bad subspace.

Methods 1, 2, and 3 require accurate solution of systems with the matrix A_V. It is a viable step for small values of m. However, when the dimension of the "bad" subspace of A is relatively big, it may be too costly. Furthermore, the iteration Method 3 involves two multiplications with A (one involved in P and one required to compute $A\mathbf{z}^{(k)}$) at each iteration step when computing the search direction vectors and is therefore particularly expensive.

Another issue to comment on is that the above methods are assumed to move the components of the eigenvectors for the smallest eigenvalues of A to become exactly zero. However, this can be sensitive to perturbations and occurs

only in exact arithmetic. As we have seen, it is a viable method for small dimensions of the subspace causing the ill-conditioning but it may be inefficient for larger dimensions.

Deflation methods have been used and analysed by [98–100], among others.

In the next section and Section 4, we present a method which move the small eigenvalues to the cluster of bigger eigenvalues which is much less dependent on having the right subspace spanned by the columns of V and which do not require exact solution of systems with A_V.

6.3. Augmented Matrix Preconditioning Methods, the Ideal Case. We now present an alternative method to handle ill-posed problems. In this method the small eigenvalues are moved to the cluster of bigger eigenvalues, instead of being deflated to zero, as in the deflation method. The method is an extension of the method presented in [101]. The presentation here is based in [25, 102]. First, we consider $B = I + VV^T$ as a (multiplicative) preconditioner to A.

In this case one must scale the column vectors appearing in V properly. A method involving an automatic scaling is based on a projection matrix. Let then

$$B = I + \sigma V A_V^{-1} V^T, \quad A_V = V^T A V. \tag{233}$$

Let $(\lambda_i, \mathbf{v}_i)_{i=1}^m$ be the eigenpairs of A for the smallest eigenvalues, $0 < \lambda_1 \le \lambda_2 \le \cdots \lambda_m$. If $V = [\mathbf{v}_1, \ldots, \mathbf{v}_m]$, we get then

$$\widetilde{\lambda}_i = \lambda_i(BA) = \begin{cases} \lambda_i + \sigma, & 1 \le i \le m, \\ \lambda_i, & m+1 \le i \le n. \end{cases} \tag{234}$$

Hence, σ determines how much the smallest eigenvalues are moved. If $\lambda_{m+1} \le \lambda_1 + \sigma$ and $\lambda_m + \sigma \le \lambda_n$, then $\lambda_{m+1} \le \widetilde{\lambda}_i \le \lambda_n$, that is, the m smallest eigenvalues have been moved to the cluster $[\lambda_{m+1}, \lambda_n]$ of bigger eigenvalues and the spectral condition number of BA is $\kappa(BA) = \lambda_n/\lambda_{m+1}$, which normally means a significant reduction, compared to $\kappa(A) = \lambda_n/\lambda_1$.

The above illustrates what can be achieved in an ideal case. In practice, the exact eigenvectors $\{\mathbf{v}_1, \ldots, \mathbf{v}_m\}$ (or the subspace spanned by them) are not known. Even if the eigenvectors are known it is not efficient to use them to form the matrix V because they are in general not sparse and the matrix vector multiplications with V will be costly. Hence, in practice other vectors which are sparse but spans about the same subspace as the smoothest eigenvectors must be used; otherwise the expense of the preconditioner would be too high. We consider therefore more general subspaces spanned by the column vectors of V. The next lemma will be useful.

Lemma 7. *Let A be s.p.d. Then,*

$$P = A^{1/2} V \left(V^T A V\right)^{-1} V^T A^{1/2}, \tag{235}$$

is an orthogonal projection, that is, $P^2 = P$ and $P^* = P$. Therefore, the only eigenvalues of P are 0 and 1.

Proof. Consider the following

$$\begin{aligned} P^2 &= A^{1/2} V \left(V^T A V\right)^{-1} V^T A^{1/2} A^{1/2} V \left(V^T A V\right)^{-1} V^T A^{1/2} \\ &= A^{1/2} V \left(V^T A V\right)^{-1} V^T A^{1/2} = P. \end{aligned} \tag{236}$$

□

The next theorem shows (what can also be expected) that the clustering can never get worse for expanding subspaces spanned by the column-vectors of V, that is, there holds a monotonicity principle.

Theorem 13 (¡xref ref-type="bibr" rid="B97"/¿])). *Let A and $\widehat{A}$ be s.p.d. matrices of order $n \times n$ and let V_k be rectangular matrices of order $n \times m_k$, $k = 1, 2$ such that* $\operatorname{rank} V_k = m_k$, *$k = 1, 2$. If* $\operatorname{Im} V_1 \subseteq \operatorname{Im} V_2$, *then for all i, $1 \le i \le n$ the following inequality holds*

$$\begin{aligned} &\lambda_i\left(\left(I + V_2\left(V_2^T \widehat{A} V_2\right)^{-1} V_2^T\right) A\right) \\ &\quad \ge \lambda_i\left(\left(I + V_1\left(V_1^T \widehat{A} V_1\right)^{-1} V_1^T\right) A\right). \end{aligned} \tag{237}$$

Proof. It is readily seen that the proposition holds if $F = V_2(V_2^T \widehat{A} V_2)^{-1} V_2^T - V_1(V_1^T \widehat{A} V_1)^{-1} V_1^T$ is negative definite. But since $\operatorname{Im} V_1 \subseteq \operatorname{Im} V_2$, there exists some matrix Q of order $m_2 \times m_1$ such that $V_1 = V_2 Q$. Then, with $D_K = V_k^T \widehat{A} V_k$, we have

$$\begin{aligned} F &= V_2\left(D_2^{-1} - Q D_1^{-1} Q^T\right) V_2^T \\ &= V_2 D_2^{-1/2}\left(I - D_2^{-1/2} Q D_1^{-1/2} Q^T D_2^{-1/2}\right) D_2^{-1/2} V_2^T, \end{aligned} \tag{238}$$

where

$$P \equiv D_2^{-1/2} Q D_1^{-1/2} Q^T D_2^{-1/2} = D_2^{-1/2} Q\left(Q^T D_2 Q\right)^{-1} Q^T D_2^{-1/2}, \tag{239}$$

is an orthogonal projector ($P^2 = P$), whose eigenvalues are 0 and 1. □

Corollary 3. *If* $\operatorname{Im} V_1 = \operatorname{Im} V_2$ *then* $I + V_2 D_2^{-1} V_2^T = I + V_1 D_1^{-1} V_1^T$.

Proof. In this case, Q in $V_1 = V_2 Q$ is invertible. Thus, $D_2^{-1/2} Q (Q^T D_2 Q)^{-1} Q^T D_2^{-1/2} = I$. □

Remark 4. The above corollary shows that the individual eigenvectors of A are not needed when constructing the matrix V; we are rather interested in the subspace spanned by them.

The most interesting case for us is when $\operatorname{Im} V \supset$ span $\{\mathbf{v}_1, \ldots, \mathbf{v}_m\}$, where v_i are the eigenvalues of A for the smallest eigenvalues $\lambda_1, \ldots, \lambda_m$. Then, the preconditioner $B = I + \sigma V A_V^{-1} V^T$ moves the smallest eigenvalues λ_i at least to $\lambda_i + \sigma$, where σ is a scaling parameter.

Theorem 14. *Let $B = I + \sigma V A_V^{-1} V^T$ and let $\lambda_1, \ldots, \lambda_m$ be the smallest eigenvalues of A. Then, for the eigenvalues of BA there holds.*

$$\widetilde{\lambda}_i \geq \begin{cases} \lambda_i + \sigma, & 1 \leq i \leq m \\ \lambda_i, & m+1 \leq i \leq n. \end{cases} \tag{240}$$

Proof. Let $V_1 = [\mathbf{v}_1, \ldots, \mathbf{v}_m]$, where $\{\mathbf{v}_i\}_1^m$ are the first m eigenvalues of A and let $V_2 = V$. Then, Lemma 7 and (234) show the result. □

It may happen that the eigenvalues are moved too far so that the maximum eigenvalue of BA is much larger that of A.

Theorem 15. *Let A be s.p.d. of order $n \times n$ and let the rectangular matrix V of order $n \times m$ be defined as $V = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m]$. Assume that* $\operatorname{rank} V = m$. *Further, define $\widetilde{A} = (I + \sigma V A_V^{-1} V^T)A$, where $A_V = V^T A V$. Then,*

$$\lambda_{\max}\left(\widetilde{A}\right) \leq \sigma + \lambda_{\max}(A). \tag{241}$$

Proof. The result follows from the following relations:

$$\begin{aligned} \lambda_{\max}\left(\widetilde{A}\right) &\leq \lambda_{\max}(A) + \sigma \sup \frac{\mathbf{x}^T V \left(V^T A_V V\right)^{-1} V^T \mathbf{x}}{\mathbf{x}^T A^{-1} \mathbf{x}} \\ &= \lambda_{\max}(A) + \sigma \sup \frac{\mathbf{x}^T A^{1/2} V \left(V^T A V\right)^{-1} V^T A^{1/2} \mathbf{x}}{\mathbf{x}^T \mathbf{x}} \\ &= \lambda_{\max}(A) + \sigma, \end{aligned} \tag{242}$$

where the last equality follows from Lemma 7 □

It follows from Theorems 6.3 and 16 that the optimal value of $\sigma = \lambda_{m+1}$, in which case $\kappa(BA) \leq (\lambda_n + \sigma)/\lambda_{m+1}$. In general, λ_{m+1} may not be known. With $\sigma = \lambda_n$, we obtain $\kappa(BA) \leq (2\lambda_n)/\lambda_{m+1}$.

The moral we can draw from the above is that the suggested technique can be a very useful means to reduce the condition number of a given matrix A if we have information about the eigenvectors corresponding to the smallest eigenvalues of A. Since in practice this is hardly ever the case, a natural step to undertake is to consider not the individual eigenvectors but the subspace spanned by some approximation of them.

In the next section, we present a generalized form of the augmented matrix preconditioner which allows for both approximate subspaces and the replacement of A_V by a simpler matrix B_V.

7. Preconditioners with an Approximate Subspace Correction Term

The preconditioner presented in the previous subsection will now be extended to include an approximate subspace correction term.

We replace first A_V with a possibly simpler matrix B_V. The resulting eigenvalue bounds are found in the next theorem.

Theorem 16. *Let A be s.p.d. Define the preconditioner B as $B = I + \sigma V B_V^{-1} V^T$, where, $\sigma > 0$, $\operatorname{Im} V \supseteq \operatorname{span}\{v_1, \ldots, v_m\}$, where v_i are the eigenvectors of A for the smallest eigenvalues and B_V is an $m \times m$ s.p.d. approximation of A_V. Then, the eigenvalues $\lambda(BA)$ of BA are bounded as follows:*

(a)

$$\begin{aligned} &\min\{\sigma \lambda_{\min}(B_V^{-1} A_V) + \lambda_1, \lambda_{m+1}\} \\ &\quad \leq \lambda(BA) \leq \sigma \lambda_{\max}(B_V^{-1} A_V) + \lambda_{\max}(A). \end{aligned} \tag{243}$$

(b) *With $\sigma = \lambda_{\max}(A)/\lambda_{\max}(B_V^{-1} A_V)$, we have*

$$\min\{\lambda_{\max}(A)/\kappa(B_V^{-1} A_V) + \lambda_1, \lambda_{m+1}\} \leq \lambda(BA) \leq 2\lambda_{\max}(A). \tag{244}$$

Proof. The minimal eigenvalue of BA can be estimated as

$$\begin{aligned} \lambda_{\min}(BA) &= \inf_x \left\{ \frac{\mathbf{x}^T (I + \sigma V B_V^{-1} V^T) \mathbf{x}}{\mathbf{x}^T A^{-1} \mathbf{x}} \right\} \\ &= \inf_x \left\{ \frac{\mathbf{x}^T \mathbf{x}}{\mathbf{x}^T A^{-1} \mathbf{x}} + \sigma \frac{\mathbf{x}^T V B_V^{-1} V^T \mathbf{x}}{\mathbf{x}^T V A_V^{-1} V^T \mathbf{x}} \cdot \frac{\mathbf{x}^T V A_V^{-1} V^T \mathbf{x}}{\mathbf{x}^T A^{-1} \mathbf{x}} \right\} \\ &= \inf_x \left\{ \frac{\mathbf{x}^T \mathbf{x}}{\mathbf{x}^T A^{-1} \mathbf{x}} + \sigma \lambda_{\min}(B_V^{-1} A V) \frac{\mathbf{x}^T V A_V^{-1} V^T \mathbf{x}}{\mathbf{x}^T A^{-1} \mathbf{x}} \right\}. \end{aligned} \tag{245}$$

Here, $\inf_x \mathbf{x}^T \mathbf{x}/\mathbf{x}^T A^{-1} \mathbf{x} = \lambda_1$ and $\inf_{x; V^T x = 0} \mathbf{x}^T \mathbf{x}/\mathbf{x}^T A^{-1} \mathbf{x} = \lambda_{m+1}$.

The lower bound equals the minimal eigenvalue of the matrix $\widehat{B}A$, where

$$\begin{aligned} \widehat{B} &= \widehat{B}(V) \equiv I + \widehat{\sigma} V A_V^{-1} V^T \\ \widehat{\sigma} &= \sigma \lambda_{\min}(N_V^{-1} A_V). \end{aligned} \tag{246}$$

By Theorem 14, we have with $V_1 = \operatorname{span}\{v_1, \ldots, v + m\}$ and V_2 a matrix satisfying $\operatorname{Im} V_2 \supseteq \operatorname{Im} V_1$, that

$$\lambda_i\left(\widehat{B}(V_2)A\right) \geq \lambda_i\left(\widehat{B}(V_1)A\right) \geq \min\{\lambda_1 + \widehat{\sigma}, \lambda_{m+1}\} \tag{247}$$

and, in particular, for $V_2 = V$

$$\begin{aligned} \lambda_{\min}(BA) &\geq \min(\lambda_1 + \widehat{\sigma}, \lambda_{m+1}) \\ &= \min\{\lambda_1 + \sigma \lambda_{\min}(B_V^{-1} A_V), \lambda_{m+1}\} \end{aligned} \tag{248}$$

which is the lower bound in part (a). The upper bound follows in a similar way. Since there is no upper bound assumed on rank V, and since $\sup_{x, V^T x = 0} \mathbf{x}^T \mathbf{x}/\mathbf{x}^T A^{-1} \mathbf{x} \leq \lambda_{\max}(A)$, we obtain

$$\lambda_{\max}(BA) \leq \sigma \lambda_{\max}(B_V^{-1} A V) + \lambda_{\max}(A). \tag{249}$$

If we let $\sigma = \lambda_{\max}(A)/\lambda_{\max}(B_V^{-1} A_V)$, we get the stated lower bound in (b) and $\lambda_{\max}(BA) \leq 2\lambda_{\max}(A)$. □

Since normally $\lambda_{\max}(A)$ and $\lambda_{\max}(B_V^{-1}A_V)$ are readily estimated, the given choice of σ is viable. It may increase the maximal eigenvalue with a factor 2, which is acceptable.

Corollary 4. *If* $\kappa(B_V^{-1}A_V) \leq \lambda_{\max}(A)\lambda_{m+1}$, *then*

$$\kappa(BA) \leq \frac{2\lambda_{\max}(A)}{\lambda_{m+1}}, \tag{250}$$

that is, the upper bound, coincides with the bound where $B_V = A_V$. *In particular, if* $\kappa(A_V) \leq \lambda_{\max}(A)/\lambda_{m+1}$, *then one can simply let* $B_V = I$ *or*

$$B_V = \operatorname{diag}(A_V). \tag{251}$$

Hence, seen that the matrix A_V in the preconditioner can be replaced with a simpler matrix B_V where the action of B_V^{-1} is cheap, without deteriorating the eigenvalue bounds.

It still remains to weaken the assumption

$$\operatorname{Im} V \supseteq \operatorname{span}\{v_1, \ldots v_m\}, \tag{252}$$

as this is not easy to satisfy in practice. Due to space limitations this will not be presented here. Instead, refer to [56].

8. Krylov Subspace Methods for Singular Systems

Singular systems, that is, with a nontrivial kernel, arise in various applications, such as in boundary value problems with pure Neumann type boundary conditions imposed on the whole boundary. Nullspaces of large dimension may arise in finite element methods using edge element methods, see, for example, [104] and in the analysis of Markov chains when stationary probability vectors of stochastic matrices are computed, see [105, 106], see also [107].

A singular system does not always have a solution and it is more appropriate to consider the least squares problem: find $\mathbf{x} \in R^n$ such that $\|\mathbf{b} - A\mathbf{x}\| \leq \|\mathbf{b} - A\mathbf{x}\|$ for all $x \in R^n$. We recall that a basic iterative solution method to solve a linear system, either has the form

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \tau_k \mathbf{r}^k, \qquad \mathbf{r}^k = \mathbf{b} - A\mathbf{x}^k, \quad k = 0, 1, \ldots, \tag{253}$$

or the preconditioned form

$$\text{solve } B\delta^k = \tau_k \mathbf{r}^k, \tag{254}$$

and update

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \delta^k, \quad k = 0, 1, \ldots. \tag{255}$$

Here, B is a preconditioner to A. Similarly, as we have seen, more involved methods, such as (generalized) CG-methods (GCG, GMRES, GCR, etc.) are based on approximations taken from a Krylov subspace

$$K(A, \mathbf{r}^0, k) = \left\{\mathbf{r}^0, A\mathbf{r}^0, \ldots, A^k\mathbf{r}^0\right\} \quad \text{or} \quad K(B^{-1}A, \mathbf{r}^0, \mathrm{k}). \tag{256}$$

In general, they are based on a minimum residual approach where, at each iteration step, we compute an updated solution that satisfies the best approximation property

$$\min_{x \in K(A, \mathbf{r}^0, k)} \|\mathbf{b} - A\mathbf{x}\| \quad \text{or} \quad \min_{x \in K(B^{-1}A, \mathbf{r}^0, k)} \|\mathbf{b} - A\mathbf{x}\|. \tag{257}$$

We will show that the convergence of such iterative solution methods can stall, or suffer a breakdown, when applied to certain singular systems. For the analysis, we will use the following properties of relevant subspaces for matrix A. In this generality, they can be stated for rectangular matrices.

Definition 2. Let $A \in \mathcal{R}^{m \times n}$. Then $R(A)$ of dimension $\leq n$, called the *range* of A, is the subspace spanned by the column vectors $\mathbf{a}_{\cdot j}$ that is

$$\mathbf{y} \in \mathcal{R}(A) \Longleftrightarrow \mathbf{y} = \sum_{j=1}^{n} \alpha_j \mathbf{a}_{\cdot j}. \tag{258}$$

Definition 3. $\mathcal{N}(A)$, of dimension $\leq n$, is the *nullspace* of $\mathbf{A}$, that is, the subspace of vectors $\underline{v} \in R^n$ s.t. $A\underline{v} = \underline{0}$.

By a classical result (see e.g., [6]), it holds $R(A)^{\perp} = \mathcal{N}(A^T)$

Definition 4. A linear system $A\mathbf{x} = \mathbf{b}$ is called *consistent* if $b \in R(A)$ and *inconsistent* otherwise. If $A\mathbf{x} = \mathbf{b}$ is consistent, there exists a solution. Cle arly, any system with a nonsingular matrix A is consistent.

To provide a general method, applicable for all types of systems, we will use a least squares type of method, that is determine an approximate solution to $\mathbf{x}$ s.t. $\min_{\mathbf{x}}\|\mathbf{b} - A\mathbf{x}\|$ (or, similarly, a preconditioned form).

In practice, the approximations are mostly computed by a Krylov subspace method, where at each step a solution $\mathbf{x}^k$ is computed such that

$$\left\|\mathbf{b} - A\mathbf{x}^k\right\| \leq \|\mathbf{b} - A\mathbf{x}\|, \quad \forall x \in \mathcal{K}(A, \mathbf{r}^0, k), \tag{259}$$

or a preconditioned form of the method. As the next Theorem shows even if the system is consistent, a breakdown can occur.

Theorem 17. *Any minimum residual Krylov subspace method may suffer a breakdown for some initial vector if and only if* $R(A) \cap \mathcal{N}(A)$ *contains a nontrivial vector.*

Proof. The sufficiency follows since if $R(A) \cap \mathcal{N}(A) \neq \{\underline{0}\}$, there exists a nonzero vector $\mathbf{x} \in \mathcal{N}(A)$ which is also in $R(A)$. Then, at some stage k, there exists a vector $\mathbf{r}^k = \mathbf{y}$, where $A\mathbf{y} = \mathbf{x}$, $\mathbf{x} \in \mathcal{N}(A)$. Hence, $A\mathbf{r}^k = A\mathbf{y} = \mathbf{x}$, which implies $A^2\mathbf{r}^k = A\mathbf{x} = 0$, so a zero vector arises in the Krylov subspace at some stage. This means that convergence stalls. On the other hand, $R(A) \cap \mathcal{N}(A) = \{\underline{0}\}$ implies the existence of nonzero vectors $A^k r_0$ of any order in the Krylov subspace, which implies that there is an improved approximate solution for each higher stage $k + 1$. □

Example 8. Let

$$A = \begin{bmatrix} 0 & 1 & -1 & 0 \\ -1 & 2 & -1 & 0 \\ 1/2 & 1/2 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{260}$$

Here, $A\underline{\mathbf{e}} = 0$, $\underline{\mathbf{e}}^T = (1, 1, , 1, 0)$, that is, $\underline{\mathbf{e}} \in \mathcal{N}(A)$.

Furthermore, there is a solution of $A\mathbf{y} = \underline{\mathbf{e}}$, for instance

$$\mathbf{y} = 1/2 \begin{bmatrix} 3 \\ 3 \\ 1 \\ 0 \end{bmatrix}. \tag{261}$$

Hence, $\underline{\mathbf{e}} \in R(A) \cap \mathcal{N}(A)$, where $\underline{\mathbf{e}} \neq \underline{0}$. Since $A^2\mathbf{r}^0 = 0$, with $\mathbf{y}$ as initial vector, the Krylov subspace for the system $A\mathbf{x} = 0$ stalls at the second step.

Corollary 5. *(a) If $\mathcal{N}(A) = \mathcal{N}(A^T)$, in particular if A is symmetric, then minimal residual type methods-based the Krylov subspace converges.*

(b) More generally, this holds if A is a $(H-)$ normal matrix.

Proof. (a) Since $R(A)^\perp = \mathcal{N}(A^T)$, it holds

$$\begin{aligned} R(A)^\perp &= \mathcal{N}(A) \quad \text{and, hence} \\ R(A) \cap \mathcal{N}(A) &= R(A) \cap R(A)^\perp = \{\underline{0}\}. \end{aligned} \tag{262}$$

(b) For a normal matrix, there exists a unitary matrix U that diagonalizes A, that is

$$U^T A U = D. \tag{263}$$

Hence,

$$U^T A^T U = D^T = D. \tag{264}$$

Therefore, if $AU\mathbf{x} = D\mathbf{x} = 0$ for some $\mathbf{x} \neq 0$, then also $A^T U\mathbf{x} = D\mathbf{x} = 0$, so

$$U\mathbf{x} \in \mathcal{N}(A), \quad U\mathbf{x} \in \mathcal{N}\left(A^T\right), \tag{265}$$

for any such vector $\mathbf{x}$. Since U is nonsingular, this implies $\mathcal{N}(A) = \mathcal{N}(A^T)$. □

Remark 5. Corollary 5 can be extended to H-normal matrices, that is matrices for which A commutes with its H-adjoint,

$$A' = H^{-1}A^*H, \tag{266}$$

for some Hermitian matrix H see, for example, [6].

A remedy to avoid breakdowns for matrices A for which the vector space $\mathcal{V} = R(A) \cap \mathcal{N}(A)$ is nontrivial, is to work in a subspace orthogonal to $\mathcal{V}$. This can be achieved by use of the augmented subspace projection method in Section 6. This method works also to avoid situations, where $R(A) \cap \mathcal{N}(A)$ contains eigenvectors to A corresponding to nearly zero eigenvalues, causing a near breakdown or, in finite precision computations, an actual breakdown. For further comments on near breakdowns, see, for example, [83–85].

9. Concluding Remarks

Some milestones in the development iterative solutions methods have been presented. By the combination of improved methods and the developments of computer hardware one can presently solve problems with a degree of freeadoms nearly billionfold compared to that in the early ages of the computer age.

There remains still, however, very difficult problems such as in multiphysics and heterogeneous media problems and various forms of inverse problems, which need further improvement of solution methods.

Some problems, such as those arising in constrained optimization and mixed finite element methods, lead to matrices on saddle pointform. Due to space limitations, they have not been discussed in this paper, however, see, for example, [108]. In the last centuries, much work has been devoted to multigrid, algebraic multigrid and multilevel iteration methods which have shown an optimal order of performance for many types of problems, for example see [58]. Also, domain decomposition methods which go back to the Schwarz alternating decomposition method, have shown developments, see, for example, [109–112]. For an early survey of domain decomposition methods, see [113]. For the same reason, they could not be discussed in this paper. Much work has also been developed to parallelization aspects of solution methods. This topic deserves a separate survey article and has also not been discussed in this paper.

References

[1] L. F. Richardson, "The approximate arithmetical solution by finite differences of physical problems involving differential equations, with an application to the stresses in a masonry dam," *Philosophical Transactions of the Royal Society of London*, vol. 210, pp. 307–357, 1911.

[2] L. F. Richardson, "How to solve differential equations approximately by arithmetic," *Mathematical Gazette*, vol. 12, pp. 415–421, 1925.

[3] R. V. Southwell, *Relaxation Methods in Engineering Science*, Oxford University Press, Oxford, UK, 1940.

[4] D. M. Young, "On Richardson's method for solving linear systems with positive definite matrices," *Journal of Mathematical Physics*, vol. 32, pp. 243–255, 1954.

[5] C. F. Gauss, "Brief an Gerling vom 26 Dec.1823, Werke," vol. 9, pp. 278–281, A translation by G.E. Forsythe, in MTAC vol. 5 255–258, 1950.

[6] O. Axelsson, *Iterative Solution Methods*, Cambridge University Press, New York, NY, USA, 1994.

[7] E. G. D'Yakonov, "On the iterative method for the solution of finite difference equations," *Doklady Akademii Nauk SSSR*, vol. 138, pp. 522–525, 1961.

[8] J. E. Gunn, "The solution of elliptic difference equations by semi-explicit iterative techniques," *SIAM Journal on Numerical Analysis*, pp. 24–45, 1964.

[9] R. E. Bank, "An automatic scaling procedure for a D'Yakanov-Gunn iteration scheme," *Linear Algebra and Its Applications*, vol. 28, pp. 17–33, 1979.

[10] R. S. Varga, *Matrix Iterative Analysis*, Prentice Hall, Englewood Cliffs, NJ, USA, 1962.

[11] J. H. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, Orlando, Fla, USA, 1970.

[12] R. Beauwens, "Factorization iterative methods, M-operators and H-operators," *Numerische Mathematik*, vol. 31, no. 4, pp. 335–357, 1978.

[13] L. Seidel, "Ueber ein verfahren, die gleichungen, auf welche die methode der kleinsten quadrate führt, sowie lineäre gleichungen überhaupt, durch successive annäherung aufzulösen," *Abhandlungen der Bayerischen Akademie der Wissenschaften. Mathematisch-Naturwissenschaftliche Abteilung*, vol. 11, pp. 81–108, 1814.

[14] S. P. Frankel, "Convergence rates of iterative treatments of partial differential equations," *Mathematical Tables and Other Aids to Computation*, vol. 4, pp. 65–75, 1950.

[15] D. M. Young, *Iterative methods for solving partial difference equations of elliptic type*, Doctoral Thesis, Harward University, 1950, Cambridge, MA.

[16] D. M. Young, *Iterative Solution of Large Systems*, Academic Press, Orlando, Fla, USA, 1971.

[17] O. Axelsson, H. Lu, and B. Polman, "On the numerical radius of matrices and its application to iterative solution methods," *Linear and Multilinear Algebra*, vol. 37, pp. 225–238, 1994.

[18] O. Axelsson, "Solution of linear systems of equations: iterative methods," in *Sparse Matrix Techniques*, V. A. Barker, Ed., LNM no. 572, pp. 1–51, Springer, Berlin, Germany, 1977.

[19] G. H. Golub and R. S. Varga, "Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and second-order Richardson Iterative Methods—part I and II," *Numerische Mathematik*, vol. 3, pp. 147–168, 1961.

[20] D. M. Young, "Second degree iterative methods for the solution of large linear systems," *Journal of Approximation Theory*, vol. 5, pp. 137–148, 1972.

[21] R. Freund, "On conjugate gradient type methods and polynomial preconditioners for a class of complex non-Hermitian matrices," *Numerische Mathematik*, vol. 57, pp. 285–312, 1990.

[22] R. Freund, "Conjugate gradient-type methods for linear systems with complex symmetric matrices," *SIAM Journal on Scientific and Statistical Computing*, vol. 13, pp. 425–448, 1992.

[23] B. Fischer and R. Freund, "Chebyshev polynomials are not always optimal," *Journal of Approximation Theory*, vol. 65, no. 3, pp. 261–272, 1991.

[24] T. A. Manteuffel, "The Tchebychev iteration for nonsymmetric linear systems," *Numerische Mathematik*, vol. 28, no. 3, pp. 307–327, 1977.

[25] M. R. Hestenes and E. Stiefel, "Methods of conjugate gradients for solving linear systems," *Journal of Research of the National Bureau of Standards. Section B*, vol. 49, pp. 409–436, 1952.

[26] G. H. Golub and D. P. O'Leary, "Some history of the conjugate gradient and Lanczos alghorithms: 1948–1976," *SIAM Review*, vol. 31, pp. 50–102, 1989.

[27] J. K. Reid, "The use of conjugate gradients for systems of linear equations posessing "Property A"," *SIAM Journal on Numerical Analysis*, vol. 9, pp. 325–332, 1972.

[28] P. Concus, G. H. Golub, and D. P. O'Leary, "A generalized conjugate gradient method for the numerical solution of elliptic partial differential equations," in *Sparse Matrix Computations*, J. R. Bunch and D. J. Rose, Eds., pp. 309–332, Academic Press, New York, NY, USA, 1976.

[29] O. Axelsson, "Optimal preconditioners based on rate of convergence estimates for the conjugate gradient method," in *Lecture Notes of IMAMM '99*, S. Mika and M. Brandner, Eds., pp. 5–56, University of West Bohemia in Pilsen, 1999.

[30] O. Axelsson and V. A. Barker, *Finite Element Solution of Boundary Value Problems. Theory and Computations*, Academic Press, Amsterdam, The Netherlands, 1984.

[31] O. Axelsson, "Condition numbers for the study of the rate of convergence of the conjugate gradient method," in *Iterative Methods in Linear Algebra II*, S. Margenov and P. S. Vassilevski, Eds., pp. 3–33, IMACS, NJ, USA, 1999.

[32] O. Nevanlinna, *Convergence of Iterations for Linear Equations*, ETH Zürich, Basel, Switzerland, 1993.

[33] L. Yu. Kolotilina, *Lecture Notes in Mathematics*, vol. 1457, Springer, Berlin, Germany, 1989.

[34] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS, NJ, USA, 1996.

[35] O. Axelsson and V. A. Barker, "Finite element solutions of boundary value problems," in *Theory and Computation*, SIAM Classics, Applied Mathematics, Philadelphia, USA, 2001.

[36] O. Axelsson and I. Kaporin, "On the sublinear and superlinear rate of convergence of conjugate gradient methods," *Numerical Algorithms*, vol. 25, no. 1–4, pp. 1–22, 2000.

[37] I. E. Kaporin, "New convergence results and preconditioning strategies for the conjugate gradient method," *Numerical Linear Algebra with Applications*, vol. 1, pp. 179–210, 1994.

[38] L. Y. Kolotilina and A. Y. Yeremin, "Sparse approximate inverse preconditionings I. Theory," *SIAM Journal on Matrix Analysis and Applications*, vol. 14, pp. 45–58, 1993.

[39] O. Axelsson and J. Karátson, "Superlinearly convergent CG methods via equivalent preconditioning for nonsymmetric elliptic operators," *Numerische Mathematik*, vol. 99, no. 2, pp. 197–223, 2004.

[40] Y. Saad and M. H. Schultz, "GMRES: a generalized minimum residual algorithm for solving nonsymmetric linear systems," *SIAM Journal on Scientific and Statistical Computing*, pp. 856–869, 1986.

[41] O. Axelsson and M. Nikolova, "Conjugate gradient minimum residual method (GCG- MR) with variable preconditioners and a relation between residuals of the GCG-MR and GCG-OR methods," *Communications in Mathematical Analysis*, vol. 1, pp. 371–388, 1997.

[42] A. Greenbaum, "Comparison of splittings used with the conjugate gradient algorithm," *Numerische Mathematik*, vol. 33, no. 2, pp. 181–193, 1979.

[43] O. Axelsson, "Stabilization of algebraic multilevel iteration methods; additive methods," *Numerical Algorithms*, vol. 21, no. 1–4, pp. 23–47, 1999.

[44] Y. Saad, "A flexible inner-outer preconditioned GMRES algorithm," *SIAM Journal on Scientific Computing*, vol. 14, pp. 461–469, 1993.

[45] V. Simoncini and D. B. Szyld, "Flexible inner-outer Krylov subspace methods," *SIAM Journal on Numerical Analysis*, vol. 40, no. 6, pp. 2219–2239, 2002.

[46] J. A. Meijerink and H. A. van der Vorst, "An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix," *Mathematics of Computation*, vol. 31, pp. 148–162, 1979.

[47] I. Gustafsson, "A class of first order factorization methods," *BIT*, vol. 18, no. 2, pp. 142–156, 1978.

[48] O. Axelsson and G. Lindskog, "On the rate of convergence of the preconditioned conjugate gradient method," *Numerische Mathematik*, vol. 48, no. 5, pp. 499–523, 1986.

[49] O. Axelsson, S. Brinkkemper, and V. P. Il'in, "On some versions of incomplete block-matrix factorization iterative

methods," *Linear Algebra and Its Applications*, vol. 58, pp. 3–15, 1984.

[50] P. Concus, G. H. Golub, and G. Meurant, "Block preconditioning for the conjugate gradient method," *Statistics and Computing*, vol. 6, pp. 220–252, 1985.

[51] O. Axelsson and B. Polman, "On approximate factorization methods for block matrices suitable for vector and parallel processors," *Linear Algebra and Its Applications*, vol. 77, pp. 3–26, 1986.

[52] R. Beauwens and M. Ben Bouzid, "On sparse block factorization, iterative methods," *SIAM Journal on Numerical Analysis*, vol. 24, pp. 1066–1076, 1987.

[53] J. Kraus, "Algebraic multilevel preconditioning of finite element matrices using local Schur complements," *Numerical linear Algebra with Applications*, vol. 13, no. 1, pp. 49–70, 2006.

[54] O. Axelsson, R. Blaheta, and M. Neytcheva, "Preconditioning of boundary value problems using elementwise Schur complements," *SIAM Journal on Matrix Analysis and Applications*, vol. 31, no. 2, pp. 767–789, 2009.

[55] M. Neytcheva, "On element-by-element Schur complement approximations," *Linear Algebra and its Applications*, 2010, In press.

[56] O. Axelsson and A. Padiy, "On the additive version of the algebraic multilevel iteration method for anisotropic elliptic problems," *SIAM Journal of Scientific Computing*, vol. 20, no. 5, pp. 1807–1830, 1999.

[57] S. McCormick, "Multilevel adaptive methods for partial differential equations," in *Frontiers in Applied Mathematics*, vol. 6, SIAM, Philadelphia, Pa, USA, 1989.

[58] P. S. Vassilevski, "Multilevel block factorization preconditioners," in *Matrix-Based Analysis and Algorithms for Solving Finite Element Equations*, Springer, New York, NY, USA, 2008.

[59] O. Axelsson, "A generalized SSOR method," *BIT*, vol. 12, no. 4, pp. 443–467, 1972.

[60] V. P. Il'in, "Incomplete factorization methods," *Soviet Journal of Numerical Analysis and Mathematical Modelling*, vol. 3, pp. 179–198, 1988.

[61] O. Axelsson, "A survey of preconditioned iterative methods for linear systems of algebraic equations," *BIT*, vol. 25, no. 1, pp. 166–187, 1985.

[62] D. W. Peaceman and H. H. Rachford Jr., "The numerical solution of parabolic and elliptic differential equations," *Journal of the Society for Industrial and Applied Mathematics*, vol. 3, pp. 28–41, 1955.

[63] G. Birkhoff, R. S. Varga, and D. M. Young, "Alternating direction implicit methods," in *Advances in Computers*, F. Alt and M. Rubinoff, Eds., vol. 3, pp. 189–273, 1962.

[64] E. Wachspress, *Iterative Solution of Elliptic Systems and Applications to the Neutron Diffussion Equations of Reactor Physics*, Englewood Cliffs, NY, USA, Prentice Hall, 1966.

[65] O. Axelsson, "A generalized conjugate gradient, least square method," *Numerische Mathematik*, vol. 51, no. 2, pp. 209–227, 1987.

[66] O. Axelsson and A. Kucherov, "Real valued iterative methods for solving complex symmetric linear systems," *Numerical Linear Algebra with Applications*, vol. 7, no. 4, pp. 197–218, 2000.

[67] O. Axelsson, "On the efficiency of a class of A-stable methods," *BIT*, vol. 14, no. 3, pp. 279–287, 1974.

[68] N. I. Buleev, "A numerical method for the solution of two-dimensional and three-dimensional equations of diffussion," *Mathematics Sbornik*, vol. 51, pp. 227–238, 1960.

[69] T. A. Oliphant, "An extrapolation process for solving linear systems," *Quarterly of Applied Mathematics*, vol. 20, pp. 257–267, 1962.

[70] T. Dupont, R. P. Kendall, and H. H. Rachford Jr., "An approximate factorization procedure for solving self-adjoint elliptic difference equations," *SIAM Journal on Numerical Analysis*, vol. 5, pp. 554–573, 1968.

[71] T. Dupont, "A factorization procedure for the solving of elliptic difference equations," *SIAM Journal on Numerical Analysis*, pp. 753–782, 1968.

[72] Z. Woźnički, "AGA two-sweep iterative methods and their application in critical reactor calculations," *Nukleonika*, vol. 23, pp. 941–968, 1978.

[73] H. S. Stone, "Iterative solution of implicit approximations of multidimensional partial differential equations," *SIAM Journal on Numerical Analysis*, vol. 5, pp. 530–558, 1968.

[74] R. R. Underwood, "An approximate factorization procedure based on the block Cholesky decomposition and its use with the conjugate gradient method," Report NRDO-11386, Nuclear Energy Division, General Electric Co., San Jose, Calif, USA, 1976.

[75] R. S. Varga, E. B. Saff, and V. Mehrman, "Incomplete factorizations of matrices and connections with H-matrices," *SIAM Journal on Numerical Analysis*, vol. 17, pp. 787–793, 1980.

[76] O. Axelsson, "A general incomplete block-matrix factorization method," *Linear Algebra and Its Applications*, vol. 74, pp. 179–190, 1986.

[77] L. Yu. Kolotilina, "On approximate inverses of block H-matrices," in *Numerical Analysis and Mathematical Modelling*, Moscow, Russia, 1989.

[78] O. Axelsson, "Preconditioning methods for block H-matrices," in *Computer Algorithms for Solving Linear Systems*, E. Spedicato, Ed., vol. 77 of *NATO ASI Series*, pp. 169–184, Springer, Berlin, Germany, 1991.

[79] L. Yu. Kolotilina and A. Yu. Yeremin, "On a family of two-level preconditionings of the incomplete block factorization type," *Soviet Journal of Numerical Analysis and Mathematical Modelling*, vol. 1, pp. 292–320, 1986.

[80] K. Takahishi, J. Fagan, and M. S. Chen, "Formation of a sparse bus impedance matrix and its application to short circuit study," in *Proceedings of the 8th Power Industry Computer Application Conference (PICA)*, pp. 63–69, Minneapolis, Minn, USA, 1973.

[81] A. M. Erisman and W. F. Tinney, "On computing certain elements of the inverse of a sparse matrix," *Communications of the ACM*, vol. 18, pp. 177–179, 1975.

[82] Z. Zlatev, *Computational Methods for General Sparse Matrices*, Kluwer Academic Publishers Group, Boston, London, 1991.

[83] A. Greenbaum and Z. Strakos, "Predicting the behaviour of finite precision Lanczos and conjugate gradient computations," *SIAM Journal on Matrix Analysis and Applications*, vol. 13, pp. 121–137, 1992.

[84] H. A. van der Vorst, "The convergence behaviour of preconditioned CG and CG-S in the presence of rounding errors," in *Preconditioned Conjugate Gradient Methods*, vol. 1457 of *Lecture Notes in Mathematics*, pp. 126–136, Springer, Berlin, Germany, 1989.

[85] Y. Notay, "On the convergence rate of the conjugate gradients in presence of rounding errors," *Numerische Mathematik*, vol. 65, pp. 301–317, 1993.

[86] R. Fedorenko, "The spead of convergence of one iterative process," *USSR Computational Mathematics and Mathematical Physics*, vol. 4, pp. 227–235, 1964.

[87] N. S. Bakhvalov, "Numerical solution of a relaxation method with natural constraints on the elliptic operator," *USSR Computational Mathematics and Mathematical Physics*, vol. 6, pp. 101–135, 1966.

[88] A. Brandt, "Multi-level adaptive solution to boundary-value problems," *Mathematics of Computation*, vol. 31, pp. 333–390, 1977.

[89] D. Braess, *Finite Elements: Theory, Fast Solvers, and Applications in Solid Mechanics*, Cambridge University Press, Cambridge, UK, 2001, 2nd edition.

[90] W. Hackbusch, *Multigrid Methods and Applications*, Springer, Berlin, Germany, 1985.

[91] J. H. Bramble, J. E. Pasciak, and J. Xu, "Parallel multilevel preconditioners," *Mathematics of Computation*, vol. 55, pp. 1–22, 1990.

[92] J Mandel, S. McCormick, and J. Ruge, "An algebraic theory for multigrid methods for variational problems," *SIAM Journal on Numerical Analysis*, vol. 25, pp. 91–110, 1988.

[93] H. Bramble, "Multigrid Methods," vol. 294 of *Pitman Research Notes in Mathematics Series*, Longman Scientific and Technical, 1993.

[94] R. E. Bank, T. F. Dupont, and H. Yserentant, "The hierarchical basis multigrid method," *Numerische Mathematik*, vol. 52, no. 4, pp. 427–458, 1988.

[95] O. Axelsson and P. S. Vassilevski, "Algebraic multilevel preconditioning methods I," *Numerische Mathematik*, vol. 56, pp. 157–177, 1989.

[96] O. Axelsson and P. S. Vassilevski, "Algebraic multilevel preconditioning methods II," *Numerische Mathematik*, vol. 27, pp. 1569–1590, 1990.

[97] O. Axelsson and M. Neytcheva, "Algebraic multilevel iteration method for Stieltjes Matrices," *Numerical Linear Algebra with Applications*, pp. 213–236, 1994.

[98] L. Mansfield, "Damped Jacobi preconditioning and coarse grid deflation for conjugate gradient iteration on parallel computers," *SIAM Journal on Scientific Computing*, vol. 12, pp. 1314–1323, 1991.

[99] R. Nicolaides, "Deflation of conjugate gradients with application to boundary value problems," *SIAM Journal on Numerical Analysis*, vol. 24, pp. 355–365, 1987.

[100] Z. Dostal, "Conjugate gradient method with preconditioning by projector," *International Journal of Computer Mathematics*, vol. 23, pp. 315–324, 1988.

[101] O. Axelsson, M. Neytcheva, and B. Polman, "An application of the bordering method to solve nearly singular systems," *Vestnik Moskovskogo Universiteta, Seria 15, Vychisl. Math. Cybern*, vol. 1, pp. 3–25, 1996.

[102] A. Padiy, O. Axelsson, and B. Polman, "Generalized augmented matrix preconditioning approach and its application to iterative solution of ill-conditioned algebraic systems," *SIAM Journal on Matrix Analysis and Applications* , vol. 22, no. 3, pp. 793–819, 2000.

[103] Y. Notay and A. van de Velde, "Coarse grid acceleration of parallel incomplete preconditioners," in *Iterative Methods in Linear Algebra II*, S. Margenov and P. Vassilevski, Eds., vol. 3 of *Computational and Applied Mathematics*, pp. 106–130, IMACS, 1996.

[104] O. Biro, K. Preis, and K. R. Richter, "On the use of the magnetic vector potential in the nodal and edge finite element analysis of 3D magnetostatic problem," *IEEE Transactions on Magnetics*, vol. 32, pp. 651–654, 1996.

[105] K. Wu, N. Nunan, J. W. Crawford, I. M. Young, and K. Ritz, "An efficient Markov chain model for the simulation of heterogeneous soil structure," *Soil Science Society of America Journal*, vol. 68, pp. 346–351, 2004.

[106] K. Tanabe, "Characterization of linear stationary iterative processes for solving a singular system of linear equations," *Numerische Mathematik*, vol. 22, pp. 349–359, 1974.

[107] A. Dax, "The convergence of linear stationary iterative processes for solving singular unstructured systems of linear equations," *SIAM Review*, vol. 32, no. 4, pp. 611–635, 1990.

[108] O. Axelsson and R. Blaheta, "Preconditioning of matrices partitioned in 2×2 block form: eigenvalue estimates and Schwarz DD for mixed FEM," *Numerical Linear Algebra with Applications*, vol. 17, pp. 787–810, 2010.

[109] H. A. Schwarz, "Über einen Grenzübergang durch alternierendes verfahren," *Vierteljahrsschrift der Naturforschenden Gesellschaft in Zürich*, vol. 15, pp. 272–286, 1870.

[110] H. A. Schwarz, "Über einige abbildungsaufgaben," *Journal für die Reine und Angewandte Mathematik*, vol. 70, pp. 105–120, 1869.

[111] A. Tosseli and O. B. Widlund, "Domain Decomposition Methods," in *Algorithms and Theory*, Springer, Berlin, Germany, 2005.

[112] R. Blaheta, "Space decomposition preconditioners and parallel solvers," in *Numerical Mathematics and Advanced Applications*, pp. 20–38, Springer, Berlin, Germany, 2004, Proceedings of ENUMATH '03.

[113] B. F. Smith, P. E. Bjorstad, and W. D. Gropp, *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge University Press, Cambridge, UK, 1996.

Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2010, Article ID 241467, 18 pages
doi:10.1155/2010/241467

Review Article

Antireflective Boundary Conditions for Deblurring Problems

Marco Donatelli and Stefano Serra-Capizzano

Dipartimento di Fisica e Matematica, Università dell'Insubria-Sede di Como, Via Valleggio 11, 22100 Como, Italy

Correspondence should be addressed to Stefano Serra-Capizzano, stefano.serrac@uninsubria.it

Received 30 June 2010; Accepted 8 July 2010

Academic Editor: Owe Axelsson

This survey paper deals with the use of antireflective boundary conditions for deblurring problems where the issues that we consider are the precision of the reconstruction when the noise is not present, the linear algebra related to these boundary conditions, the iterative and noniterative regularization solvers when the noise is considered, both from the viewpoint of the computational cost and from the viewpoint of the quality of the reconstruction. In the latter case, we consider a reblurring approach that replaces the transposition operation with correlation. For many of the considered items, the anti-reflective algebra coming from the given boundary conditions is the optimal choice. Numerical experiments corroborating the previous statement and a conclusion section end the paper.

1. Introduction

Formation of a blurred signal/image is typically modeled as a linear system:

$$A\mathbf{f} = \mathbf{g}, \tag{1}$$

where $\mathbf{f}$ is the true object, $\mathbf{g}$ is the blurred object, and A models the blurring process. Obtaining an accurate deblurring model (i.e., the matrix A) requires essentially two main pieces of information:

(1) identification of the blur operator, called a point spread function (PSF),

(2) choosing an appropriate boundary condition (BC), assuming that the observed image is always finite.

The identification of the blur operator is related to the infinite dimensional problem and it decides the essential structure of the matrix A. In the spatially invariant case, due to the shift invariance of the blurring, for image deblurring we deduce a two-level Toeplitz structure (i.e., a two-level Toeplitz matrix or in a different language a block Toeplitz matrix with Toeplitz blocks). The choice of the BCs influences small sections of A by a low-rank plus low-norm term. However, these correction matrices have a substantial impact in two important directions:

(a) precision of the reconstruction especially close to the boundaries (presence of ringing effects);

(b) cost of the computation for recovering the "true" image from the blurred one with or without noise.

On the other hand, the involved correction matrices do not modify significantly the spaces of ill-conditioning (where the coefficient matrix A shows small eigenvalues). This happen because the global matrix is of convolution type. Therefore, the eigenvectors look globally like the Fourier vectors $\mathbf{m}_k$ (they are exactly the Fourier vectors when periodic BCs are imposed). Conversely, the small rank matrices induced by the chosen BCs are not generic since they are necessarily localized in space. More specifically, in one dimension there exists $k \ll n$, n being the size of the matrix, such that the low-rank correction belongs to the span of canonical matrices $E_{i,j} = \mathbf{e}_i \mathbf{e}_j^T$ with $i, j \leq k$ and/or $i, j \geq n + 1 - k$, $\mathbf{e}_k$ being the kth vector of the canonical basis. In that case, we observe $\|E_{i,j}\mathbf{m}_k\|_2 \ll \|\mathbf{m}_k\|_2$ with $\|\cdot\|_2$ being the classical Euclidean norm and hence the term $E_{i,j}$ is unable to modify significantly the characterization in frequencies of the eigenspaces. In actuality, due to the relation $\|E_{i,j}\mathbf{m}_k\|_2 \ll \|\mathbf{m}_k\|_2$, the effect induced by any $E_{i,j}$ is more or less uniformly distributed in all the Fourier directions.

For image deblurring, a classical choice is to use zero (Dirichlet) BCs (see Andrews and Hunt [1], pp. 211–220) where we assume that the values of the image $\mathbf{f}$ outside the domain of consideration are zero. The corresponding blurring matrix is two-level Toeplitz, which is known to be computationally expensive to invert when direct solvers are used (see for instance the review [2]): the fastest available direct Toeplitz solver is of $O(n^4)$ operations for an n^2-by-n^2 two-level Toeplitz matrix [3]. In addition, we have no good news when applying iterative methods due to the negative results in [4–6], where it is proved that we should not expect the classical matrix algebra preconditioners to lead to optimal iterative solvers in the ill-conditioned case. More specifically, we lose the strong cluster at unity of the preconditioned matrix sequence [4, 5] and the spectral boundedness of the preconditioned matrix sequence and its inverse [6]. Therefore, by the convergence analysis of Axelsson and coworkers (see e.g., the pioneering paper [7]), we cannot expect optimal convergence (i.e., a convergence rate independent of the matrix size) both for Krylov type methods and for classical stationary solvers (with the partial exception of multigrid type techniques, see [8, 9] for a recent adaptation in the context of image deblurring). In image restoration problems, we consider regularized problems so that the previous statement is less relevant; however, for linear systems associated to the Tikhonov regularization, the optimality could guarantee a convergence speed independent of the regularizing parameter (for a discussion on this kind of robustness see [9, 10]). Moreover, an image has boundaries, and if the true image is not close to zero at the boundaries, it means that Dirichlet BCs may introduce an artificial discontinuity, which in turn results in the well-known Gibbs phenomenon (observed as a ringing effect in the reconstructed image). Therefore, we face substantial problems both with respect to (a) and (b). A possibility for drastically reducing the computational cost is to impose periodic BCs since this implies that A is a two-level circulant matrix which can be efficiently inverted by means of few fast Fourier transforms (FFTs). In this case, we have a good solution to problem (b) (see Gonzalez and Woods, [11] p. 258) but often the image at the right (upper) boundary has nothing to do with the image at the left (lower) boundary. Again this implies that we are introducing an artificial discontinuity which in turn leads to ringing effects. An important proposal in the direction of reducing the ringing effect and of designing fast algorithms is the one by Ng et al. [12] (see also Strang, [13] page 145, and Chan et al. [14]). In particular, they consider Neumann (reflective) BCs, which means that the data outside the domain of consideration is taken as a reflection of the data inside. The approach is fast for circularly symmetric PSFs since the corresponding blurring matrix belongs to a special two-level algebra simultaneously diagonalized by the fast cosine transform (DCT III). Because a two-level DCT III requires only real multiplications and can be done at half of the cost of a single FFT (see Rao and Yip, [15] pp. 59-60), inversion of these matrices is substantially faster than those obtained from zero BCs and is comparable to (or little better than) the case of periodic BCs. Thus, for circularly symmetric PSFs, requirement (b) is satisfied when using reflective BCs. Moreover, the artificial boundary discontinuities are eliminated, and therefore the ringing effects are strongly reduced. However, although reflection guarantees continuity, it generally fails to guarantee continuity of the normal derivative except for the nongeneric case where the normal derivative is zero at the boundary. Therefore if the image is smooth, then the reflective BCs only save the continuity but introduce an artificial discontinuity of the normal derivative (though requirement (b) is substantially improved with respect to periodic and Dirichlet BCs). In the image processing literature, other methods have also been proposed to assign boundary values as a local image mean, or are obtained by extrapolation methods in order to insure continuity (see Lagendijk and Biemond [16], p. 22, and the references therein). However, these techniques that meet the approximation requirement (a), generally do not satisfy the computational requirement (b) so we still have to face a difficult linear algebra problem.

Here we follow the approach in [17] in which new BCs are proposed that can further reduce ringing effects and that, under the assumption of symmetry of the PSF, leads to a matrix A belonging to the sine algebra of type I for which fast (real) algorithms are available. More precisely, we consider the use of antireflective BCs (AR-BCs) where the data outside the domain of consideration are taken as an antireflection of the data inside. Consider the original signal $\widetilde{\mathbf{f}} = (\ldots, f_{-m+1}, \ldots, f_0, f_1, \ldots, f_n, f_{n+1}, \ldots, f_{n+m}, \ldots)^T$ and the normalized blurring function given by

$$h = (\ldots, 0, 0, h_{-m}, h_{-m+1}, \ldots, h_0, \ldots, h_{m-1}, h_m, 0, 0, \ldots)^T,$$
$$\sum_{j=-m}^{m} h_j = 1. \tag{2}$$

The blurred signal $\mathbf{g}$ is the convolution of h and $\widetilde{\mathbf{f}}$ and consequently $g_i = \sum_{j=-\infty}^{\infty} h_j f_{i-j}$. The deblurring problem is to recover the vector $\mathbf{f} = (f_1, \ldots, f_n)^T$ given the blurring function h and a blurred signal $\mathbf{g} = (g_1, \ldots, g_n)^T$ of finite length, that is,

$$\mathbf{g} = \widetilde{A}\widetilde{\mathbf{f}},$$
$$\widetilde{A} = \begin{pmatrix} h_m & \cdots & h_0 & \cdots & h_{-m} & & & & \\ & h_m & & h_0 & & h_{-m} & & 0 & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots & & \\ & & & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & 0 & & & h_m & & h_0 & & h_{-m} \\ & & & & & h_m & \cdots & h_0 & \cdots & h_{-m} \end{pmatrix}. \tag{3}$$

Thus the blurred signal $\mathbf{g}$ is determined not by $\mathbf{f}$ only, but also by $(f_{-m+1}, \ldots, f_0)^T$ and $(f_{n+1}, \ldots, f_{n+m})^T$ and the linear system (3) is underdetermined. To overcome this, we make

certain assumptions (called BCs) on the unknown boundary data $f_{-m+1},\ldots,f_0$ and $f_{n+1},\ldots,f_{n+m}$ in such a way that the number of unknowns equals the number of equations.

In terms of the objects defined so far, we recall that zero (Dirichlet) BCs means $f_j = f_{n+j} = 0$ for all j in (3) so that a Toeplitz structure is encountered. If we consider periodic BCs we set $f_j = f_{n+j}$, for all j in (3) and the matrix system in (3) becomes n-by-n circulant so that it can be diagonalized by the discrete Fourier matrix and the above system can be solved by using three FFTs (one for finding the eigenvalues of the blurring matrix and two for solving the system). For the Neumann BCs, we assume that the data outside f are a reflection of the data inside f (refer to [12]) so that $f_{1-j} = f_j$ for $j = 1,\ldots,m$ and $f_{n+j} = f_{n+1-j}$ for all $j = 1,\ldots,m$ in (3). Thus (3) becomes $Cf = g$, where C is neither Toeplitz nor circulant but it is a special n-by-n Toeplitz plus Hankel matrix which is diagonalized by the discrete cosine transform provided that the blurring function h is symmetric, that is, $h_j = h_{-j}$ for all j in (2). It follows that the above system can be solved by using three transforms DCT III in $O(n\log n)$ operations (refer to [12]). The latter approach is computationally interesting since a DCT III requires only real operations and is about twice as fast as the FFT (see [15], pp. 59-60), and this is true in two dimensions as well. With the help of a different Toeplitz plus Hankel structure, we establish similar results for the antireflective BCs both in one dimension and two dimensions. It is worth finally to remark that La Spina has analyzed [18] the BCs associated with all the known trigonometric algebras: the interesting facts are two, first some matrix algebras do not lead to any boundary condition, second the highest precision is reached by the classical DCT III algebra and by the classical sine transform algebra of type I which ensures only the continuity of the signal. Therefore, in this sense we can claim that among the known algebras the antireflective one is that related to the most precise reconstructions.

The paper is organized as follows. In the Section 2 we examine the antireflective BCs, the related algebra, and the related transforms also from a computational viewpoint. At the end of Section 2 we briefly consider the multilevel extension while in Section 3 we study some regularization techniques specifically adapted to the features of the antireflective algebra. Finally, Section 4 contains numerical experiments both with Tikhonov like solvers (with reblurring) and with Krylov techniques with early termination as stopping criterion. A final section of conclusions and future problems end the paper.

2. The Algebra of AR Matrices

In this section we describe the AR-BCs and the algebra $\mathcal{AR}_n \equiv \mathcal{AR}$, $n \geq 3$, of matrices arising from the imposition of AR-BCs.

2.1. The Antireflective BCs. When defining the antireflective boundary conditions, we assume that the data outside $\mathbf{f}$ are an antireflection of the data inside $\mathbf{f}$. More precisely, if x is a point outside the domain and x^* is the closest boundary point, then we have $x = x^* - \delta x$ and the quantity $f(x)$ is approximated by $f(x^*) - (f(x^* + \delta x) - f(x^*))$ so that we impose

$$\begin{aligned} f_{1-j} &= f_1 - \left(f_{j+1} - f_1\right) = 2f_1 - f_{j+1}, \\ &\qquad \text{for all } j = 1,\ldots,m, \\ f_{n+j} &= f_n - \left(f_{n-j} - f_n\right) = 2f_n - f_{n-j}, \\ &\qquad \text{for all } j = 1,\ldots,m \end{aligned} \tag{4}$$

in (3). If we define the vector $\mathbf{z}$ whose components are $z_j = 2\sum_{k=j}^{m} h_k$ for $j \leq m$ and zero otherwise and if we define the vector $\mathbf{w}$ whose components are $w_{n+1-j} = 2\sum_{k=j}^{m} h_{-k}$ for $j \leq m$ and zero otherwise, then (3) becomes

$$\widetilde{A}\widetilde{\mathbf{f}} = \left[\mathbf{z}\mathbf{e}_1^T - (0 \mid T_l)\widetilde{J} + T - (T_r \mid 0)\widehat{J} + \mathbf{w}\mathbf{e}_n^T\right]\widetilde{\mathbf{f}} = \mathbf{g}, \tag{5}$$

where $\mathbf{e}_k$ is the kth vector of the canonical basis, $\widetilde{J} = \left(\begin{smallmatrix} 0 & 0 \\ 0 & J \end{smallmatrix}\right)$, $\widehat{J} = \left(\begin{smallmatrix} J & 0 \\ 0 & 0 \end{smallmatrix}\right)$ with J denoting the $n-1$ dimensional flip matrix having entries $[J]_{s,t} = 1$ if $s + t = n + 1$ and zero otherwise, and where the matrices T_l, T, and T_r are given by

$$\begin{pmatrix} h_m & \cdots & h_1 \\ & \ddots & \ddots \\ & & h_m \\ & & \\ 0 & & \end{pmatrix}, \quad \begin{pmatrix} h_0 & \cdots & h_{-m} & & 0 \\ \vdots & \ddots & \ddots & \ddots & \\ h_m & \ddots & \ddots & \ddots & h_{-m} \\ & \ddots & \ddots & \ddots & \vdots \\ 0 & & h_m & \cdots & h_0 \end{pmatrix}, \quad \begin{pmatrix} & & 0 \\ & & \\ h_{-m} & & \\ \vdots & \ddots & \\ h_{-1} & \cdots & h_{-m} \end{pmatrix}. \tag{6}$$

Therefore, the matrices $(0 \mid T_l)\widetilde{J}$ and $(T_r \mid 0)\widehat{J}$ involved in (5) take the form

$$\begin{pmatrix} 0 & h_1 & \cdots & h_m & 0 & \cdots & 0 \\ & \vdots & \iddots & & & & \\ & h_m & & & & & \\ \vdots & & & O & & & \vdots \\ & & & & & & \\ & & & & & & \\ 0 & & & \cdots & & & 0 \end{pmatrix}, \quad \begin{pmatrix} 0 & & & \cdots & & & 0 \\ & & & & & & \\ & & & & & & \\ \vdots & & & O & & & \vdots \\ & & & & & h_{-m} & \\ & & & & \iddots & \vdots & \\ 0 & \cdots & 0 & h_{-m} & \cdots & h_{-1} & 0 \end{pmatrix}. \tag{7}$$

We remark that the coefficient matrix A in (5) is neither Toeplitz nor circulant, but it is a Toeplitz plus Hankel plus 2 rank correction matrix, where the correction is placed at the first and the last column. We will show that the linear system (5) can be reduced to an $(n-2)$ by $(n-2)$ new system whose coefficient matrix can always be diagonalized by the discrete sine transform DST I (associated with the τ class) matrix provided that the blurring function h is symmetric, that is, $h_j = h_{-j}$ for all j in (2). It follows that (5) can be solved by using three FSTs in $O(n \log n)$ operations. This approach is computationally attractive as FST requires only real operations and is about twice as fast as the FFT and hence solving a problem with the AR BCs is twice as fast as solving a problem with the periodic BCs, has the same cost as solving a problem with the reflective BCs, and one gains one order of precision due to the C^1 continuity. Moreover, all these remarks stand in two dimensions as well and indeed an abstract treatment of the two-level and multilevel settings is reported in Section 2.5.

Finally it is worth mentioning that the use of AR BCs has been considered by several authors (see e.g., [18–26]). In particular in [24] the mean BCs have been introduced as a slight variation of the AR BCs. The order of approximation is the same but the hidden constants are smaller so that the mean BCs are generally more precise than the AR BCs. However, the resulting matrices do not form an algebra so that we cannot define a new transform and this is a confirmation of the negative result found in [18].

2.2. The τ Algebra. Let Q be the type I sine transform matrix of order n (see [27]) with entries

$$[Q]_{i,j} = \sqrt{\frac{2}{n+1}} \sin\left(\frac{ji\pi}{n+1}\right), \quad i,j = 1,\ldots,n. \tag{8}$$

It is known that the real matrix Q is orthogonal and symmetric ($Q^{-1} = Q^T = Q$). For any n-dimensional real vector $\mathbf{v}$, the matrix-vector multiplication $Q\mathbf{v}$ (DST-I transform) can be computed in $O(n \log(n))$ real operations by using the algorithm FST-I. Let τ be the space of all the matrices that can be diagonalized by Q:

$$\tau = \{QDQ : D \text{is a real diagonal matrix of size } n\}. \tag{9}$$

Let $X = QDQ \in \tau$, then $QX = DQ$. Consequently, if we let $\mathbf{e}_1$ denote the first column of the identity matrix, then the relationship $QX\mathbf{e}_1 = DQ\mathbf{e}_1$ implies that the eigenvalues $[D]_{i,i}$ of X are given by $[D]_{i,i} = [Q(X\mathbf{e}_1)]_i/[Q\mathbf{e}_1]_i$, $i = 1,\ldots,n$. Therefore, the eigenvalues of X can be obtained by applying a DST-I transform to the first column of X and, in addition, any matrix in τ is uniquely determined by its first column.

Now we report a characterization of the τ class, which is important for analyzing the structure of AR-matrices. Let us define the shift of any vector $\mathbf{h} = [h_0,\ldots,h_{n-1}]^T$ as $\sigma(\mathbf{h}) = [h_1, h_2,\ldots,h_{n-1}, 0]^T$. According to a Matlab-like notation, we define $T(\mathbf{x})$ to be the n-by-n symmetric Toeplitz matrix whose first column is $\mathbf{x}$ and $H(\mathbf{x},\mathbf{y})$ to be the n-by-n Hankel matrix whose first and last column are $\mathbf{x}$ and $\mathbf{y}$, respectively. Every matrix of the class (9) can be written as (see [27])

$$T(\mathbf{h}) - H(\sigma^2(\mathbf{h}), J\sigma^2(\mathbf{h})), \tag{10}$$

where $\mathbf{h} = [h_0,\ldots,h_{n-1}]^T \in \mathbb{R}^n$ and J is the flip matrix. The structure defined by (10) means that matrices in the τ class are special instances of Toeplitz-plus-Hankel matrices.

Moreover, the eigenvalues of the τ matrix in (10) are given by the cosine function $h(y)$ evaluated at the grid points vector $G_n = [k\pi/(n+1)]_{k=1}^n$, where

$$h(y) = \sum_{|j|\le n-1} h_j \exp(\mathbf{i}jy), \tag{11}$$

$\mathbf{i}^2 = -1$, and $h_j = h_{|j|}$ for $|j| \le n-1$. The τ matrix in (10) is usually denoted by $\tau(h)$ and is called the τ matrix generated by the function or *symbol* $h = h(\cdot)$ (see the seminal paper [27] where this notation was originally proposed).

2.3. The AR-Algebras $\mathcal{AR}$. Let $h = h(\cdot)$ be a real-valued cosine polynomial of degree l and let $\tau_k(h) \equiv Q\,\mathrm{diag}(h(G_k))Q$ (note that $\tau_k(h)$ coincides with the matrix in (10)-(11), when $l \le k-1$). Hence the Fourier coefficients of h are such that $h_i = h_{-i} \in \mathbb{R}$ with $h_i = 0$ if $|i| > l$, and for $k = n-2$ we can define the one-level $\mathrm{AR}_n(\cdot)$ operator

$$\mathrm{AR}_n(h) = \begin{bmatrix} h(0) & & \\ \mathbf{v}_{n-2}(h) & \tau_{n-2}(h) & J\mathbf{v}_{n-2}(h) \\ & & h(0) \end{bmatrix}, \tag{12}$$

where $\mathbf{v}_{n-2}(h) = \tau_{n-2}((\phi(h))(\cdot))\mathbf{e}_1$ and

$$(\phi(h))(y) = \frac{h(y) - h(0)}{2(\cos(y) - 1)}. \tag{13}$$

It is interesting to observe that $h(y) - h(0)$ has a zero of order at least 2 at zero (since h is a cosine polynomial) and therefore $\phi(h) = (\phi(h))(\cdot)$ is still a cosine polynomial of degree $l-1$, whose value at zero is $-h''(0)/2$ (in other words the function is well defined at zero).

As proved in [28], with the above definition of the operator $\mathrm{AR}_n(\cdot)$, we have

(1) $\alpha\mathrm{AR}_n(h_1) + \beta\mathrm{AR}_n(h_2) = \mathrm{AR}_n(\alpha h_1 + \beta h_2)$,

(2) $\mathrm{AR}_n(h_1)\mathrm{AR}_n(h_2) = \mathrm{AR}_n(h_1 h_2)$,

for real α and β and for cosine functions $h_1 = h_1(\cdot)$ and $h_2 = h_2(\cdot)$.

These properties allow us to define $\mathcal{AR}$ as the algebra (closed under linear combinations, product, and inversion) of matrices $\mathrm{AR}_n(h)$, with h being a cosine polynomial. By standard interpolation arguments it is easy to see that $\mathcal{AR}$ can be defined as the set of matrices $\mathrm{AR}_n(h)$, where h is a cosine polynomial of degree at most $n-3$. Therefore, it is clear that $\dim(\mathcal{AR}) = n-2$. Moreover, the algebra $\mathcal{AR}$ is commutative thanks to item 2, since $h_1(y)h_2(y) = h_2(y)h_1(y)$ at every y. Consequently, if matrices of the form $\mathrm{AR_n}(h)$ are diagonalizable, then they must have the same set of eigenvectors [29]. This means there must exist an "antireflective transform" that diagonalizes the matrices in $\mathcal{AR}$. Unfortunately this transform fails to be unitary, since the matrices in $\mathcal{AR}$ are generically not normal. However the AR-transform and its inverse are close in rank to orthogonal linear mappings and only moderately ill conditioned.

Following the analysis given in [17], if the blurring function (the PSF) $\mathbf{h}$ is symmetric (i.e., $h_i = h_{-i}$, for all $i \in \mathbb{Z}$), if $h_i = 0$ for $|i| \geq n-2$ (degree condition), and if $\mathbf{h}$ is normalized so that $\sum_{i=-m}^{m} h_i = 1$, then the structure of the $n \times n$ antireflective blurring matrix A is

$$A = \begin{bmatrix} z_0 & 0^T & 0 \\ z_1 & & z_m \\ \vdots & \widehat{A} & \vdots \\ z_m & & z_1 \\ 0 & 0^T & z_0 \end{bmatrix}, \tag{14}$$

where $A_{1,1} = A_{n,n} = 1$, $z_i = h_i + 2\sum_{k=i+1}^{m} h_k$, $\widehat{A}$ has order $n-2$ and

$$\widehat{A} = T(\mathbf{h}) - H\left(\sigma^2(\mathbf{h}), J\sigma^2(\mathbf{h})\right). \tag{15}$$

According to the brief discussion of Section 2.2, relation (15) implies that $\widehat{A} = \tau_{n-2}(h)$ with $h(y) = h_0 + 2\sum_{k=1}^{m} h_k \cos(ky)$ (see (10) and (11)). Moreover in [28] it is proved that $A = \mathrm{AR}_n(h)$.

2.4. Eigenvalues and Eigenvectors of AR-BC Matrices. We first describe the spectrum of AR-BC matrices, under the usual mild degree condition (i.e., the PSF $\mathbf{h}$ has finite support), with symmetric, normalized PSFs. Then we describe the eigenvector structure and we introduce the AR-transform.

The spectral structure of any AR-BC matrix, with symmetric PSF $\mathbf{h}$, is concisely described in the following result.

Theorem 1 (see [28]). *Let the blurring function (PSF) $\mathbf{h}$ be symmetric (i.e., $h_s = h_{-s}$), normalized, and satisfying the usual degree condition. As a consequence the eigenvalues of the $n \times n$ AR-BCs blurring matrix A in (14), $n \geq 3$, are given by $h(0) = 1$ with multiplicity two and $h(G_{n-2})$.*

The proof can be easily derived by (12) which shows that the eigenvalues of $\mathrm{AR}_n(h)$ are $h(0)$ with multiplicity 2 and those of $\tau_{n-2}(h)$, that is, $h(G_{n-2})$, with multiplicity 1 each.

Here we will determine the eigenvectors of every matrix $\mathrm{AR}_n(h)$. In particular, we show that every AR-BCs matrix is diagonalizable, and we demonstrate independence of the eigenvectors from the symbol h. With reference to the notation in (8)–(11), calling $\mathbf{q}_j^{(n-2)}$ the j th column of Q_{n-2}, and $y_j^{(n-2)}$ the jth point of G_{n-2}, $j = 1, \ldots, n-2$, we have

$$\begin{aligned} \mathrm{AR}_n(h) \begin{bmatrix} 0 \\ \mathbf{q}_j^{(n-2)} \\ 0 \end{bmatrix} &= \begin{bmatrix} h(0) & & \\ \mathbf{v}_{n-2}(h) & \tau_{n-2}(h) & J\mathbf{v}_{n-2}(h) \\ & & h(0) \end{bmatrix} \begin{bmatrix} 0 \\ \mathbf{q}_j^{(n-2)} \\ 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 \\ \tau_{n-2}(h)\mathbf{q}_j^{(n-2)} \\ 0 \end{bmatrix} = h\left(y_j^{(n-2)}\right) \begin{bmatrix} 0 \\ \mathbf{q}_j^{(n-2)} \\ 0 \end{bmatrix}, \end{aligned} \tag{16}$$

since $\mathbf{q}_j^{(n-2)}$ is an eigenvector of $\tau_{n-2}(h)$ and $h(y_j^{(n-2)})$ is the related eigenvalue. Due to the centrosymmetry of the involved matrix, if $[1, \mathbf{p}^T, 0]^T$ is an eigenvector of $\mathrm{AR}_n(h)$ related to the eigenvalue $h(0)$, then the other is its flip, that is, $[0, (J\mathbf{p})^T, 1]^T$. Let us look for this eigenvector by imposing the equality

$$\mathrm{AR}_n(h) \begin{bmatrix} 1 \\ \mathbf{p} \\ 0 \end{bmatrix} = h(0) \begin{bmatrix} 1 \\ \mathbf{p} \\ 0 \end{bmatrix}, \tag{17}$$

which is equivalent to seeking a vector $\mathbf{p}$ that satisfies

$$\mathbf{v}_{n-2}(h) + \tau_{n-2}(h)\mathbf{p} = h(0)\mathbf{p}. \tag{18}$$

Since $\mathbf{v}_{n-2}(h) = \tau_{n-2}(\phi(h))\mathbf{e}_1$ by definition of the operator $\mathbf{v}_{n-2}(\cdot)$ (see (12) and the lines below), and because of the algebra structure of τ_{n-2} and thanks to the above relation, we deduce that the vector $\mathbf{p}$ satisfies the relation

$$\tau_{n-2}(h - h(0))\left[-L_{n-2}^{-1}\mathbf{e}_1 + \mathbf{p}\right] = 0, \tag{19}$$

where L_{n-2} is the discrete one-level Laplacian, that is, $L_{n-2} = \tau_{n-2}(2 - 2\cos(\cdot))$. Therefore, by (19), the solution is given by $\mathbf{p} = L_{n-2}^{-1}\mathbf{e}_1$. If $\tau_{n-2}(h - h(0))$ is invertible (as it happens for every nontrivial PSF, since the unique maximum of the function is reached at $y = 0$, which is not a grid point of G_{n-2}), then the solution is unique. Hence, independently of h, we have

$$\begin{aligned} &\mathrm{AR}_n(h) \begin{bmatrix} 1 & & \\ \mathbf{p} & Q_{n-2} & J\mathbf{p} \\ & & 1 \end{bmatrix} \\ &\quad = \begin{bmatrix} 1 & & \\ \mathbf{p} & Q_{n-2} & J\mathbf{p} \\ & & 1 \end{bmatrix} \begin{bmatrix} h(0) & & \\ & \mathrm{diag}(h(G_{n-2})) & \\ & & h(0) \end{bmatrix}. \end{aligned} \tag{20}$$

Now we observe that the jth eigenvector is unitary, $j = 2, \ldots, n-1$, because Q_{n-2} is unitary: we wish to impose the same condition on the first and the last eigenvector. The interesting fact is that $\mathbf{p}$ has an explicit expression. By using standard finite difference techniques, it follows that $p_j = 1 - j/(n-1)$ so that the first eigenvector is exactly the sampling of the function $1 - x$ on the grid $j/(n-1)$ for $j = 0, \ldots, n-1$. Its Euclidean norm is $\alpha_n = \sqrt{\sum_{j=0}^{n-1} j^2}/(n-1) \sim \sqrt{n/3}$, where, for nonnegative sequences β_n, γ_n, the relation $\gamma_n \sim \beta_n$ means $\gamma_n = \beta_n(1+o(1))$. In this way, the (normalized) AR-transform can be defined as

$$T_n = \begin{bmatrix} \alpha_n^{-1} & & \\ \alpha_n^{-1}\mathbf{P} & Q_{n-2} & \alpha_n^{-1}J\mathbf{P} \\ & & \alpha_n^{-1} \end{bmatrix}. \tag{21}$$

Remark 2. With the normalization condition in (21), all the columns of T_n are unitary. However orthogonality is only partially fulfilled since it holds for the central columns, while the first and last columns are not orthogonal to each other, and neither one is orthogonal to the central columns. We can solve the first problem: the sum of the first and of the last column (suitably normalized) and the difference

of the first and the last column (suitably normalized) become orthonormal, and are still eigenvectors related to the eigenvalue $h(0)$. However, since $\mathbf{q}_1^{(n-2)}$ has only positive components and the vector space generated by the first and the last column of T_n contains positive vectors, it follows that T_n cannot be made orthonormal just by operating on the first and the last column. Indeed, we do not want to change the central block of T_n since it is related to a fast $O(n\log(n))$ real transform and hence, necessarily, we cannot get rid of this quite mild lack of orthogonality.

Remark 3. There is a suggestive functional interpretation of the transform T_n. When considering periodic BCs, the transform of the related matrices is the Fourier transform: its jth column vector, up to a normalizing scalar factor, can be viewed as a sampling, over a suitable uniform gridding of $[0, 2\pi]$, of the frequency function $\exp(-\mathbf{i}jy)$. Analogously, when imposing reflective BCs with a symmetric PSF, the transform of the related matrices is the cosine transform: its j th column vector, up to a normalizing scalar factor, can be viewed as a sampling, over a suitable uniform gridding of $[0, \pi]$, of the frequency function $\cos(jy)$. Here the imposition of the antireflective BCs can be functionally interpreted as a linear combination of sine functions and of linear polynomials (whose use is exactly required for imposing C^1 continuity at the borders).

The previous observation becomes evident in the expression of T_n in (21). Indeed, by defining the one-dimensional grid $\widetilde{G}_n = [0, G_{n-2}^T, \pi]^T = [j\pi/(n-1)]_{j=0}^{n-1}$, which is a subset of $[0, \pi]$, we infer that the first column of T_n is given by $\alpha_n^{-1}(1 - y/\pi)|_{\widetilde{G}_n}$, the jth column of T_n, $j = 2, \ldots, n-1$, is given by $\sqrt{2/(n-1)}(\sin(jy))|_{\widetilde{G}_n}$, and finally the last column of T_n is given by $\alpha_n^{-1}(y/\pi)|_{\widetilde{G}_n}$, that is,

$$T_n = \left[\, 1 - \frac{y}{\pi},\ \sin(y),\ \ldots,\ \sin((n-2)y),\ \frac{y}{\pi}\,\right]\Big|_{\widetilde{G}_n} \cdot \Delta_n,$$
$$\Delta_n = \operatorname{diag}\left(\alpha_n^{-1},\ \sqrt{\frac{2}{n-1}}\, I_{n-2},\ \alpha_n^{-1}\right). \tag{22}$$

Finally, it is worth mentioning that the inverse transform is also described in terms of the same block structure since

$$T_n^{-1} = \begin{bmatrix} \alpha_n & & \\ -Q_{n-2}\mathbf{P} & Q_{n-2} & -Q_{n-2}J\mathbf{P} \\ & & \alpha_n \end{bmatrix} \tag{23}$$

Theorem 4 ($AR_n(\cdot)$ Jordan Canonical Form). *With the notation and assumptions of Theorem 1, the $n \times n$ AR-BCs blurring matrix A in* (14)*, $n \geq 3$, coincides with*

$$AR_n(h) = T_n \operatorname{diag}\left(h\left(\widehat{G}_n\right)\right) T_n^{-1}, \tag{24}$$

where T_n and T_n^{-1} are defined in (22) *and* (23)*, while $\widehat{G}_n = [0,\ G_{n-2}^T,\ 0]^T$.*

2.5. Multilevel Extension. Here we provide some comments on the extension of our findings to d-dimensional objects with $d > 1$. When $d = 1$, $\mathbf{h}$ is a vector, when $d = 2$, $\mathbf{h}$ is a 2D array, when $d = 3$, $\mathbf{h}$ is a 3D tensor and so forth.

With reference to Section 2.3 we propose a (canonical) multidimensional extension of the algebras $\mathcal{AR}$ and of the operators $\mathrm{AR}_n(\cdot)$, $n = (n_1, \ldots, n_d)$: the idea is to use tensor products. If $h = h(\cdot)$ is d-variate real-valued cosine polynomial, then its Fourier coefficients form a real d-dimensional tensor which is strongly symmetric, that is symmetric with respect to every direction, that is, $h_j = h_{|j|}$ for all $j \in \mathbb{Z}^d$. In addition, $h(\mathbf{y})$, $\mathbf{y} = (y_1, \ldots, y_d)$, can be written as a linear combination of independent terms of the form $m(\mathbf{y}) = \prod_{j=1}^d \cos(\alpha_j y_j)$ where any α_j is a nonnegative integer. Therefore, we define

$$\mathrm{AR}_n(m(\mathbf{y})) = \mathrm{AR}_{n_1}(\cos(\alpha_1 y_1)) \otimes \cdots \otimes \mathrm{AR}_{n_d}(\cos(\alpha_d y_d)), \tag{25}$$

where $\otimes$ denotes Kronecker product, and we force

$$\mathrm{AR}_n(\alpha h_1 + \beta h_2) = \alpha \mathrm{AR}_n(h_1) + \beta \mathrm{AR}_n(h_2) \tag{26}$$

for every real α and β and for every d-variate real-valued cosine polynomials $h_1 = h_1(\cdot)$ and $h_2 = h_2(\cdot)$. It is clear that the request that $\mathrm{AR}_n(\cdot)$ is a linear operator (for $d > 1$, we impose this property in (26) by definition) is sufficient for defining completely the operator in the d-dimensional setting.

With the above definition of the operator $\mathrm{AR}_n(\cdot)$, we have

(1) $\alpha \mathrm{AR}_n(h_1) + \beta \mathrm{AR}_n(h_2) = \mathrm{AR}_n(\alpha h_1 + \beta h_2)$,

(2) $\mathrm{AR}_n(h_1)\mathrm{AR}_n(h_2) = \mathrm{AR}_n(h_1 h_2)$,

for real α and β and for cosine functions $h_1 = h_1(\cdot)$ and $h_2 = h_2(\cdot)$.

The latter properties of algebra homomorphism allows to define a commutative algebra $\mathcal{AR}$ of the matrices $\mathrm{AR}_n(h)$, with $h(\cdot)$ being a d-variate cosine polynomial. By standard interpolation arguments it is easy to see that $\mathcal{AR}$ can be defined as the set of matrices $\mathrm{AR}_n(h)$, where h is a d-variate cosine polynomial of degree at most $n_j - 3$ in the jth variable for every j ranging in $\{1, \ldots, d\}$: we denote the latter polynomial set by $\mathcal{P}_{n-2e}^{(d,\mathrm{even})}$, with e being the vector of all ones. Here we have to be a bit careful in specifying the meaning of algebra when talking of polynomials. More precisely, for $h_1, h_2 \in \mathcal{P}_{n-2e}^{(d,\mathrm{even})}$ the product $h_1 \cdot h_2$ is the unique polynomial $h \in \mathcal{P}_{n-2e}^{(d,\mathrm{even})}$ satisfying the following interpolation condition:

$$h(y) = z_y, \quad z_y \equiv h_1(y)h_2(y),\ \forall y \in G_{n-2}^{(d)}. \tag{27}$$

If the degree of h_1 plus the degree of h_2 in the j th variable does not exceed $n_j - 2$, $j = 1, \ldots, d$, then the uniqueness of the interpolant implies that h coincides with the product between polynomials in the usual sense. The uniqueness holds also for $d \geq 2$ thanks to the tensor form of the grid $G_{n-2}^{(d)}$ (see [28] for more details). The very same idea applies when considering inversion. In conclusion, with this careful

definition of the product/inversion and with the standard definition of addition, $\mathcal{P}_{n-2e}^{(d,\text{even})}$ has become an algebra, showing the vector-space dimension equal to $(n_1-2)\cdot(n_2-2)\cdots(n_d-2)$ which coincides with that of $\mathcal{AR}_n$.

Without loss of generality and for the sake of notational clarity, in the following we assume $n_j = n$ for $j = 1,\ldots,d$. Thanks to the tensor structure emphasized in (25)–(26), and by using Theorem 4 for every term $\text{AR}_n(\cos(\alpha_j y_j))$, $j = 1,\ldots,d$, of $\text{AR}_n(m)$ the d-level extension of such a theorem easily follows. More precisely, if h is a d-variate real-valued cosine symbol related to a d-dimensional strongly symmetric and normalized mask $\mathbf{h}$, then

$$\text{AR}_n(h) = T_n^{(d)} D_n \left(T_n^{(d)}\right)^{-1}, \qquad T_n^{(d)} = T_n \otimes \cdots \otimes T_n, \tag{28}$$

(d times) where D_n is the diagonal matrix containing the eigenvalues of $\text{AR}_n(h)$. The description of D_n in d dimensions is quite involved when compared with the case $d = 1$, implicitly reported in Theorem 1.

For a complete analysis of the spectrum of $\text{AR}_n(h)$ we refer the reader to [28]. Here we give details on a specific aspect. More precisely we attribute a correspondence in a precise and simple way among eigenvalues and eigenvectors, by making recourse only to the main d-variate symbol $h(\cdot)$. Let $\mathbf{x}_n = \mathbf{x}_n^{(1)} \otimes \mathbf{x}_n^{(2)} \otimes \cdots \otimes \mathbf{x}_n^{(d)}$ be a column of $T_n^{(d)}$, with $\mathbf{x}_n^{(j)} \in \{\alpha_n^{-1}[1,\ \mathbf{p}^T,\ 0]^T,\ \alpha_n^{-1}\ [0,\ (J\mathbf{p})^T,\ 1]^T\}$ or $\mathbf{x}_n^{(j)} = [0,\ \mathbf{q}_{s_j}^T,\ 0\ ]^T$, $1 \le s_j \le n-2$ and $\mathbf{q}_{s_j}$ is the (s_j) th column of Q_{n-2}, for $j = 1,\ldots,\ d$. Let

$$\begin{aligned}\mathcal{F}_{\mathbf{x}_n} &= \left\{ j \mid \mathbf{x}_n^{(j)} = \alpha_n^{-1}\left[1,\ \mathbf{p}^T,\ 0\right]^T \text{ or } \mathbf{x}_n^{(j)} = \alpha_n^{-1}\left[0,\ (J\mathbf{p})^T,\ 1\right]^T \right\} \\ &\subset \{1,\ldots,d\},\end{aligned} \tag{29}$$

with $\mathbf{x}_n$ being the generic eigenvector, that is, the generic column of $T_n^{(d)}$. The eigenvalue related to $\mathbf{x}_n$ is

$$\lambda = h\left(y_1^{(n)},\ldots,y_d^{(n)}\right), \tag{30}$$

where $y_j^{(n)} = 0$ for $j \in \mathcal{F}_{\mathbf{x}_n}$ and $y_j^{(n)} = \pi\nu_j/n - 1$ for $j \notin \mathcal{F}_{\mathbf{x}_n}$. Defining the d-dimensional grid

$$\hat{\text{G}}_n^{(d)} = \hat{G}_n \otimes \cdots \otimes \hat{G}_n, \quad d \text{ times}, \tag{31}$$

we can evaluate the d-variate function h on $\hat{G}_n^{(d)}$ by $h(\text{reshape}(\hat{G}_n^{(d)}, n))$, where

$$\text{reshape}(X, n) \tag{32}$$

arranges the entries of X in a d-dimensional array of length n along each direction according to Matlab notation. Using this notation the following compact and elegant result can be stated (its proof is omitted since it is simply the combination of the eigenvalue analysis in [28], of Theorem 4, and of the previous tensor arguments).

Theorem 5 ($AR_n(\cdot)$ Jordan Canonical Form). *The $n^d \times n^d$ AR-BCs blurring matrix A, obtained when using a strongly symmetric d-dimensional mask h such that $h_i = 0$ if $|i_j| \ge n-2$ for some $j \in \{1,\ldots,d\}$ (the d-dimensional degree condition), $n \ge 3$, coincides with*

$$\begin{aligned}&AR_n(h) \\ &= T_n^{(d)} \,\text{diag}\left(\text{reshape}\left(h\left(\text{reshape}\left(\hat{G}_n^{(d)}, n\right)\right), n^d\right)\right)\left(T_n^{(d)}\right)^{-1},\end{aligned} \tag{33}$$

where $T_n^{(d)}$ and $\hat{G}_n^{(d)}$ are defined in (28) *and* (31).

It is worth observing that the matrix $\text{AR}_n(h)$ spectrally analyzed in the previous theorem is exactly the same matrix arising from the imposition of AR-BCs applied separately in every direction, when h is the multivariate cosine symbol coming from the d-D tensor mask $\mathbf{h}$ defining the shift-invariant d-dimensional blurring operator.

3. Regularization by Reblurring

When the observed signal (or image) is noise-free, then there is a substantial gain of the reflective boundary conditions (R-BCs) with respect to both the periodic and zero Dirichlet BCs and, at the same time, there is a significant improvement of the AR-BCs with regard to the R-BCs (see [17, 30]). Since the deconvolution problem is ill posed (components of the solution related to high frequency data errors are greatly amplified) regardless of the chosen BCs, it is evident that we have to regularize the problem. Two classical methods, that is, Tikhonov regularization, with direct or iterative solution of the Tikhonov linear system, and regularization iterative solvers, with early termination, for normal equations (CG [31] or Landweber method [32]) can be used. We observe that in both the cases, the coefficient matrix involves a shift of A^TA and that the righthand-side is given by $A^T\mathbf{g}$. Quite surprisingly, the AR-BCs may be spoiled by this approach at least for $d = 1$ and if we compute explicitly the matrix A^TA and the vector $A^T\mathbf{g}$, see [33]: more in detail, even in presence of a moderate noise and a strongly symmetric PSF, the accuracy of AR-BCs restorations becomes worse in some examples than the accuracy of R-BCs restorations (see [33]). The reason of this fact relies upon the properties of the matrix A^T, and we give some insights in the following.

3.1. The Reblurring Operator. A key point is that, for zero Dirichlet, periodic and reflective BCs, when the kernel h is symmetric, the matrix A^T is still a blurring operator since $A^T = A$, while, in the case of the AR-BCs matrix, A^T cannot be interpreted as a blurring operator. A (normalized) blurring operator is characterized by nonnegative coefficients such that every row sum is equal to 1 (and it is still acceptable if it is not higher than 1): in the case of A^T with AR-BCs the row sum of the first and of the last row can be substantially larger than 1. This means that modified signal $A^T\mathbf{g}$ may have artifacts at the borders and this seems to be a potential motivation for which a reduction of the reconstruction quality occurs.

Furthermore, the structure of the matrix A^TA is also spoiled and, in the case of images ($d = 2$) we lose the $O(n^2 \log(n))$ computational cost for solving a generic system $A^TA\mathbf{x} = \mathbf{b}$. The cost of solving such a type of linear systems is proportional to n^3 by using for example, Shermann-Morrison formulae (which by the way can be numerically unstable [34]). Dealing with higher dimensions, the scenario is even worse [35], because in the d-dimensional setting the solution of the normal equation linear system is asymptotic to $n^{3(d-1)}$, where n^d is the size of the matrix A. In order to overcome these problems (which arise only with the most precise AR-BCs for strongly symmetric PSFs), we replace A^T by A', where A' is the matrix obtained by imposing the current BCs to the center-flipped PSF (i.e., in the 2D case, to the PSF rotated by 180 degrees).

For the sake of simplicity we first consider a strongly symmetric PSF so that the associated normal equations can be read as $A^2\mathbf{f} = A\mathbf{g}$, whenever zero Dirichlet, periodic or reflective BCs are considered. Therefore, the observed image $\mathbf{g}$ is reblurred. The reblurring is the key of the success of classical regularization techniques (Tikhonov or CG, Landweber for the normal equations) since also the noise is blurred and this makes the contribution of the noise less evident. We notice that the two systems $A^2\mathbf{f} = A\mathbf{g}$ and $A\mathbf{f} = \mathbf{g}$ are algebraically equivalent if A is invertible: in any case, if A is also positive definite, the first represents the minimization of the functional $\|A\mathbf{f} - \mathbf{g}\|_2^2$ while the second represents the minimization of the functional $\|A^{1/2}\mathbf{f} - A^{-1/2}\mathbf{g}\|_2^2$ so that the first can be considered the blurred version of the second and in fact the first approach is uniformly better than the second. On these grounds, in the case of AR-BCs, since $A^T \neq A$, we can replace A^T by A which is again a low-pass filter (see [33]). In this way, we overcome also the computational problems.

In order to provide a general reblurring approach also in the case of nonsymmetric PSFs, we consider the correlation operation instead of the transposition (see [36]). In the discrete 2D case, the correlation performs the same operation of the convolution, but rotating the mask (the PSF in our case) of 180 degrees. Moreover, we note that in the continuous case over an infinite domain, the correlation and the adjoint are exactly the same operation, provided that the convolution kernel is real. Indeed, let K be the convolution operator with shift-invariant kernel $k(s)$, then $[Kf](x) = \int k(x-y)f(y)\mathrm{d}y$. Since the PSF (and then k) is real (and then real valued), the adjoint operator K^* is $[K^*f](x) = \int k(y-x)f(y)\mathrm{d}y$ which is a correlation operator. We remark that here the convolution and the correlation use the same kernel except for the sign of the variable (i.e., $k(\mathbf{s})$ vs $k(-\mathbf{s})$), and, in the 2D case, the change of sign in the variable $\mathbf{s}$ of the kernel can be viewed as a 180 degrees rotation of the PSF mask.

By virtue of these arguments, in order to overcome the problems arising with the normal equations approach for AR-BCs 2D restorations, we propose to replace A^T by A' (the reblurring matrix), where A' is the matrix obtained by imposing the BCs to the PSF rotated by 180 degrees. Using Matlab notation, if H is a $q \times q$ PSF, its 180 degrees rotated version is $H' = \mathrm{fliplr}(\mathrm{flipud}(H)) = J_qHJ_q$, where J_q is the flip matrix defined as $[J_q]_{i,j} = 1$ if $i + j = q + 1$ for $i, j = 1, \ldots, q$, and zero otherwise. For a d-dimensional problem, A' is obtained by imposing the BCs to the PSF flipped with respect to the origin, or, in other words, to the new PSF where all the coefficients are flipped with respect to every variable.

In this way A' has the same computational properties of A (it belongs to $\mathcal{AR}$ in the case of AR BCs) and it is certainly a low-pass filter. In the reblurring approach the normal equations are replaced by

$$A'A\mathbf{f} = A'\mathbf{g}. \tag{34}$$

Furthermore, using the Tikhonov regularization in the reblurred version, we propose to use

$$[A'A + \mu L'L]\mathbf{f} = A'\mathbf{g}, \tag{35}$$

with L being any discrete. differential operator with AR-BCs. In general $A'A$ is nonsymmetric, but the asymptotic eigenvalue analysis in [28, Section 3.3] has shown that the spectrum is clustered around the positive real interval given by the range of $|h|^2$ if h is the symbol associated to the PSF. Such a cluster is strong if the decay of the PSF coefficients is fast enough, as it occurs in real world PSFs. The latter statements give a reasonable support to the applicability of the CGLS when A^T is replaced by A' and, indeed, in our numerical tests such a regularizing method has always worked perfectly. From the viewpoint of the modeler, the previous considerations can be summarized in the following motivation. The image restoration problem is the restriction in the FOV of an infinite dimensional problem. We can follow two ways to design the linear system to solve:

(1) to impose BCs and then to look at a least-squares solution,

(2) to formulate a least-squares solution on the infinite dimensional problem, and then to impose the BCs to the two infinite dimensional operators K and K^*, separately.

A third possibility is to formulate a least-squares solution on the infinite dimensional problem, and then to impose the BCs to this minimum problem: a difficulty in this case is that, even without noise, the resulting system is not equivalent in an algebraic sense to the original equations $A\mathbf{f} = \mathbf{g}$. In the first case we resort to the normal equations in the finite dimensional space. On the contrary, in the second case we apply the BCs to K and K^* in the infinite dimensional normal equations (where the adjoint operator K^* performs a correlation operation) and then we obtain (34). More precisely, the discretization of K and K^* in the continuous equation $K^*Kf = K^*g$ with any fixed BCs gives (34).

3.2. Linear Algebra and Computational Issues. We note that in the 1D case $A'_n = J_nA_nJ_n$. In the d-dimensional case, let $n = (n_1, n_2, \ldots, n_d)$ be the partial dimensions of the problem, whose total size is $\prod_{i=1}^{d} n_i$. By flipping each variable, we obtain

$$A'_n = J_nA_nJ_n, \quad J_n = \bigotimes_{i=1}^{d} J_{n_i}. \tag{36}$$

For the analysis of properties of the reblurring scheme (34) with respect to all the different BCs, we now study the discretization of the continuous operator K. Let us consider the Toeplitz d-level matrix $T_n(\phi)$ of partial dimensions $n = (n_1, \ldots, n_d) \in \mathbb{N}_+^d$ and generating function ϕ [2], which is defined as

$$T_n(\phi) = \sum_{|j| \le n-e} a_j Z_n^{[j]} = \sum_{|j_1|<n_1} \cdots \sum_{|j_d|<n_d} a_{(j_1,\ldots,j_d)} Z_{n_1}^{[j_1]} \otimes \cdots \otimes Z_{n_d}^{[j_d]} \tag{37}$$

$(e = (1, \ldots, 1) \in \mathbb{N}_+^d)$ by means of the Fourier coefficients of ϕ

$$a_k = \frac{1}{(2\pi)^d} \int_{[-\pi,\pi]^d} \phi(x) \exp^{-\mathbf{i}\langle k|x\rangle} \mathrm{d}x, \quad \mathbf{i}^2 = -1,\ k \in \mathbb{Z}^d. \tag{38}$$

Here $\langle k \mid x\rangle = \sum_{i=1}^d k_i x_i$ and for $j \in \mathbb{Z}$, $m \in N_+$, $Z_m^{[j]} \in \mathbb{R}^{m\times m}$ is the matrix whose (s,r) th entry is 1 if $s - r = j$, and 0 elsewhere. As it is well known for multilevel Toeplitz matrices $T_n^H(\phi) = T_n(\overline{\phi})$, where $\overline{\phi}$ is the conjugate of the function ϕ, and the Fourier coefficients of $\overline{\phi}$ are the same of ϕ, but conjugated and flipped. Moreover, since $T_n(\overline{\phi}) = J_n \overline{T_n(\phi)} J_n$, if $A_n = T_n(\phi)$ is real then $A_n^T = J_n A_n J_n = A_n'$. This means that for Dirichlet BCs (D-BCs) and periodic BCs (P-BCs) the reblurring approach is exactly equal to the classical normal equations approach, since in these two cases the corresponding blurring matrix A_n is multilevel Toeplitz: indeed, concerning P-BCs, we notice that the resulting multilevel circulant structure is a special instance of the multilevel Toeplitz case. Unfortunately, in the case of Hankel matrices (or multilevel mixed Toeplitz-Hankel matrices with at least one Hankel level) this is no longer true in general. However, a sufficient and necessary condition to have $A_n' = A_n^T$ is $J_n A_n = (J_n A_n)^T$ (or equivalently $A_n J_n = (A_n J_n)^T$), which is a multilevel antidiagonal symmetry called persymmetry. Therefore in the case of R-BCs, where the matrix A_n involves nested Hankel parts, in general $A_n' \neq A_n^T$, while $A_n' = A_n^T$ only when the PSF is strongly symmetric since in this case $J_n A_n = (J_n A_n)^T$. Dealing with the AR-BCs, the situation is even more involved, since $A'_n \neq A_n^T$ also for strongly symmetric PSFs, owing to the low-rank correction term. Hence, we can state that the reblurring is a new proposal not only for the AR-BCs, but also for all those BCs for which $J_n A_n \neq (J_n A_n)^T$. As a nontrivial and unexpected example, it is important to stress that the imposition of R-BCs with nonstrong symmetric PSFs implies $J_n A_n \neq (J_n A_n)^T$, that is, $A_n' \neq A_n^T$.

We provide now a computational motivation for the choice of using A' as an alternative to A^T : A' is the usual operation which has to be implemented to perform the adjoint operation in the Fourier domain. Indeed, the convolution with prescribed BCs can be implemented by first enlarging the image according to the considered BCs and then by computing the matrix vector product by a simple circular convolution operation, see [37]. More precisely, let X and H be two matrices such that X represents an $n \times n$ image and H is the discrete $q \times q$ 2D PSF, $q = 2m + 1$, which leads to the matrix blurring A. By using the Matlab notation $\mathbf{x} = X(:)$ (i.e., the vector $\mathbf{x}$ is the column-stacked version of X), the product $A\mathbf{x}$ in the 2D case can be implemented

(1) by using an enlarged image $\widetilde{X}$, which is the $(n + q - 1) \times (n + q - 1)$ image X extended at the boundaries according to the imposed BCs,

(2) computing $H * \widetilde{X}$, where the symbol "$*$" denotes the circular convolution operator (H should be zero padded to have the same size of $\widetilde{X}$),

(3) and then taking the inner part of the result having the same size of X.

The circular convolution can be computed using the 2D discrete Fourier transform (DFT2) and its inverse (IDFT2), since we have

$$H * \widetilde{X} = \mathrm{IDFT2}\Big(\mathrm{DFT2}(H) \odot \mathrm{DFT2}\Big(\widetilde{X}\Big)\Big), \tag{39}$$

where "$\odot$" is the componentwise product. If $\mathcal{C}_k(f)$ denotes the block circulant matrix with circulant blocks, of block size k with blocks of size k and generating function f, then (39) represents the same operation as $\mathcal{C}_{n+q-1}(\phi)\widetilde{\mathbf{x}}$, where $\widetilde{\mathbf{x}} = \widetilde{X}(:)$ (according to a 2D ordering). Conversely, it is well known that the operation corresponding to the product with the adjoint operator, in the Fourier domain gives rise to

$$Y = H' * \widetilde{X} = \mathrm{IDFT2}\Big(\overline{\mathrm{DFT2}(H)} \odot \mathrm{DFT2}\Big(\widetilde{X}\Big)\Big), \qquad H'_{i,j} = H_{-i,-j}, \quad i, j \in \mathbb{Z}, \tag{40}$$

where the overline symbol denotes the complex conjugation. As a result, since the transform $H \mapsto H'$ is equivalent to the transform $\phi \mapsto \overline{\phi}$ (because $\phi(x,y) = \sum_{i,j\in\mathbb{Z}} H_{i,j} \exp^{\mathbf{i}\langle(i,j)|(x,y)\rangle}$), and since $\mathcal{C}_{n+q-1}(\overline{\phi}) = \mathcal{C}_{n+q-1}^T(\phi)$, if $\mathbf{y} = Y(:)$, then it follows that $\mathbf{y} = \mathcal{C}_{n+q-1}^T(\phi)\widetilde{\mathbf{x}}$. Therefore, for any of the considered BCs, the inner part of $\mathbf{y}$ is exactly $A'\mathbf{x}$. Here it is worthwhile to specify exactly what we mean for inner part: if the vector $\mathbf{y}$ is partitioned in $n+q-1$ blocks of size $n + q - 1$, $q = 2m + 1$, then for inner part we mean that we are excluding the first and the last m blocks and, in any of the remaining blocks, we are deleting the first and the last m entries. More generally, if the PSF is arbitrary (e.g. nonsymmetric) that is, the nonzero coefficients of the PSF have first index belonging to $[-m_1^-, m_1^+]$ and second index in the range $[-m_2^-, m_2^+]$, then we have to delete the first m_1^- and the last m_1^+ blocks and, in any of the other blocks, we have to exclude the first m_2^- and the last m_2^+ entries.

Since the DFT and its inverse can be computed in $O(n^2 \log(n))$ arithmetic operations using FFTs, the previous approach is implemented in the Matlab toolbox RestoreTools [37]. We have added the AR-BCs in such a toolbox for the matrix vector product, suitable for iterative regularizing methods. This code has been used for the numerical tests of Section 3 and it is downloadable from the homepage "http://scienze-como.uninsubria.it/mdonatelli/software.html".

3.3. Filtering Methods for AR-BCs Matrices. As mentioned in Section 1, regardless of the imposed BCs, matrices A that arise in signal and image restoration are typically severely ill conditioned, and regularization is needed in order to compute a stable approximation of the solution of (1). A class of regularization methods is obtained through spectral filtering [38, 39]. Specifically, if the spectral decomposition of A is

$$A = T_n \ \mathrm{diag}(\mathbf{d}) \ T_n^{-1}, \quad T_n = \begin{bmatrix} \mathbf{t}_1 & \mathbf{t}_2 & \cdots & \mathbf{t}_n \end{bmatrix}, \ T_n^{-1} = \begin{bmatrix} \widetilde{\mathbf{t}}_1^T \\ \widetilde{\mathbf{t}}_2^T \\ \vdots \\ \widetilde{\mathbf{t}}_n^T \end{bmatrix}, \tag{41}$$

with $\mathbf{d} = h(\widehat{G}_n)$, then a spectral filter solution is given by

$$\mathbf{f}_{\mathrm{reg}} = \sum_{i=1}^{n} \phi_i \frac{\widetilde{\mathbf{t}}_i^T \mathbf{g}}{d_i} \mathbf{t}_i, \tag{42}$$

where ϕ_i are filter factors that satisfy

$$\phi_i \approx \begin{cases} 1, & \text{if } d_i \text{ is large,} \\ 0, & \text{if } d_i \text{ is small.} \end{cases} \tag{43}$$

The small eigenvalues correspond to eigenvectors with high frequency components, and are typically associated with the noise space, while the large eigenvalues correspond to eigenvectors with low-frequency components, and are associated with the signal space. Thus filtering methods attempt to reconstruct signal space components of the solution, while avoiding reconstruction of noise space components.

For example, the filter factors for two well-known filtering methods, truncated spectral value decomposition (TSVD) and Tikhonov regularization, are

$$\phi_i^{\mathrm{tsvd}} = \begin{cases} 1, & \text{if } d_i \geq \delta, \\ 0, & \text{if } d_i < \delta, \end{cases} \quad \phi_i^{\mathrm{tik}} = \frac{d_i^2}{d_i^2 + \lambda}, \quad \lambda > 0, \tag{44}$$

where the problem dependent *regularization parameters* δ and λ must be chosen [39]. Several techniques can be used to estimate appropriate choices for the regularization parameters when the SVD is used for filtering (i.e., d_i are the singular values), including generalized cross validation (GCV), L-curve, and the discrepancy principle [38, 40, 41].

In our case, the notation in (44) defines a slight abuse of notation, because the eigenvalues d_i are not the singular values: in fact the Jordan canonical form (CF) in (24) is different from the singular value decomposition (SVD), since the transform T_n is not orthogonal (indeed it is a rank-2 correction of a symmetric orthogonal matrix). Therefore, note that the use of ϕ_i^{tsvd} in (42) defines the filtering of the eigenvalues in the Jordan CF instead of the more classical filtering of the singular values in the SVD. However, we note that in general computing the SVD can be computationally very expensive, especially in the multidimensional case and also in the strongly symmetric case. Moreover, quite surprisingly, a recent and quite exhaustive set of numerical tests, both in the case of signals and images (see [42]), has shown that the truncated Jordan CF is more or less equivalent to the truncated SVD in terms of quality of the restored object: indeed this is a delicate issue that deserves more attention in the future.

Our final aim is to compute (42) in a fast and stable way. This is the classic approach implemented for instance with periodic BCs by using three FFTs. In our case we employ the AR-transform (21), its inverse (23), and a fast algorithm for computing the eigenvalues described in [28].

Algorithm 6. (1) $\widetilde{\mathbf{g}} = T_n^{-1}\mathbf{g}$.

(2) $\mathbf{d} = [h(0), \ \widehat{\mathbf{d}}^T, \ h(0)]^T$, where $\widehat{\mathbf{d}} = [d_2, \ \ldots, \ d_{n-1}]^T$ are the eigenvalues of $\tau_{n-2}(h)$ that can be computed by a fast sine transform (FST).

(3) $\widetilde{\mathbf{f}} = (\phi \,./\, \mathbf{d}) \odot \widetilde{\mathbf{g}}$, where the dot operations are component-wise.

(4) $\mathbf{f}_{\mathrm{reg}} = T_n\widetilde{\mathbf{f}}$.

The product $T_n\widetilde{\mathbf{f}}$ can be clearly computed in a fast and stable way by one FST. Indeed for all $\mathbf{x} \in \mathbb{R}^n$

$$T_n\mathbf{x} = \alpha_n^{-1}x_1 \begin{bmatrix} 1 \\ \mathbf{p} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ Q_{n-2}\mathbf{x}(2:n-1) \\ 0 \end{bmatrix} + \alpha_n^{-1}x_n \begin{bmatrix} 0 \\ J\mathbf{p} \\ 1 \end{bmatrix}, \tag{45}$$

where $\mathbf{x}(2:n-1)$ in Matlab notation is the vector $\mathbf{x}$ with components indexed from 2 to $n-1$. A similar strategy can be followed for computing the matrix-vector product $T_n^{-1}\mathbf{g}$. Instead of $\alpha_n^{-1}\mathbf{p}$ there is $\mathbf{u} = -Q_{n-2}\mathbf{p}$ and instead of $\alpha_n^{-1}J\mathbf{p}$ there is $\mathbf{w} = -Q_{n-2}J\mathbf{p}$. Recalling that $\mathbf{p} = L_{n-2}^{-1}\mathbf{e}_1$ the two vectors $\mathbf{u}$ and $\mathbf{w}$ can be explicitly computed obtaining $u_i = (2n-2)^{-1/2}\cot(i\pi/(2n-2))$, for $i = 1, \ldots, n-2$ and $\mathbf{w} = \mathrm{diag}_{i=1,\ldots,n-2}(-1)^{i+1}\mathbf{u}$.

Remark 7. As discussed in Remark 3, there is a natural interpretation in terms of frequencies when considering one-dimensional periodic and reflective BCs. The eigenvalue obtained as a sampling of the symbol h at a grid-point close to zero, that is, close to the maximum point of h, has an associated eigenvector that corresponds to low-frequency (signal space) information, while the eigenvalue obtained as a sampling of the symbol h at a grid-point far away from zero (and, in particular, close to π), has an associated eigenvector that corresponds to high-frequency (noise space) information. Concerning antireflective BCs, the same situation occurs when dealing with the frequency eigenvectors $\sqrt{2/(n-1)}(\sin(jy))|_{\widetilde{G}_n}$, $j = 2, \ldots, n-1$. The other two exceptional eigenvectors generate the space of linear polynomials and therefore they correspond to low-frequency information: this intuition is well supported by the fact that the related eigenvalue is $h(0)$, that is, the maximum and the infinity norm of h, and by the fact that AR-BCs are more precise than other classical BCs.

Remark 8. For $d = 1$ and with reference to the previous sections, we have proved that, thanks to the definition of a (fast) AR-transform, it is possible to define a truncated spectral decomposition which is computationally very effective and, surprisingly enough, quite competitive when compared with the celebrated but costly truncated SVD in terms of restoration quality. However, we are well aware that the real challenge is represented by a general extension to the multidimensional setting. Indeed, except for more involved multi-index notations, all the techniques are plainly generalized in the multilevel setting, maintaining a cost proportional to three d-level FSTs of size $(n-2)^d$, and the key tool is the simplified eigenvalue-eigenvector correspondence concisely indicated in Theorem 5. In reality, regarding the previous Algorithm 6 the only difficult task is the computation in step (2), where we have to compute the eigenvalues in the right order. For this task we refer to [28], where an algorithm is proposed and studied: more specifically the related procedure in [28] is based on a single d-level FST of size $(n-2)^d$ plus lower order computations.

3.4. Convergence of the Reblurring Approach. We remark that the antireflective transform can be defined also by the eigenvector matrix

$$V_n = \left[\begin{array}{c|c|c} & 0 & \\ 1 & Q & \boldsymbol{\ell} \\ & 0 & \end{array}\right], \tag{46}$$

where

$$1 = \frac{1}{\sqrt{n}}\begin{bmatrix}1\\1\\\vdots\\1\end{bmatrix}, \qquad \boldsymbol{\ell} = \frac{\sqrt{3}}{\sqrt{n(n^2-1)}}\begin{bmatrix}1-n\\3-n\\\vdots\\n-1\end{bmatrix}. \tag{47}$$

Note that 1 and $\boldsymbol{\ell}$ differ from the corresponding eigenvectors $[1, \mathbf{p}^T, 0]^T$ and $[0, J\mathbf{p}^T, 1]^T$ used in Section 2.4, but they are a linear combination of them. They have been chosen here to form an *orthonormal* basis of the grid samples of all linear polynomials and this property will be useful in the following.

According to Theorem 4 the spectral decomposition of $\mathrm{AR}_n(h)$ can now be written as

$$\mathrm{AR}_n(h) = V_n \Lambda V_n^{-1}, \tag{48}$$

where the diagonal entries λ_{jj} of Λ are given by

$$\lambda_{jj} = \begin{cases} h\left(\dfrac{j-1}{n-1}\pi\right), & 1 \le j < n, \\ h(0), & j = n. \end{cases} \tag{49}$$

Here we prove that the reblurring approach for Tikhonov regularization in (35) where $L = I$ is a regularization method. For a complete analysis we need to compute the SVD of V_n.

We use the notation $\mathbf{y} \doteq \mathbf{z}$ if the two vectors $\mathbf{y}$ and $\mathbf{z}$ depend on n and for each entry y_i of $\mathbf{y}$ and the corresponding entry z_i of $\mathbf{z}$ there holds $y_i/z_i \to 1$ as $n \to \infty$.

Theorem 9 (see [43]). *The two dominant singular values of V_n are given by*

$$\sigma_1 \doteq \sigma_2 \doteq \sqrt{2}, \tag{50}$$

where $\sqrt{2}$ is, in fact, a strict upper bound, and the two minimal singular values are given by

$$\sigma_{n-1} \doteq \frac{\sqrt{3}}{\sqrt{n}} \quad \text{and} \quad \sigma_n \doteq \frac{1}{\sqrt{n}}, \tag{51}$$

respectively. Fix $1' = 1(2:n-1)$, $\boldsymbol{\ell}' = \boldsymbol{\ell}(2:n-1)$, $\mathbf{a} = Q1'$, and $\mathbf{b} = Q\boldsymbol{\ell}'$. The corresponding right singular vectors are

$$\begin{aligned} \mathbf{v}_1 &\doteq \frac{1}{\sqrt{2}}\begin{bmatrix}1\\\mathbf{a}\\0\end{bmatrix}, \qquad \mathbf{v}_2 \doteq \frac{1}{\sqrt{2}}\begin{bmatrix}0\\\mathbf{b}\\1\end{bmatrix}, \\ \mathbf{v}_{n-1} &\doteq \frac{1}{\sqrt{2}}\begin{bmatrix}0\\\mathbf{b}\\-1\end{bmatrix}, \qquad \mathbf{v}_n \doteq \frac{1}{\sqrt{2}}\begin{bmatrix}-1\\\mathbf{a}\\0\end{bmatrix}, \end{aligned} \tag{52}$$

and the left singular vectors are

$$\begin{aligned} \mathbf{u}_1 &\doteq \begin{bmatrix}\dfrac{1}{(2\sqrt{n})}\\1'\\\dfrac{1}{(2\sqrt{n})}\end{bmatrix}, \qquad \mathbf{u}_2 \doteq \begin{bmatrix}\dfrac{l_1}{2}\\\boldsymbol{\ell}'\\\dfrac{l_n}{2}\end{bmatrix}, \\ \mathbf{u}_{n-1} &\doteq \frac{1}{\sqrt{2}}\begin{bmatrix}1\\\sqrt{\dfrac{3}{n}}\,\boldsymbol{\ell}'\\-1\end{bmatrix}, \qquad \mathbf{u}_n \doteq \frac{1}{\sqrt{2}}\begin{bmatrix}-1\\\dfrac{1'}{\sqrt{n}}\\-1\end{bmatrix}, \end{aligned} \tag{53}$$

respectively. The remaining singular values are equal to one, and the corresponding left and right singular vectors have homogeneous boundary values.

Now we are in the position to determine the condition number of the antireflective transform to first order.

Corollary 10. *The condition number of the antireflective transform satisfies*

$$\mu(V_n) \doteq \sqrt{2n}, \quad n \longrightarrow \infty. \tag{54}$$

Remark 11. It is important to note that the ill-conditioned subspace of V_n^{-1} has dimension two, independent of n, since V_n has two singular values that decay like $1/\sqrt{n}$ while all others are between one and two. Also, V_n^{-1} only amplifies vectors that fail to be orthogonal to $\mathcal{U} = \mathrm{span}\{\mathbf{u}_{n-1}, \mathbf{u}_n\}$. According to Theorem 9 the vectors from $\mathcal{U}$ are essentially zero, except for their two boundary values.

Convergent bounds for the Tikhonov reblurring approach, can be obtained with the usual remedy from the theory of ill-posed problems, which consists in so-called smoothness assumptions, the most simple one being as follows.

Assumption 12. Let f be itself a blurred version of a continuous signal w, that is,

$$f(x) = \int_{\mathbb{R}} k(x-y)w(y)dy, \quad x \in \mathbb{R}, \tag{55}$$

where w satisfies the same BCs as f (i.e., periodic, reflective, or antireflective ones).

On the grounds of Assumption 12 we may therefore assume that

$$\mathbf{f} = A\mathbf{w}, \quad \text{for some } \mathbf{w} \in \mathbb{R}^n \tag{56}$$

with a moderate bound

$$|||\mathbf{w}||| \leq \|\mathbf{w}\|_\infty \leq \varrho, \tag{57}$$

where $|||\mathbf{x}||| := \|\mathbf{x}\|_2/\sqrt{n}, \mathbf{x} \in \mathbb{R}^n$. Since the observed object is usually affected by noise, instead of the blurred object $\mathbf{g}$ we have to work with the blurred and noisy object $\tilde{\mathbf{g}} = \mathbf{g} + \mathbf{e}$, where $\|\mathbf{e}\|_\infty < \varepsilon$. In this way, for $L = I$ (35) becomes $(A'A + \mu I)\mathbf{f} = A'\mathbf{g}$.

Theorem 13. *Let the exact solution* $\mathbf{f}$ *of* (1) *satisfy* (56) *with* (57). *Furthermore the total error of the reblurring strategy with AR BCs satisfies*

$$|||\mathbf{f}_\alpha^\varepsilon - \mathbf{f}||| = O\left(\sqrt{\varepsilon\varrho}\right), \tag{58}$$

for $\alpha = \alpha(\varepsilon) = \varepsilon/\varrho$, *where the constant in the* $O(\cdot)$*-notation is independent of the dimension* n.

Note that the upper bound from Theorem 13 is the same as for Tikhonov regularization with reflective or periodic BCs; only the constant hidden in the $O(\cdot)$-notation may be somewhat larger for the reblurring strategy.

4. Numerical Results

The section is devoted to numerical experiments, with reference to the Tikhonov regularization in the reblurred version and to the classical conjugate gradient (CG) regularization with early termination. In both cases the use of antireflective BCs improves the quality of the restored image, without penalizing the computational cost. Another promising approach not discussed here both from the viewpoint of the quality and of the complexity is that based on a regularized version of multigrid-type techniques (see [44, 45]): also this idea can be successfully implemented in combination with the choice of AR BCs.

In our numerical experiments we use Matlab 6.5 and the toolbox RestoreTools [37] suitably extended for dealing with AR-BCs. The relative restoration error (RRE) is $\|\widehat{\mathbf{f}} - \mathbf{f}\|_2/\|\mathbf{f}\|_2$, where $\widehat{\mathbf{f}}$ is the computed approximation of the true image $\mathbf{f}$. The signal-to-noise ratio (SNR) is computed as $20\log_{10}\|\mathbf{g}_b\|_2/\|\boldsymbol{\nu}\|_2$, where $\mathbf{g}_b$ is the blurred image without noise and $\boldsymbol{\nu}$ is the noise vector [32].

Table 1: RRE for the test problem in Figure 1.

Noise	Reflective	AR
10%	0.1284	0.1261
1%	0.1188	0.1034
0.1%	0.1186	0.0989

4.1. Reblurring and Tikhonov Regularization. Let us begin with an example illustrating the approach discussed in Section 3.3 for a 2-dimensional imaging problem. We do not take an extensive comparison of the AR-BCs with other classic BCs, like periodic or reflective, since the topic and related issues have been already widely discussed in several works (see e.g., [19, 33, 46]), where the advantage on some classes of images, in terms of the restored image quality, of the application of AR-BCs has been emphasized. Here we present only a 2D image deblurring example with Gaussian blur and various levels of white Gaussian noise.

The true and the observed images are in Figure 1, where the observed image is affected by a Gaussian blur and 1% noise. We compare the AR-BCs only with the reflective BCs since for this test other BCs like periodic or Dirichlet do not produce satisfactory restorations. In Figure 2 we observe a better restoration and reduced ringing effects at the edges for AR-BCs with respect to reflective BCs. Restored images in Figure 2 are obtained with the minimum relative restoration error varying several values of the regularization parameter λ.

From Table 1, we note that for the 10% noise case, all of the approaches give comparable restorations. On the other hand, decreasing the noise, that is, passing to 1% and then to 0.1% noise, the AR-BCs improve the restoration while the reflective BCs are not able to do that, due to the barrier of the ringing effects.

4.2. Reblurring and CG Regularization. In the following tests, the reblurring strategy will be applied with R-BCs and AR-BCs when the PSF is not necessarily strongly symmetric. Indeed, in the case of D-BCs and P-BCs, the reblurring approach is equal to the classical normal equations, while, in the case of R-BCs and AR-BCs, this is no longer true in general.

We provide two problems using iterative regularization by CG.

Test I: Cameraman. The first test is reported in Figure 3. The true image is a cameraman and the 61×61 PSF is associated with a Gaussian distribution in the square domain $[-30, 30] \times [-30, 30]$, with variance equal to four. We add a Gaussian noise.

Test II: Astronomy. We are dealing with a nonsymmetric experimental 256×256 PSF developed by US Air Force Phillips Laboratory, Lasers and Imaging Directorate, Kirtland Air Force Base, New Mexico, widely used in literature (see e.g. [12, 47]). The true object is the image of Saturn in Figure 4; a Poissonian noise is added, as it is customary when dealing with astronomical images.

Figure 1: Test problem with Gaussian blur and 1% white Gaussian noise.

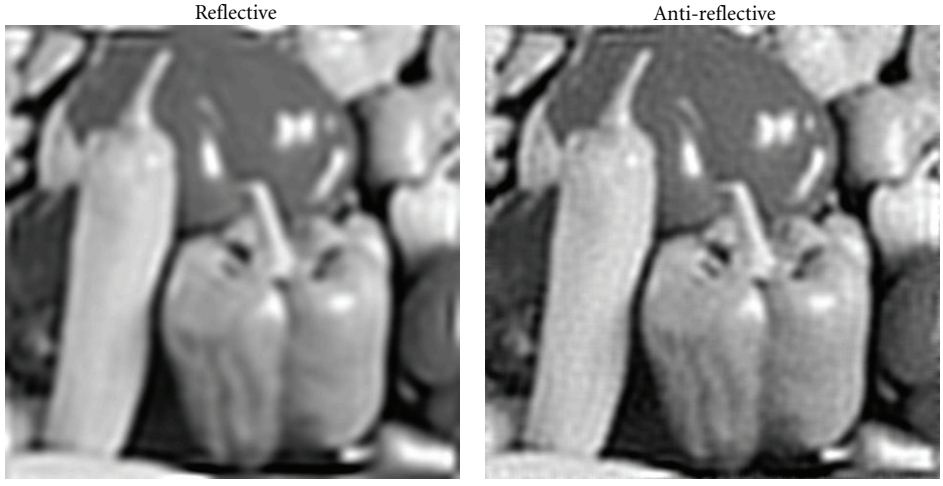


Figure 2: Restored images for the test problem in Figure 1.

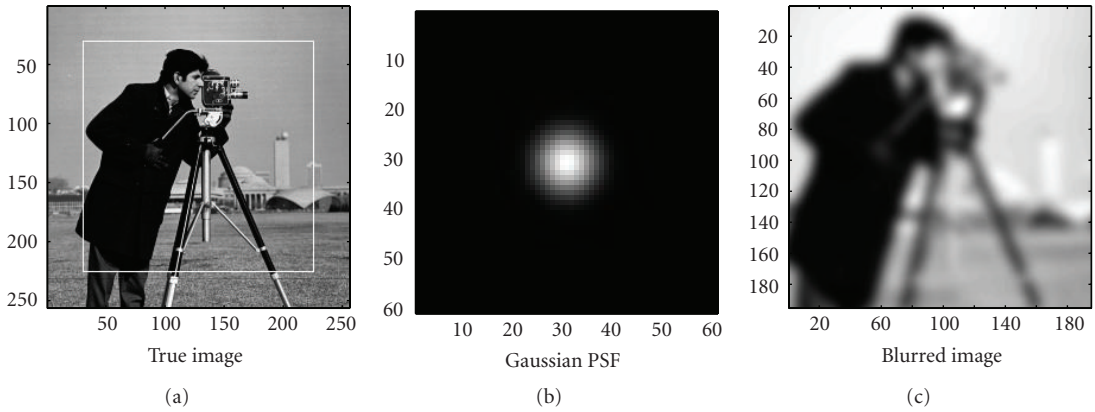


Figure 3: Test I: true image, Gaussian PSF, and the blurred image without noise.

Table 2: Test I: best RREs and L^2 norm of the residuals for CG with reblurring varying the SNR (SNR = ∞ means 0% of noise).

SNR	Relative restoration errors			L^2 norm of the residuals ($\|\mathbf{g} - A\hat{\mathbf{f}}\|^2$)		
	Periodic	Reflective	Antireflective	Periodic	Reflective	Antireflective
∞	0.2275	0.1993	0.1831	1.2363	0.0113	0.0004
50	0.2276	0.1996	0.1850	1.2728	0.0480	0.0374
40	0.2278	0.2007	0.1921	1.6078	0.3766	0.3654
30	0.2300	0.2088	0.2051	4.9395	3.7346	3.7133
20	0.2487	0.2382	0.2378	38.2357	37.2100	37.2149
10	0.3836	0.3814	0.3823	379.2445	376.0303	376.0419

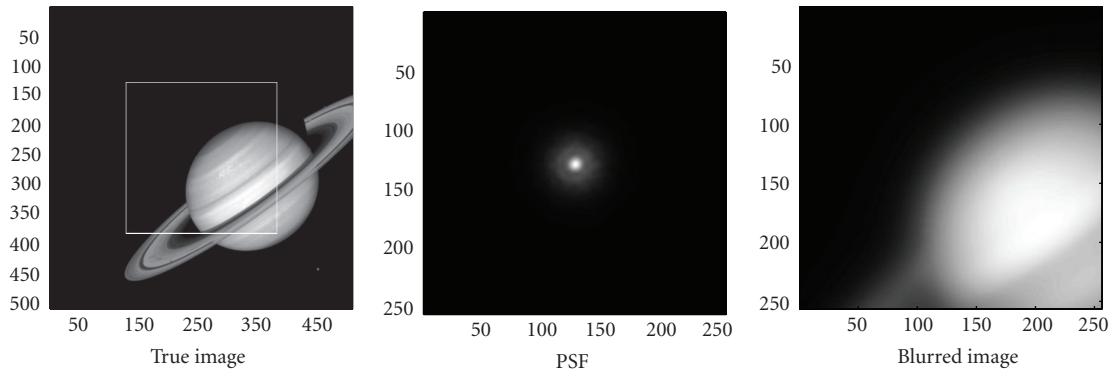


Figure 4: Test II: true image, PSF, and the blurred image without noise.

Table 3: Test II: best RREs for CG with reblurring varying the SNR (SNR = ∞ means 0% of noise) within 200 iterations.

SNR	Periodic	Reflective	Antireflective
∞	0.2415	0.1529	0.0604
50	0.2416	0.1530	0.0616
40	0.2418	0.1532	0.0737
30	0.2430	0.1575	0.1052
20	0.2574	0.1838	0.1605
10	0.3689	0.3329	0.3289

We show the results corresponding only to P-BCs, R-BCs, and AR-BCs. For shortness, we do not report the reconstructions coming from D-BCs, since the related restorations are usually not better than those with P-BCs.

Tables 2 and 3 show the best RREs for various levels of noise. In Table 2 we also report the L^2 norm of the residuals, that is, $\|\mathbf{g} - A\hat{\mathbf{f}}\|_2$, where $\mathbf{g}$ is the observed image, $\hat{\mathbf{f}}$ is the computed approximation, and A is the coefficient matrix constructed according to the considered BCs: the latter measure is the sum of square errors and it represents, up to the scaling of the variance, the χ^2 statistical measure of the error. As already pointed out, the choice of the BCs is important mainly for low levels of noise, that is, for high values of SNR. Indeed, in the last row of these tables (SNR = 10), the errors due to noise start to dominate the restoration process and therefore the choice of particular BCs is not relevant for the restoration accuracy. In the other rows, where the noise is lower, the choice of the BCs becomes crucial. In particular, the AR-BCs improve substantially the quality of the restorations with respect to the other BCs. This is especially evident in Test II (see Table 3). The reason of the observed high improvement is due to the shape of the PSF, since, basically, the more the support of the PSF is large, the more the ringing effects (and hence the BCs) become dominating.

To emphasize the quality of the restored images, we consider the reconstruction in the case of Test I for a fixed SNR equal to 40. In Figure 5, we report the restored images and in Figure 6 the residuals of the computed solutions for each pixel divided by the variance of the noise. The last one should have a normal distribution in the case of a good restoration since we add a Gaussian noise. In Figure 5 is evident the reduction of the ringing effects passing from P-BCs to R-BCs (ringings such as the horizontal white line on the top and the horizontal black line on the bottom disappear) and from R-BCs to AR-BCs (ringings such as the two vertical white lines in the bottom left disappear). Indeed, in the same figure we note a higher level of detail in the case of AR-BCs, especially concerning the face of the cameramen. The image restored with R-BCs is smoother when compared with the one restored with AR-BCs, where we can see the "freckles" effect, typical of the L^2 norm restoration. Indeed the CGLS method computes the least-squares solution that is well known to be affected by such a phenomenon [39]. When passing to the R-BCs, the considered effect is less evident: indeed it seems that the slightly greater ringing

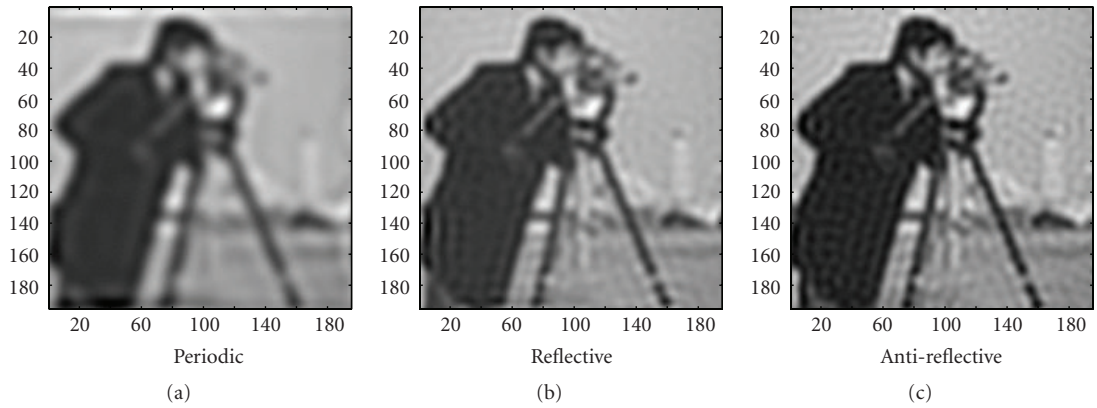


Figure 5: Restored images with SNR = 40.

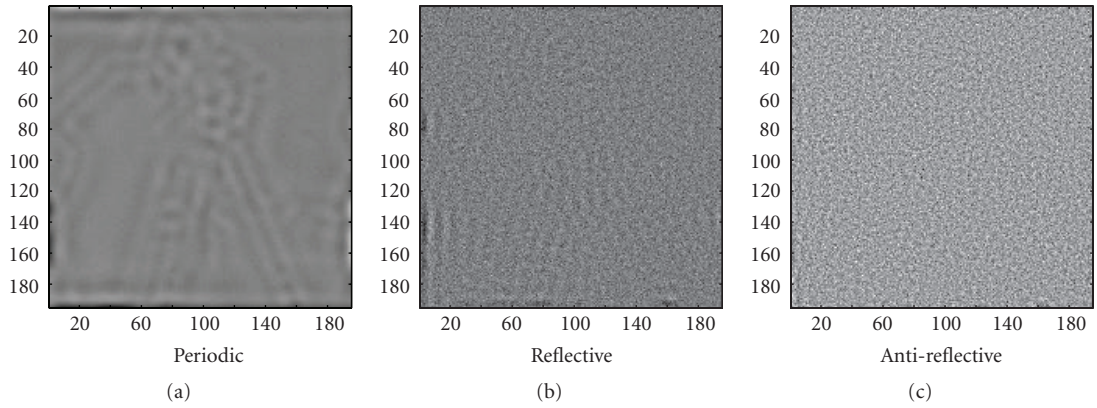


Figure 6: Residuals divided by the variance of the noise for the restored images with SNR = 40.

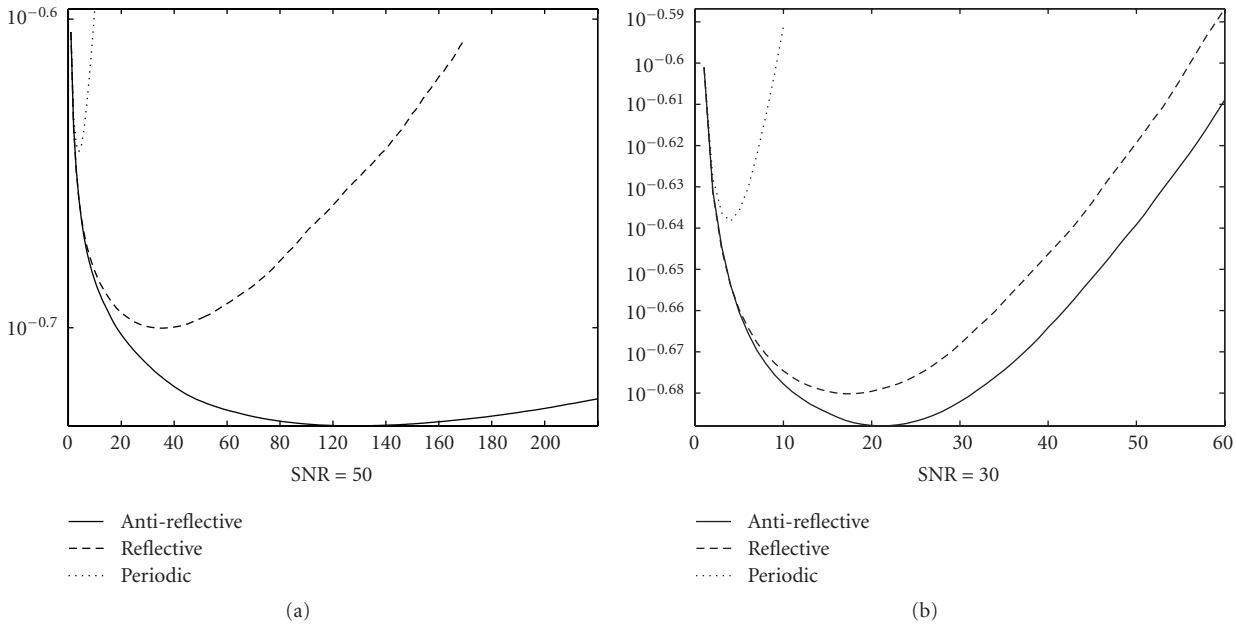


Figure 7: Test I: RREs at each CG iteration with different BCs: dotted line: Periodic, dashed line: Reflective, solid line: Antireflective.

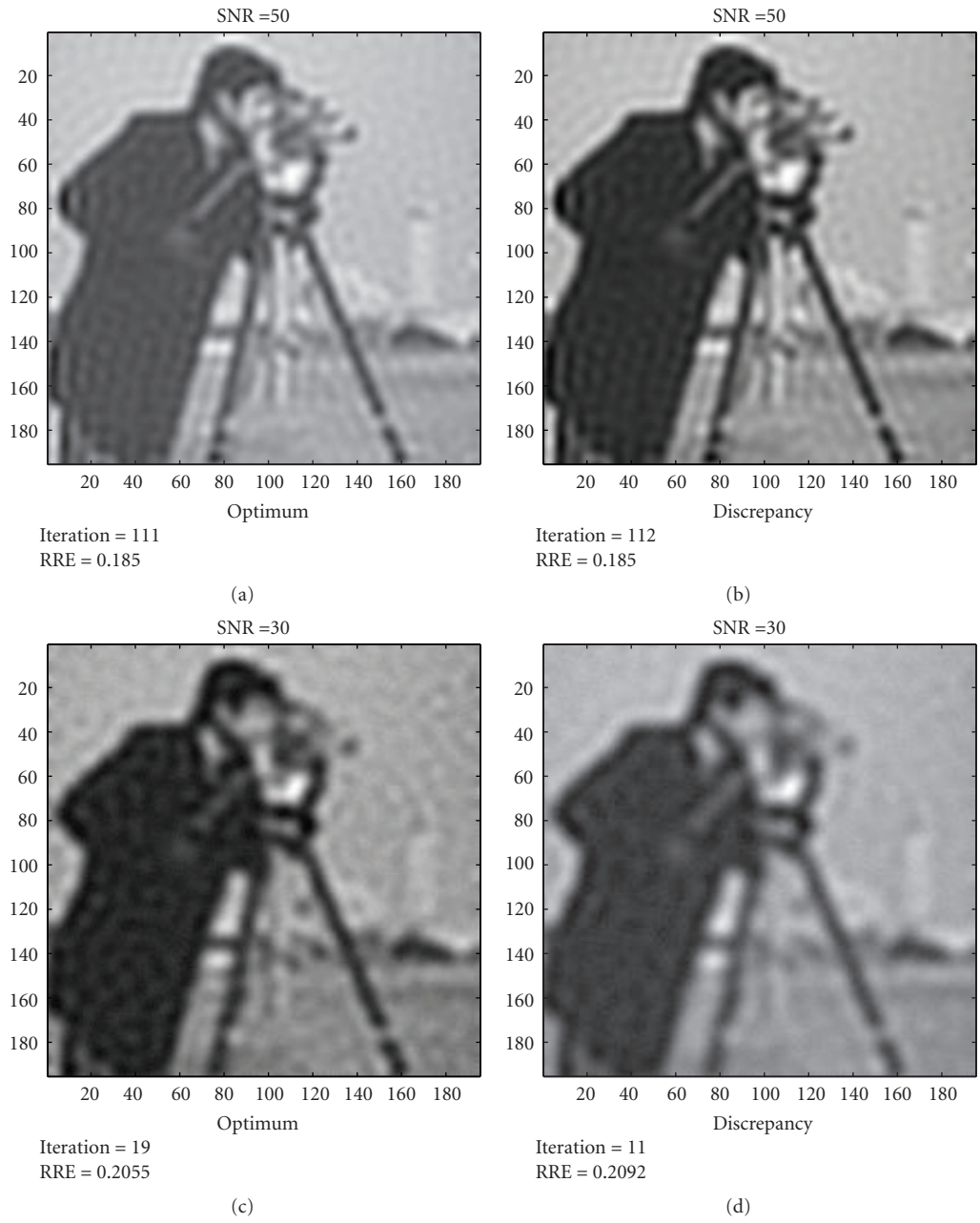


Figure 8: Discrepancy principle: images restored at the optimum value iteration and at which computed with the discrepancy principle.

effects smooth the image reducing the "freckles", but also reducing some details like, for example, the eye of the cameraman. The previous comments suggest that using AR-BCs in connection with regularization methods related to other norms, like the Total Variation [48], could lead to a reconstruction with sharper edges. In Figure 6, we observe a normal distribution of the scaled residuals only with the AR-BCs, while, also with the R-BCs, some further errors corresponding to the ringing effects emerge at the boundary and at the edges of the image: this means that the imposition of the R-BCs is not good enough as model at least for this example. On the other hand, with the AR-BCs, this kind of error seems to disappear and the scaled residual seems to follow a normal distribution. This confirms the goodness of the restoration obtained with the AR-BCs.

Two convergence histories, that is, the RREs at any iteration, are plotted in Figure 7 for two different values of the SNR. It should be stressed that the AR-BCs give the best results and the lowest level of RRE. Such behavior is again more evident considering Test II. Indeed, in such case the RREs with AR-BCs continue to decrease even after the first 200 iterations. On the other hand, the restorations with R-BCs start to deteriorate after the very first iterations. In addition, we notice that AR-BCs curves are in general quite flat. This is a very useful feature since the estimation of the optimal stopping value, which is well known to be a

crucial and difficult task, can be done with low precision. Indeed, in order to stress the applicability of the AR-BCs to real problems, we consider for the Test I the discrepancy principle widely used with iterative methods [31]. Since we know the L^2 norm of the error, we stop the CG when the L^2 norm of the residual becomes lower than the L^2 norm of the error. Such a criterion seems to work quite well for the AR-BCs as we can see in Figure 8. The restored images are good enough with respect to the optimal solution and also the stopping iteration, at least in this example, is close to the optimal one. On the other hand, such criterion is not always effective for the other BCs in this example. For instance, the stopping iteration for R-BCs is greater than 1000 in the case of SNR = 50 and it is 13 for SNR = 30. However, an analysis of the stopping criterion, in connection with AR-BCs, should be further investigated in the future.

Finally, we remark that also for Test II the CG applied to the linear system, (34) works without breakdown, both with R-BCs and AR-BCs. Therefore, it is possible that the applicability of the CG to (34) is a general property, which does not depend on the particular choice of the BCs.

5. Conclusions

In this contribution we have dealt with the use of antireflective BCs for deblurring problems where the considered issues have been: the precision of the reconstruction when the noise is not present, the linear algebra related to these BCs, the computational costs, and the reconstruction quality associated with iterative and noniterative regularizing solvers when the noise is considered. For many of the considered items, the antireflective algebra coming from the given BCs is the optimal choice: for instance in the work of La Spina it is proven that no one of the trigonometric algebras can be associated with BCs of the same precision as the antireflective. Numerical experiments corroborating the previous statements have been reported and discussed: in this direction it remains an open problem to understand why the CG works without any numerical problem even if the antireflective structure is non symmetric (an event not normal as emphasized by the Jordan decomposition).

Acknowledgment

The paper was partially supported by MIUR 2008, Grant number 20083KLJEZ.

References

[1] H. Andrews and B. Hunt, *Digital Image Restoration*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1977.

[2] R. H. Chan and M. Ng, "Conjugate gradient methods for Toeplitz systems," *SIAM Review*, vol. 38, no. 3, pp. 427–482, 1996.

[3] N. Kalouptsidis, G. Carayannis, and D. Manolakis, "Fast algorithms for block Toeplitz matrices with Toeplitz entries," *Signal Processing*, vol. 6, no. 1, pp. 77–81, 1984.

[4] S. Serra-Capizzano and E. Tyrtyshnikov, "Any circulant-like preconditioner for multilevel matrices is not superlinear," *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 2, pp. 431–439, 2000.

[5] S. Serra-Capizzano, "Matrix algebra preconditioners for multilevel Toeplitz matrices are not superlinear," *Linear Algebra and Its Applications*, vol. 343/344, pp. 303–319, 2002.

[6] D. Noutsos, S. Serra-Capizzano, and P. Vassalos, "Matrix algebra preconditioners for multilevel Toeplitz systems do not insure optimal convergence rate," *Theoretical Computer Science*, vol. 315, no. 2-3, pp. 557–579, 2004.

[7] O. Axelsson and G. Lindskog, "On the rate of convergence of the preconditioned conjugate gradient method," *Numerische Mathematik*, vol. 48, no. 5, pp. 499–523, 1986.

[8] G. Fiorentino and S. Serra-Capizzano, "Multigrid methods for symmetric positive definite block toeplitz matrices with nonnegative generating functions," *SIAM Journal of Scientific Computing*, vol. 17, no. 5, pp. 1068–1081, 1996.

[9] M. Donatelli, "A multigrid for image deblurring with Tikhonov regularization," *Numerical Linear Algebra with Applications*, vol. 12, no. 8, pp. 715–729, 2005.

[10] R. H. Chan, M. Donatelli, S. Serra-Capizzano, and C. Tablino-Possio, "Application of multigrid techniques to image restoration problems," in *Advanced Signal Processing Algorithms, Architectures, and Implementations*, F. Luk, Ed., vol. 4791 of *Proceedings of SPIE*, pp. 210–221, Seattle, Wash, USA, 2002.

[11] R. Gonzalez and R. Woods, *Digital Image Processing*, Addison-Wesley, Reading, Mass, USA, 1992.

[12] M. K. Ng, R. H. Chan, and W.-C. Tang, "A fast algorithm for deblurring models with Neumann boundary conditions," *SIAM Journal of Scientific Computing*, vol. 21, no. 3, pp. 851–866, 1999.

[13] G. Strang, "The discrete cosine transform," *SIAM Review*, vol. 41, no. 1, pp. 135–147, 1999.

[14] R. H. Chan, T. F. Chan, and C. Wong, "Cosine transform based preconditioners fortotal variation minimization problems in image processing," in *Proceedings of the 2nd IMACS International Symposium on Iterative Methods in Linear Algebra II*, vol. 3 of *IMACS Series in Computational and Applied Mathematics*, pp. 311–329, June 1995.

[15] K. Rao and P. Yip, *Discrete Cosine Transform: Algorithms, Advantages, Applications*, Academic Press, New York, NY, USA, 1990.

[16] R. Lagendijk and J. Biemond, *Iterative Identification and Restoration of Images*, Kluwer Academic, Dordrecht, The Netherlands, 1991.

[17] S. Serra-Capizzano, "A note on antireflective boundary conditions and fast deblurring models," *SIAM Journal of Scientific Computing*, vol. 25, no. 4, pp. 1307–1325, 2003.

[18] G. La Spina, "Algebre di matrici associate a trasformate trigonometriche ed il loro ruolo nel restauro di immagini digitali," M.S. thesis in Mathematics, Department of Mathematics, University of Pisa, Pisa, Italy, 2009.

[19] M. Christiansen and M. Hanke, "Deblurring methods using antireflective boundary conditions," *SIAM Journal on Scientific Computing*, vol. 30, no. 2, pp. 855–872, 2007.

[20] D. Fan and J. Nagy, "Synthetic boundary conditions for image deblurring," *Linear Algebra and Its Applications*. In press.

[21] M. Donatelli, "Fast transforms for high order boundary," *BIT*, vol. 50, no. 3, pp. 559–576, 2010.

[22] R. Liu and J. Jia, "Reducing boundary artifacts in image deconvolution," in *Proceedings of the IEEE International Conference on Image Processing (ICIP '08)*, pp. 505–508, October 2008.

[23] A. Meister, "Deconvolution from Fourier-oscillating error densities under decay and smoothness restrictions," *Inverse Problems*, vol. 24, no. 1, pp. 101–120, 2008.

[24] Y. Shi and Q. Chang, "Acceleration methods for image restoration problem with different boundary conditions," *Applied Numerical Mathematics*, vol. 58, no. 5, pp. 602–614, 2008.

[25] V. S. Sizikov, M. V. Rimskikh, and R. K. Mirdzhamolov, "Reconstructing blurred noisy images without using boundary conditions," *Journal of Optical Technology*, vol. 76, no. 5, pp. 279–285, 2009.

[26] D.-H. Xu, R. Xu, and S. Wang, "Accuate numerical method for total variation-based image deblurring," in *Proceedings of the 7th International Conference on Machine Learning and Cybernetics (ICMLC '08)*, pp. 2765–2769, Kunming, China, July 2008.

[27] D. Bini and M. Capovani, "Spectral and computational properties of band symmetric toeplitz matrices," *Linear Algebra and Its Applications*, vol. 52-53, pp. 99–125, 1983.

[28] A. Aricò, M. Donatelli, and S. Serra-Capizzano, "Spectral analysis of the anti-reflective algebra," *Linear Algebra and Its Applications*, vol. 428, no. 2-3, pp. 657–675, 2008.

[29] R. Horn and C. Johnson, *Matrix Analysis*, Cambridge University Press, Cambridge,UK, 1999.

[30] M. Donatelli, C. Estatico, J. Nagy, L. Perrone, and S. Serra-Capizzano, "Anti-reflective boundary conditions and fast 2D deblurring models," in *Advanced Signal Processing Algorithms, Architectures, and Implementations*, Proceedings of SPIE, pp. 380–389, San Diego, Calif, USA, August 2003.

[31] H. Engl, M. Hanke, and A. Neubauer, *Regularization of Inverse Problems*, Kluwer Academic, Dordrecht, The Netherlands, 1996.

[32] M. Bertero and P. Boccacci, *Introduction to Inverse Problems in Imaging*, Institute of Physics Publishing, London, UK, 1988.

[33] M. Donatelli and S. Serra-Capizzano, "Anti-reflective boundary conditions and re-blurring," *Inverse Problems*, vol. 21, no. 1, pp. 169–182, 2005.

[34] G. Golub and C. Van Loan, *Matrix Computation*, The Johns Hopkins University Press, Baltimore, Md, USA, 1983.

[35] A. Aricò, M. Donatelli, and S. Serra-Capizzano, "The anti-reflective algebra: structural and computational analysis with application to image deblurring and denoising," *Calcolo*, vol. 45, no. 3, pp. 149–175, 2008.

[36] E. O. Brigham, *The Fast Fourier Transform and Applications*, Prentice Hall, Englewood Cliffs, NJ, USA, 1988.

[37] J. Nagy, K. Palmer, and L. Perrone, "Restore tools: an object oriented Matlab packagefor image restoration," 2002, http://www.mathcs.emory.edu/~nagy/RestoreTools/.

[38] P. C. Hansen, *Rank-Deficient and Discrete Ill-Posed Problems*, SIAM, Philadelphia, Pa, USA, 1997.

[39] P. C. Hansen, J. Nagy, and D. P. O'Leary, *Deblurring Images Matrices, Spectra and Filtering*, SIAM, Philadelphia, Pa, USA, 2006.

[40] H. Engl, M. Hanke, and A. Neubauer, *Regularization of Inverse Problems*, Kluwer Academic, Dordrecht, The Netherlands, 2000.

[41] C. R. Vogel, *Computational Methods for Inverse Problems*, SIAM, Philadelphia, Pa, USA, 2002.

[42] C. Tablino-Possio, "Truncated decompositions and filtering methods with reflective/anti-reflective boundary conditions: a comparison," in *Matrix Methods: Theory, Algorithms, Applications*, V. Olshevsky and E. Tyrtyshnikov, Eds., pp. 382–408, World Scientific, Singapore, 2010.

[43] M. Donatelli and M. Hanke, "On the condition number of the antireflective transform," *Linear Algebra and Its Applications*, vol. 432, no. 7, pp. 1772–1784, 2010.

[44] M. Donatelli and S. Serra-Capizzano, "On the regularizing power of multigrid-type algorithms," *SIAM Journal of Scientific Computing*, vol. 27, no. 6, pp. 2053–2076, 2006.

[45] S. Morigi, L. Reichel, F. Sgallari, and A. Shyshkov, "Cascadic multiresolution methods for image deblurring," *SIAM Journal on Imaging Sciences*, vol. 1, no. 1, pp. 51–74, 2008.

[46] M. Donatelli, C. Estatico, A. Martinelli, and S. Serra-Capizzano, "Improved image deblurring with anti-reflective boundary conditions and re-blurring," *Inverse Problems*, vol. 22, no. 6, pp. 2035–2053, 2006.

[47] J. G. Nagy, K. Palmer, and L. Perrone, "Iterative methods for image deblurring: a Matlab object-oriented approach," *Numerical Algorithms*, vol. 36, no. 1, pp. 73–93, 2004.

[48] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D*, vol. 60, pp. 259–268, 1992.

Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2010, Article ID 930218, 16 pages
doi:10.1155/2010/930218

Research Article

Smoothing and Regularization with Modified Sparse Approximate Inverses

T. Huckle and M. Sedlacek

Technische Universität München, Boltzmannstraße 3, 80748 Garching, Germany

Correspondence should be addressed to T. Huckle, huckle@in.tum.de

Received 20 September 2010; Accepted 22 September 2010

Academic Editor: Owe Axelsson

Sparse approximate inverses M which satisfy $\min_M \|AM - I\|_F$ have shown to be an attractive alternative to classical smoothers like Jacobi or Gauss-Seidel (Tang and Wan; 2000). The static and dynamic computation of a SAI and a SPAI (Grote and Huckle; 1997), respectively, comes along with advantages like inherent parallelism and robustness with equal smoothing properties (Bröker et al.; 2001). Here, we are interested in developing preconditioners that can incorporate probing conditions for improving the approximation relative to high- or low-frequency subspaces. We present analytically derived optimal smoothers for the discretization of the constant-coefficient Laplace operator. On this basis, we introduce probing conditions in the generalized Modified SPAI (MSPAI) approach (Huckle and Kallischko; 2007) which yields efficient smoothers for multigrid. In the second part, we transfer our approach to the domain of ill-posed problems to recover original information from blurred signals. Using the probing facility of MSPAI, we impose the preconditioner to act as approximately zero on the noise subspace. In combination with an iterative regularization method, it thus becomes possible to reconstruct the original information more accurately in many cases. A variety of numerical results demonstrate the usefulness of this approach.

1. Introduction

For applying an iterative solution method to an ill-conditioned system of linear equations $Ax = b$ with sparse matrix $A \in \mathbb{R}^{n\times n}$, $x, b \in \mathbb{R}^n$, it is often crucial to include an efficient preconditioner. Here, the original problem $Ax = b$ is replaced by the preconditioned system $MAx = Mb$ or $Ax = A(My) = b$. Often, used preconditioners as Jacobi, Gauss-Seidel, or Incomplete LU (ILU) decomposition are of unsatisfactory quality or strongly sequential. In a parallel environment, both the computation of the preconditioner M as well as the application of the preconditioner on any given vector v should be efficient. Furthermore, an iterative solver applied on $AMy = b$ or $MAx = Mb$ should converge much faster than for $Ax = b$ (e.g., it holds $\mathrm{cond}(MA) \ll \mathrm{cond}(A)$).

The first two conditions can be easily satisfied by using a sparse matrix M as approximation to A^{-1}. Note that the inverse of a sparse system A is nearly dense, but in many cases, the entries of A^{-1} are rapidly decaying, so most of the entries are very small (see Demko et al. [1]).

Sparse approximate inverses can be computed by minimizing $AM - I$ in the Frobenius norm, where I denotes the identity matrix. In this Frobenius norm minimization, we can include further approximation conditions, described by the Modified SPAI (MSPAI) [2, 3] method. This additional feature allows us to control the approximation property of the preconditioner. So, by means of probing vectors, we can choose subspaces for which the preconditioner satisfies certain conditions. This is especially important for iterative solution methods that differ between high-frequency and low-frequency components, for example, smoothing in multigrid or regularization techniques based on iterative solvers.

The outline of the paper is the following. In Section 1, we will give a survey of SPAI and MSPAI and a short description of multigrid methods and iterative solvers for regularization problems. In Section 2, we show that for multigrid methods the smoothing property can be greatly improved by using MSPAI in comparison to SAI or SPAI smoothers. With a different subspace approach, we focus on

the reconstruction of signals associated to ill-posed problems in Section 3. We present numerical results throughout the paper to demonstrate the impact of our approach at the corresponding parts. A conclusion with a short outlook closes the discussion.

Benson and Frederickson [4] were the first to propose the computation of an explicit approximation $M \approx A^{-1}$ to the inverse of a system matrix $A \in \mathbb{R}^{n\times n}$. For an a priori prescribed sparsity pattern $\mathcal{P}$, this can be done in a static way by solving

$$\min_{\mathcal{P}(M)=\mathcal{P}} \|AM - I\|_F^2 = \sum_{k=1}^{n} \min_{\mathcal{P}(m_k)=\mathcal{P}_k} \|Am_k - e_k\|_2^2, \quad (1)$$

with m_k the kth column of the preconditioner M and e_k the kth column of the identity matrix I. $\mathcal{P}(A)$ denotes the pattern $\mathcal{P}$ of A and $\mathcal{P}_k$ the pattern of the kth column of $\mathcal{P}$. This well-known approach of providing a SAI preconditioner, naturally leads to inherent parallelism which is one of its main advantages. Each of the n small least squares problems regarding one column can be computed independently of one another.

Using both the index set $\mathcal{J}_k$, which is implicitly given by $\mathcal{P}_k$ and contains the indices j such that $m_k(j) \neq 0$, and its corresponding so-called shadow $\mathcal{I}_k$, that is the indices of nonzero rows in $A(:, \mathcal{J}_k)$, each subproblem in (1) is related to a small matrix $\widehat{A}_k := A(\mathcal{I}_k, \mathcal{J}_k)$ if $\mathcal{P}_k$ is sparse. We refer to the reduced sparse column vectors as $\widehat{m}_k := m_k(\mathcal{J}_k)$ and $\widehat{e}_k := e_k(\mathcal{I}_k)$, respectively. The solution of every reduced problem $\min_{\widehat{m}_k} \|\widehat{A}\widehat{m}_k - \widehat{e}_k\|_2$ can be obtained, for example, by using QR decomposition using Householder or the modified Gram-Schmidt algorithm.

1.1. The SPAI Algorithm. The SPAI algorithm is an additional feature in this Frobenius norm minimization that introduces different strategies for choosing new profitable indices in m_k to improve on an already computed approximation. We assume that by solving (1) for a given index set $\mathcal{J}_k$, we already have determined an optimal solution $m_k(\mathcal{J}_k)$ inducing the sparse vector m_k with residual r_k. Dynamically, we want to augment new entries in m_k and solve (1) for this enlarged index set $\widetilde{\mathcal{J}}_k$ such that we derive a reduction in the norm of the new residual $\widetilde{r}_k = A(\widetilde{\mathcal{I}}_k, \widetilde{\mathcal{J}}_k)m_k(\widetilde{\mathcal{J}}_k) - e_k(\widetilde{\mathcal{I}}_k)$.

Following Cosgrove et al. [5] and Grote and Huckle [6], in SPAI, we test one possible new index $j \in \mathcal{J}_{\text{new}}$ out of a given set of possible new indices $\mathcal{J}_{\text{new}}$ to improve m_k. Therefore, we can consider the reduced 1D minimization problem

$$\min_{\lambda_j} \left\| A\left(m_k + \lambda_j e_j\right) - e_k \right\|_2 = \min_{\lambda_j} \left\| \lambda_j A_j + r_k \right\|_2. \quad (2)$$

The solution of (2) is given by

$$\lambda_j = -\frac{r_k^T A e_j}{\left\| Ae_j \right\|_2^2}, \quad (3)$$

and leads to an improved squared residual norm

$$\rho_j^2 = \|r_k\|^2 - \frac{\left(r_k^T A e_j\right)^2}{\left\| Ae_j \right\|_2^2}. \quad (4)$$

For improving m_k, we only have to consider indices j in rows of A that are related to nonzero entries in the old residual r_k; otherwise, they do not lead to a reduction of the residual norm. Thus, we have to determine those column indices j, which satisfy $r_k^T A e_j \neq 0$. Let us denote the index set of nonzero entries in r_k by $\mathcal{L}$. By $\widetilde{\mathcal{J}}_i$, we denote the set of new indices that are related to the nonzero elements in the ith row of A, and by $\mathcal{J}_{\text{new}} := \bigcup_{i\in\mathcal{L}} \widetilde{\mathcal{J}}_i$ the set of all possible new indices that will lead to a reduction of the residual norm. The one or more newly added indices $\mathcal{J}_c$ are chosen to be a subset of $\mathcal{J}_{\text{new}}$ that corresponds to the maximal reduction in r_k. For the enlarged index set $\mathcal{J}_k \cup \mathcal{J}_c$, we have to update the QR decomposition of the related least squares submatrix and solve the new column m_k.

It is possible to influence the sparsity and approximation quality during the computation of the SPAI indirectly by different parameters, for example, how many entries are to be added in one step, how many pattern updates are to be done, which residual norm should be reached, or which initial pattern $\mathcal{P}$ is to be used. Note that SPAI can also be applied on dense systems to compute a sparse preconditioner.

1.2. Modified SPAI. Holland et al. [7] have generalized the SPAI ansatz allowing a sparse target matrix B on the right hand side in the form $\min_{\mathcal{P}(M)=\mathcal{P}} \|AM - B\|_F$. This approach is useful in connection with some kind of two-level preconditioning: first, compute a standard sparse preconditioner B for A, and then improve this preconditioner by an additional Frobenius norm minimization with target B. From an algorithmic point of view, the minimization with sparse target matrix B, instead of I, introduces no additional difficulties. Simply, $\mathcal{P}(M)$ should be chosen more carefully with respect to A and B.

In [2], we combine this approach with classical probing techniques [8–10], which are, for example, applied to preconditioning Schur complements. In contrast to classical probing, our basic formulation

$$\min_{\mathcal{P}(M)=\mathcal{P}} \|CM - B\|_F = \min_{\mathcal{P}(M)=\mathcal{P}} \left\| \begin{pmatrix} C_0 \\ \rho e^T \end{pmatrix} M - \begin{pmatrix} B_0 \\ \rho f^T \end{pmatrix} \right\|_F, \quad (5)$$

with sparse matrices C_0 and B_0, is not restricted to special probing subspaces as it allows any choice of e and f. The resulting preconditioner M satisfies both $C_0M \approx B_0$ and $e^TM \approx f^T$. We refer to the first n rows of (5), that is, $C_0M - B_0$, as *full approximation part* and to the additional rows as *probing part*. The weight $\rho \geq 0$ enables us to control how much emphasis is put on the *probing constraints*, and the matrices $e, f \in \mathbb{R}^{n\times k}$ represent the k-dimensional subspace on which the preconditioner should be optimal. Choosing $\rho = 0$, $C_0 = A$, and $B_0 = I$ in (5) leads to the classical SPAI formulation. Setting $C_0 = I$ and $B_0 = A$, we end up with

a formulation computing explicit sparse approximations to A. In this case, the derived approximation on A can have a considerably fewer number of nonzeros (nnz) than A but choosing $f^T = e^T A$, the preconditioner M will have a similar action on e^T as A.

Furthermore, it is also possible to include individual probing conditions for each column m_k. As a new approach, we use probing masks defined by sparse row vectors s_k, $k \in \{1, \ldots, n\}$, containing the same pattern as m_k, that is, $\mathcal{P}(s_k) = \mathcal{P}(m_k)$, and add the condition $\min_{\hat{m}_k} |\hat{s}_k \hat{m}_k - f_k|$ to the Frobenius norm minimization of the kth column. Corresponding to $\hat{m}_k, \hat{s}_k$ denotes the reduced form of s_k. The masks for each column of M can be stored in a sparse rectangular matrix S, whereby the individual probing conditions result from $\operatorname{diag}(SM) \approx f$. Compared to MSPAI using global probing vectors e, f, this approach gives considerably more freedom for choosing probing conditions individually for each column of the preconditioner. Note that, for example, for a tridiagonal pattern, the mask $\hat{s}_k = (1, 0, -1)$ can be used to enforce a quasisymmetry in the column vector m_k such that $m_{k-1,k} \approx m_{k+1,k}$.

The field of applications using MSPAI is versatile: we can improve preconditioners resulting from ILU, IC, FSAI, FSPAI, or AINV (see [11] for an overview) by adding probing information. We also overcome the main drawbacks of MILU and classical probing such as the restriction to certain vectors like $(1, 1, \ldots, 1)^T$ as probing subspace and the rather difficult efficient implementation on parallel computers. The numerical examples in [2, 3] demonstrate MSPAI's effectiveness for preconditioning various PDE matrices and preconditioning Schur complements arising from domain decompositions.

1.3. Multigrid. The crucial observation leading to multigrid (MG) methods is the following: applying a stationary iterative solver like Gauss-Seidel iteration gives a satisfactory reduction of the error on the subspace related to high-frequency components. Therefore, Gauss-Seidel iteration is considered as a smoother in a first step of the MG algorithm. The error mainly contains smooth components and can be projected to a smaller linear system. This reduced system can be tackled recursively by the same approach based on smoothing steps and projection on an again reduced system. On the coarsest level, the small linear system can be solved explicitly. Afterwards, the coarse solutions have to be prolongated step by step back to the finer levels including postsmoothing steps.

Here, we are mainly interested in the smoother. To derive a convergent method the smoother has to reduce the error in the high-frequency subspace. For a given matrix A and an approximate inverse smoother M, the iteration k is described by $x^{(k+1)} = x^{(k)} + M(b - Ax^{(k)})$, and thus the error is given by $I - MA$. As the eigenvalues and eigenvectors are analytically well-known for a discretization of the constant coefficient Laplace operator, it is possible to fully discuss the convergence behavior of MG in this special case. For analyzing the smoothing property, the eigenvalues are separated into high- and low-frequency eigenvectors. The projection $P = U^T(I - MA)U$ on the high-frequency subspace U gives the smoothing factor defined as the spectral radius of P. In the constant coefficient case, the smoothing factor can also be described as $1 - ma$, where a and m are generating functions representing A and M [12, 13]. Note that the technique of generating functions is similar to Local Fourier Analysis (LFA) used in the standard multigrid literature [14]. In 2D the functions are defined in $x, y \in [0, \pi]$, where the high-frequency domain is given by the difference G between the two squares $[0, \pi]^2$ and $[0, \pi/2]^2$ with corners at

$$\left(0, \frac{\pi}{2}\right), \quad \left(\frac{\pi}{2}, \frac{\pi}{2}\right), \quad \left(\frac{\pi}{2}, 0\right), \quad (\pi, 0), \quad (\pi, \pi), \quad (0, \pi). \tag{6}$$

Thus, the smoothing factor is given by

$$\mu := \max_{x,y \in G} \{ \, | \, 1 - m(x, y)a(x, y) \, | \, \}. \tag{7}$$

In this paper, we want to use MSPAI as a smoother. SAI was already considered by Tang and Wan in [15] and SPAI by Bröker et al. in [16].

1.4. Iterative Regularization. For ill-posed problems, as they arise in image restoration, regularization techniques are important in order to recover the original information. Let us consider the model problem

$$x \xrightarrow{\text{blur}} Hx \xrightarrow{\text{noise}} Hx + \eta = b, \tag{8}$$

where x is the original image, H is the blur operator, η is a vector representing the noise, and b is the observed image. We want to recover x as good as possible and as fast as possible. Because H may be extremely ill-conditioned or even singular, and, because of the presence of noise, (8) cannot be solved directly. Consequently, to solve $Hx = b$ on the signal subspace, a regularization technique has to be applied. One of the classical methods is the Tikhonov regularization [17] which solves

$$\min_x \left\{ \|Hx - b\|_2^2 + \gamma \|b\|_2^2 \right\} \Longleftrightarrow \left(H^T H + \gamma I\right) x = H^T b, \tag{9}$$

instead of (8) for a fixed regularization parameter $\gamma \geq 0$.

Another regularization method is based on an iterative solver such as the Conjugate Gradient (CG) method [8, 18] for spd matrices or CG on the normal equations in the general unsymmetric case. The usual observation which coincides with the CG convergence analysis is that in the first iterations, the error is reduced relative to large eigenvalues. In later steps, the eigenspectrum related to noise and small eigenvalues dominates the evolution of the approximate solution. Therefore, the restoration has to stop after a few iterations before the method starts to reduce the error relative to the noise space.

Preconditioning usually should accelerate the convergence without destroying the quality of the reconstruction [19–21]. Structured preconditioners like Toeplitz or circulant matrices are considered typically whenever spatially

invariant blur operators are treated. For general H, a preconditioner like ILU will lead to faster convergence, but the quality of the reconstruction will deteriorate, because the preconditioner also improves the solution relative to the unwanted noise subspace. Therefore, it is even more demanding to develop preconditioners for general H, for example, for spatially variant blur.

The application of the preconditioner can have three positive effects:

(i) reduce the necessary number of iterations,

(ii) result in a better reconstruction of the original vector, and

(iii) result in a flat convergence curve such that it is easier to find the best reconstruction.

In general, we have to expect that not all three conditions can be reached simultaneously. Therefore, we have to present different preconditioners depending on the application.

Using preconditioners within iterative methods to restore original data has been successfully proposed by Nagy, for example, in [22, 23], mostly in connection with nearly structured problems. Furthermore, for the analysis and solution of discrete ill-posed systems there are several MATLAB packages such as *Regularization Tools* developed by Hansen [24] and *RestoreTools* developed by Nagy et al. [25].

2. MSPAI Smoothing

2.1. A 1D Model Problem. To derive approximate inverse smoothers, we consider the standard 1D discretized Laplace operator with constant coefficients which is of the form $A_1 x = b$, with $A_1 := \text{tridiag}(-1/2,\ 1,\ -1/2)$. The matrix A_1 is related to the generating function or symbol

$$a_1(x) = 1 - \frac{e^{ix} + e^{-ix}}{2} = 1 - \cos(x). \tag{10}$$

The high-frequency part of A_1 is represented by (10) for $x \in [\pi/2, \pi] =: I_1$. Hence, the optimal smoothing parameter ω in the Jacobi smoother $1 - \omega a_1(x)$ is found by solving the problem

$$\min_{\omega} \max_{x \in I_1} |1 - \omega a_1(x)| = \min_{\omega} \max_{x \in I_1} |s(x)| \overset{\omega = 2/3}{=} \frac{1}{3}. \tag{11}$$

The solution can be found by replacing (11) with the maximum over the two boundary values $\min_\omega \max\{|s(\pi/2)|, |s(\pi)|\}$.

2.1.1. Analytical Derivation of the Optimal Smoother. As approximate inverse smoother, we choose a trigonometric polynomial of the same degree

$$m_1(x) = a + 2b\cos(x), \tag{12}$$

and a tridiagonal Toeplitz matrix $M = \text{tridiag}(b, a, b)$, respectively. The smoothing condition for (12) can be written as

$$\min_{a,b} \max_{x \in I_1} |1 - m_1(x)a_1(x)| = \min_{a,b} \max_{x \in I_1} |s(x)|, \tag{13}$$

or using the boundary values of I_1 as

$$\begin{aligned} \min_{a,b} \max \Big\{ &\Big| 1 - m_1\Big(\frac{\pi}{2}\Big) a_1\Big(\frac{\pi}{2}\Big) \Big|, |1 - m_1(\pi)a_1(\pi)|, \\ &|1 - m_1(u)a_1(u)| \Big\} \\ = \min_{a,b} \max &\{|a-1|, |2(a-2b)-1|, |1-m_1(u)a_1(u)|\}, \end{aligned} \tag{14}$$

in which $u \in I_1$ is the local extreme point of the function $1 - m_1(x)a_1(x)$ with derivative equal to zero. We thus obtain for $u = 1/2 - a/4b$ the quadratic condition

$$s(u) = \frac{(a+2b)^2}{8b} - 1, \tag{15}$$

and the overall solution $(a, b) = (16/17, 4/17)$ which leads to

$$M3_{\text{opt}} := \text{tridiag}\Big(\frac{4}{17}, \frac{16}{17}, \frac{4}{17}\Big). \tag{16}$$

The related smoothing factor for this optimal preconditioner $m(x)$ is $1/17 = 0.0588$, which is significantly smaller than the optimal smoothing factor of $1/3 = 0.3\overline{33}$ related to $\omega = 2/3$ for the Jacobi smoother.

2.1.2. Individual Probing Masks. The minimization conditions (14) on a and b can be also seen as conditions for the entries of the matrix directly. A column k of M is described by the three values $(m_{k-1,k}, m_{k,k}, m_{k+1,k})^T =: \hat{m}_k$, where the main diagonal entry is related to a and the upper and lower entries to b. Therefore, we can translate the two linear minimization conditions of (14) into individual probing masks $\hat{s}_k$ on entries of M directly

$$\begin{aligned} S3_{L1} &: \min_{\hat{m}_k} \big| (0,1,0)\hat{m}_k - 1 \big|, \\ S3_{L2} &: \min_{\hat{m}_k} \Big| (-1,1,-1)\hat{m}_k - \frac{1}{2} \Big|. \end{aligned} \tag{17}$$

Notice that we refer to individual probing conditions as local probing conditions. There are several possibilities to eliminate and replace the quadratic condition (15) by linear conditions that can be related to masks itself. In one possible method, we assume that the linear conditions of (14) are satisfied exactly. Inserting $a = 1$ into the second condition yields $b = 1/4$. We replace the denominator of the quadratic condition with this value and get the new linear condition

$$\begin{aligned} &\min_{a,b} \left| \frac{(a+2b)^2}{8 \cdot 1/4} - 1 \right| \\ &\quad = \min_{a,b} \Big| (a+2b)^2 - 2 \Big| \longrightarrow \min_{a,b} \Big| a + 2b - \sqrt{2} \Big|, \end{aligned} \tag{18}$$

which is related to the probing condition with isotropic probing mask

$$S3_{L3} : \min_{\hat{m}_k} \Big| (1,1,1)\hat{m}_k - \sqrt{2} \Big|. \tag{19}$$

Table 1: Smoothing factors by using $M3_{\text{opt}}$ (16), SPAI, and MSPAI with local probing conditions for the constant coefficient system A_1 of order $n = 10^3$.

$M3_{\text{opt}}$	SPAI	Weight	Local probing conditions	
		ρ	$\rho S3_{L1} \wedge 0.7S3_{L2}$	$\rho S3_{L3}$
0.0588	0.250	1.0	0.134	0.083
		2.0	0.107	0.077
		10^1	0.095	0.075
		10^2		

We call a mask $\hat{s}_k$ to be isotropic if the minimization $\min_{\hat{m}_k} |\hat{s}_k \hat{m}_k - f_k|$ can be described by one probing vector $e \in \mathbb{R}^n$ with $e(\mathcal{J}_k) = \alpha \hat{s}_k^T$, $\alpha \in \mathbb{R}$ and $k \in \{1, \ldots, n\}$, in the form $\min_{\mathcal{P}(M)=\mathcal{P}} \|e^T M - f^T\|_2$. Here, for instance, $\hat{s}_k = (-1, 1, -1)$ of $S3_{L2}$ is related to $e = (1, -1, 1, -1, \ldots)^T$ and $\hat{s}_k = (1, 1, 1)$ of $S3_{L3}$ to $e = (1, 1, \ldots, 1)^T$.

In our following numerical experiments, we are interested in a comparison between MSPAI with probing conditions, the optimal smoother $M3_{\text{opt}}$ (16), and the tridiagonal preconditioner given by SPAI. The SPAI matrix is derived by the minimization over the Frobenius norm in which the reduced Least Squares problem for a typical column of M is given by

$$\min_{\hat{m}_k} \left\| \begin{pmatrix} -0.5 & 0 & 0 \\ 1 & -0.5 & 0 \\ -0.5 & 1 & -0.5 \\ 0 & -0.5 & 1 \\ 0 & 0 & -0.5 \end{pmatrix} \hat{m}_k - \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \right\|_2. \tag{20}$$

The inner columns of the solution of (20) have Toeplitz structure described by the matrix tridiag $(2/5, 6/5, 2/5)$ related to the generating function $m(x) = 6/5 + (4/5)\cos(x)$. Preconditioning with the SPAI matrix results in the smoothing factor 0.250. Indeed, this is a degradation compared to the smoothing factor 0.0588 for the optimal-derived smoother $M3_{\text{opt}}$ but still an improvement compared to Jacobi. Note that MSPAI minimizes the 2-norm of a combination of conditions while $M3_{\text{opt}}$ is derived by minimizing the 1-norm. Therefore, to derive efficient smoothing factors by MSPAI, it is important to find a good combination and weighting of probing conditions.

Let us consider a numerical experiment for the matrix A_1 of order $n = 10^3$. Using an approximate inverse preconditioner satisfying the individual probing conditions, we achieve highly reduced smoothing factors in comparison to SPAI and Jacobi. Table 1 shows that depending on the probing mask and weight ρ, reductions down to $\mu = 0.075$ are possible for $S3_{L3}$ weighed with $\rho = 10^2$. Moreover, the choice of the subspace weight is stable; that is, increasing values lead to a saturation of the achievable smoothing factors.

2.1.3. Global Probing Vectors. Considering the conditions $S3_{L2}$ and $S3_{L3}$ with isotropic masks it is possible to derive conditions with low- and high-frequency global probing vectors $e_S := (1, 1, \ldots, 1)^T$ and $e_{N1} := (1, -1, 1, -1, \ldots)^T$ for the generalized Frobenius norm minimization (5) of MSPAI

$$\min_{\mathcal{P}(M)=\mathcal{P}} \left\| e_S^T M - \frac{1}{2} e_S^T \right\|_2, \qquad \min_{\mathcal{P}(M)=\mathcal{P}} \left\| e_{N1}^T M - \sqrt{2} e_{N1}^T \right\|_2, \tag{21}$$

respectively. In the following, we use a different approach and derive global probing conditions with respect to the optimal derived preconditioner $M3_{\text{opt}}$ (16). This can also be applied to general probing vectors like $e_{N2} := (1, 0, -1, 0, 1, \ldots)^T$ and $e_{N3} := (0, 1, 0, -1, 0, 1, \ldots)^T$, representing additional high-frequency subspaces. For a given probing vector e, we observe that $e^T M3_{\text{opt}} = \alpha e^T$ with $\alpha \in \mathbb{R}$. As we want to find the smoother M, based on MSPAI with the same action on e^T, we define probing conditions $e^T M \approx \alpha e^T$ and obtain

$$\begin{aligned} S3_{G1} &: \min_{\mathcal{P}(M)=\mathcal{P}} \left\| e_S^T M - \frac{24}{17} e_S^T \right\|_2, \\ S3_{G2} &: \min_{\mathcal{P}(M)=\mathcal{P}} \left\| e_{N1}^T M - \frac{8}{17} e_{N1}^T \right\|_2, \\ S3_{G3} &: \min_{\mathcal{P}(M)=\mathcal{P}} \left\| e_{N2}^T M - \frac{16}{17} e_{N2}^T \right\|_2, \\ S3_{G4} &: \min_{\mathcal{P}(M)=\mathcal{P}} \left\| e_{N3}^T M - \frac{16}{17} e_{N3}^T \right\|_2. \end{aligned} \tag{22}$$

Figure 1(a) shows the impact of using these global probing conditions for matrix A_1. Inducing MSPAI to satisfy $S3_{G1}$ leads to a strong reduction of the smoothing factor in comparison to SPAI, close to the optimal value of $M3_{\text{opt}}$. A combination of global conditions can lead to an efficient smoother as well. Again the smoothing factor stays stable for increasing values of ρ.

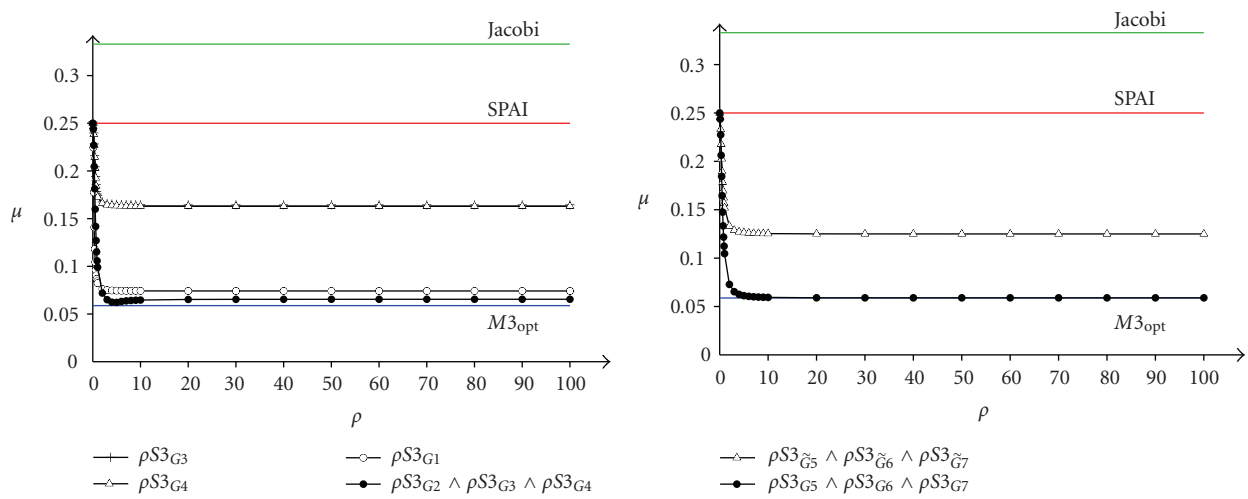
In a similar way, it is possible to derive global probing conditions with action on A, that is, $\min_{\mathcal{P}(M)=\mathcal{P}} \|e^T AM - \alpha e^T\|_2$. For probing vectors, for example, e_{N1}, e_{N2}, and e_{N3}, the approximation $e^T A \approx \text{const} \cdot e^T$ holds, and therefore $e^T AM \approx \text{const} \cdot e^T M$ up to some boundary perturbations. Note that for an approximate inverse preconditioner which should nearly be exact on e the condition $e^T AM \approx e^T$ should be satisfied.

As a generalization, we are interested in the impact of using global probing conditions when considering the 1D tridiagonal matrix B_1 with varying coefficients and kth row defined by

$$\begin{aligned} (B_1)_{k,:} &:= (0, \ldots, 0, -b_{k-1}, b_{k-1} + b_k, -b_k, 0, \ldots, 0), \\ &\text{with } b := \left(-\frac{1}{2} - \frac{j}{2n} \right)_{j=0,\ldots,n}. \end{aligned} \tag{23}$$

We denote the exact conditions $\min_{\mathcal{P}(M)=\mathcal{P}} \|e^T B_1 M - (16/17)e^T\|_2$ for the probing vectors e_{N1}, e_{N2}, and e_{N3} as $S3_{G5}$, $S3_{G6}$, and $S3_{G7}$. The approximate conditions for $\alpha = 1$ are indicated by $S3_{\tilde{G}5}$, $S3_{\tilde{G}6}$, and $S3_{\tilde{G}7}$.

Figure 1(b) shows that both the exact and approximate probing on the high-frequency part of the system lead to significant improvement compared to SPAI and Jacobi, similar to the 1D problem A_1.



(a) MSPAI with global probing conditions for A_1

(b) MSPAI using global probing vectors with action on B_1

FIGURE 1: Smoothing factor μ against subspace weight ρ. (a) considers MSPAI using global probing conditions for the constant coefficient system A_1 of size $n = 10^3$. (b) shows MSPAI using global probing vectors with action on B_1 with varying coefficients of size $n = 10^3$. MSPAI is compared to Jacobi, SPAI, and $M3_{\text{opt}}$ (16).

2.2. A 2D Model Problem. We consider the block tridiagonal matrix

$$A_2 := \frac{A_1 \otimes I + I \otimes A_1}{2} = \text{blocktridiag}\left(0, -\frac{1}{4}, 0 \,\middle|\, -\frac{1}{4}, 1, -\frac{1}{4} \,\middle|\, 0, -\frac{1}{4}, 0\right), \tag{24}$$

with $A_2 \in \mathbb{R}^{n^2 \times n^2}$ related to the generating function

$$a_2(x, y) = 1 - \frac{\cos(x) + \cos(y)}{2}. \tag{25}$$

As before the smoothing corresponds to the rectangle $I_2 := \{(x, y) \mid x \in [\pi/2, \pi] \wedge y \in [0, \pi]\}$. Hence, the solution of $\min_\omega \max_{x,y \in I_2} |1 - \omega a_2(x, y)|$ yields the optimal Jacobi smoother with smoothing factor $3/5 = 0.6$ for $\omega = 4/5$.

2.2.1. Analytical Derivation of the Optimal Smoother. As approximate inverse smoother, we use the trigonometric polynomial $m_2(x, y) = a + 2b(\cos(x) + \cos(y))$. Via the minimization $1 - m_2(x, y)a_2(x, y)$ over I_2, the solution for the optimal smoothing preconditioner is given by

$$M5_{\text{opt}} := \text{blocktridiag}\left(0, \frac{8}{41}, 0 \,\middle|\, \frac{8}{41}, \frac{48}{41}, \frac{8}{41} \,\middle|\, 0, \frac{8}{41}, 0\right), \tag{26}$$

with smoothing factor $9/41 = 0.2195$.

2.2.2. Individual Probing Masks. Analogously to the 1D case, the minimization can be derived by considering

$$\min_{a,b} \left| 1 - (a + 2bh(x, y)) \left(1 - \frac{h(x, y)}{2}\right) \right|, \quad \text{with } \cos(x) + \cos(y) =: h(x, y) \in [-2, 1]. \tag{27}$$

The corners of I_2 result in the minimization conditions

$$\min_{a,b} \left\{ |2a - 8b - 1|, \left| \frac{a}{2} + b - 1 \right|, \left| \frac{a}{2} + b + \frac{a^2}{16b} - 1 \right| \right\}. \tag{28}$$

To get rid of the quadratic condition, we make use of the fact that the linear conditions in (28) have the same absolute value for the optimal a and b. We obtain $a = 6b$ and the quadratic term can be replaced either by $6ba/b = 6a$ or $(6b)^2/b = 36b$. Consequently, we end up with two additional linear conditions

$$\min_{a,b} \left\{ \left| \frac{7a}{8} + b - 1 \right|, \left| \frac{a}{2} + \frac{13b}{4} - 1 \right| \right\}. \tag{29}$$

Once again, we can see the minimizations (28) and (29) on a and b as conditions for the entries of the smoother directly. A column k of M is now described by the five degrees of freedom $(0, m_{k-n,k}, 0 | m_{k-1,k}, m_{k,k}, m_{k+1,k} | 0, m_{k+n,k}, 0)^T =: \hat{m}_k$. Thus, we gain the following local probing conditions for the 5-point stencil A_2:

$$\begin{aligned} S5_{L1} &: \min_{\hat{m}_k} \left| (0, -2, 0 | -2, 2, -2 | 0, -2, 0) \hat{m}_k - 1 \right|, \\ S5_{L2} &: \min_{\hat{m}_k} \left| \left(0, \frac{1}{4}, 0 \,\middle|\, \frac{1}{4}, \frac{1}{2}, \frac{1}{4} \,\middle|\, 0, \frac{1}{4}, 0\right) \hat{m}_k - 1 \right|, \\ S5_{L3} &: \min_{\hat{m}_k} \left| \left(0, \frac{1}{4}, 0 \,\middle|\, \frac{1}{4}, \frac{7}{8}, \frac{1}{4} \,\middle|\, 0, \frac{1}{4}, 0\right) \hat{m}_k - 1 \right|, \\ S5_{L4} &: \min_{\hat{m}_k} \left| \left(0, \frac{13}{16}, 0 \,\middle|\, \frac{13}{16}, \frac{1}{2}, \frac{13}{16} \,\middle|\, 0, \frac{13}{16}, 0\right) \hat{m}_k - 1 \right|. \end{aligned} \tag{30}$$

It is possible to combine the conditions $S5_{L2}$ and $S5_{L4}$ to obtain an isotropic mask to which the low-frequency probing vector e_S corresponds to. To derive this, we add the diagonal

and subdiagonal values of the weighed conditions $r \cdot S5_{L2}$ and $s \cdot S5_{L4}$ and set them to be equal. The isotropy condition leads to the equation $(r+s)/2 = r/4 + 13s/16$ with the solution $r = 5s/4$. We obtain $a = b = (r+s)/2 = 9s/8$ and $r + s = 5s/4 + s = 9s/4$ for the right hand side. This leads to the additional individual isotropic probing condition

$$S5_{L5} : \min_{\widehat{m}_k} \left| (0,1,0|1,1,1|0,1,0)\widehat{m}_k - 2 \right|. \tag{31}$$

Let us consider A_2 of size $n = 1024$. Following Figure 2(a) we can see that MSPAI satisfying individual probing conditions reduces the smoothing factor in comparison to SPAI with factor 0.339. Utilizing more degrees of freedom with a combination of probing masks, the smoothing factor can be reduced further towards the optimal value of $M5_{\text{opt}}$.

2.2.3. Global Probing Vectors. Similar to the 1D model problem, we observe that $(e \otimes f)^T M5_{\text{opt}} = \alpha(e \otimes f)^T$ for certain e, f, and α. Therefore, we define global probing conditions by using $(e \otimes f)^T M = \alpha(e \otimes f)^T$. Regarding the 2D case, we use global vectors resulting from the Kronecker products $\overline{e}_S := e_S \otimes e_S$, $\overline{e}_{N1} := e_{N1} \otimes e_{N1}$, $\overline{e}_{N2} := e_{N2} \otimes e_{N2}$, and $\overline{e}_{N3} := e_{N3} \otimes e_{N3}$. We obtain

$$\begin{aligned}
S5_{G1} &: \min_{\mathcal{P}(M)=\mathcal{P}} \left\| \overline{e}_S^T M - \frac{80}{41}\overline{e}_S^T \right\|_2, \\
S5_{G2} &: \min_{\mathcal{P}(M)=\mathcal{P}} \left\| \overline{e}_{N1}^T M - \frac{16}{41}\overline{e}_{N1}^T \right\|_2, \\
S5_{G3} &: \min_{\mathcal{P}(M)=\mathcal{P}} \left\| \overline{e}_{N2}^T M - \frac{48}{41}\overline{e}_{N2}^T \right\|_2, \\
S5_{G4} &: \min_{\mathcal{P}(M)=\mathcal{P}} \left\| \overline{e}_{N3}^T M - \frac{48}{41}\overline{e}_{N3}^T \right\|_2.
\end{aligned} \tag{32}$$

Like MSPAI with individual probing masks an approximation on the subspace spanned by the global probing vectors reduces the smoothing factor compared to SPAI and Jacobi (see Figure 2(b)). The global condition $S5_{G1}$ leads to a stable smoothing factor of 0.252. Similar to the 1D model problem it is also possible to derive global probing conditions with action on A_2, not considered here.

2.2.4. 9-Point Stencil of the 2D Model Problem. As a second 2D example, we consider the 9-point stencil of A_2, which is the block tridiagonal matrix

$$A_3 := \text{blocktridiag}\left(-\frac{1}{8}, -\frac{1}{8}, -\frac{1}{8} \,\middle|\, -\frac{1}{8}, 1, -\frac{1}{8} \,\middle|\, -\frac{1}{8}, -\frac{1}{8}, -\frac{1}{8}\right), \tag{33}$$

related to the generating function

$$a_3(x,y) = 1 - \frac{h(x,y)}{4}, \tag{34}$$

with $h(x,y) := \cos(x) + \cos(y) + \cos(x-y) + \cos(x+y)$.

We use the pattern of $a_3(x,y)$ for our approximate inverse smoother

$$m_3(x,y) = a + 2b(\cos(x) + \cos(y) + \cos(x-y) + \cos(x+y)). \tag{35}$$

With $h(x,y) \in [-1,2]$, this yields the minimization problem

$$\min_{a,b} \left| 1 - (a + 2bh(x,y)) \left(1 - \frac{h(x,y)}{4}\right) \right|. \tag{36}$$

The optimal solution is given by $(a,b) = (160/153, 16/153)$ which leads to

$$M9_{\text{opt}} = \text{blocktridiag}\left(\frac{16}{153}, \frac{16}{153}, \frac{16}{153} \,\middle|\, \frac{16}{153}, \frac{160}{153}, \frac{16}{153} \,\middle|\, \frac{16}{153}, \frac{16}{153}, \frac{16}{153}\right), \tag{37}$$

with smoothing factor $1/17 = 0.0588$.

We consider the boundary values $h(x,y) = -1$ and $h(x,y) = 2$ to get linear conditions. A third, quadratic condition can be deduced by setting the derivative of (36) to zero and inserting its solution $h'(x,y) = 2 - (a/4b)$ into (36). We obtain the following minimization conditions at high-frequency values:

$$\min_{a,b} \left\{ \left| \frac{3(a-4b)}{2} - 1 \right|, \left| \frac{3(a+2b)}{4} - 1 \right|, \left| \frac{a}{2} + 2b + \frac{a^2}{32b} - 1 \right| \right\}. \tag{38}$$

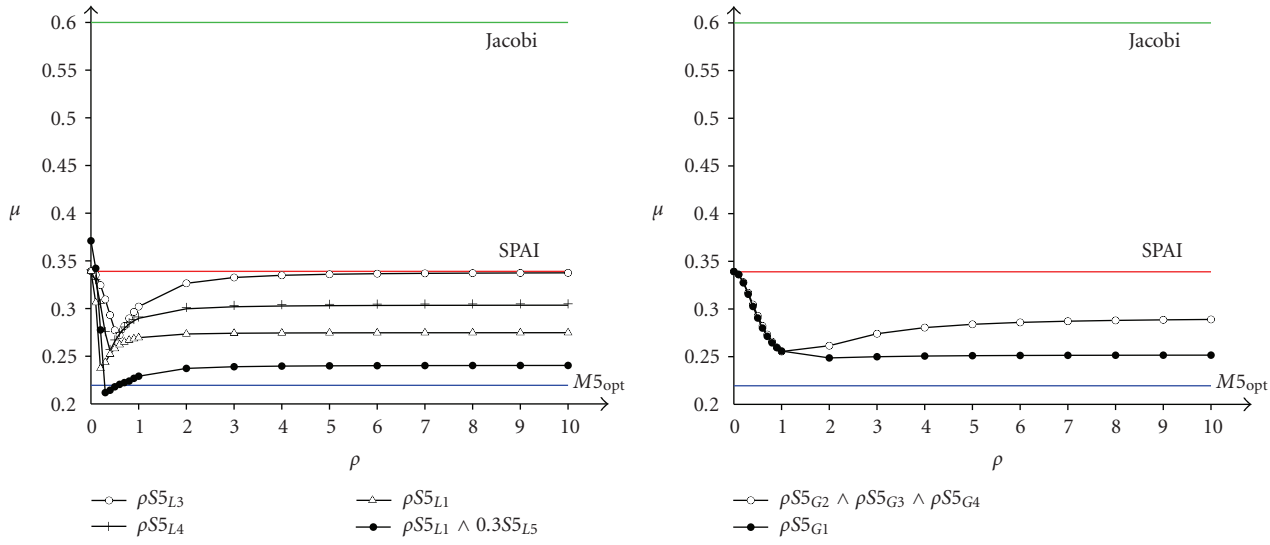
The direct translation into linear probing masks and their combination reveals the individual probing conditions

$$\begin{aligned}
S9_{L1} &: \min_{\widehat{m}_k} \left| \left(-\frac{3}{4}, -\frac{3}{4}, -\frac{3}{4} \,\middle|\, -\frac{3}{4}, \frac{3}{2}, -\frac{3}{4} \,\middle|\, -\frac{3}{4}, -\frac{3}{4}, -\frac{3}{4}\right) \cdot \widehat{m}_k - 1 \right|, \\
S9_{L2} &: \min_{\widehat{m}_k} \left| \left(\frac{3}{16}, \frac{3}{16}, \frac{3}{16} \,\middle|\, \frac{3}{16}, \frac{3}{4}, \frac{3}{16} \,\middle|\, \frac{3}{16}, \frac{3}{16}, \frac{3}{16}\right)\widehat{m}_k - 1 \right|, \\
S9_{L3} &: \min_{\widehat{m}_k} \left| \left(\frac{1}{4}, \frac{1}{4}, \frac{1}{4} \,\middle|\, \frac{1}{4}, \frac{13}{16}, \frac{1}{4} \,\middle|\, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\right)\widehat{m}_k - 1 \right|, \\
S9_{L4} &: \min_{\widehat{m}_k} \left| \left(\frac{41}{64}, \frac{41}{64}, \frac{41}{64} \,\middle|\, \frac{41}{64}, \frac{1}{2}, \frac{41}{64} \,\middle|\, \frac{41}{64}, \frac{41}{64}, \frac{41}{64}\right)\widehat{m}_k - 1 \right|, \\
S9_{L5} &: \min_{\widehat{m}_k} \left| (1,1,1|1,1,1|1,1,1)\widehat{m}_k - \frac{20}{11} \right|.
\end{aligned} \tag{39}$$

In Table 2, we can see that in comparison to SPAI, all individual probing conditions lead to a reduced smoothing factor and are stable for increasing values of ρ. It is almost feasible to reach the optimal value 0.0588. Again, it is possible to derive global probing conditions, for example, with action on A_3, by using Kronecker products of the 1D probing vectors. By observing $(e \otimes f)^T AM9_{\text{opt}} = \alpha(e \otimes f)^T$ similar to Section 2.2.3, we can define conditions with $(e \otimes f)^T AM = \alpha(e \otimes f)^T$ for some e, f, and α.

3. MSPAI in Regularization

An optimal preconditioner for iterative regularization methods should treat the large eigenvalues and have no effect on



(a) MSPAI using individual probing conditions

(b) MSPAI using global probing conditions

Figure 2: Smoothing factor μ against subspace weight ρ. MSPAI using individual (a) and global (b) probing conditions compared to Jacobi, SPAI, and $M5_{\text{opt}}$ (26) for the 2D constant-coefficient model problem A_2 of size $n = 1024$.

Table 2: Smoothing factors by using $M9_{\text{opt}}$ (37), SPAI, and MSPAI with local probing conditions for the 2D 9-point stencil A_3 of order $n = 1024$.

$M9_{\text{opt}}$	SPAI	Weight	Local probing conditions				
		ρ	$\rho S9_{L1}$	$\rho S9_{L2}$	$\rho S9_{L3}$	$\rho S9_{L4}$	$\rho S9_{L5}$
0.0588	0.1657	0.2	0.147	0.161	0.150	0.116	0.095
		1.0	0.129	0.136	0.077	0.066	0.069
		2.0	0.130	0.129	0.063	0.063	
		10^1	0.126	0.126	0.070	0.062	0.068
		10^2	0.130	0.127	0.071		

the smaller eigenvalues not amplifying the noise. Since SPAI is an efficient smoother and satisfies the first condition, we propose MSPAI probing as regularizing preconditioner for general H to suppress a reconstruction on the noise space. Following [20], such a preconditioner M should have the following properties:

(i) $M \approx |H|^{-1}$ on the signal subspace with $|H| = \sqrt{H^T H}$, and

(ii) $M \approx I$ or $M \approx 0$ on the noise subspace.

For circulant matrices, the eigendecomposition is known, and, therefore, these conditions can be satisfied by manipulating the eigenvalues. Most of the preconditioners make use of properties of structured matrices. For general matrices, this is usually not possible, and we thus use the probing facility of MSPAI in order to derive a different approximation quality on the signal or noise subspace, respectively.

(i) For the signal space, we could use the vector $e_S := (1, 1, \ldots, 1)^T$ representing smooth components, and therefore, the important part of the signal subspace. In this paper, we expect that the signal subspace is already taken into account by SPAI itself, and thus we omit this probing possibility.

(ii) For the noise space, we use $e_{N1} := (1, -1, 1, -1, \ldots)^T$, $e_{N2} := (1, 0, -1, 0, 1, \ldots)^T$, and $e_{N3} := (0, 1, 0, -1, 0, \ldots)^T$ as typical vectors related to fast oscillations. In case of using probing masks, $\hat{s}_k = (-1, 1, -1)$ and $\hat{s}_k = (1, 0, -1)$ are related to e_{N1} and e_{N2}, e_{N3}, respectively.

For higher dimensional problems, probing vectors typically result from a Kronecker product of 1D probing vectors. The conditions in MSPAI are given by $AM \approx I$ in order to derive a good preconditioner and fast convergence on the signal subspace and $\rho e_N^T M \approx 0$ with $\rho > 0$ for the noise subspace in order to avoid a deterioration of the reconstruction by the preconditioner.

3.1. A 1D Model Problem. Let us consider the matrix $H_1 := \text{tridiag}(1/2, 1, 1/2)$ related to the symbol

$$h_1(x) = 1 + \frac{e^{ix} + e^{-ix}}{2} = 1 + \cos(x). \tag{40}$$

As approximate inverse preconditioner, we choose the trigonometric polynomial of symbol $m_4(x) = a + 2b\cos(x)$ related to a Toeplitz matrix $M = \text{tridiag}(b, a, b)$. The entries a and b should be determined such that the preconditioner M acts both nearly as the inverse on the signal subspace and as zero on the noise subspace. For $x \in \{0,\ \pi/2,\ \pi\}$, this leads to the minimization conditions

$$\min_{a,b}\left\{|m_4(0)h_1(0)-1|, \left|m_4\left(\frac{\pi}{2}\right)h_1\left(\frac{\pi}{2}\right)-1\right|, \rho|m_4(\pi)|\right\} = \min_{a,b}\{|2(a+2b)-1|, |a-1|, \rho|a-2b|\}. \tag{41}$$

By using the factor ρ, we introduce a weighting between the signal and the noise conditions. We obtain the optimal solution of (41) by solving the equality

$$(2a+4b-1) = (a-1) = -\rho(a-2b). \tag{42}$$

Thus, we are able to derive the optimal regularizing preconditioner

$$M3_\rho := \text{tridiag}\left(\frac{\rho-0.5}{2+5\rho}, \frac{2+2\rho}{2+5\rho}, \frac{\rho-0.5}{2+5\rho}\right). \tag{43}$$

In the limiting case, for only preconditioning on the signal space ($\rho = 0$), we get $M3_0 = \text{tridiag}(-1/4, 1, -1/4)$, and for only regularizing on the noise subspace ($\rho = \infty$) we get $M3_\infty = \text{tridiag}(1/5, 2/5, 1/5) = (2/5)H_1 = (2/5)H_1^T$. Therefore, the preconditioner for $\rho = \infty$ is equivalent to the normal equations. With the parameter ρ, we can choose between preconditioning on the signal subspace or suppression of noise.

Again, the conditions (41) can be seen as conditions for the entries of the preconditioner directly, where a column k is described by the three degrees of freedom $\hat{m}_k := (m_{k-1,k}, m_{k,k}, m_{k+1,k})^T$. Both the first and the second condition are covered by SPAI and are not used in MSPAI, as we can think of SPAI approximation as acting mainly on the signal subspace. The translation of the last condition of (41) into an individual probing condition with probing mask $\hat{s}_k$ yields

$$R3_{L1} : \min_{\hat{m}_k} |(-1, 1, -1)\hat{m}_k|. \tag{44}$$

It is possible to transfer this individual regularization condition to the global condition $e^T M \approx 0$ with probing vector $e_{N1} = (1, -1, 1, -1, \ldots)^T$, representing high-frequency noisy components. Thus, we force M to act as approximately zero on the noise subspace. We refer to the high-frequency global probing condition as

$$R3_{G1} : \min_{\mathcal{P}(M)=\mathcal{P}} \left\|e_{N1}^T M\right\|_2. \tag{45}$$

As a first numerical example, we examine the given 1D blur operator H_1^{2k} with $k > 0$ for a given vector x representing the original data. We use the MSPAI regularization property to reconstruct two different signals perturbed by random noise. By using powers of H, we make the problem more ill-conditioned and thus enforce the difference between the reconstruction results of all methods. In order to be able to use CG as iterative method, we ensure to have a symmetric and positive definite (spd) preconditioner via corresponding powers of MM^T after the computation of M; that is, the preconditioner is $(MM^T)^k$. We implement the CG algorithm without a stopping criterion to iterate to the maximum number of specified iterations, that is, to observe the semiconvergent behavior. We keep track of the reconstruction error between the original signal x and the reconstruction $\tilde{x}$ as well as of the residual in each iteration. If not mentioned otherwise, we refer to CG as using CG without preconditioner and to PCG as CG using a preconditioner, for example, MSPAI with properties on a certain subspace. Notice that in general, an appropriate stopping criterion should be employed [26] when using an iterative regularization method.

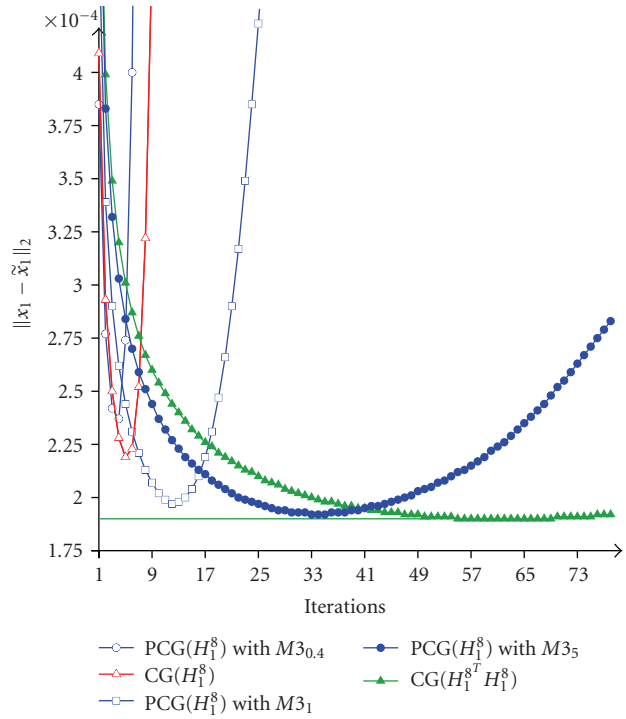


Figure 3: Reconstruction error against the iteration for the 1D operator H_1^8 and original data x_1 of size $n = 10^3$ affected by random noise of order 0.01%. CG is compared to PCG using the optimal preconditioner $M3_\rho$ (43) for different weights ρ.

Let us consider in a first example the ill-posed problem with operator H_1^8 and original data x_1 of size $n = 10^3$ affected by random noise of order 0.01%. Here, x_1 denotes the smooth signal

$$x_1 := \left(\sin\left(\frac{4\pi j}{n}\right)\right)_{j=1,\ldots,n}. \tag{46}$$

The measurement of the error $\|x_1 - \tilde{x}_1\|_2$ within each iteration shows that for increasing values of ρ the reconstruction is stable, and the region of optimal reconstruction quality

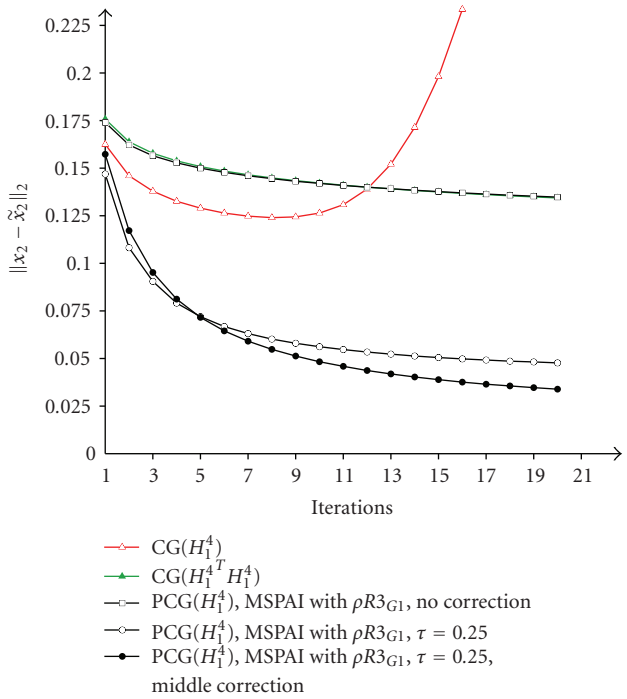


Figure 4: Reconstruction error against the iteration for the 1D blur operator H_1^4 and the original data x_2 of size $n = 10^3$ affected by random noise of order 0.1%. CG is compared to CG for the normal equations and to PCG using MSPAI satisfying $R3_{G1}$ with $\rho = 10^2$.

Table 3: Optimal reconstruction error $\|x_1 - \tilde{x}_1\|_2$ for the 1D operator H_1^8 and original data x_1 of size $n = 10^3$ affected by random noise of order 0.01%.

Regularization method		Optimal value	Reached at iteration
CG(H_1^8)		$2.1870 \cdot 10^{-4}$	5
CG($H_1^{8T} H_1^8$)		$1.8998 \cdot 10^{-4}$	62
PCG(H_1^8) with $M3_\rho$	$\rho = 0.4$	$2.3699 \cdot 10^{-4}$	4
	$\rho = 1$	$1.9746 \cdot 10^{-4}$	12
	$\rho = 5$	$1.9236 \cdot 10^{-4}$	34
	$\rho = 10^2$	$1.9226 \cdot 10^{-4}$	48

Table 4: Optimal reconstruction error $\|x_2 - \tilde{x}_2\|_2$ for the 1D blur operator H_1^4 and the original data x_2. The problem has size $n = 10^3$ and is affected by random noise of order 0.1%. $\rho = 10^2$.

Regularization method		Optimal value	Reached at iter.
Tikhonov ($\gamma = 0.3$)		0.1286	—
CG(H_1^4)		0.1240	8
PCG(H_1^4)$\rho R3_{G1}$	no boundary correction	0.1210	130
	$\tau = 0.25$	0.0433	58
	$\tau = 0.25$, middle correction	0.0263	52

is broader and smoother compared to the unpreconditioned CG (see Figure 3). Moreover, by putting more weight to the regularization property of $M3_\rho$ it is possible to approximate the distribution of CG using normal equations. However, we achieve almost similar reconstruction quality in fewer iterations. Refer to Table 3 for details. Hence, with this family of preconditioners, we can steer the iteration to faster convergence (small ρ) or slightly better reconstruction quality ($\rho > 1$).

In our second 1D example, we observe the behavior for the blur operator H_1^4 and a signal which has strongly increasing entries near the boundaries and an additional nonanalytic point in the middle

$$x_2 := \left(\frac{1}{j} - \frac{1}{n+1-j} + \frac{1}{n/2+1/4-j} + \sin\left(\frac{4j\pi}{n}\right)\right)_{j=1,\ldots,n}. \tag{47}$$

The problem has size $n = 10^3$ and is perturbed by random noise of order 0.1%. Having a closer look at the error between the original and the reconstructed signal vector shows that the reconstruction is very good for interior components, different from $n/2$, but deteriorates near the boundary. Therefore, we introduce some correction at the boundary by changing the components $(e_{N1})_1 = \tau$ and $(e_{N1})_n = \pm\tau$, in which τ is a weight factor of heuristic choice, and in most cases $\tau \in [0, 1]$. Additionally, we correct our probing subspace in the middle by $(e_{N1})_{n/2} = 0$ and $(e_{N1})_{n/2\pm1} = 1/2(e_{N1})_{n/2\pm1}$ again with heuristic weight 1/2. Following Figure 4, we can see the typical CG behavior of reaching the optimal value after a few iterations but afterwards deteriorating the reconstruction very fast by reducing the error relative to the noise subspace. Using a MSPAI preconditioner the optimum is reached after more iterations but in a stable and smooth manner with slightly smaller value. Table 4 shows the optimal reached reconstruction error with its corresponding number of iterations. Compared to the unpreconditioned CG, the reconstruction $\tilde{x}_2$ is by a factor of 2.86 times more accurate when using boundary corrections and 4.72 times when using both corrections within MSPAI.

Likewise, it is possible to use subspace corrections within mask probing. We change the individual probing condition $R3_{L1}$ for the first and last column of the preconditioner, in order to take into account the missing value -1 which lies outside the vector. This lost information can be incorporated by using the probing mask $(1 - \tau, -1 + (-1 + \tau)) = (1 - \tau, \tau - 2)$ for the first and the last column. Note that similarly, interior discontinuities can also be treated by modifying the related masks. Furthermore, also for $M3_\rho$ it is possible to build in similar corrections, for example, by weighting the nondiagonal entries. Various experiments revealed that using such corrections at discontinuities of the data vector x within the probing subspaces lead to better reconstruction, that is, smaller reconstruction errors compared to the unpreconditioned case. We will use similar techniques in Section 3.3.

As a more general problem, we consider the reconstruction of x_1 based on the blur operator B_2 of size $n = 10^3$ affected by random noise of order 1% and 0.1%, respectively.

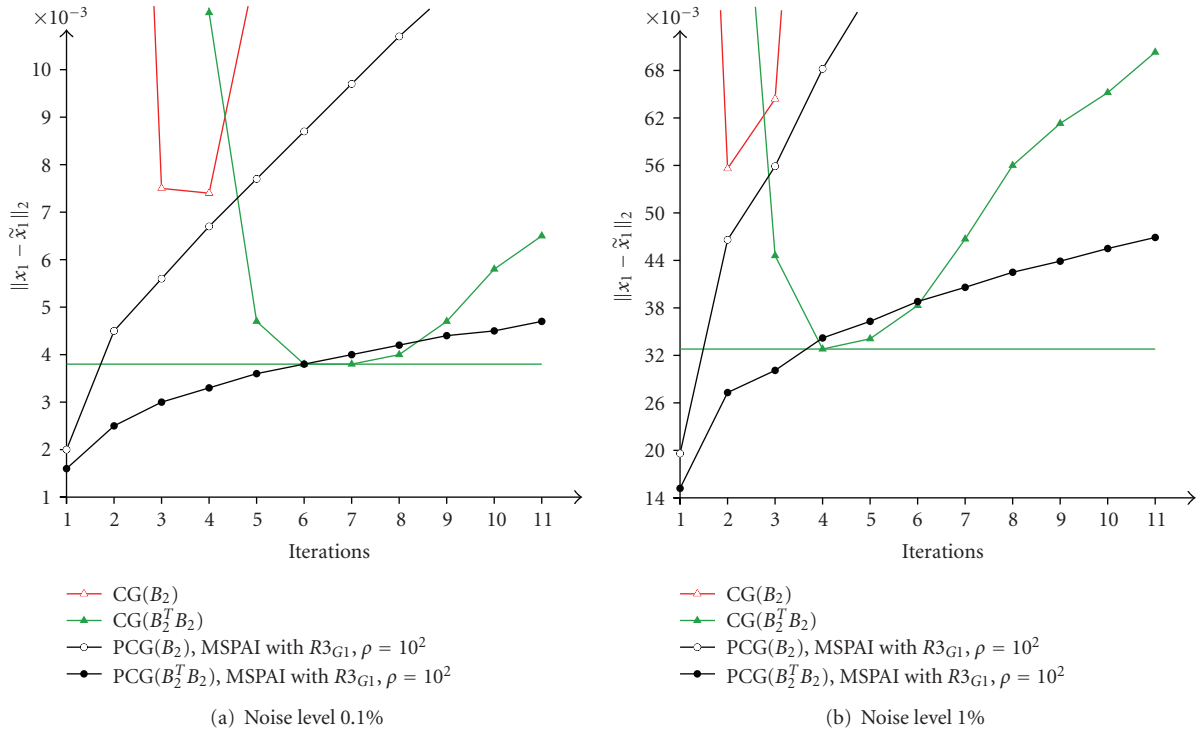


FIGURE 5: Reconstruction error against the iteration for the 1D blur operator B_2 and the original data x_1 of size $n = 10^3$ affected by random noise of order 0.1% (a) and 1% (b), respectively. CG is compared to CG using the normal equations, PCG for B_2 and $B_2^T B_2$ using MSPAI with $R3_{G1}$.

B_2 denotes the tridiagonal system with varying coefficients whose kth row is given by

$$(B_2)_{k,:} := (0,\dots,0,w_{k-1},w_{k-1}+w_k,w_k,0,\dots,0), \quad \text{for } w := \left(\frac{2j^2}{n^2}+1\right)_{j=0,\dots,n}, \tag{48}$$

with $k \in \{1,\dots,n\}$. Following Figure 5, CG using normal equations yields both better reconstruction after slightly more iterations and smoother convergence compared to the unpreconditioned CG. In case of applying MSPAI satisfying the global probing condition $R3_{G1}$ to CG(B_2), we are able to reconstruct x_1 with smaller error and in a smooth way. These positive effects can be enforced by using it for CG with normal equations. The optimal reconstruction is achieved in fewer iterations, is much better, and the convergence curve much smoother.

3.2. A 2D Model Problem. In 2D, we consider the matrix

$$H_2 := \frac{H_1 \otimes I + I \otimes H_1}{2} = \text{blocktridiag}\left(0,\frac{1}{4},0 \,\middle|\, \frac{1}{4},1,\frac{1}{4} \,\middle|\, 0,\frac{1}{4},0\right), \tag{49}$$

with its corresponding symbol $h_2(x,y) = 1 + \cos(x)/2 + \cos(y)/2$ and preconditioner $m_5(x,y) = a + 2b(\cos(x) + \cos(y))$. With $\cos(x) + \cos(y) =: h(x,y) \in [-1,2] =: I_3$, we have to minimize

$$\min_{a,b} \max_{x,y \in I_3} \left| 1 - (a + 2bh(x,y)) \left(1 - \frac{h(x,y)}{2}\right) \right|, \tag{50}$$

relative to the signal subspace and $\min_{a,b}\rho|a - 4b|$ for $h(x,y) = -2$ relative to the noise subspace. This yields the optimal block tridiagonal preconditioner

$$M5_\rho = \text{blocktridiag}\left(0, 2\rho - \frac{3}{2}, 0 \,\middle|\, 2\rho - \frac{3}{2}, 9 + 8\rho, 2\rho - \frac{3}{2} \,\middle|\, 0, 2\rho - \frac{3}{2}, 0\right). \tag{51}$$

The individual probing masks for a reduced inner column of the preconditioner $\hat{m}_k := (0, m_{k-n,k}, 0 | m_{k-1,k}, m_{k,k}, m_{k+1,k} | 0, m_{k+n,k}, 0)^T$ as well as the high-frequency probing vector for the 2D case can be derived via Kronecker products of the 1D probing vectors. Thus, we obtain the individual and global probing condition

$$R5_{L1} : \min_{\hat{m}_k} \left| ((-1,1,-1) \otimes (-1,1,-1))\hat{m}_k \right|, \qquad R5_{G1} : \min_{\mathcal{P}(M)=\mathcal{P}} \left\| (e_{N1} \otimes e_{N1})^T M \right\|_2, \tag{52}$$

respectively.

We observe the behavior for the 2D problem with blur operator H_2^4 and original data x_1 of size $n = 50^2$ affected by

Table 5: Optimal reconstruction error $\|p - \widetilde{p}\|_2$ at given iteration for the `blur` problem of [24] invoked with `blur(150,4,1)` for noise of order 0.01%, 0.1%, 1%, and 10%. Subspace weight is $\rho = 1$.

Regularization method	Noise level 0.01%		Noise level 0.1%	
	$\|p - \widetilde{p}\|_2$	at it.	$\|p - \widetilde{p}\|_2$	at it.
CG(G)	25.637	3	41.828	1
CG(G^TG)	24.123	19	37.807	2
PCG(G), MSPAI with $R5_{G1,G2,G3}$	23.620	6	39.007	1
PCG(G^TG), MSPAI with $R5_{G1,G2,G3}$	23.778	51	38.023	3
Regularization method	Noise level 1%		Noise level 10%	
	$\|p - \widetilde{p}\|_2$	at it.	$\|p - \widetilde{p}\|_2$	at it.
CG(G)	591.0	1	$13.820 \cdot 10^3$	1
CG(G^TG)	82.5	1	$1.253 \cdot 10^3$	1
PCG(G), MSPAI with $R5_{G1,G2,G3}$	189.8	1	$4.104 \cdot 10^3$	1
PCG(G^TG), MSPAI with $R5_{G1,G2,G3}$	69.7	1	$0.962 \cdot 10^3$	1

random noise of order 0.1%. Again, for $M5_\rho$, with increasing values of ρ, we approximate the convergence of CG using normal equations (refer to Figure 6). Applying MSPAI with $R5_{G1}$ to PCG it is almost possible to reach the value of CG($H_2^{4T}H_2^4$) but in less iterations. Higher values of ρ lead to smooth and broad convergence curves similar to $M5_\rho$. CG has its optimal value after 9 iterations with error $2.219 \cdot 10^{-3}$, MSPAI using $R5_{G1}$, $\rho = 1$ after 12 iterations with value $1.496 \cdot 10^{-3}$, and CG using normal equations reaches the error $1.282 \cdot 10^{-3}$ after 44 iterations.

3.3. Examples from the Regularization Toolbox. We are interested in the impact of using sparse approximate inverse preconditioners on some problems of the MATLAB package Regularization Tools Version 4.1 for analysis and solution of discrete ill-posed problems, developed by Hansen [24].

In our first example, we consider the `deriv2` example which is a discretization of a first kind Fredholm integral equation. We choose `deriv2(n,case)` with case = 2. Our problem has size $n = 2 \cdot 10^3$ and is affected with random noise of order 0.001% and 0.01%, respectively. Note that the system matrix K is dense. We apply both CG and PCG to K^TK and force the MSPAI to act as approximately zero on the noise subspace by using the high-frequency probing conditions $R3_{L1}$ and $R3_{L2}$: $\min_{\widehat{m}_k} |(1, 0, -1)\widehat{m}_k|$ simultaneously. We weigh the subspace with $\rho = 10^3$ and apply the symmetric preconditioner $M + M^T$ to the normal equations. To avoid deterioration at the boundary, we adjust M by resetting the values $M_{1,1} = M_{2,2}$, $M_{2,1} = M_{3,2}$, $M_{n,n} = M_{n-1,n-1}$, and $M_{n-1,n} = M_{n-2,n-1}$.

Following Figure 7, we are able to achieve better reconstruction $\widetilde{g}$ of the original data g and in fewer iterations when applying MSPAI. For other noise levels and for `deriv(n,1)`, we observed similar behavior.

Let us focus on the `blur` [24] test problem as a second example which is deblurring images degraded by atmospheric turbulence blur. The matrix G is an n^2-by-n^2 symmetric, doubly block Toeplitz matrix that models blurring of an n-by-n image by a Gaussian point spread function. The parameter σ controls the width of G and thus the amount of smoothing and ill-posedness. $G = A_1 \otimes A_1$ is symmetric block banded and possibly positive definite depending on n and σ. We choose the problem to be of size $n = 150$ with bandwidth 4 and $\sigma = 1$, that is, we invoke `blur(150,4,1)` and G is of size $150^2 \times 150^2$. The original data vector is denoted by p.

We compare the unpreconditioned CG to PCG both for G and for the normal equations G^TG. In case of preconditioning G, MSPAI is symmetrized via $M + M^T$ and for G^TG the preconditioner M^TM is applied. For MSPAI, we impose the blocktridiagonal pattern $\widehat{m}_k := (0, m_{k-n,k}, 0|m_{k-1,k}, m_{k,k}, m_{k+1,k}|0, m_{k+n,k}, 0)^T$. In view of the structure of G, we build the high-frequency subspace by Kronecker products of oscillatory probing vectors in the regularizing global conditions $R5_{G1}$ and the new ones $R5_{G2}$: $\min_{\mathcal{P}(M)=\mathcal{P}} \|(e_{N1} \otimes e_{N2})^T M\|_2$ and $R5_{G3}$: $\min_{\mathcal{P}(M)=\mathcal{P}} \|(e_{N2} \otimes e_{N1})^T M\|_2$, all weighed with $\rho = 1$.

Following Figure 8 and Table 5, we obtain better reconstruction $\widetilde{p}$ when applying MSPAI in contrast to the unpreconditioned CG for G or G^TG. We observed similar behavior for other values of *band* and σ.

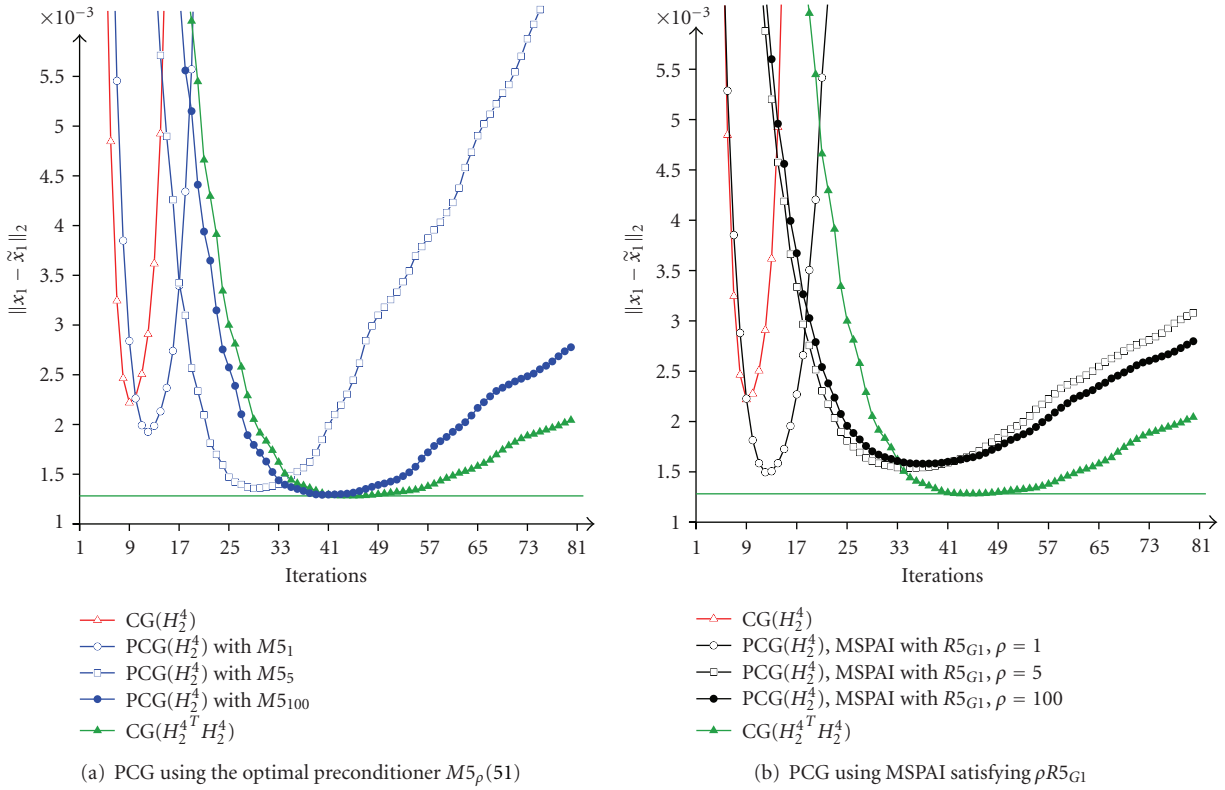
As a last example we reduce the 2D `blur` example of Hansen to the 1D case. For the blur operator we take the 1D analogon of G and reduce the 2D right hand side p to appropriate size. The consideration of example H_1^4 with original data x_2 (Table 4 and Figure 4) shows that the preconditioner should take also into account the behavior of the original or blurred data vector. Smoothing, for example, with $M = \text{tridiag}(1/2, 1, 1/2)$ makes sense to remove noisy components only as long as the data is continuous. At discontinuities, smoothing would cause additional errors. Therefore, we use a modified tridiagonal smoothing preconditioner with jth row $(0, \ldots, 0, r_{j-1}, 1, r_j, 0, \ldots, 0)$. Here, $r_j \approx 1/2$ near continuous components x_j, but $r_j \approx 0$ near discontinuities. In case that we know the original data x we define M_r with jth row

$$(M_r)_{j,:} := \left(0, \ldots, 0, r_{j-1}, 1, r_j, 0, \ldots, 0\right),$$
$$\text{for } r := \left(\frac{1}{2} \cdot \frac{1}{1 + \rho\left|x_j - x_{j+1}\right|}\right)_{j=1,\ldots,n-1}. \tag{53}$$

Otherwise we define the preconditioner $M_{\widetilde{r}}$ with jth row

$$(M_{\widetilde{r}})_{j,:} := \left(0, \ldots, 0, \widetilde{r}_{j-1}, 1, \widetilde{r}_j, 0, \ldots, 0\right),$$
$$\text{via } \widetilde{r} := \left(\frac{1}{2} \cdot \frac{1}{1 + \left(\rho\left|b_j - b_{j+1}\right|\right)^k}\right)_{j=1,\ldots,n-1}. \tag{54}$$

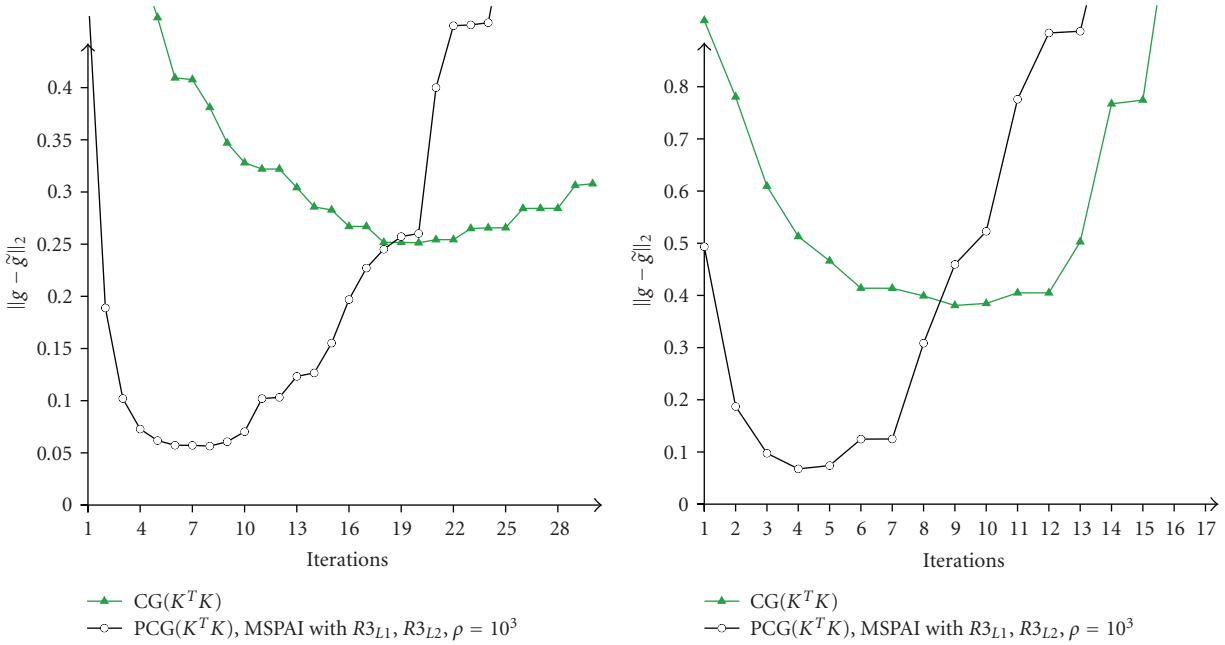
The parameters ρ and k have to be chosen in such a way that discontinuities are revealed as good as possible.



(a) PCG using the optimal preconditioner $M5_\rho(51)$

(b) PCG using MSPAI satisfying $\rho R5_{G1}$

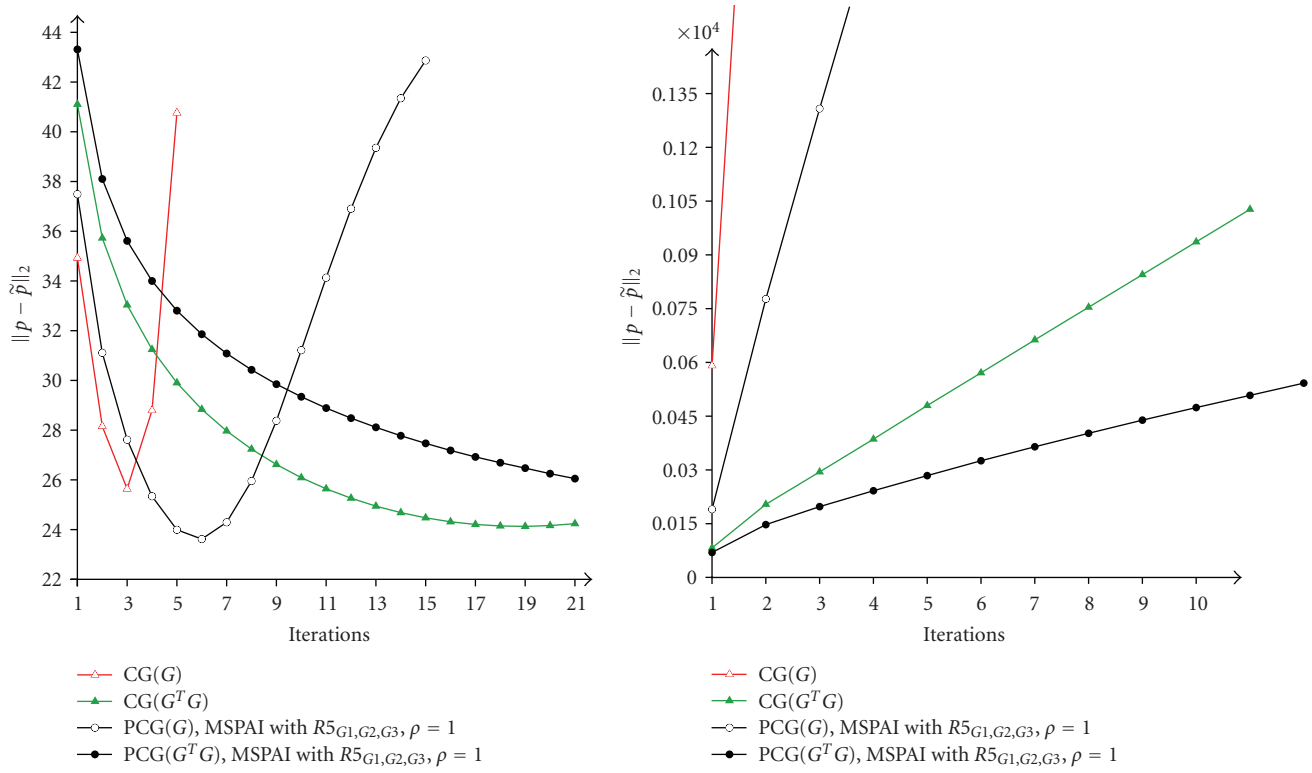
FIGURE 6: Reconstruction error against the iteration for the 2D problem with blur operator H_2^4 and original data x_1 of size $n = 50^2$ affected by random noise of order 0.1%. CG is compared to CG for the normal equations and to $M5_\rho$ (a) and MSPAI satisfying $\rho R5_{G1}$ (b), respectively.



(a) Noise level 0.001%

(b) Noise level 0.01%

FIGURE 7: Reconstruction error against the iteration for the `deriv2` problem of [24] invoked with `deriv2(2000,2)` and affected by random noise of order 0.001% (a) and 0.01% (b), respectively. CG is compared to PCG using MSPAI with $R3_{L1}$, $R3_{L2}$, and $\rho = 10^3$ for the normal equations. MSPAI is symmetrized via $M + M^T$.



(a) Noise level 0.01%

(b) Noise level 1%

FIGURE 8: Reconstruction error against the iteration for the `blur` problem of [24] invoked with `blur(150,4,1)` and affected by random noise of order 0.01% (a) and 1% (b), respectively. CG is compared to CG using normal equations, PCG using MSPAI with $R5_{G1}$, $R5_{G2}$, $R5_{G3}$, and PCG using MSPAI for normal equations with $\rho = 1$. We use $M + M^T$ as preconditioner within PCG(G) and M^TM for PCG(G^TG).

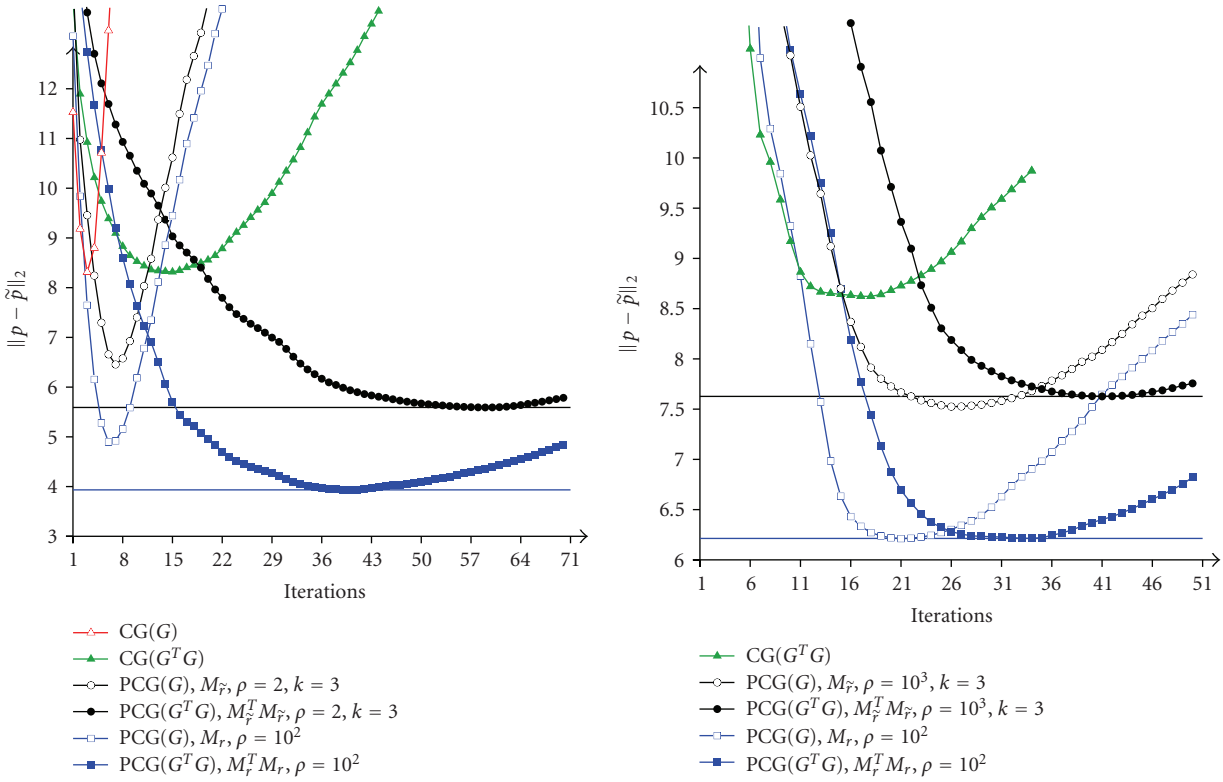
Following Figure 9, the reconstruction is strongly improved if the preconditioner is adjusted relative to the discontinuities of x. Also, using only the observed data improves the reconstruction. Therefore, we could also consider an iterative process, where a first approximation x_1 on x is used to define the tridiagonal preconditioner M_1 delivering an improved approximation x_2 which again gives a new preconditioner M_2, and so on.

4. Conclusion

We have considered the derivation of preconditioners with special behavior on certain subspaces. For this purpose, analytic minimization problems in functions have been translated into MSPAI minimizations for vectors based on masks. Such mask-based probing conditions can be different for each column m_k of the preconditioner M and can be written in the form $\min_{\hat{m}_k} |\hat{s}_k \hat{m}_k - f_k|$ with reduced vectors $\hat{s}_k$, $\hat{m}_k$, and scalar $f_k \in \mathbb{R}$. Mask probing has the advantage that for each sparse column m_k we can use a different sparse probing vector s_k. Furthermore, we have introduced probing conditions based on probing vectors that are global for the whole matrix M in the form $\min_{\mathcal{P}(M)=\mathcal{P}} \|e^TM - f^T\|_2$ and $\min_{\mathcal{P}(M)=\mathcal{P}} \|e^TAM - f^T\|_2$, respectively, with $e, f \in \mathbb{R}^n$. The probing vectors are related to the low-frequency or smooth subspace, represented by $e_S = (1, 1, \ldots, 1)^T$, or to the high-frequency oscillatory subspace, represented, for example, by $e_{N1} = (1, -1, 1, -1, \ldots)^T$, $e_{N2} = (1, 0, -1, 0, 1, \ldots)^T$, or $e_{N3} = (0, 1, 0, -1, 0, \ldots)^T$.

For multigrid methods, we have shown that the smoothing property of approximate inverses like SAI or SPAI can be improved significantly by using MSPAI with appropriate probing masks or probing vectors. In special cases, we can analytically determine the optimal approximate inverse smoother and its corresponding smoothing factor. SPAI is far from being optimal in this class, but including convenient individual or global probing conditions, we can nearly reach the optimal smoothing factor with MSPAI. Moreover, an increasing weighting of the subspace leads to stable behavior. Our tests on systems with varying coefficients and for the 2D case demonstrate that even the usage of global probing conditions with action on A, only satisfying the approximation $e^TAM \approx e^T$, reduces the smoothing factor in comparison to SPAI and the damped Jacobi method.

A different subspace approach becomes necessary during the recovery process of blurred signals. Here, our main focus is to allow a better and more stable reconstruction of the original signal. Applying a preconditioner in iterative regularization can easily lead to a deterioration of the reconstruction by approximating the inverse also in the noise subspace, or by removing high-frequency components



(a) G with $\sigma = 1$

(b) G with $\sigma = 10$

FIGURE 9: Reconstruction error against iterations for the 1D `blur` problem of size $n = 10^3$ and band $= 4$ affected by random noise of order 0.1%. G has $\sigma = 1$ (a) and $\sigma = 10$ (b), respectively. CG is compared to PCG using M_r and $M_{\tilde{r}}$, respectively, for G and G^TG.

in the original signal. Therefore, preconditioners have to be developed very carefully. We derive approximate inverse preconditioners analytically based on generating functions, or by applying MSPAI with probing masks or probing vectors. These preconditioners allow to incorporate filtering for noise reduction, and they can be adjusted both to the system matrix, for example, the blur operator, and to the data vector x. So, the deterioration of the reconstruction at discontinuities of x can be reduced by modifying the probing conditions relative to the variation of the signal data. We show that these preconditioners can be used for faster convergence or better reconstruction. The application to more general problems is an interesting and important task which will be investigated in the future.

References

[1] S. Demko, W. F. Moss, and P. W. Smith, "On approximate-inverse preconditioners," *Mathematics of Computation*, vol. 43, pp. 491–499, 1984.

[2] T. Huckle and A. Kallischko, "Frobenius norm minimization and probing for preconditioning," *International Journal of Computer Mathematics*, vol. 84, no. 8, pp. 1225–1248, 2007.

[3] A. Kallischko, *Modified sparse approximate inverses (MSPAI) for parallel preconditioning*, Ph.D. thesis, Technische Universität München, 2008.

[4] M. W. Benson and P. O. Frederickson, "Iterative solution of large sparse linear systems arising in certain multidimensional approximation problems," *Utilitas Mathematica*, vol. 22, pp. 127–140, 1982.

[5] J. D. F. Cosgrove, J. C. Díaz, and A. Griewank, "Approximate inverse preconditionings for sparse linear systems," *International Journal of Computer Mathematics*, vol. 44, pp. 91–110, 1992.

[6] M. J. Grote and T. Huckle, "Parallel preconditioning with sparse approximate inverses," *SIAM Journal of Scientific Computing*, vol. 18, no. 3, pp. 838–853, 1997.

[7] R. M. Holland, A. J. Wathen, and G. J. Shaw, "Sparse approximate inverses and target matrices," *SIAM Journal of Scientific Computing*, vol. 26, no. 3, pp. 1000–1011, 2005.

[8] O. Axelsson, *Iterative Solution Methods*, Cambridge University Press, New York, NY, USA, 1994.

[9] O. Axelsson and B. Polman, "A robust preconditioner based on algebraic substructuring and two-level grids," in *Robust Multigrid Methods*, W. Hackbusch, Ed., vol. 23, pp. 1–26, Friedrich Vieweg & Sohn, 1988.

[10] T. F. C. Chan and T. P. Mathew, "The interface probing technique in domain decomposition," *SIAM Journal on Matrix Analysis and Applications*, vol. 13, pp. 212–238, 1992.

[11] M. Benzi and M. Tůma, "Comparative study of sparse approximate inverse preconditioners," *Applied Numerical Mathematics*, vol. 30, no. 2, pp. 305–340, 1999.

[12] U. Grenander and G. Szegö, *Toeplitz Forms and Their Applications*, Chelsea Publishing, NewYork, NY, USA, 1984.

[13] T. Huckle, "Compact fourier analysis for designing multigrid methods," *SIAM Journal on Scientific Computing*, vol. 31, pp. 644–666, 2008.

[14] U. Trottenberg, C. W. Oosterlee, and A. Schüller, *Multigrid*, Academic Press, San Diego, Calif, USA, 2001.

[15] W.-P. Tang and W. L. Wan, "Sparse approximate inverse smoother for multigrid," *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1236–1252, 2000.

[16] O. Bröker, M. J. Grote, C. Mayer, and A. Reusken, "Robust parallel smoothing for multigrid via sparse approximate inverses," *SIAM Journal of Scientific Computing*, vol. 23, no. 4, pp. 1396–1417, 2002.

[17] A. N. Tikhonov, "Solution of incorrectly formulated problems and regularization method," *Soviet Mathematics. Doklady*, vol. 4, pp. 1035–1038, 1963.

[18] H. W. Engl, M. Hanke, and A. Neubauer, *Regularization of Inverse Problems*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1996.

[19] M. Hanke, "Iterative regularization techniques in image reconstruction," in *Proceedings of the Conference on Mathematical Methods in Inverse Problems for Partial Differential Equations*, Mt. Holyoke, Mass, USA, 1998.

[20] M. Hanke, J. G. Nagy, and R. J. Plemmons, "Preconditioned iterative regularization for ill-posed problems," in *Numerical Linear Algebra and Scientific Computing*, pp. 141–163, de Gruyter, Berlin, Germany, 1993.

[21] J. G. Nagy, R. J. Plemmons, and T. C. Torgersen, "Iterative image restoration using approximate inverse preconditioning," *IEEE Transactions on Image Processing*, vol. 5, no. 7, pp. 1151–1162, 1996.

[22] J. G. Nagy, "Accelerating convergence of iterative image restoration algorithms," Tech. Rep. TR-2007-020, Emory University, Atlanta, Ga, USA, 2007, Proceedings of the Advanced Maui Opticaland Space Surveillance Technologies (AMOS) Conference.

[23] J. G. Nagy and D. P. O'Leary, "Restoring images degraded by spatially variant blur," *SIAM Journal of Scientific Computing*, vol. 19, no. 4, pp. 1063–1082, 1998.

[24] P. C. Hansen, "Regularization tools: a Matlab package for analysis and solution of discrete ill-posed problems," *Numerical Algorithms*, vol. 6, no. 1, pp. 1–35, 1994.

[25] J. G. Nagy, K. Palmer, and L. Perrone, "Iterative methods for image deblurring: a Matlab object-oriented approach," *Numerical Algorithms*, vol. 36, no. 1, pp. 73–93, 2004.

[26] P. C. Hansen, "Analysis of discrete ill-posed problems by means of the l-curve," *SIAM Review*, vol. 34, pp. 561–580, 1992.

Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2010, Article ID 153540, 9 pages
doi:10.1155/2010/153540

Research Article

Generalized Superposition Modulation and Iterative Demodulation: A Capacity Investigation

Christian Schlegel, Marat Burnashev, and Dmitri Truhachev

Department of Computing Science, University of Alberta, Edmonton, AB, Canada T6G 2E8

Correspondence should be addressed to Christian Schlegel, schlegel@ualberta.ca

Received 1 March 2010; Revised 30 June 2010; Accepted 12 August 2010

Academic Editor: Peter Hoeher

Modulation with correlated signal waveforms is considered. Such correlation arises naturally in a number of modern communications systems and channels, for example, in code-division multiple-access (CDMA) and multiple-antenna systems. Data entering the channel in parallel streams either naturally or via inverse multiplexing is transmitted redundantly by adding additional signal waveforms populating the same original time-frequency space, thus not requiring additional bandwidth or power. The transmitted data is spread over a frame of N signaling intervals by random permutations. The receiver combines symbol likelihood values, calculates estimated signals and iteratively cancels mutual interference. For a random choice of the signal waveforms, it is shown that the capacity of the expanded waveform set is nondecreasing and achieves the capacity of the Gaussian multiple access channel as its upper limit when the number of waveforms becomes large. Furthermore, it is proven that the iterative demodulator proposed here can achieve a fraction of 0.995 or better of the channel capacity irrespective of the number of transmitted data streams. It is also shown that the complexity of this iterative demodulator grows only linearly with the number of data streams.

1. Introduction

Modulation is the process of injectively mapping elements of a discrete set, called the messages, onto functions of time, called the signals, for the purpose of information transmission. The signals form a (finite-dimensional) Hilbert space, called the signal space. Geometric representations of the signals are often called signal constellations. Basic modulation methods prefer the use of orthogonal bases of the signal space as the signals themselves, since demodulation can be accomplished by projection onto these bases. For example, equidistant m-ary pulse-amplitude modulation (PAM) uses discrete amplitudes on each basis [1].

Signals experience distortion during transmission which is modeled probabilistically, mainly due to the addition of noise. The received signals are therefore no longer identical with the transmitted signals. The demodulation problem is that of mapping a received signal back to a message such that the probability of the demodulated message not equalling the original transmitted message is minimized. Under the assumption of additive white Gaussian noise, picking the message whose signal is closest to the received signal using the natural Euclidean distance metric is optimal (if noise is correlated, for example, a generalized metric needs to be used) [1]. This is referred to as maximum-likelihood (ML) decoding since it minimizes the message error probability.

However, ML decoding quickly becomes practically infeasible by the "curse of combinatorics," and other methods are needed to be considered. Shannon [2] showed that every transmission channel has a maximum possible transmission rate which it can support, called the Shannon capacity, and that there exist coding and decoding methods which can operate to within ϵ of this capacity at arbitrarily low error rates. Shannon's nonconstructive proofs did not require ML decoding, opening the door to possibly low-complexity capacity-achieving signaling methods. Unfortunately, to achieve capacity requires continuous input alphabets which is highly impractical. Discrete modulations,

mapped onto orthogonal bases, such as PAM, cannot achieve the Shannon capacity on the Gaussian channel. Certain high-dimensional discrete constellations, such as lattices, have been reported to achieve capacity, but in many ways their regular (discrete) structure is lost in the process [3].

In this paper we pursue another approach, abandoning the use of orthogonal bases as signals. In many practical situations the signals utilized are correlated, either by design, or by the effects that occur during transmission. An example of the former is (random) code-division multiple-access (CDMA) [4], and an example of the latter is multiple-antenna transmission (MIMO) [5]. In both cases the signals are densely correlated-which makes efficient demodulation extremely difficult. If the correlation pattern is sparse, that is, if any given signal waveform interferes with only a few other (neighboring) signal waveforms, sequence detection algorithms like the Viterbi algorithm can be used efficiently. A number of modulation methods based on superposition of individual data streams have been proposed (see [6–8]). When a number of independent signals add up in the channel, they can sometimes be decoded sequentially. Onion-peeling decoding starts from the largest power signal, decodes it treating the rest of the signals as noise and subtracts the result from the composite received signal. The decoding then continues analogously with the second strongest signal to the weakest. A number of methods based on successive decoding have been proposed and studied for various types of signals (data streams), including binary [9]. Channel capacity can be approached in the case when powers and rates of the signals follow specific precise arrangements, which is, however, challenging to accomplish in practice.

In this paper we assume a random correlation among the signals by postulating that these signals correspond to random vectors in signal space. The CDMA and MIMO channels are practical examples of such random channels [5, 10, 11]. Transmission relies on repeating the symbols of a message with random delays. Each time the symbol is modulated onto a new signal. While this increases the number of signals utilized, it allows for a very efficient iterative demodulation method to be used. This iterative demodulator forms the first stage of a two-stage receiver, where the second stage is a conventional forward error control (FEC) decoder for individual (binary) data streams, That is, the iterative first stage efficiently separates the correlated data streams. Specific adaptations of generalized modulation have recently been proposed for both CDMA [12] and MIMO channels [13].

Our contributions in this paper are two-fold. First we show that using random signals incurs no capacity loss, and furthermore, that regular-spaced PAM-type modulation on these random signals can achieve the Shannon capacity. We then discuss transmission using redundant signaling and an iterative demodulation method for which we show that it can operate close to the Shannon capacity over the entire range of operational interest. In showing this, we will only assume that we have capacity-achieving binary error control codes available, a very reasonable assumption given the current state-of-the art in error control coding [14].

2. Modulation

Generally, a discrete data stream $\mathbf{d}$ is mapped onto signals from a finite set of such signals according to some mapping rule. In the ubiquitous pulse-amplitude (PAM) modulation, a discrete amplitude x_r for each value of d_r from $\mathbf{d}$ is first selected then used to multiply the rth signal. Most commonly, one of 2^B amplitude levels is selected for each B-bit data symbol. In the case of 8-PAM, for example, with $B = 3$, the discrete equispaced amplitudes shown in Figure 1 are used on each signal. This signal constellation can be interpreted as the superposition of three simple binary constellations, where Bit 1 has 4 times the power of Bit 0, and Bit 2 has 16 times its power.

In general, any properly labeled 2^B-PAM modulation can be written as the superposition of B binary antipodal amplitudes, that is,

$$v = \sum_{j=0}^{B-1} 2^j b_j, \tag{1}$$

where $b_j \in \{-1, 1\}$. If an entire sequence $\mathbf{v}$ of 2^B-ary PAM symbols is considered, it may be viewed as the superposition of B binary modulated data streams with powers $P_0, 4P_0, 16P_0, \ldots, 4^{B-1}P_0$ on binary data streams which make up the PAM symbol sequence $\mathbf{v}$.

This viewpoint is quite productive in that it suggests a capacity-achieving demodulation method for large constellations based on cancellation. Consider the case where the highest-power uncanceled data stream considers all lower power data streams as noise. Its maximum rate is then given by the mutual information

$$C_j = I\left(b_j;\ y \mid b_j + 1, \ldots, b_{B-1}\right), \tag{2}$$

where y is the output of the channel. As long as the rate on the jth binary data stream $R_j < C_j$, it can be correctly decoded using a binary Shannon-capacity-achieving code.(While no class of binary codes with nonexponential decoding complexity exist which can provably achieve the capacity on a binary-input channel, codes which can achieve this capacity "practically" with "implementable" complexities have recently emerged from intense research. The most popular representatives are turbo codes and low-density parity-check codes. Both utilize iterative message passing decoding algorithms [14].) By virtue of (1), knowledge of $b_{j+1}, \ldots, b_{B-1}$ implies that these data streams can be canceled from the received signal, and C_j is the capacity of the jth binary data stream. This thought model leads to a successive decoding and canceling method which can achieve the mutual information rate

$$C_{\text{symmetric}} = I(v; y) = \sum_{j=0}^{B-1} C_j \tag{3}$$

by the chain rule of mutual information. $C_{\text{symmetric}}$ is of course not equal to the capacity of the additive white Gaussian noise channel $y = v+n$, since the input distribution of v is uniform, rather than being Gaussian distributed as

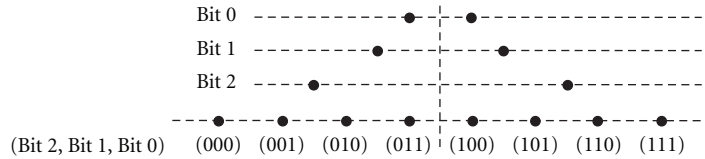


Figure 1: Pulse-amplitude modulation as superposition of binary antipodal modulation with geometric power distribution.

required to achieve the channel capacity. In fact, $C_{\text{symmetric}}$ loses the so-called shaping gain of 1.52 dB with respect to the capacity of the Gaussian channel [14].

3. Main Result

In this paper, we propose a generalized PAM modulation method which operates with random signals, rather than the orthogonal bases implied by the discussion in the previous section. We present a two-stage demodulator/decoder which remedies the difficulties of the onion-peeling method discussed above. Specifically, the demodulator consists of an iterative demodulator which operates in parallel and achieves a signal-to-noise ratio (SNR) improvement on each of the binary data streams. These are then decoded using external binary error control codes. The latency issue is basically confined to that of the parallel demodulator and that of the follow-up error control decoder. The iterative demodulator is based on cancelation. This means that its complexity, and that of the entire decoder scales linearly with the number of data streams.

Our main result is that we will prove that such an iterative demodulator/decoder can achieve a cumulative data rate $R_{\mathbf{d}}$ per dimension such that

$$R_{\mathbf{d}} \geq 0.995 C_{\text{GMAC}} - 0.54 \left[\frac{\text{bits}}{\text{dimension}}\right], \tag{4}$$

where C_{GMAC} is the Shannon capacity of the Gaussian multiple-access channel. That is, our system can achieve a fraction of 0.995 of channels information theoretic capacity, irrespective of system size.

In order for the iterative demodulator to function, we require that the number of signals in the signal space is increased, but not the power or spectral resources.

4. System Model

4.1. Signaling. We are considering communication of multiple data streams $\mathbf{d}_k$ using random signals $\mathbf{s}_k$ of dimension N. If N is sufficiently large, the number of useful signals is arbitrarily large. A set of K data symbols $d_{k,l}$ from the data streams $\mathbf{d}_k$ is transmitted at each time interval l. There are basically two ways to do this. Conventionally each symbol $d_{k,l}$ is directly modulated onto an individual signal $\mathbf{s}_{k,l}$. In this paper, however, we propose an alternative where we duplicate each symbol $d_{k,l}$ M-fold. These duplicates are then modulated onto separate signals $\mathbf{s}_{k,\pi_{k,m}(l)}$ at M random time intervals within a certain signal block, where $\pi_{k,m}(l)$ is the random location within the block where the mth copy of symbol $d_{k,l}$ is located. The function $\pi_{k,m}(l)$ is a permutation function with inverse $\pi_{k,m}^{-1}(l)$. Even though we have increased the number of signals by a factor M, scaling the power with M, and requiring that the signal set $\{\mathbf{s}_{k,\pi_{k,m}(l)}\}$ occupies the original N-dimensional signal space, this will not affect total power or the total spectrum utilization. (Another form of modulation based on randomly correlated signals called "partitioned transmission" has been recently proposed in [15]. Partition signalling creates redundancy and sparseness in transmitted data by partitioning K existing N-dimensional signal waveforms and permuting the resulting partitions. Generalized modulation relies on populating the signals space with additional N-dimensional signal waveforms. The latter gives an opportunity to create the requited level M of redundancy independently of signal dimensionality N. Further, near capacity operation with generalized modulation does not require $N \to \infty$.) A diagram of this modulator is given in Figure 2.

We make the convenient, but in no way necessary assumption that the channel is block-synchronous, that is, that the signal waveforms at time interval l interfere only within that time interval, and that there is no correlation of signal waveforms between time intervals. With this we can write the channel in the linear matrix form

$$\mathbf{y}_l = \mathbf{S}_l \mathbf{W}^{1/2} \mathbf{x}_l + \eta_l, \tag{5}$$

where the $N \times KM$ matrix $\mathbf{S}_l$ contains the signal vectors as columns. The capacity per dimension of this channel is well known [10] and is given by

$$C_{\mathbf{S}} = \frac{1}{2N} \log\det\left(\mathbf{I} + \frac{1}{\sigma^2}\mathbf{S}\mathbf{W}\mathbf{S}^T\right), \tag{6}$$

where

$$\mathbf{W} = \text{diag}\left(\frac{P_1}{M}, \frac{P_1}{M}, \ldots, \frac{P_1}{M}, \frac{P_2}{M}, \ldots, \frac{P_K}{M}\right) \tag{7}$$

is a $KM \times KM$ diagonal matrix with the powers used for transmission of the different signal vectors. We now assume that the signals $\mathbf{s}_{k,l}$ are chosen randomly from the signal space(the individual components $s_{k,l,n}$, $n = 1, \ldots, N$, of signals $\mathbf{s}_{k,l}$ can, for example, be selected randomly out of the set $\{-1/\sqrt{N}, 1/\sqrt{N}\}$ picking each entry with probability 1/2. However, other random selections satisfying (8) are also possible) such that the mutual pairwise expected correlation between signals is

$$\mathrm{E}\left[\mathbf{s}_{j,l}^* \mathbf{s}_{j',l}\right]^2 = \frac{1}{N}; \quad j \neq j'. \tag{8}$$

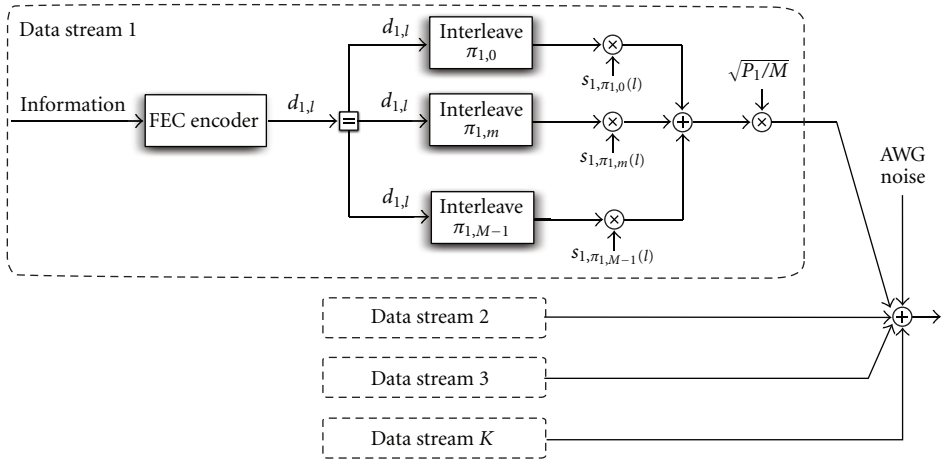


Figure 2: Modulator with superimposed binary data streams.

This model captures among others the random code-division multiple-access channel and the isotropic multiple-antenna channel model.

The capacity $C_{\overline{\mathbf{S}}}$ of this random vector channel is given by the expectation over $\mathbf{S}$ in (6). Using Jensen's inequality

$$C_{\overline{\mathbf{S}}} \leq \mathrm{E}_{\mathbf{S}}\left[\frac{1}{2N}\log\det\left(\mathbf{I}+\frac{1}{\sigma^2}\mathbf{S}\mathbf{S}^T\right)\right] \tag{9}$$

with equality if and only if $\mathbf{W} = \mathbf{I}$ (see [16]). That is to say that an equal distribution of powers over the signals maximizes the capacity of the random vector channel (5).

We now investigate the information theoretic impact of increasing the signal population as proposed by the M-fold duplication. The following lemma addresses this issue.

Lemma 1. *Keeping the transmit power* $\mathrm{tr}(\mathbf{W}) = P$ *constant, the capacity* $C_{\overline{\mathbf{S}}}$ *as a function of* K *and* N *approaches the capacity of the Gaussian multiple-access channel in the limit, that is,*

$$C_{\overline{\mathbf{S}}} \longrightarrow C_{\mathrm{GMAC}} = \frac{1}{2}\log\left(1+\frac{\sum_{k=1}^{K}P_k}{N\sigma^2}\right). \tag{10}$$

It approaches this limit from below as $M, K \to \infty$, *that is,* $C_{\overline{\mathbf{S}}} < C_{\mathrm{GMAC}}$ *for all* $K/N < \infty$.

Proof. See Appendix A. □

Lemma 1 reveals useful information in several ways. Firstly, it guarantees that the signaling strategy presented above, that is, the addition of extra random signal waveforms, incurs no capacity loss, and secondly, in the limit, arbitrary power assignments become capacity achieving, not only the equal power assignment.

4.2. Demodulation. The first stage of the demodulation process starts with matched filtering of the received signal with respect to each transmitted signal waveform in each time interval. Given the received signal embedded in Gaussian noise as

$$\mathbf{y}_r = \sum_{k=1}^{K}\sum_{m=0}^{M-1}\sqrt{\frac{P_k}{M}}d_{k,\pi_{k,m}^{-1}(r)}\mathbf{s}_{k,\pi_{k,m}^{-1}(r)} + \mathbf{n}_r, \tag{11}$$

these matched filter outputs are given by

$$\begin{aligned} z_{k,l} = \mathbf{s}_{k,l}^{*}\cdot\mathbf{y}_r &= \sum_{k'=1}^{K}\sum_{m'=0}^{M-1}\sqrt{\frac{P_k}{M}}d_{k',l'}\ \mathbf{s}_{k,l}^{*}\cdot\mathbf{s}_{k',l'} + n_r, \\ l = \pi_{k,m}^{-1}(r), &\qquad l' = \pi_{k',m'}^{-1}(r), \end{aligned} \tag{12}$$

where n_r is the sampled noise of variance σ^2, and $\mathbf{s}_{k,l}^{*}\cdot\mathbf{s}_{k',l'} = \rho_{m,m',l,l'}^{k,k'}$ is the correlation value between the target signal and a given interfering signal. The matched filter outputs in (12) consist of $d_{k,l}$ and an interference and noise term, which is given by

$$I_{k,m,r} = \sum_{\substack{k'=1\\k'\neq k}}^{K}\sum_{m'=0}^{M-1}\sqrt{\frac{P_{k'}}{M}}\rho_{m,m',l,l'}^{k,k'}d_{k',l'} + n_r. \tag{13}$$

At this point the graphical illustration shown in Figure 3 may prove helpful, which shows how the different symbols and signals combine to generate the sequence of received signal vectors $\mathbf{y}_r$. Note that in the interference equation (13) above, self-interference is not included. Apart from unnecessarily complicating the notation, this self-interference is negligible as shown below. Furthermore, in many cases it is not difficult to ensure that the signal vectors used for the different signals from a user k impinging on channel symbol $\mathbf{y}_r$ are orthogonal, that is,

$$\mathbf{s}_{k,\pi_{k,m}^{-1}(r)}^{*}\cdot\mathbf{s}_{k,\pi_{k,m'}^{-1}(r)} = 0, \quad \forall k, \tag{14}$$

and cause no self-interference. In [12], for example, different time intervals are used for the duplicate signals to accomplish this. The graphical representation reveals the similarity with graph-based error control codes, in particular with fountain

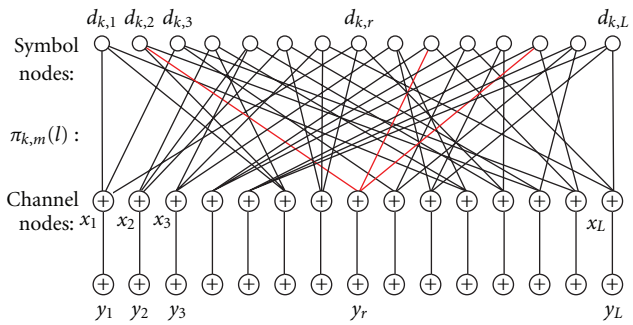


FIGURE 3: Signaling arrangement for the kth data stream. Symbols aggregating on the same channel node may use orthogonal signal waveforms. The actual received signal vectors $\mathbf{y}_r$ are the superposition of K such data streams, causing the channel nodes to have large message degrees.

codes [17]. Consequently, we will explore a demodulation method based on message passing.

Iterative demodulation follows the following message-passing principle. At the channel nodes, updated matched filter output signals are computed at each iteration by subtracting interference to generate

$$z_{k,l,m}^{(i)} = \mathbf{s}_{k,l}^{*} \cdot \left(\mathbf{y}_r - \sum_{\substack{k'=1 \\ (k' \neq k)}}^{K} \sum_{m'=0}^{M-1} \sqrt{\frac{P_{k'}}{M}} \widetilde{d}_{k',l',m'}^{(i-1)} \cdot \mathbf{s}_{k',l'} \right), \quad (15)$$

where $\widetilde{d}_{k',l',m'}$ is a soft symbol estimate of the m'th copy of $d_{k,l}$. Note that, following the extrinsic principle, the m different estimates for the same symbol are not necessarily identical (see below). These soft symbol estimates, in turn, are computed at the symbol nodes from the M matched filter signals for each copy of $d_{k,l}$. While $d_{k,l}$ can, in general, be any complex integer, we will concentrate on the basic binary case were $d_{k,l} \in \{-1, 1\}$. We will show later how to build larger modulation alphabets from this basic binary case using the binary decomposition of PAM signals.

In the binary case, the soft symbols are calculated as

$$\widetilde{d}_{k,l,m}^{(i)} = \tanh\left(\sum_{\substack{m'=0 \\ (m' \neq m)}}^{M-1} \sqrt{\frac{P_k}{M}} \frac{z_{k,l,m'}^{(i)}}{\sigma_{k,i}^2} \right) \quad (16)$$

which is the optimal *local* minimum-variance estimate of $d_{k,l}$ given that interference and noise combined form a Gaussian random variable with power $\sigma_{k,i}^2$. The variance of the symbol estimates (16) will be required in the analysis in Section 5. Defining this variance at iteration i as $\sigma_{d,k,i}^2 = \mathrm{E}|d_k - \widetilde{d}_{k,m}^{(i)}|^2$, and assuming that correlation between interference experienced by different replicas of the same symbol is negligible due to sufficiently large interleaving, it can be calculated adapting the development in [18] for CDMA as

$$\sigma_{d,k,i}^2 = \mathrm{E}\left(1 - \tanh\left(\mu + \sqrt{\mu}\xi\right)\right)^2 = g(\mu), \quad \forall i, \quad (17)$$

where $\xi \sim \mathcal{N}(0,1)$ and $\mu = (M-1)P_k/(M\sigma_i^2)$ from (16), and $\sigma_{k,i}^2 = \sigma_i^2$, for all k. The function $g(\mu)$ has no closed form, but the following bounds are quite tight [19]:

$$g(\mu) \leq 1(1+\mu);\ \mu < 1, \quad (18)$$

$$g(\mu) \leq \pi Q\left(\sqrt{\mu}\right);\ \mu \geq 1, \quad (19)$$

where $Q(\cdot)$ is the complementary error function. The final output signal after I iterations is $z_{k,l}^{(I)} = \sum_{m=0}^{M-1} z_{k,l,m}^{(I)}$, which is passed to binary error control decoders for data stream k. The final signal-to-noise/interference ratio (SINR) of $z_{k,l}^{(I)}$ is what primarily matters for the error performance of these error control decoders.

After I detection iterations of the first stage the data is passed to the second stage of demodulation. The second stage of the reception is the error control decoding which is executed for each of the K data streams individually. SINR for data stream k equals P_k/σ_I^2 and it can be argued that the residual noise and interference is Gaussian [15]. Ultimately the information rate (i.e., the rate of the error control code) of stream k should satisfy

$$R_k \leq C_{\mathrm{BIAWGN}}\left(\frac{P_k}{\sigma_I^2}\right) \quad (20)$$

for error-free decoding at the second stage. Here by $C_{\mathrm{BIAWGN}}(x)$ we denote the capacity of the binary-input real-valued output AWGN channel with SNR x.

5. Generalized Modulation

The discussion above treated the case of binary modulation on the different signal waveforms, however, as illustrated in Section 2, we can create the regular-spaced PAM modulations with geometrically scaled binary modulations using powers

$$P_0 4^b, \quad 0 \leq b \leq B-1. \quad (21)$$

We assume that there are K_b data streams with powers $P_0 4^b$. Thus, the total number of streams equals $K = \sum_{b=0}^{B-1} K_b$.

Assuming large enough interleavers, the evolution of the interference in this iterative demodulator can be captured with a standard density evolution analysis. Since the average correlation between signal waveforms $\mathrm{E}[(\rho_{\mathrm{m},m',l,l'}^{k,k'})^2] = 1/N$ (see (8)), the interference and noise on stream k is given by

$$\sigma_{k,i}^2 = \frac{1}{N} \sum_{\substack{k'=1 \\ (k' \neq k)}}^{K} P_{k'} \sigma_{d,k',i-1}^2 + \sigma^2 \leq \frac{1}{N} \sum_{k=1}^{K} P_k \sigma_{d,k,i-1}^2 + \sigma^2 = \sigma_i^2 \quad (22)$$

which is common to all streams. The upper bound in (22) contains the self-interference term for $k' = k$, which, however, becomes negligible as K and M grow. Using (17) in (22) and the PAM power distribution we obtain

$$\frac{\sigma_i^2}{P_0} = \sum_{b=0}^{B-1} K_b \frac{4^{b-1}}{N} g\left(\frac{M-1}{M} \frac{4^{b-1}}{\sigma_{i-1}^2/P_0}\right) + \frac{\sigma^2}{P_0}. \quad (23)$$

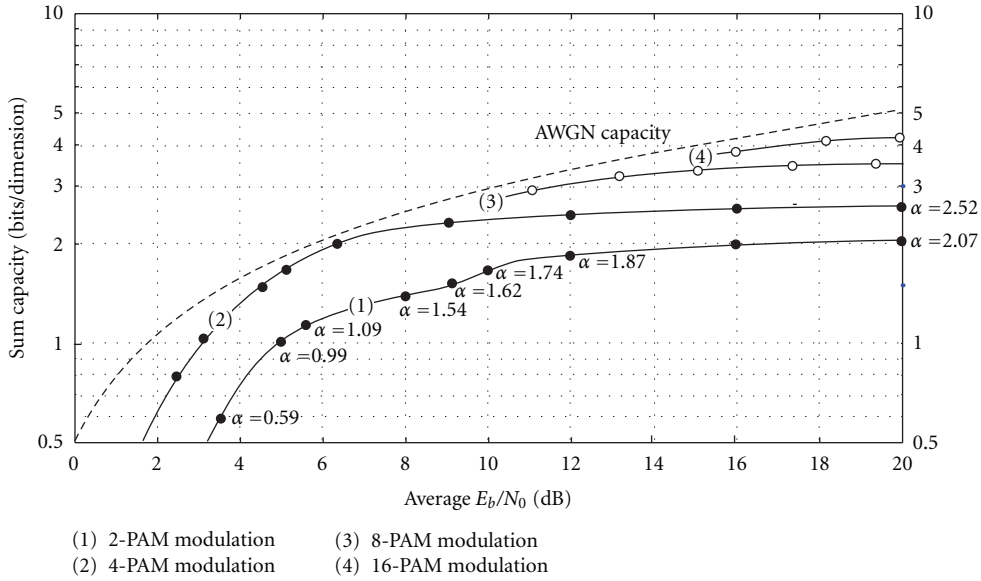


Figure 4: Achievable spectral efficiencies using iterative demodulation of various PAM constellations.

The next theorem proves that generalized PAM modulation used with the two-stage demodulation described above can closely approach the channel capacity.

Theorem 1. *Consider generalized* PAM *modulation* (21) *with* B *levels and* $K_b = 0.995N$ *data streams per level for* $b = 0, 1, \ldots, B-1$ *giving a total number of streams* $K = 0.995BN$. *One assumes that each data stream is encoded with a binary error control code which is capacity achieving on the binary-input* AWGN *channel, that is,*

$$R_k = C_{\text{BIAWGN}}\left(\frac{P_k}{\sigma_\infty^2}\right), \tag{24}$$

and let $M \to \infty$. *Then the resulting spectral efficiency per dimension*

$$R_d = \frac{1}{N}\sum_{k=1}^{K} R_k \geq 0.995 C_{\text{GMAC}} - 0.54. \tag{25}$$

Proof. See Appendix B. □

We note that for $B_1 < B_2$ the corresponding capacity approaching power profiles $P_0, P_0 4, \ldots, P_0 4^{B_1-1}$ and $P_0, P_0 4, \ldots, P_0 4^{B_2-1}$ coincide for $b \leq B_1 - 1$. The importance of these results is that new data streams can always be added without affecting decodability of the existing streams.

The gap between achieved spectral efficiency and the channel capacity can also be introduced in terms of average E_b/N_0, instead of data stream power profile. Average E_b/N_0 for the power profile used in Theorem 1 can be upper bounded as

$$\frac{E_b}{N_0} = \frac{1}{2R_d}\eta\sum_{b=0}^{B-1}\gamma_0 4^b \leq \frac{2\eta(4^B-1)}{3\eta(B-1+0.6706)} \tag{26}$$

from (B.14), and therefore the corresponding capacity of AWGN channel $C(E_b/N_0)$, using

$$\frac{E_b}{N_0} = \frac{2^{2C(E_b/N_0)}-1}{2C(E_b/N_0)} \tag{27}$$

can be upper bounded as

$$C(E_b/N_0) \leq B + 0.76. \tag{28}$$

As a result we obtain

$$\eta C\left(\frac{E_b}{N_0}\right) - R_d\left(\frac{E_b}{N_0}\right) \leq 1.09. \tag{29}$$

In Figure 4 we plot the achievable spectral efficiencies for the proposed generalized PAM modulation (21) for $B = 1, 2, 3, 4$ levels and assume ideal posterror control decoding with rates satisfying (24). Such performance can be closely approached with appropriate standard error control codes, which are very well developed for the binary case [20, 21]. We assume that $K_b = \alpha N/B$, for $b = 0, 1, 2, \ldots, B-1$, where parameter $\alpha \in (0, \infty)$. Each curve corresponds to fixed B and plotted as a function of average E_b/N_0 which is in turn the function of α. We can observe that spectral efficiency of generalized PAM modulation exceeds the capacity of the same PAM modulation using orthogonal waveforms. This is because the number of allowable correlated signal waveforms K exceeds the number of available orthogonal dimensions N. This advantage is most noticeable for 2-PAM, where the maximum achievable capacity of 2.08 is more than twice the number of orthogonal dimensions. For higher PAM constellations, the capacity per level $\alpha_b = K_b/N = \alpha/B \to 1$ rapidly from above. Note that for $\alpha = \eta = 0.995$ the gap between the performance curves and the capacity curve satisfies (29). Specifically, point $\alpha = \eta$ for $B = 1$ gives $E_b/N_0 = 4.72$ dB, for $B = 2$ gives $E_b/N_0 = 7.74$ dB, for $B = 3$ gives $E_b/N_0 = 11.94$ dB, and for $B = 4$ gives $E_b/N_0 = 16.63$ dB.

6. Conclusions

We have presented and analyzed a two-stage iterative demodulation methodology for generalized PAM constellations using correlated random signals rather than the usual orthogonal bases. The method operates by introducing redundant duplicate copies of the data symbols modulated onto extra signals. An exponential power distribution, inherently present in PAM modulations, allows this two-stage iterative demodulator to achieve 99.5% of the Shannon capacity using binary capacity-achieving error control codes for each data stream. This generalized PAM modulation format was shown to approach the channel capacity over a wide range of operating SNRs, and can exceed the capacity of traditional PAM constellations on orthogonal signals.

Appendices

A. Proof of Lemma 1

Decompose the argument of (6) as

$$\begin{aligned}\det\left(\mathbf{I}+\frac{1}{\sigma^2}\mathbf{S}\mathbf{W}\mathbf{S}^T\right) &= \det\left(\left(1+\frac{\sum_{k=1}^{K}P_k}{N\sigma^2}\right)\mathbf{I}+\mathbf{B}\right)\\ &= \left(1+\frac{\sum_{k=1}^{K}P_k}{N\sigma^2}\right)^N \det(\mathbf{I}+\mathbf{F}),\end{aligned} \tag{A.1}$$

where $\mathbf{F}$ is the matrix of the off-diagonal elements with

$$F_{ij} = \kappa\sum_{m=1}^{M}\sum_{k=1}^{K}\frac{P_k}{M}b_{m,k}, \tag{A.2}$$

where $\kappa = \left(\sigma^2+\sum_{k=1}^{K}P_k\right)^{-1}$ and $b_{m,k}\in\{\pm 1\}$ with uniform probabilities. The entries B_{ij} have zero mean and variance

$$\mathrm{var}\left(F_{ij}\right) = \frac{1}{M^2}\frac{\sum_{k=1}^{K}P_k^2}{\left(\sum_{k=1}^{K}P_k\right)^2}, \tag{A.3}$$

which vanishes as (i) $M\to\infty$ or (ii) $K\to\infty$. Condition (ii), however, requires the Lindeberg condition to hold on the set $\{P_k\}$.

Using (i), or (ii), the elements in $\mathbf{F}$ are sufficiently small to apply Jacobi's formula, that is,

$$\det(\mathbf{I}+\mathbf{F}) = (1-\mathrm{tr}(\mathbf{F}))+O\left(F_{ij}^2\right). \tag{A.4}$$

Since $\mathrm{tr}(\mathbf{F})=0$, and the second moment of F_{ij} vanishes, the limit value of the Lemma is proven.

Using Hadamard's inequality it is straightforward to show that

$$C_{\overline{\mathbf{S}}} < C_{\mathrm{GMAC}}, \tag{A.5}$$

and the limit is approached from below. While $\det(\mathbf{I}+\mathbf{F})\to 1$ in probability, $C_{\overline{\mathbf{S}}}\to C_{\mathrm{GMAC}}$ almost surely.

B. Proof of Theorem 1

Let us define $\gamma = P_0/\sigma_i^2$, $\gamma' = P_0/\sigma^2$, and $\gamma_\infty = P_0/\sigma_\infty^2$. Consider $B=\infty$ here for simplicity and define $\eta = 0.995$. Convergence defined by (23) (for σ_i^2, $i=0,1,\ldots$) follows from

$$1 > \sum_{b=0}^{\infty}\eta\gamma 4^b g\left(\gamma 4^b\right)+\frac{\gamma}{\gamma'},\quad \text{for } \gamma\in(0,\gamma_\infty], \tag{B.1}$$

and $K_b=\eta N$. Success of the demodulation stage happens if γ_∞ is close to γ'. This means that the interstream interference is canceled almost entirely. Let us choose a somewhat arbitrary lowest power P_0 such that $\gamma'=4$ and prove that $\gamma_\infty > 1.79$.

Let us define the functions

$$t(x) = xg(x), \tag{B.2}$$

$$f(\gamma) = \sum_{b=0}^{\infty}\eta t\left(\gamma 4^b\right)+\frac{\gamma}{\gamma'} = \sum_{b=0}^{\infty}\eta t\left(\gamma 4^b\right)+\frac{\gamma}{4}. \tag{B.3}$$

The function $t(x)$ monotonically increases for $0<x<x_0$ and monotonically decreases for $x>x_0$, where $x_0\approx 1.508$. To find an upper bound on $f(\cdot)$, we consider the terms $t(\gamma 4^b)$ for very small and very large arguments separately, that is,

$$\begin{aligned}f(\gamma) &= \eta\sum_{b=0}^{\infty}t\left(\gamma 4^b\right)+\frac{\gamma}{4}\\ &= \eta\sum_{b\ \text{s.t.}\,\gamma 4^b<A_1}t\left(\gamma 4^b\right)+\eta\sum_{b\ \text{s.t.}\,\gamma 4^b>A_2}t\left(\gamma 4^b\right)\\ &\quad+\eta\sum_{b\ \text{s.t.}\,A_1\le\gamma 4^b\le A_2}t\left(\gamma 4^b\right)+\frac{\gamma}{4}.\end{aligned} \tag{B.4}$$

Using the fact that $g(x)\le 1$ for any x we obtain for any $b_1\ge 0$

$$\sum_{b=0}^{b_1}t\left(\gamma 4^b\right) \le \gamma\sum_{b=0}^{b_1}4^b = \frac{\gamma\left(4^{b_1+1}-1\right)}{3} < \frac{\gamma 4^{b_1+1}}{3}. \tag{B.5}$$

From (19) we obtain

$$\begin{aligned}\sum_{b=b_2}^{\infty}t\left(\gamma 4^b\right) &\le \gamma\pi\sum_{b=b_2}^{\infty}4^b Q\left(2^b\sqrt{\gamma}\right)\\ &\le \sqrt{\frac{\gamma\pi}{2}}\sum_{b=b_2}^{\infty}2^b e^{-\gamma 4^b/2} \le \sqrt{\frac{\gamma\pi}{2}}\frac{2^{b_2}e^{-\gamma 4^{b_2}/2}}{\left(1-e^{-3\gamma 4^{b_2}/2}\right)}\end{aligned} \tag{B.6}$$

for b_2 such that $\gamma 4^{b_2}>1$. The last inequality in (B.6) is computed by upper bounding the sum by the geometrical progression using the inequality

$$\frac{t\left(\gamma 4^{b+1}\right)}{t\left(\gamma 4^b\right)} = 2e^{-3\gamma 4^b/2} \le e^{-3\gamma 4^{b_2}/2}. \tag{B.7}$$

Using (B.5) we get for any A_1

$$\sum_{b\ \text{s.t.}\,\gamma 4^b<A_1}t\left(\gamma 4^b\right) < \frac{4A_1}{3}. \tag{B.8}$$

Analogously, (B.6) gives

$$\sum_{b\ \text{s.t.}\gamma 4^b > A_2} t\left(\gamma 4^b\right) \le \sqrt{\frac{\pi A_2}{2}} \frac{e^{-A_2/2}}{\left(1 - 2e^{-3A_2/2}\right)}. \tag{B.9}$$

By choosing $A_1 = 0.00003$ and $A_4 = 24$, we compute numeric values of the bounds (B.8) and (B.9) for the tails as

$$\sum_{b\ \text{s.t.}\gamma 4^b < A_1} t\left(\gamma 4^b\right) < 4 \cdot 10^{-5}, \tag{B.10}$$

$$\sum_{b\ \text{s.t.}\gamma 4^b > A_2} t\left(\gamma 4^b\right) < 4 \cdot 10^{-5}. \tag{B.11}$$

Let us define

$$\overline{f}_{10}(\gamma) = 8 \cdot 10^{-5} + \eta \sum_{b=0}^{9} t\left(\gamma 4^b\right) + \frac{\gamma}{4}. \tag{B.12}$$

It follows from (B.4), (B.10), and (B.11) that for any γ

$$f(\gamma) < \overline{f}_{10}(\gamma). \tag{B.13}$$

We also notice that it is enough to consider $\gamma \in [A_1, \gamma']$. Numerical calculation shows that the only root $\overline{\gamma}$ of $\overline{f}_{10}(\gamma) - 1$ on the interval $\gamma \in [A_1, 4]$ equals 1.79374. Thus, $\gamma_\infty \ge \overline{\gamma} = 1.79374$ due to (B.13).

We calculate the spectral efficiency (or sum-rate per dimension) as follows:

$$\begin{aligned} R_{\mathrm{d}} &= \sum_{b=0}^{B-1} \eta C_{\mathrm{BIAWGN}}\left(\gamma_\infty 4^b\right) \\ &= \eta(0.6859 + 0.9835 + B - 2 - \epsilon) \ge \eta(B - 1 + 0.6706), \end{aligned} \tag{B.14}$$

where we use a bound from [15]

$$1 - C_{\mathrm{BIAWGN}}(x) \le \frac{2\pi^{3/2}}{\ln 2(\pi^2 - 8)} e^{-1/2x} < 10e^{-1/2x} \tag{B.15}$$

to upper bound ϵ as

$$\begin{aligned} \epsilon &= \sum_{b=2}^{B-1} \eta\left(1 - C_{\mathrm{BIAWGN}}\left(\gamma_\infty 4^b\right)\right) \\ &\le \sum_{b=2}^{B-1} 10e^{-(\gamma_\infty 4^b)/2} \le 10^{-5}, \quad \text{for any } B. \end{aligned} \tag{B.16}$$

The capacity of the additive Gaussian channel corresponding to power profile (21) with $K_b = \eta N$ streams per level can be calculated as follows

$$C_{\mathrm{GMAC}} = \frac{1}{2}\log_2\left(1 + \eta \sum_{b=0}^{B-1} \gamma_0 4^b\right) = \frac{1}{2}\log_2\left(1 + 4\eta \sum_{b=0}^{B-1} 4^b\right), \tag{B.17}$$

$$= \frac{1}{2}\log_2\left(\frac{1 + 4\eta(4^B - 1)}{3}\right) \le B - 1 + 1.21. \tag{B.18}$$

Combining (B.18) and (B.14), we obtain (25).

References

[1] J. Proakis, *Digital Communications*, McGraw-Hill, New York, NY, USA, 4th edition, 2001.

[2] C. E. Shannon, "A Mathematical Theory of Communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, 1948.

[3] H.-A. Loeliger, "Averaging bounds for lattices and linear codes," *IEEE Transactions on Information Theory*, vol. 43, no. 6, pp. 1767–1773, 1997.

[4] A. J. Viterbi, *CDMA: Principles of Spread Spectrum Communication*, Addison-Wesley, New York, NY, USA, 1995.

[5] G. J. Foschini, "Layered space-time architecture for wireless communication in a fading environment when using multi-element antennas," *Bell Labs Technical Journal*, vol. 1, no. 2, pp. 41–59, 1996.

[6] L. Ping, J. Tong, X. Yuan, and Q. Guo, "Superposition coded modulation and iterative linear MMSE detection," *IEEE Journal on Selected Areas in Communications*, vol. 27, no. 6, pp. 995–1004, 2009.

[7] T. Wo and P. Hoeher, "Superposition mapping with application to bit-interleaved coded modulation," in *Proceedings of the 8th International ITG Conference on Source and Channel Coding (SCC '10)*, January 2010.

[8] T. Wo and P. Hoeher, "Iterative processing for superposition mapping," This Special Issue, 2010.

[9] L. Duan, B. Rimoldi, and R. Urbanke, "Approaching the AWGN channel capacity without active shaping," in *Proceedings of IEEE International Symposium on Information Theory (ISIT '97)*, p. 374, Ulm, Germany, July 1997.

[10] C. Schlegel and A. Grant, *Coordinated Multiple User Communications*, Springer, Berlin, Germany, 2006.

[11] E. Telatar, "Capacity of multi-antenna Gaussian channels," *European Transactions on Telecommunications*, vol. 10, no. 6, pp. 585–595, 1999.

[12] C. Schlegel, "CDMA with partitioned spreading," *IEEE Communications Letters*, vol. 11, no. 12, pp. 913–915, 2007.

[13] Z. Bagley, C. Schlegel, D. Truhachev, and L. Krzymien, "Partitioned-mapping for variable rank MIMO channels," in *Proceedings of the Allerton Conference*, September 2006.

[14] C. Schlegel and L. Perez, *Trellis and Turbo Coding*, IEEE/Wiley, New York, NY, USA, 2004.

[15] D. Truhachev, C. Schlegel, and L. Krzymien, "A two-stage capacity-achieving demodulation/decoding method for random matrix channels," *IEEE Transactions on Information Theory*, vol. 55, no. 1, pp. 136–146, 2009.

[16] E. Biglieri and G. Taricco, *Transmission and Reception with Multiple Antennas: Theoretical Foundations*, Now Publishers, Hanover, Mass, USA, 2004.

[17] J. W. Byers, M. Luby, and M. Mitzenmacher, "A digital fountain approach to asynchronous reliable multicast," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 8, pp. 1528–1540, 2002.

[18] C. Schlegel, Z. Shi, and M. Burnashev, "Optimal power/rate allocation and code selection for iterative joint detection of coded random CDMA," *IEEE Transactions on Information Theory*, vol. 52, no. 9, pp. 4286–4294, 2006.

[19] M. Burnashev, C. Schlegel, W. Krzymien, and Z. Shi, "Characteristics analysis of successive interference cancellation methods," *Problemy Peredachi Informatsii*, vol. 40, no. 4, pp. 297–317, 2004.

[20] R. G. Gallager, *Low-Density Parity-Check Codes*, MIT Press, Cambridge, Mass, USA, 1963.

[21] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: turbo-codes," *IEEE Transactions on Communications*, vol. 44, no. 9, pp. 1261–1271, 1996.

Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2010, Article ID 706464, 13 pages
doi:10.1155/2010/706464

Research Article

Iterative Processing for Superposition Mapping

Tianbin Wo, Meelis Noemm, Dapeng Hao, and Peter Adam Hoeher

Information and Coding Theory Laboratory, University of Kiel, Kaiserstrasse 2, 24143 Kiel, Germany

Correspondence should be addressed to Tianbin Wo, wtb@tf.uni-kiel.de

Received 12 March 2010; Accepted 27 June 2010

Academic Editor: Christian Schlegel

Superposition mapping (SM) is a modulation technique which loads bit tuples onto data symbols simply via linear superposition. Since the resulting data symbols are often Gaussian-like, SM has a good theoretical potential to approach the capacity of Gaussian channels. On the other hand, the symbol constellation is typically nonbijective and its characteristic is very different from that of conventional mapping schemes like QAM or PSK. As a result, its behavior is also quite different from conventional mapping schemes, particularly when applied in the framework of bit-interleaved coded modulation. In this paper, a comprehensive analysis is provided for SM, with particular focus on aspects related to iterative processing.

1. Introduction

According to Shannon's information theory, the capacity of a Gaussian channel corresponds to the maximum mutual information between channel inputs and outputs [1, 2]. Given a power constraint, this maximum can be achieved if and only if the channel inputs are Gaussian distributed. Since Shannon's seminal work in 1948, many approaches have been proposed to map binary bits onto Gaussian-like distributed symbols. Among these approaches, the most well-known ones are Huffman decoding [3] and signal shaping [4]. By using a Huffman decoder to map bits onto a signal constellation with a well-designed distribution, for example, the Maxwell-Boltzmann distribution [5], one can maximize the mutual information between channel input and output for a given symbol cardinality and average power. Unfortunately, this approach demands variable-rate transmission and is consequently undesirable for practical applications. In contrast, signal shaping techniques are fixed-rate transmission schemes and for this reason have attracted interest in the field of real-world implementations. There are two popular methods for signal shaping, namely trellis shaping [6] and shell mapping [7, 8]. Both trellis shaping and shell mapping share a common idea, that is to construct a high-dimensional uniform constellation which results in low-dimensional nonuniform constituent constellations. They are both able to deliver a shaping gain of about 1.0 dB [9] without any additional effort with respect to channel coding, but subject to the assumption of noniterative demapping and decoding. In case of iterative soft-in soft-out (SISO) demapping and decoding, the block-wise mapping manner inherent in popular shaping techniques presents a bottleneck to the design of receiver algorithms.

To overcome the drawbacks of Huffman decoding and signal shaping, several researchers proposed different transmission schemes employing linear superposition to generate Gaussian-like symbols [10–12]. Without loss of generality, we denote the key component of such techniques as *superposition mapping* (SM). The characteristic feature of SM is that the conversion from binary bits to data symbols is done by a certain form of superposition instead of bijective (one-to-one) mapping. Since the output of a superposition mapper is often Gaussian-like, the necessity of doing active signal shaping is eliminated. On the other hand, superimposed signal components interfere with each other, and the resulting relationship between bit tuples and data symbols is often nonbijective. To guarantee a perfect reconstruction at the receiver side, channel coding and interleaving are typically mandatory, and iterations between the decoder and demapper are essential. This incurs a structure well known as bit-interleaved coded modulation (BICM) [13]. For an easy reference, we may call such a transmission technique bit-interleaved coded modulation with superposition mapping (BICM-SM).

In this paper, we provide a comprehensive study on the performance of BICM-SM, with particular focus on aspects related to iterative demapping and decoding. Theoretical as well as numerical results are provided. Via mutual information analysis, the capacity-achieving potential of superposition mapping is demonstrated. Due to the superimposed signal structure and the nonuniform signal distribution, the requirements of a superposition mapper on the channel code are essentially different from that of a uniform bijective mapper like PSK/QAM. Through EXIT chart analysis, we will show that repetition coding is in fact superior for coded SM systems. In addition to these analytical investigations, two practical issues are also considered. A novel tree-pruning concept is proposed to reduce the demapping complexity, and a baseband clipping scheme together with an iterative soft compensation mechanism is presented to reduce the signal peak-to-average power ratio (PAPR), while keeping the system performance degradation at an acceptable level.

The remainder of this paper is organized as follows. Section 2 introduces the Gaussian channel model under investigation. Section 3 provides a brief description of superposition mapping and related information theoretical aspects. Section 4 treats the application of superposition mapping in the framework of bit-interleaved coded modulation. In Section 5, an extensive study is carried out on iterative demapping and decoding process of coded SM systems. Particular focus is put on the convergence behavior of the iterative receiver given different channel codes. In Section 6, several approaches for low-complexity demapping are discussed, and in Section 7 a novel method for PAPR control is proposed. Finally, Section 8 summarizes the paper and suggests some interesting topics for future work.

2. Gaussian Channel

The additive white Gaussian noise (AWGN) channel is perhaps the most important channel model with continuous outputs. Though simple, it models the fundamental effects of communication in a noisy environment. The discrete-time complex AWGN channel model can be written as

$$y[k] = x[k] + z[k], \quad z[k] \sim \mathcal{CN}(0, \sigma_z^2), \tag{1}$$

where k is the time index. The mutual information between the channel input and output is given by

$$I(x; y) = h(y) - h(y \mid x) = h(y) - h(z), \tag{2}$$

where $h(\cdot)$ denotes differential entropy. Since the normal distribution maximizes the entropy for a given variance, the maximum of $I(x; y)$ is achieved when y is Gaussian distributed. Strictly speaking, y can only be Gaussian if x is Gaussian. In practice, however, it often suffices if x is Gaussian-like, given a reasonable signal-to-noise ratio (SNR).

3. Superposition Mapping

Figure 1 shows the general structure of superposition mapping. After serial-to-parallel (S/P) conversion, each code bit

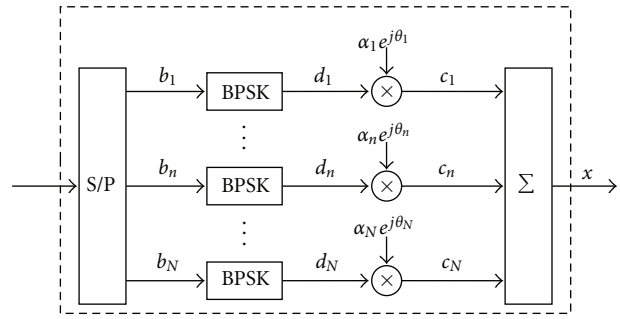


Figure 1: Superposition mapping (SM).

b_n is first converted into a binary antipodal symbol d_n. (Due to superposition mapping, binary antipodal component symbols bring no limit with respect to the overall bandwidth efficiency.) Then, power allocation and phase allocation are performed. Afterwards, the complex component symbols c_n (called "chips") are linearly superimposed to create a finite-alphabet data symbol. SM can be described by the following equation:

$$x = \sum_{n=1}^{N} c_n = \sum_{n=1}^{N} d_n \alpha_n e^{j\theta_n}, \quad d_n \in \{\pm 1\}, \tag{3}$$

where N is called the bit load of the superposition mapper. By tuning the amplitudes α_n and the phases θ_n, different superposition mapping schemes can be obtained. In case of equal power allocation, that is, when all amplitudes α_n are the same, all bits b_n are equally protected. It was shown in [14] that, given a BICM system with standard coding approaches, direct superposition mapping (DSM) with equal power allocation suffers from limited supportable bandwidth efficiency, and DSM with unequal power allocation suffers from power efficiency degradation. In contrast, phase-shifted superposition mapping (PSM) provides virtually unlimited supportable bandwidth efficiency and desirable power efficiency. For this reason, this paper will exclusively focus on PSM, while most of the analysis can easily be extended to the case of DSM.

3.1. Phase-Shifted Superposition Mapping. PSM is characterized by the following power and phase allocation:

$$\begin{aligned} \alpha_n &= \alpha \quad \text{for } 1 \le n \le N, \\ \theta_n &= \frac{\pi(n-1)}{N} \quad \text{for } 1 \le n \le N. \end{aligned} \tag{4}$$

All chips are allocated with the same magnitude, and each chip is allocated with a unique phase uniformly drawn over the interval $[0, \pi)$. Substituting (4) into (3), we obtain

$$\begin{aligned} \mathrm{Re}\{x\} &= \sum_{n=1}^{N} \mathrm{Re}\{c_n\} = \sum_{n=1}^{N} \alpha d_n \cos\left(\frac{\pi(n-1)}{N}\right), \\ \mathrm{Im}\{x\} &= \sum_{n=1}^{N} \mathrm{Im}\{c_n\} = \sum_{n=1}^{N} \alpha d_n \sin\left(\frac{\pi(n-1)}{N}\right). \end{aligned} \tag{5}$$

Since $|c_n|^2 \equiv \alpha^2$, the total energy spent for each bit is constant. Hence, PSM is characterized by a high power efficiency provided by superposition mapping with equal power allocation. On the other hand, from (5) one can see that unequal power allocation is actually done in the real and imaginary dimension, respectively, due to the individual phase shift. This increases the cardinality of the output symbol and greatly enhances the supportable bandwidth efficiency. As a consequence, PSM does not suffer from limited supportable bandwidth efficiency.

Figure 2 illustrates the PSM constellation for different bit loads, and Figure 3 shows the probability distribution of one quadrature component of the PSM outputs. For $N = 4$, the PSM constellation looks like circular 16-QAM, while for larger N, the constellation points tend to be geometrically Gaussian-like distributed. (The central limit theorem cannot be applied here, since $\{c_1, \ldots, c_N\}$ are independent but not identically distributed due to individual phase shifts.) It can be easily proven that the real part and the imaginary part of x are statistically independent. Therefore, the constellation points of PSM are approximately circular Gaussian distributed, though in no case perfect.

3.2. An Information Theoretical View. Given the distribution of x, the mutual information (MI) $I(x; y)$ can be numerically evaluated. Figure 4 provides MI curves for PSM as well as for square QAM. Comparing Figure 4(a) with Figure 4(b), it can be seen that PSM outperforms QAM in the left region, and it is almost capacity achieving, which is due to the Gaussian-like symbol distribution. The gap between the capacity curve and the slope for high-order QAM in Figure 4(b) corresponds to the ultimate shaping gain [9]. On the other hand, PSM is worse than QAM in the right region, which is due to a smaller minimum distance between distinct constellation points. Last but not least, in most cases, PSM and QAM provide roughly the same entropy per symbol, given an identical bit load N, which can be seen by comparing the smooth sections of the MI curves. An important message from the above observations is that, with PSM, signal shaping is no longer necessary to approach the capacity of the Gaussian channel. As a matter of fact, the mutual information given PSM input symbols can be arbitrarily close to the capacity at any SNR, as long as the bit load N is large enough.

3.3. Rate Limit of PSM. Following (1), the limit of coding rate for PSM transmission over the AWGN channel is given by

$$R \leqslant \frac{I(x;y)}{N} \leqslant \frac{H(x)}{N} \lessapprox 1. \tag{6}$$

Hence, the bandwidth efficiency of PSM is virtually unlimited and the required spreading factor $(1/R)$ should not be very large. The bandwidth efficiency limit at about 2 bits/symbol per signal dimension reported in [15] is in fact due to the adopted Gaussian-approximation-based demapping algorithm rather than the mapping scheme itself.

4. Bit-Interleaved Coded Modulation with Superposition Mapping

Bit-interleaved coded modulation [13] is widely recognized as a promising technique to approach the channel capacity at high SNRs. Typically, higher-order QAM is used to achieve a high bandwidth efficiency, while an interleaver is placed between the encoder and the signal mapper to exploit the bit diversity of QAM. With the same structure but replacing QAM with superposition mapping (SM), we obtain the BICM-SM transmission scheme under investigation.

4.1. General Structure. Figure 5 shows the transmitter and receiver structure of BICM-SM, where SD stands for superposition demapping. Since a superposition mapper is often a nonbijective (many-to-one) mapper, iterations between the demapper and the decoder are not only necessary but mandatory. A superposition demapper usually can not work properly without utilizing the redundancy introduced by the channel code. This is quite different in comparison with the demapper of conventional mapping schemes, like a PSK/QAM demapper, which does not necessarily have to cooperate with the channel decoder.

4.2. Soft-In Soft-Out Demapping. The superposition demapper needs to provide soft decisions of each code bit given the channel outputs and the a priori information from the decoder. Given an AWGN channel, an APP soft-input soft-output (SISO) demapper for SM can be described as follows. Let

$$\mathbf{b}_{\sim n} \doteq [b_1, b_2, \ldots, b_{n-1}, b_{n+1}, \ldots, b_N] \tag{7}$$

collect the bits excluding b_n loaded on one symbol, we have

$$\begin{aligned} \mathrm{LLR}(b_n) &\doteq \ln \frac{p(y \mid b_n = 0)}{p(y \mid b_n = 1)} \\ &= \ln \frac{\sum_{\mathbf{b}_{\sim n}} p(y \mid b_n = 0, \mathbf{b}_{\sim n}) P(\mathbf{b}_{\sim n})}{\sum_{\mathbf{b}_{\sim n}} p(y \mid b_n = 1, \mathbf{b}_{\sim n}) P(\mathbf{b}_{\sim n})}, \end{aligned} \tag{8}$$

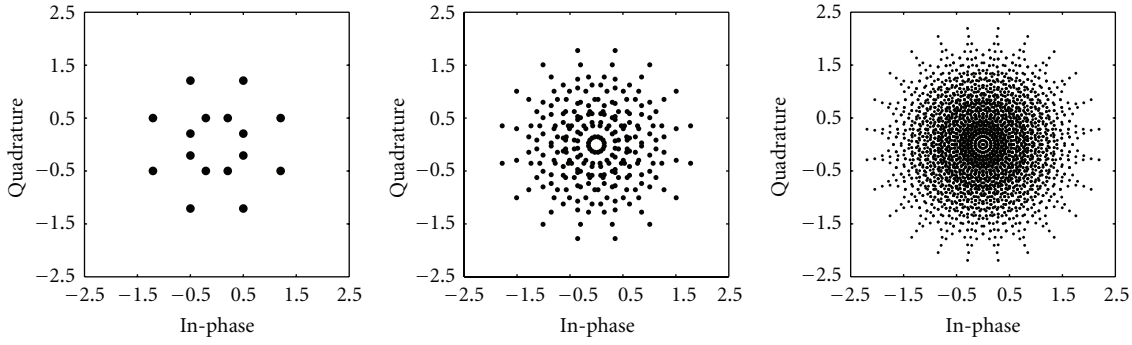
where

$$P(\mathbf{b}_{\sim n}) = \prod_{i=1, i \neq n}^{N} P(b_i), \tag{9}$$

assuming statistical independence of interleaved code bits. The APP demapping algorithm, in spite of its high complexity, is the first choice for theoretical analyses, as it provides the best performance. In the following measurements, APP demapping is applied, if not specifically addressed.

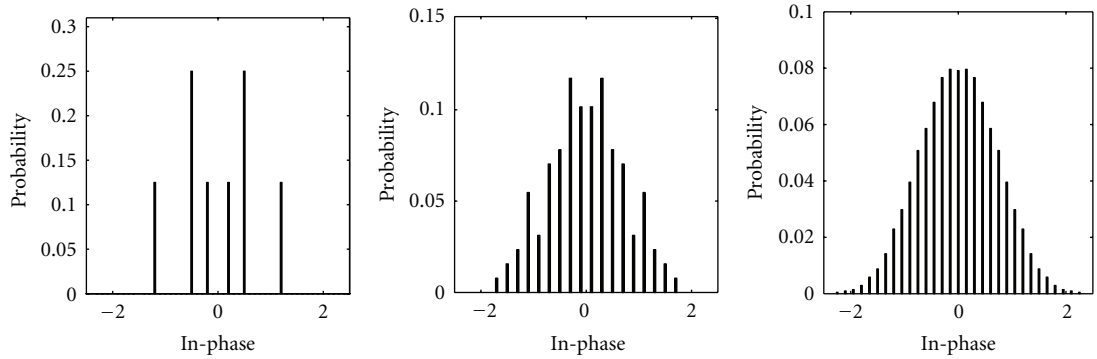
5. Iterative Processing for Superposition Mapping

Due to the geometrically nonuniform symbol distribution, we observe that PSM is better than square QAM in the sense of mutual information over the Gaussian channel. As long as the bit load N is large enough, PSM can efficiently



(a) $N = 4$ (b) $N = 8$ (c) $N = 12$

Figure 2: Constellation diagrams for PSM, $\alpha = \sqrt{1/N}$.



(a) $N = 4$ (b) $N = 8$ (c) $N = 12$

Figure 3: Distribution of $\mathrm{Re}\{x\}$ for PSM, $\alpha = \sqrt{1/N}$.

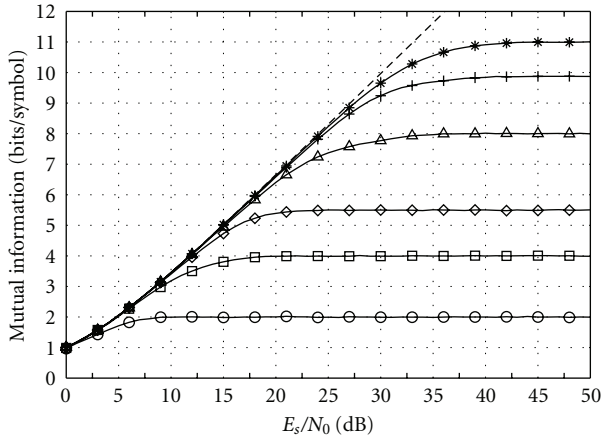
fill up the 1.53 dB gap, well known as the ultimate shaping gain in the terminology of signal shaping, between the channel capacity and the mutual information with uniform signalling. Nevertheless, due to linear superposition with equal power allocation, the behavior of a PSM demapper, in the scenario of iterative demapping and decoding is very different from that of a demapper of conventional mapping schemes like PSK/QAM. This in fact implies some different concepts for the code design. In this section, we will have an in-depth study on the iterative demapping and decoding process of coded PSM systems via an EXIT chart analysis [16, 17].

5.1. First Impression on the Performance. To start, let us first have a look at the performance of coded PSM. We consider two type of codes, low-density parity check (LDPC) codes, which are known to be capacity-achieving for binary-input Gaussian channels, and repetition codes, considered to be weak codes as they provide no coding gain at all on the AWGN channel. As shown in Figure 6, the true situation is in fact opposite to the common understanding, that is, repetition-coded PSM can even outperform LDPC-coded PSM, particularly at large bit loads N. Another noteworthy phenomenon is that the performance of LDPC-coded PSM degrades severely as the bit load N increases. All these observations indicate that the classical purely parity-check-based channel codes do not really fit with a superposition mapper, which is to be studied in the next sections.

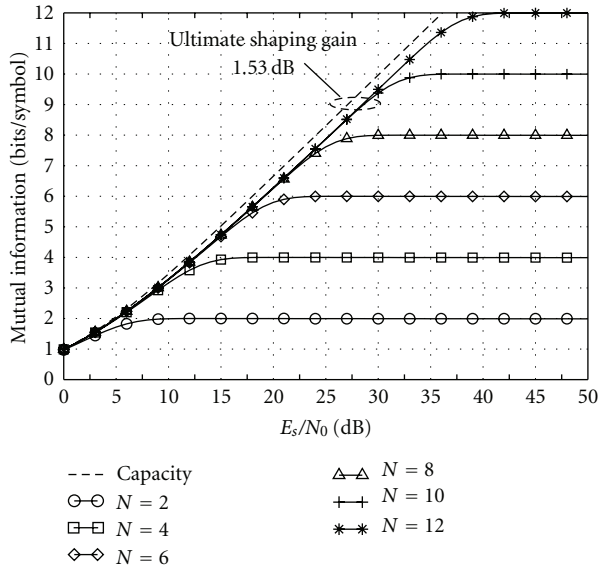
5.2. MI Transfer Characteristics of APP Demapper. To start an analysis on PSM, we will have a look at the mutual information (MI) transfer characteristics of the SISO APP demapper. Square QAM with Gray labeling will be used as reference for comparison.

Let us begin the discussion with the starting points in the left side of Figure 7. Compared to PSM, QAM with Gray labeling has considerably higher starting points, given the same bit load. The starting points show how reliable will be the output of the demapper given only the channel output and no a priori information from the decoder. For PSM with large N, the starting points are very low. Hence, the demapper output in the first iteration(s) is very weak. This comes from the fact that the constellation diagram, Figure 2, is densely populated in the central region, which leads to a reduced minimum distance and in some cases even to nonbijective mapping. This effect can be clearly observed in Figure 7(b) for both PSM and QAM, where the starting points improve as the SNR increases.

The slope of the curves presents the most obvious difference. For QAM with Gray labeling, the curves are



(a) PSM



(b) QAM

Figure 4: Mutual information of PSM/QAM over the AWGN channel.

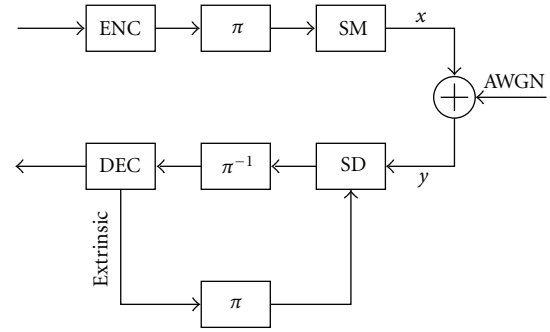


Figure 5: Bit-interleaved coded modulation with superposition mapping.

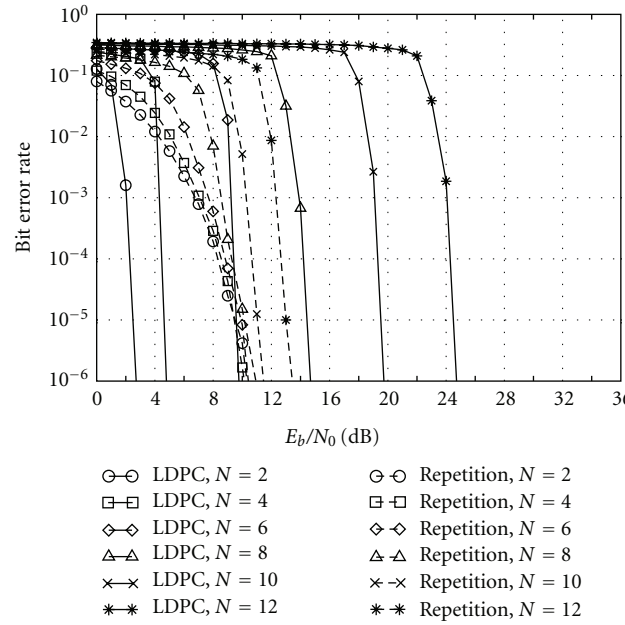
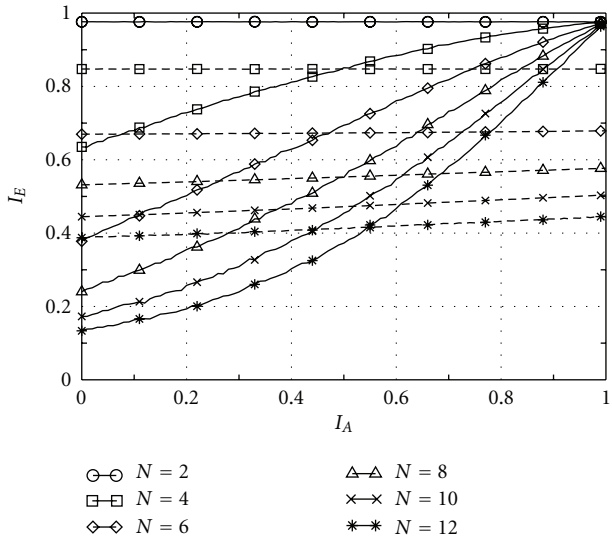


Figure 6: BER performance of PSM with rate 1/2 $(3, 6)$-regular LDPC codes and rate 1/2 regular repetition codes.

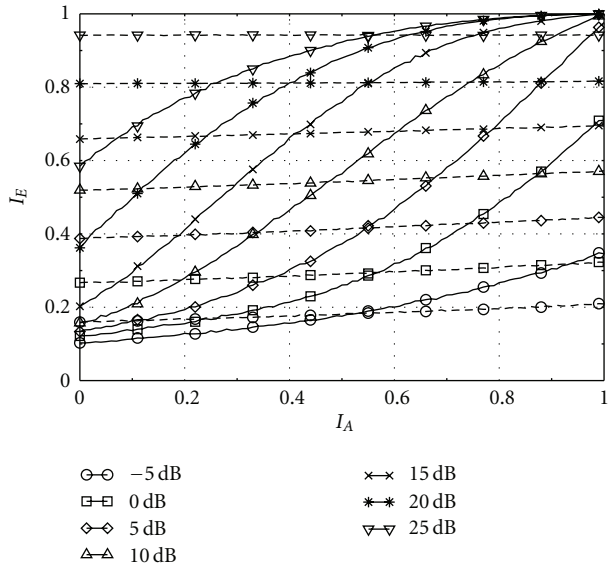
more or less horizontal, with modest slopes for $N \geq 8$. On the other hand, for PSM the curves have steep slopes, excluding $N = 2$. For $N > 6$, the curves are even convex. The slope of the curves characterizes the importance or usefulness of iterations. Hence in case of PSM, as the feedback from the decoder becomes more reliable from iteration to iteration, the demapper output improves. On the contrary for QAM, usually only a few iterations are sufficient and the gain by feedback of extrinsic information is noticeably smaller.

At last we come to the ending points of the MI transfer plots. Figure 7(a) presents an interesting fact. In the case of PSM, given the same E_c/N_0, for all selections of N, the curves end in the same point. This relates to the fact that all code bits are converted to binary antipodal symbols. If the feedback information is very reliable, then we are left with a detection scenario for BPSK transmission over a Gaussian channel. For QAM, the ending points differ significantly and for $N > 2$ the points are always lower than the corresponding ones for PSM. This can be explained by the fact that for QAM bits are unequally protected.

5.3. Convergence Behavior of Coded PSM. For coded modulation, an important issue is the convergence of the iterative demapping and decoding process. Whether an iterative receiver can converge or not is determined by the channel SNR and, more importantly, by the relation between the decoder characteristics and the demapper characteristics. As a commonly used tool, an EXIT chart can elegantly demonstrate the suitability of a channel code for a given signal mapper. By means of an EXIT chart analysis, we will see that there is indeed a strong reason for the good performance of repetition-coded PSM systems, and we will also see that the requirements of superposition mapping on channel codes are essentially different with respect to conventional mapping schemes.



(a) $E_c/N_0 = 5$ dB



(b) $N = 12$

Figure 7: MI transfer characteristics of PSM demapper (solid lines) and that of Gray labeled QAM demapper (dashed lines).

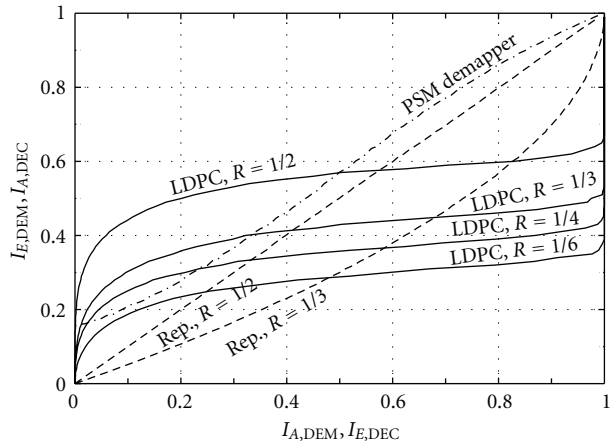


Figure 8: EXIT chart for coded PSM, $N = 12$, $E_c/N_0 = 10$ dB.

Let us assume that the SNR per code bit is $E_c/N_0 = 10$ dB and the bit load is $N = 12$. Given this setup, the MI transfer curve of a PSM demapper is plotted as a dashed dot curve in Figure 8. The MI transfer curves for several decoders are also plotted, with the horizontal and vertical axes swapped. To guarantee convergence, the MI transfer curve of the decoder must be below the MI transfer curve of the demapper, so as to open the tunnel for iterative refining of the soft decisions. Otherwise, the iterative process will get stuck at a certain point and no additional gain is achievable by further iterations.

As one can see from Figure 8, given a regular LDPC code with column weight 3, one needs to drop the code rate to as low as 1/6 in order to open the tunnel. The main problem is in the left region. As already explained in Section 5.2, the MI transfer curve of a PSM demapper with large N starts from a rather low point but ends at a rather high point. Meanwhile, the MI transfer curve of an LDPC code always possesses a wide quasi-flat section, mainly due to the nature of parity checks. This situation makes the tunnel difficult to open in the left region but often unnecessarily wide in the right region, which clearly explains the poor performance of LDPC-coded PSM at large bit loads N, as shown in Figure 6. As a matter of fact, the classical strong codes, such as Turbo codes and LDPC codes, purely built upon parity checks, do not fit with PSM very well. To fully exploit the capacity-achieving potential of superposition mapping, new codes and new code design concepts are necessary. Nevertheless by checking Figure 7(b) carefully, one recognizes that the MI transfer characteristics of LDPC codes perfectly match with that of QAM, which explains the excellent performance of LDPC-coded QAM systems.

In contrast to parity-check-based codes, a simple rate 1/2 repetition code can already effectively open the tunnel for convergence, which can also be validated by the good performance of repetition-coded PSM in Figure 6. Repetition codes come with zero coding gain, in the sense that lowering the ending point of the MI transfer curve is only possible via an equivalent amount of degradation in the E_c/N_0. Nevertheless, due to the special property of PSM, this issue does not present a big problem. For reasonable SNRs, the MI transfer curves of PSM always end at a very high point. Note that, for PSM with large bit loads, the primary task of channel coding is no longer to combat the additive noise but to guarantee the separation of superimposed binary chips. As long as all chips are perfectly separated, the BER performance of PSM will asymptotically approach that of BPSK, while the performance of uncoded BPSK is already very good at $E_c/N_0 = 10$ dB. Above all, the difficult part is the first few iterations. A decoder has to be able to deliver reliable feedback given very weak inputs in order to let the iterative demapping and decoding go on properly. With respect to this concern, however, a repetition code is the best choice. One may call the checks of a repetition code as equality checks.

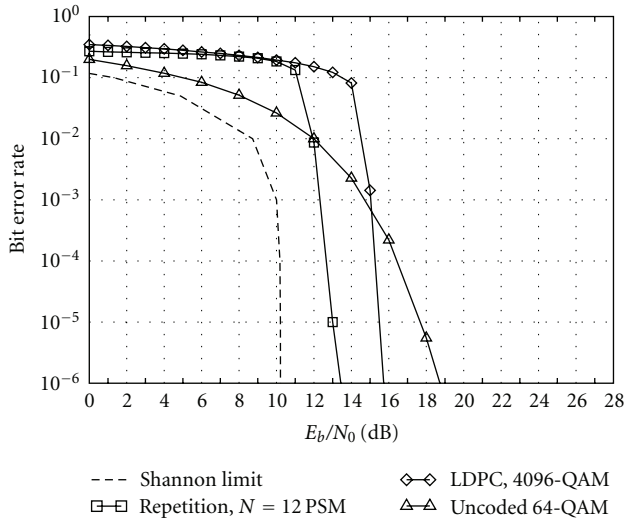


Figure 9: BER versus E_b/N_0, bandwidth efficiency 6 bits/symbol, rate 1/2 repetition code, and rate 1/2 $(6000, 12000, 3, 6)$-regular LDPC code.

There is effectively only one info bit involved in each equality check. As a result, the feedback information of an equality check can be as strong as needed at the very beginning of iterative processing, if the coding rate is low enough, illustrated by the MI transfer curve of a rate 1/3 repetition code in Figure 8. Note that the MI transfer curve of the PSM demapper is often concave, which creates a severe problem for parity-check-based codes, but this is not a problem for repetition codes, because the curves for repetition codes are also concave for $R < 1/2$. Therefore, although a pure repetition code cannot be optimal for PSM, it is definitely a good choice among others.

To locate the performance of repetition-coded PSM, we compare it to several systems with the same bandwidth efficiency in Figure 9. For an information rate of 6 bits per channel symbol, uncoded 64-QAM has a distance of about 9 dB to the Shannon limit, while a rate 1/2 LDPC-coded 4096-QAM system shortens this distance by 3 dB. In comparison, rate 1/2 repetition-coded PSM with $N = 12$ is in fact just 3 dB away from the Shannon limit, in spite of zero coding gain. Although the codes for both PSM and QAM have not yet been optimized, these results give a rough impression about their relative performance.

6. Alternative Demappers

APP demapping is a good choice for system analysis but the computational complexity is very high. Hence, two alternative demappers are brought out and a brief analysis is given.

6.1. Gaussian Approximation Approach. The Gaussian approximation (GA) is well known from multiuser systems. This concept has been widely used in multiple-access/multiplexing systems [18, 19]. The main idea is that

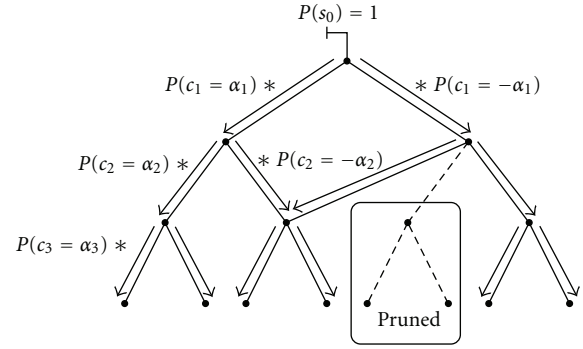


Figure 10: Illustration of the update of a priori values and the pruning concept.

the interfering superimposed symbols, together with the Gaussian noise

$$v_n = y - c_n = \sum_{i=1, i\neq n}^{N} c_i + z, \tag{10}$$

are approximated by a single Gaussian random variable. Hence, the LLR of c_n can be obtained via

$$\begin{aligned} \mathrm{LLR}(c_n) &\doteq \ln \frac{p(y \mid c_n = +\alpha_n)}{p(y \mid c_n = -\alpha_n)} \\ &\approx -\frac{(y - \alpha_n - \mu_{v_n})^2}{\sigma_{v_n}^2} + \frac{(y + \alpha_n - \mu_{v_n})^2}{\sigma_{v_n}^2} \\ &= 4 \cdot \frac{\mathrm{Re}\{\alpha_n^* \cdot (y - \mu_{v_n})\}}{\sigma_{v_n}^2}, \end{aligned} \tag{11}$$

where the mean and variance of the interfering signal are updated by feedback information from the decoder. According to the turbo principle, as the feedback information improves from iteration to iteration, the demapper estimates get more reliable. The complexity increases linearly with the bit load N, which makes Gaussian approximation very attractive for practical implementations. However, the approximation is not always good enough and can severely limit the performance. Details on this will be given in Section 6.5.

6.2. Tree-Based Approach. In [12], SM is modelled by a tree diagram. Since a tree diagram is a special case of a Markov process, the Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm [20] can be used for SISO demapping of SM to reduce the computational complexity. Let s_n denote the state at the nth level. By the property of superposition mapping, the relationship between the states of two neighboring levels is simply given by

$$s_n = s_{n-1} + c_n. \tag{12}$$

The a priori distribution of the states is evaluated recursively according to

$$P(s_n) = \sum_{c_n} P(s_{n-1} = s_n - c_n) P(c_n) \tag{13}$$

through the tree by propagating the values from root to leaves. The a priori information is given via the branches to the lower level and multiplied with the corresponding chip probability, as illustrated in Figure 10. Now, the extrinsic LLR of the nth chip can be obtained via

$$\begin{aligned}\mathrm{LLR}(c_n) &\doteq \ln \frac{p(y \mid c_n = +\alpha_n)}{p(y \mid c_n = -\alpha_n)} \\ &= \ln \frac{\sum_{s_{n-1}} p(y \mid s_n = s_{n-1} + \alpha_n) P(s_{n-1})}{\sum_{s_{n-1}} p(y \mid s_n = s_{n-1} - \alpha_n) P(s_{n-1})}.\end{aligned} \tag{14}$$

The likelihood of the received value given the state at the nth level can be calculated by a backward recursion through the tree. Starting from the leaves,

$$p(y \mid s_N) = \frac{1}{\pi\sigma_z^2} \exp\left(\frac{|y - s_N|^2}{\sigma_z^2}\right), \tag{15}$$

one calculates $p(y \mid s_n)$ recursively by adding together the likelihood values from the branches that have been multiplied with the corresponding chip probabilities. This approach cleverly utilizes the probability propagation and hence considerably reduces the computational complexity. However, the complexity is still exponentially increasing. Inspired by the tree structure, we would like to introduce a novel method of how to reduce the size of the tree without big sacrifices concerning the performance.

6.3. Pruned Tree-Based Approach. The core idea is based on the fact that if at some level of the tree two nodes are close to each other, then during the subsequent tree expansion the corresponding subtrees will always keep the distance relation. (This means that the superposition values for nodes are similar.) In other words, the two subtrees are the same, just one is a shifted version of the other. In the special case that two nodes have exactly the same superposition value, then the subtrees are identical. Such subtrees are redundant and can be pruned. Figure 10 illustrates one possible merging scenario. Considering the demapping process, nothing is changed. All the bit paths still exist, the only difference compared to the full tree is that all pruned node values are shifted by a certain constant. In order to control the pruning rate (and hence the computational complexity), we introduce a merging threshold and denote it by ϵ. Now starting from the root, for every level the squared difference values of all nodes are checked if they fulfil

$$\left|s_n^i - s_n^j\right|^2 < \epsilon, \tag{16}$$

where s_n^i and s_n^j denote two node values from that level. If two nodes are in the range of ϵ, then s_n^j is merged to s_n^i and pruned. With a single merging operation actually 2^{N-n} constellation point pairs are merged, that differ by $s_n^i - s_n^j$, because a node at level n, if expanded, would relate to 2^{N-n} leaves. Hence, the tree is reduced by the given sub-tree from node s_n^j to the leaves, resulting in $2^{N-n+1} - 1$ pruned nodes. Once the whole tree is processed, all nodes in the range of ϵ are merged and the tree will be stored for permanent use.

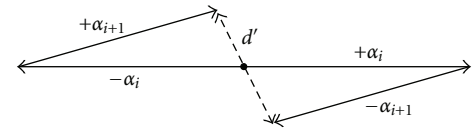


Figure 11: Distance between bit sequences "01" and "10".

Table 1: Complexity reduction of the tree with given ϵ.

N	ϵ	Total nodes	Nodes left	$\lvert\mathcal{X}'\rvert$
6	0.1786	127	100	42
8	0.0761	511	258	90
10	0.0391	2047	634	196
12	0.0227	8191	1678	468
14	0.0143	32767	3303	804
16	0.0096	131071	6269	1364

However, there are a few problems. First, with multiple mergers, a node may drift quite far from its original value, maximally by

$$\max(d) < \epsilon \cdot (N - n + 1), \tag{17}$$

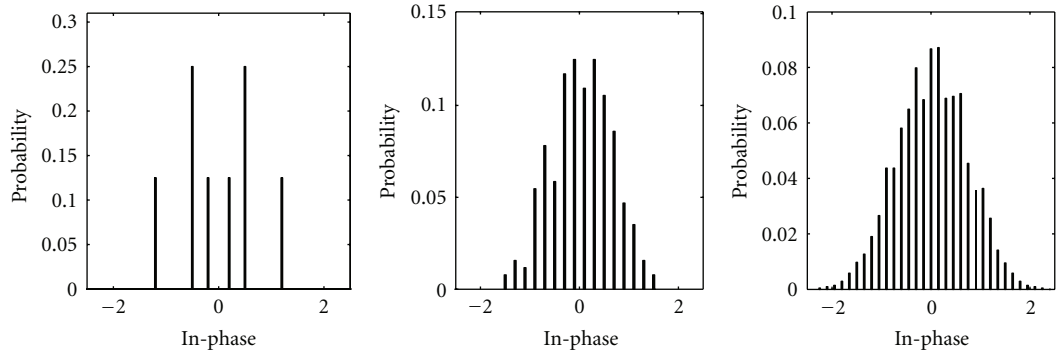
if the remained node always gets merged to another node. The second issue comes from detection. After pruning the tree, some constellation points do not exist in the tree anymore. Hence, given a cut symbol is transmitted, an increased SNR will not lead to a better demapping performance, as such signal point does not exist in the tree anymore. To overcome this problem, we should adjust the transmitter to fit to the receiver. Therefore, the same pruned tree should be used to select the transmit symbol, given the coded bits. One takes N coded bits and follows the tree from root to leaf, while choosing the branch according to the bit. This way the ambiguity between the mapper and demapper is removed, as both sides are working with the same constellation space.

The parameter ϵ gives a certain freedom to balance between complexity and performance. Clearly if one would prune the tree too much, the performance would degrade significantly. From simulations, a certain point was found after that catastrophic merging takes place. This happens if ϵ is large enough so that any code bit sequences, "01" and "10", for any N, get merged. The distance d' between these two bit sequences is shown in Figure 11 by the dashed line.

Hence, to avoid such short span mergers, ϵ should be chosen smaller than d'^2. Given the amplitude factor $\alpha = 1/\sqrt{N}$, the boundary values are listed in Table 1. The formula to calculate d'^2 for a fixed N is

$$\begin{aligned}d'^2 &= (2(c_n - c_{n+1}))^2 \\ &= 4\left(\alpha\cos\left(\frac{\pi(n-1)}{N}\right) - \alpha\cos\left(\frac{\pi n}{N}\right)\right. \\ &\qquad \left. + \alpha\sin\left(\frac{\pi(n-1)}{N}\right) - \alpha\sin\left(\frac{\pi n}{N}\right)\right)^2.\end{aligned} \tag{18}$$

The node reduction is significant. For $N = 16$ already approximately 95% of the tree is pruned and the rate



(a) $N = 4$ (b) $N = 8$ (c) $N = 12$

FIGURE 12: Distribution of $\mathrm{Re}\{x\}$ for pruned tree, $\alpha = \sqrt{1/N}$.

increases with increasing N. In the last column, we can see the symbol cardinality corresponding to the pruned tree, $|\mathcal{X}'|$. For $N = 16$, it is about 2% of 2^{16}. Clearly the mapping is no longer bijective. The new constellation diagram and probability distribution are shown in Figures 13 and 12. As stated before, after pruning is finished, all nodes in the range of ϵ are merged. Hence, the central region is no longer geometrically densely populated, as seen in Figure 2, but instead probabilistically nonuniform. Correspondingly, there are many bit sequences merged to constellation points in the central region and less near the border.

6.4. Complexity Comparison. As mentioned before, the Gaussian approximation demapper has a very low complexity that increases linearly with N. APP and full tree-based demapper both have exponentially increasing complexity, but the latter already shows a noticeable reduction. For APP, to calculate the LLRs for each chip, the demapping algorithm has to go through 2^N constellation points, meaning for each symbol it visits $N \cdot 2^N$ points. However, for tree-based demapping, due to the tree structure the complexity to calculate the LLR for each chip varies. For the top level one needs to consider just 2 nodes. Totally, considering the forward and backward recursion and the LLR calculation, a tree-based demapper visits roughly $5 \cdot 2^N$ points. Hence, already the full tree reduces the complexity compared to APP demapping. As can be seen from Table 1, the pruned tree significantly reduces the whole tree size, leading to a proportionally smaller computational complexity.

6.5. Performance. The Gaussian approximation-based demapper has a very low complexity, but it is associated with a severe performance degradation. In Figure 14, the MI transfer characteristics for both pruned tree and GA demapper are shown. Clearly GA does not work so well. In [21] it has been pointed out that for a rate 1/2 repetition code, GA can only support bit loads up to $N = 4$ and after that presents a very high error floor. The pruned tree approach (Using ϵ values from Table 1) shows much smaller degradation in the MI transfer characteristics with respect to GA. The comparison between the BER performance of pruned tree and GA demapper is given in Figure 15. For $N > 4$, the GA demapper makes a severe problem for convergence. Comparing Figure 15 with Figure 6, we see that tree pruning brings some performance degradation. This is easy to understand, since a certain amount of information is lost during pruning. Nevertheless, tree pruning does not make a big problem for convergence.

7. PAPR Control and Compensation

The output symbols of a superposition mapper inherently exhibit a large peak-to-average power ratio (PAPR). High peaks occur when the superimposed chips have similar polarities. The PAPR of the output symbols is defined as

$$\mathrm{PAPR} = \frac{\max\left\{|x|^2\right\}}{E\left\{|x|^2\right\}}, \tag{19}$$

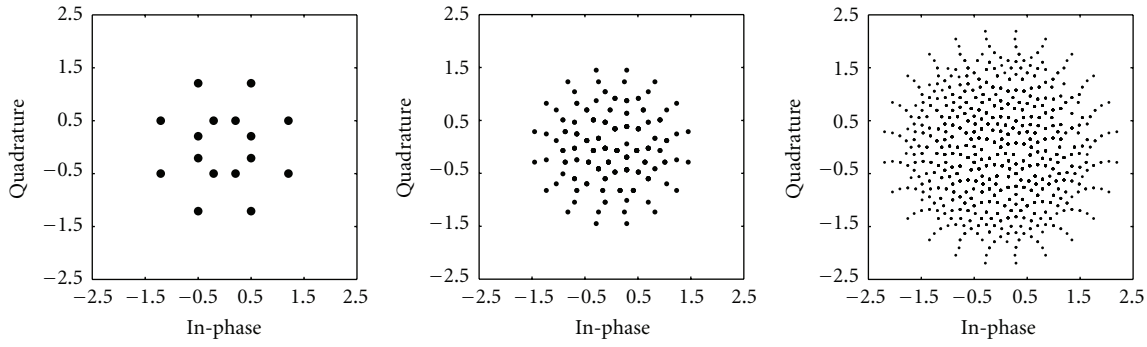
where $E\{|x|^2\}$ denotes the average symbol power (for simplicity, we use P_x in the following). For PSM ($\alpha_n = 1/\sqrt{N}, \theta_n = \pi(n-1)/N$), the PAPR value can be calculated as

$$\begin{aligned}\mathrm{PAPR} &= \frac{1}{N}\left|\sum_{n=1}^{N} e^{j\pi(n-1)/N}\right|^2 \\ &\approx (0.4N + 0.1) \quad \text{for } N \geqslant 4,\end{aligned} \tag{20}$$

where the first equation comes from the fact that the highest peak happens when the component bits are all zeros or all ones. From (20), it is clear that the PAPR grows linearly with the bit load N. Consequently, the power amplifier has to operate with a large back-off, which reduces the power efficiency.

Clipping at the transmitter side is an efficient and simple method to reduce the PAPR. As shown in Figure 16, if the output symbol has a larger magnitude than the given clipping threshold A, the clipper (CLP) will limit the magnitude of the symbol while keeping its phase as follows:

$$\overline{x} = \begin{cases} x, & \text{if } |x| \leq A, \\ A \cdot \dfrac{x}{|x|}, & \text{otherwise.} \end{cases} \tag{21}$$



(a) $N = 4$ (b) $N = 8$ (c) $N = 12$

Figure 13: Constellation diagrams for pruned tree, $\alpha = \sqrt{1/N}$.

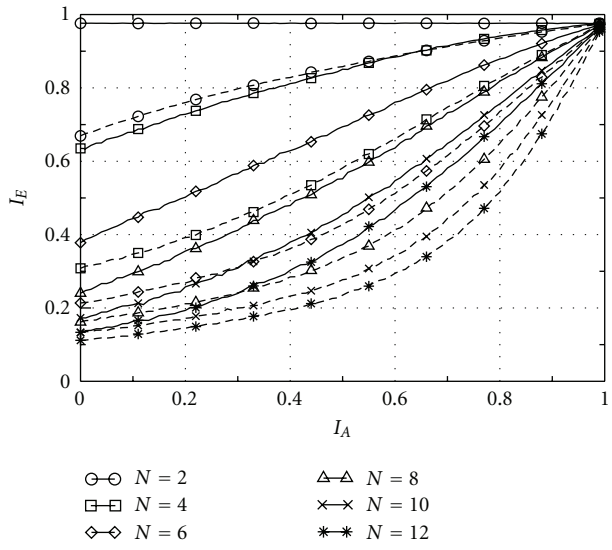


Figure 14: MI transfer characteristics for pruned tree demapper (solid lines) and for GA demapper (dashed lines), $E_c/N_0 = 5$ dB.

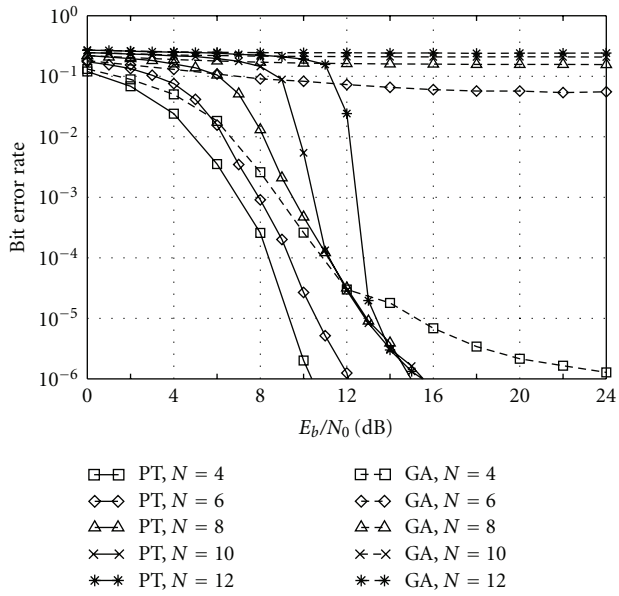


Figure 15: BER performance of PSM with pruned tree (PT) and GA demapper, rate 1/2 repetition code.

The clipping process is usually characterized by the clipping ratio (CR) γ, which is defined as

$$\gamma = \frac{A}{\sqrt{P_x}}. \tag{22}$$

Certainly, the power of clipped symbols decreases with γ. As shown in [22], the average output power of the clipper is given by

$$P_{\overline{x}} = \left(1 - e^{-\gamma^2}\right) P_x. \tag{23}$$

The power loss due to clipping can be compensated by subsequent normalization. To preserve the same average power compared to the unclipped symbol, all symbols must be amplified by a factor $a \geq 1$:

$$a = \sqrt{\frac{P_x}{P_{\overline{x}}}} = \sqrt{\frac{1}{1 - e^{-\gamma^2}}}. \tag{24}$$

As a result, more power is allocated to the unclipped symbols after normalization. This process can be treated as a special form of unequal error protection, which makes the unclipped symbols more robust to the channel noise. The PAPR of the clipped symbol remains the same after normalization since the symbols within the burst are multiplied by the same factor a.

With a subsequent pulse shaping filter, there will be no bandwidth expansion for the signal in continuous-time domain. However, new peaks may grow since filtering usually causes overlapping between the consequent symbols.

The distortion due to clipping can be described by a clipping noise z_{cl}. The clipped symbol $\overline{x}$ can be modeled as a summation of the unclipped symbol x and the clipping noise z_{cl} as follows:

$$\overline{x} = x + z_{\text{cl}}. \tag{25}$$

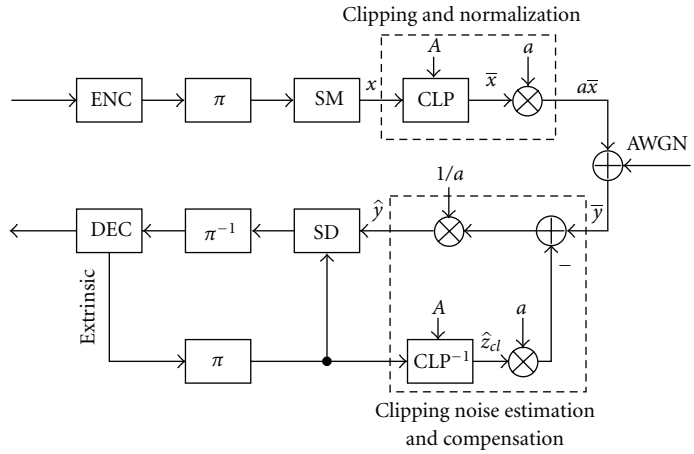


Figure 16: Coded PSM system with PAPR control and compensation.

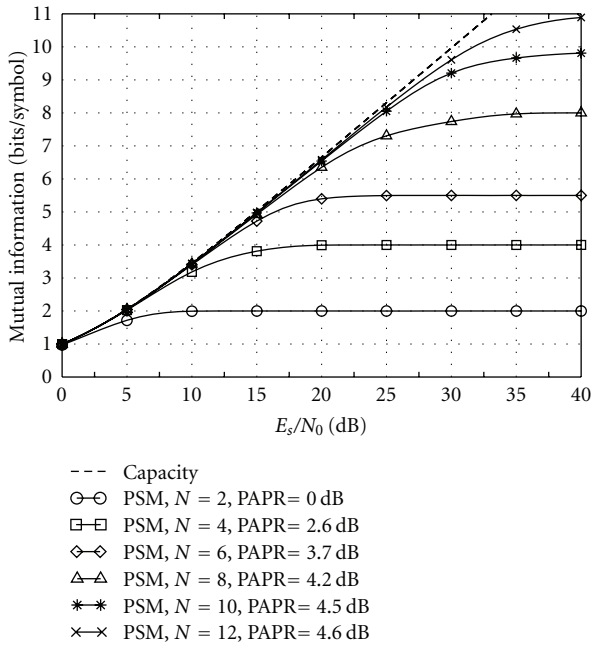


Figure 17: Mutual information of peak-limited PSM over the AWGN channel.

After normalization and for the AWGN channel, the received signal $\overline{y}$ can be written as follows:

$$\begin{aligned}\overline{y} &= a \cdot \overline{x} + z \\ &= a \cdot (x + z_{\mathrm{cl}}) + z.\end{aligned} \tag{26}$$

7.1. Mutual Information of Clipped PSM. The mutual information of clipped PSM is given in Figure 17, in which the PSM symbols are clipped to have the same PAPR as conventional QAM with the same bit load. Compared to the unclipped PSM in Figure 4(a), the mutual information of clipped PSM does not degrade significantly. Revisiting Figure 4(b), we see that even with the same PAPR, PSM still outperforms QAM in the slope region.

7.2. Clipping Noise Estimation and Compensation. To compensate the distortions introduced by clipping, clipping noise should be estimated and removed from the received signal. Assuming that the clipping threshold A and the normalization factor a are known at the receiver side, the clipping noise can be estimated, for example, as follows [23]:

$$\hat{z}_{\mathrm{cl}}^{\mathrm{OPT}} = \sum_{x \in \mathcal{X}} P(x) \cdot z_{\mathrm{cl},x}, \tag{27}$$

where $\mathcal{X}$ denotes the symbol alphabet and $z_{\mathrm{cl},x}$ is the clipping noise corresponding to symbol x.

Equation (27) is optimal in the sense of minimizing the mean square error of the estimated clipping noise. Unfortunately, the computational complexity of this algorithm grows exponentially with bit load N. As an alternative algorithm, soft compensation algorithm is proposed by Tong et al. in [24, 25]. The clipping noise is treated as an equivalent Gaussian noise sample, and a look-up table method is used to speed up the detection. Another alternative is the soft reconstruction algorithm (SRA). The idea is to use decoder feedback to reconstruct the soft transmitted symbol then repeat the clipping process as in the transmitter and derive the clipping noise. In the following, we will briefly introduce the SRA algorithm proposed in [26] and extend it to the case of APP demapping.

Let L_{c_n} denote the extrinsic LLR of binary chips from the channel decoder, then its corresponding soft binary chip can be calculated as follows:

$$\mu_{c_n} = \tanh\left(\frac{L_{c_n}}{2}\right). \tag{28}$$

Using the superposition mapping rule, the soft symbol μ_x can be obtained by superimposing the binary soft chips with known power and phase allocation as follows:

$$\mu_x = \sum_{n=1}^{N} \alpha_n e^{j\theta_n} \mu_{c_n}. \tag{29}$$

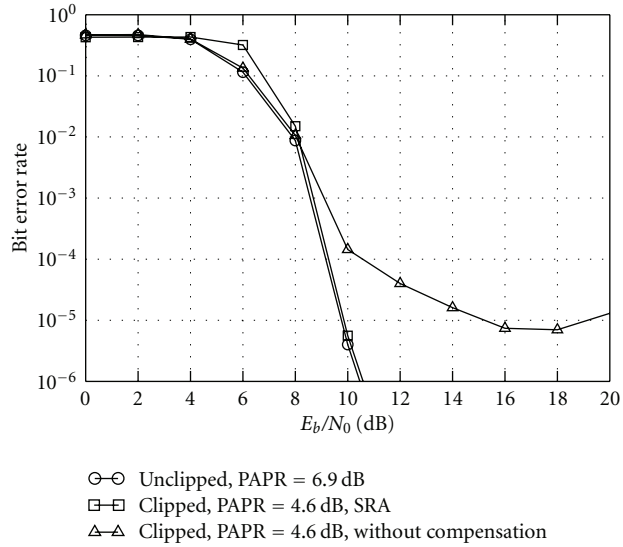


Figure 18: BER performance of clipped PSM with rate 1/4 repetition code and bit load $N = 12$.

With the knowledge of the reconstructed symbol μ_x and the clipping threshold A, the estimated clipping noise can be obtained as follows:

$$\hat{z}_{\text{cl}}^{\text{SRA}} = \begin{cases} 0, & \text{if } |\mu_x| \le A, \\ \mu_x \cdot \left(\dfrac{A}{|\mu_x|} - 1 \right), & \text{otherwise.} \end{cases} \tag{30}$$

Then the received signal is refined by subtracting the estimated clipping noise as follows:

$$\hat{y} = \frac{\overline{y} - a \cdot \hat{z}_{\text{cl}}^{\text{SRA}}}{a}. \tag{31}$$

The refined signal can be used in the APP demapper. Compared to the optimal solution in (27), the proposed SRA method has only a linear complexity.

Figure 18 shows the BER performance for the clipped PSM transmission over the AWGN channel. After clipping, the PAPR of the output symbol is reduced from 6.9 dB to 4.6 dB. It can be seen that an error floor exits even at the high SNR region if clipping noise is not compensated. With the SRA algorithm, the SNR loss due to clipping is reduced to 0.15 dB at 10^{-5}. This result demonstrates that the SRA algorithm can efficiently compensate the performance loss due to clipping while introducing only marginal complexity overhead.

8. Conclusions and Future Work

In this paper, we have carried out an extensive study on phase-shifted superposition mapping (PSM), particularly on its behavior in the scenario of iterative demapping and decoding. It is shown that PSM is a quasi-bijective mapping scheme and provides no rate limit in a coded system. Due to a geometrically Gaussian-like symbol distribution, PSM has a good potential to approach the capacity of Gaussian channels. Analogous to conventional mapping schemes like PSK/QAM, PSM can easily be applied in a bit-interleaved coded modulation (BICM) scenario, albeit with different requirements on the channel code. Via an EXIT chart analysis, it is found that for PSM with large bit loads a simple repetition code can already outperform a classical LDPC code. Though surprising, the reason for this phenomenon is indeed simple. Because of superimposing binary chips with identical power, the main task of the channel code is to guarantee a perfect separation of superimposed chips instead of combating the noise. A repetition code simply works better than a parity-check code with respect to this purpose. Numerical results show that rate 1/2 regular-repetition-coded PSM with bit load $N = 12$ can outperform regular-LDPC-coded QAM with the same bandwidth efficiency and is indeed just 3 dB away from the Shannon limit at a BER of 10^{-6}.

Besides theoretical concerns, several practical issues are also treated in this contribution. Using a tree diagram to represent the constellation evolution process of PSM and pruning those nodes that are close enough to each other, a dramatic complexity reduction can be achieved with an acceptable performance degradation. Furthermore, by baseband clipping at the transmitter side in conjunction with iterative soft compensation at the receiver side, the peak-to-average power ratio of PSM symbols can be controlled to a reasonable level without much sacrificing the BER performance.

There are at least two interesting topics for future work. A repetition code can give an excellent performance for a PSM system, particularly in the initial iterations. Nevertheless, in late-stage iterations parity-check codes can provide some gain over repetition codes. Therefore, finding good hybrid repetition and parity-check codes deserves to be an interesting topic for superposition mapping. Second, the good performance of the pruned-tree-based SISO demapping algorithm provides an important hint. In the PSM constellation, some points are close to each other and some not. This certainly brings some penalty in case of noisy channels. It is worthwhile to check if one can use a tree to refine the PSM constellation to achieve additional performance improvement.

Acknowledgment

This work has been supported by the German Research Foundation (DFG) under Contracts nos. HO 2226/10-1 and HO 2226/9-2.

References

[1] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, 1948.

[2] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, John Wiley & Sons, New York, NY, USA, 2nd edition, 2006.

[3] F. R. Kschischang and S. Pasupathy, "Optimal nonuniform signaling for Gaussian channels," *IEEE Transactions on Information Theory*, vol. 39, no. 3, pp. 913–929, 1993.

[4] R. F. H. Fischer, *Precoding and Signal Shaping for Digital Transmission*, John Wiley & Sons, New York, NY, USA, 2002.

[5] E. Schrödinger, *Statistical Thermodynamics*, Cambridge University Press, Cambridge, UK, 1962.

[6] G. D. Forney Jr., "Trellis shaping," *IEEE Transactions on Information Theory*, vol. 38, no. 2, pp. 281–300, 1992.

[7] G. R. Lang and F. M. Longstaff, "A leech lattice modem," *IEEE Journal on Selected Areas in Communications*, vol. 7, no. 6, pp. 968–973, 1989.

[8] P. Fortier, A. Ruiz, and J. M. Cioffi, "Multidimensional signal sets through the shell construction for parallel channels," *IEEE Transactions on Communications*, vol. 40, no. 3, pp. 500–512, 1992.

[9] D. G. Forney Jr. and G. Ungerboeck, "Modulation and coding for linear Gaussian channels," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2384–2415, 1998.

[10] L. Duan, B. Rimoldi, and R. Urbanke, "Approaching the AWGN channel capacity without active shaping," in *Proceedings of IEEE International Symposium on Information Theory (ISIT '97)*, p. 374, Ulm, Germany, June 1997.

[11] H. Schoeneich and P. A. Hoeher, "Adaptive interleave-division multiple access—a potential air interface for 4G bearer services and wireless LANs," in *Proceedings of the International Conference on Wireless and Optical Communications and Networks (WOCN '04)*, Muscat, Oman, June 2004.

[12] X. Ma and L. Ping, "Coded modulation using superimposed binary codes," *IEEE Transactions on Information Theory*, vol. 50, no. 12, pp. 3331–3343, 2004.

[13] G. Caire, G. Taricco, and E. Biglieri, "Bit-interleaved coded modulation," *IEEE Transactions on Information Theory*, vol. 44, no. 3, pp. 927–946, 1998.

[14] T. Wo and P. A. Hoeher, "Superposition mapping with application in bit-interleaved coded modulation," in *Proceedings of the International ITG Conference on Source and Channel Coding (SCC '10)*, Siegen, Germany, January 2010.

[15] H. Schoeneich, *Adaptiver Interleave-Division Mehrfachzugriff (IDMA) mit Anwendung in der Mobilfunkkommunikation*, Ph.D. dissertation, University of Kiel, Kiel, Germany, 2007.

[16] S. ten Brink, "Convergence of iterative decoding," *Electronics Letters*, vol. 35, no. 13, pp. 1117–1118, 1999.

[17] S. ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Transactions on Communications*, vol. 49, no. 10, pp. 1727–1737, 2001.

[18] L. Ping, L. Liu, and W. K. Leung, "A simple approach to near-optimal multiuser detection: Interleave-division multiple access," in *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC '03)*, pp. 391–396, New Orleans, La, USA, March 2003.

[19] H. Schoeneich and P. A. Hoeher, "A hybrid multiple access scheme approaching single user performance," in *Proceedings of the Workshop on Signal Processing in Communications (SPC '03)*, pp. 163–168, Baiona, Spain, September 2003.

[20] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, vol. 20, no. 2, pp. 284–287, 1974.

[21] M. Noemm, T. Wo, and P. A. Hoeher, "Multilayer APP detection for IDM," *Electronics Letters*, vol. 46, no. 1, pp. 96–97, 2010.

[22] H. Ochiai and H. Imai, "Performance analysis of deliberately clipped OFDM signals," *IEEE Transactions on Communications*, vol. 50, no. 1, pp. 89–101, 2002.

[23] S. M. Kay, *Fundamentals of Statistical Signal Processing*, vol. 1 of *Estimation Theory*, Prentice Hall, Upper Saddle River, NJ, USA, 1st edition, 1993.

[24] J. Tong, L. Ping, and X. Ma, "Superposition coding with peak-power limitation," in *Proceedings of the IEEE International Conference on Communications (ICC '06)*, pp. 1718–1723, Istanbul, Turkey, June 2006.

[25] J. Tong, L. Ping, and X. Ma, "Superposition coded modulation with peak-power limitation," *IEEE Transactions on Information Theory*, vol. 55, no. 6, pp. 2562–2576, 2009.

[26] D. Hao and P. A. Hoeher, "Iterative estimation and cancellation of clipping noise for multi-layer IDMA systems," in *Proceedings of the International ITG Conference on Source and Channel Coding (SCC '08)*, Ulm, Germany, January 2008.

Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2010, Article ID 609509, 12 pages
doi:10.1155/2010/609509

Research Article

Low-Complexity Gaussian Detection for MIMO Systems

Tianbin Wo and Peter Adam Hoeher

The Information and Coding Theory Lab, University of Kiel, Kaiserstrasse 2, 24143 Kiel, Germany

Correspondence should be addressed to Tianbin Wo, wtb@tf.uni-kiel.de

Received 13 March 2010; Accepted 30 August 2010

Academic Editor: Christian Schlegel

For single-carrier transmission over delay-spread multi-input multi-output (MIMO) channels, the computational complexity of the receiver is often considered as a bottleneck with respect to (w.r.t.) practical implementations. Multi-antenna interference (MAI) together with intersymbol interference (ISI) provides fundamental challenges for efficient and reliable data detection. In this paper, we carry out a systematic study on the interference structure of MIMO-ISI channels, and sequentially deduce three different Gaussian approximations to simplify the calculation of the global likelihood function. Using factor graphs as a general framework and applying the Gaussian approximation, three low-complexity iterative detection algorithms are derived, and their performances are compared by means of Monte Carlo simulations. After a careful inspection of their merits and demerits, we propose a graph-based iterative Gaussian detector (GIGD) for severely delay-spread MIMO channels. The GIGD is characterized by a strictly linear computational complexity w.r.t. the effective channel memory length, the number of transmit antennas, and the number of receive antennas. When the channel has a sparse ISI structure, the complexity of the GIGD is strictly proportional to the number of nonzero channel taps. Finally, the GIGD provides a near-optimum performance in terms of the bit error rate (BER) for repetition encoded MIMO systems.

1. Introduction

In single-carrier mobile transmission systems not exploiting a guard interval, there are two sources of intersymbol interference (ISI): static ISI due to pulse shaping and receive filtering, and dynamic ISI due to the time-varying delay spread of the physical channel. Static ISI degrades the receiver performance, but can be avoided or limited by proper signal design. Dynamic ISI is particularly severe if the delay spread exceeds the symbol period, which is likely the case for high-rate data transmission. Dynamic ISI, however, provides a diversity gain in the time domain (fast fading) and the frequency domain (multipath fading). In addition to ISI, MIMO-ISI channels are characterized by another type of interference, namely, multi-antenna interference (MAI), which is caused by the simultaneous transmission of data streams via multiple antennas. MAI together with ISI manifests a fundamental challenge for efficient and reliable data detection. On the other hand, MAI provides a diversity gain in the spatial domain, from an information theoretical point of view.

There are two obvious facts that impede a practical implementation of high-rate single-carrier transmission over MIMO-ISI channels. First, with increasing signal bandwidth the effective channel memory length increases, which degrades the system performance in case of linear or decision-feedback equalization. Second, state-space-based detectors, such as the Viterbi algorithm [1, 2] and the BCJR algorithm [3], provide an excellent performance since they benefit from the diversity gain of dynamic ISI and MAI, but their computational complexity is typically prohibitive. Therefore, multi-carrier transmission schemes, particularly orthogonal frequency-division multiplexing (OFDM) [4], are often applied to circumvent the problem of ISI. An important question is if it is truly impossible to implement a single-carrier transmission system with reasonable performance and complexity for MIMO-ISI channels. We will try to answer this question by proposing a new detection algorithm, called graph-based iterative Gaussian detector (GIGD).

As the detection complexity of MIMO-ISI channels is mainly caused by multi-antenna interference and intersymbol interference, we will first carry out a systematic study on

the interference structure and try to find the opportunities of easy treatment. Based on the knowledge obtained from this study, we deduce three different Gaussian approximations, namely, joint Gaussian approximation (JGA), grouped joint Gaussian approximation (GJGA), and independent Gaussian approximation (IGA), to simplify the calculation of the global likelihood function and sequentially reduce the data detection complexity. The JGA is already well known [5–10], while the GJGA and the IGA are new approaches proposed by the authors. Corresponding to these three Gaussian approximations, three low-complexity iterative parallel soft interference cancellation [5, 11] algorithms, namely, joint Gaussian detector (JGD), grouped joint Gaussian detector (GJGD), and graph-based iterative Gaussian detector (GIGD), will be described by utilizing factor graphs [12, 13] as a general framework. From the JGD to the GJGD, and from the GJGD to the GIGD, the detection complexity is reduced dramatically in each step.

For severely delay-spread MIMO-ISI channels, we propose the GIGD as a promising solution. Applying the independent Gaussian approximation, the GIGD has a computational complexity strictly linear w.r.t. the number of nonzero channel taps, the number of transmit antennas, and the number of receive antennas. Meanwhile, the performance loss incurred by the independent Gaussian approximation can be well compensated by using a repetition code. More importantly, the GIGD shows a satisfying capability in exploiting the frequency/time/space diversity provided by the MIMO-ISI fading channels.

The remainder of this paper is organized as follows. Section 2 introduces a conventional output-oriented channel model as well as a symbol-oriented channel model. Section 3 provides a deep insight into the interference structure of MIMO-ISI channels, and Section 4 gives a brief introduction on factor graphs and message passing algorithms. Section 5 revises the known joint Gaussian detector, Section 6 derives a grouped joint Gaussian detector, and Section 7 proposes a graph-based iterative Gaussian detector. Numerical results by means of Monte Carlo simulations are provided in Section 8 and Section 9 to assess and compare the performance of the three Gaussian detectors. Finally, conclusions are drawn in Section 10.

2. Channel Model

In this section, we will first introduce a conventional MIMO-ISI channel model, and then convert it into a symbol-oriented channel model to facilitate the mathematical derivation of the new algorithms.

2.1. Output-Oriented Channel Model. The equivalent discrete-time model of a MIMO-ISI channel (including transmit and receive filters, physical channel and symbol-rate sampling) can be written in complex baseband notation as

$$y_n[k] = \sum_{m=1}^{N_T} \sum_{l=0}^{L} h_{n,m}^{l} \cdot x_m[k-l] + w_n[k], \quad 1 \le n \le N_R, \quad (1)$$

where N_R denotes the number of receive (Rx) antennas, N_T the number of transmit (Tx) antennas, L the effective memory length of all subchannels, and $k \in \{0, 1, \ldots, K-1\}$ the discrete time index with K denoting the block length. $y_n[k] \in \mathbb{C}$ is the channel output sample at the nth Rx antenna at time index k, and $x_m[k]$ is the channel input symbol at the mth Tx antenna at time index k. $h_{n,m}^{l} \in \mathbb{C}$ marks the lth tap of the subchannel connecting the nth Rx antenna and the mth Tx antenna. $w_n[k]$ represents a complex additive white Gaussian noise (AWGN) sample at the nth Rx antenna at time index k with zero mean and variance σ_w^2. By convention, the single-sided noise spectral density in the passband is denoted by N_0. Noting that $w_n[k]$ is a complex noise sample, we have $\sigma_w^2 = N_0/2 + N_0/2 = N_0$. Throughout this paper, the signal-to-noise ratio per info bit will be defined as E_b/N_0, where E_b stands for the energy used for transmitting one info bit. In case of coded transmission, we have $E_b = E_s/R$ with $E_s \doteq \mathrm{E}\{|x_m[k]|^2\}$ denoting the energy used for transmitting one symbol and R denoting the coding rate.

We assume that all channel taps are constant within each data burst while varying independently from burst to burst. Moreover, we assume that the fading processes of channel taps all have the same average power and are mutually independent. This equal delay power profile is often used for the purpose of equalizer test, for example, in the 3GPP GSM standard, since it is the most challenging case for linear equalization. Nevertheless, we will show that low-complexity high-performance data detection is in fact possible for this type of MIMO-ISI channel, by means of the receiver algorithm proposed in this paper.

If we take a second look at (1), we may recognize that it is actually an output-oriented channel model, that is, this channel model explains how a channel output sample is formed given multiple channel inputs. Such kind of channel model is convenient to derive state-space-based detection algorithms, but inconvenient for the derivation of factor-graph-based detection algorithms, which requires a channel model that explicitly states the information spread of a data symbol over multiple channel outputs.

2.2. Symbol-Oriented Channel Model. Let us consider an arbitrary data symbol $x_m[k]$. Due to multiple Rx antennas and delay spread, there will be in total $N_R(L+1)$ channel outputs containing the information of $x_m[k]$. From now on, we call these channel outputs the observations of symbol $x_m[k]$. To facilitate the following mathematical elaboration, we collect the observations of $x_m[k]$ into a matrix

$$\mathbf{Y}[k] = \begin{bmatrix} y_1[k] & y_1[k+1] & \ldots & y_1[k+L] \\ y_2[k] & y_2[k+1] & \ldots & y_2[k+L] \\ \vdots & \vdots & \ddots & \vdots \\ y_{N_R}[k] & y_{N_R}[k+1] & \ldots & y_{N_R}[k+L] \end{bmatrix} \quad (2)$$

which may be termed the observation matrix of $x_m[k]$. Note that $\mathbf{Y}[k]$ is shared by all $x_m[k]$ for $m = 1, 2, \ldots, N_T$. Hence, there is no necessity for $\mathbf{Y}[k]$ to have a subscript m making this distinction.

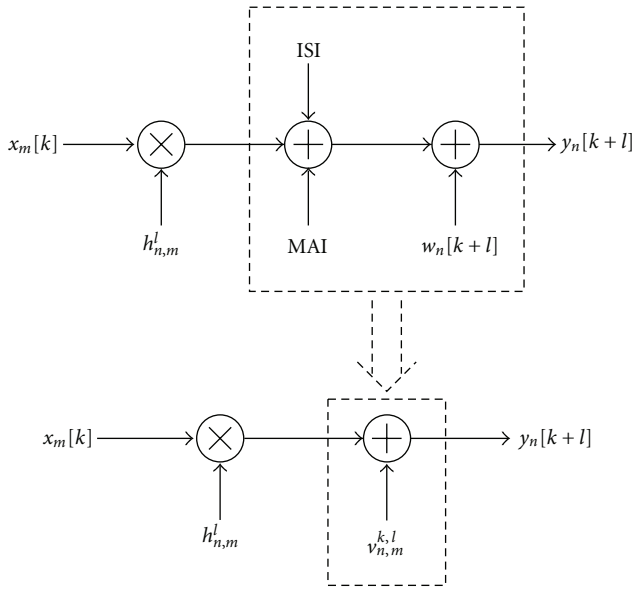


Figure 1: Relationship between a data symbol and one of its observations.

Revisiting (1), we find that the relationship between $x_m[k]$ and one of its observations $y_n[k+l]$ ($1 \le n \le N_R$, $0 \le l \le L$) can be written as

$$\begin{aligned} y_n[k+l] &= \sum_{i=1}^{N_T}\sum_{j=0}^{L} h^j_{n,i}x_i[k+l-j] + w_n[k+l] \\ &= h^l_{n,m}\, x_m[k] + \underbrace{\sum_{j=0, j\neq l}^{L} h^j_{n,m}\, x_m[k+l-j]}_{\text{ISI}} \\ &\quad + \underbrace{\sum_{i=1, i\neq m}^{N_T}\sum_{j=0}^{L} h^j_{n,i}x_i[k+l-j]}_{\text{MAI}} + \underbrace{w_n[k+l]}_{\text{AWGN}}. \end{aligned} \tag{3}$$

Defining the summation of ISI, MAI, and AWGN as an effective noise term $v^{k,l}_{n,m}$, the relationship between $x_m[k]$ and $y_n[k+l]$ can be simplified as

$$y_n[k+l] = h^l_{n,m}x_m[k] + v^{k,l}_{n,m}, \tag{4}$$

c.f. Figure 1. Combining (2) and (4), we obtain the following symbol-oriented channel model:

$$\mathbf{Y}[k] = \mathbf{H}_m x_m[k] + \mathbf{V}^k_m, \tag{5}$$

with

$$\begin{aligned} \mathbf{H}_m &= \begin{bmatrix} h^0_{1,m} & h^1_{1,m} & \cdots & h^L_{1,m} \\ h^0_{2,m} & h^1_{2,m} & \cdots & h^L_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ h^0_{N_R,m} & h^1_{N_R,m} & \cdots & h^L_{N_R,m} \end{bmatrix} \\ \mathbf{V}^k_m &= \begin{bmatrix} v^{k,0}_{1,m} & v^{k,1}_{1,m} & \cdots & v^{k,L}_{1,m} \\ v^{k,0}_{2,m} & v^{k,1}_{2,m} & \cdots & v^{k,L}_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ v^{k,0}_{N_R,m} & v^{k,1}_{N_R,m} & \cdots & v^{k,L}_{N_R,m} \end{bmatrix} \end{aligned} \tag{6}$$

being the channel matrix of the mth Tx antenna and the effective noise matrix in $\mathbf{Y}[k]$ w.r.t. $x_m[k]$, respectively.

With this new channel model, it is clear that all information about $x_m[k]$ that we can extract from the channel outputs is fully represented by the following global likelihood function:

$$p(\mathbf{Y}[k] \mid x_m[k]) = p\left(\mathbf{V}^k_m = \mathbf{Y}[k] - \mathbf{H}_m x_m[k]\right). \tag{7}$$

Now, the question is how to calculate this likelihood function in an efficient manner. According to (7), the key for this task is the probability density function (PDF) of the effective noise matrix, that is, $p(\mathbf{V}^k_m)$. As a matter of fact, the main differences between the three Gaussian detectors to be described are in their way of dealing with $p(\mathbf{V}^k_m)$.

3. Statistical Properties of the Effective Noise Matrix

From (3), (4), and (5), we see that the effective noise matrix $\mathbf{V}^k_m$ consists of multi-antenna interference, intersymbol interference, and additive noise samples. Due to the large amount of variables involved in $\mathbf{V}^k_m$, an exact calculation of $p(\mathbf{V}^k_m)$ typically incurs a prohibitive complexity. Therefore, reasonable approximations are necessary to make things easier. In this section, we will carefully study the statistical properties of the effective noise matrix and try to find a way towards complexity reduction.

3.1. Distribution of Effective Noise Samples. Noting that each effective noise sample $v^{k,l}_{n,m}$ is a sum of $N_T(L+1)$ independent random variables, its probability density function may be approximated by a complex Gaussian distribution:

$$p\left(v^{k,l}_{n,m}\right) \approx \frac{1}{\pi\sigma_v^2}\exp\left(-\frac{\left|v^{k,l}_{n,m} - \mu_v\right|^2}{\sigma_v^2}\right), \tag{8}$$

where μ_v and σ_v^2 are defined as

$$\mu_v \doteq \mathrm{E}\left\{v^{k,l}_{n,m}\right\}, \qquad \sigma_v^2 \doteq \mathrm{E}\left\{\left|v^{k,l}_{n,m} - \mu_v\right|^2\right\}. \tag{9}$$

(Here, we neglect the correlation between the real part and the imaginary part. Concerning this issue, interested readers may refer to [7].) According to the rule of thumb, as long as

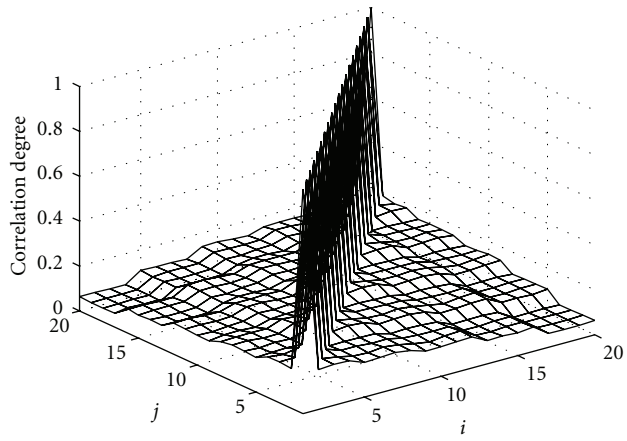


Figure 2: Average magnitude of correlation coefficient between effective noise samples, $N_T = N_R = 4$, $L = 4$, BPSK mapping, and $E_b/N_0 = 4$ dB.

$N_T(L+1) \geq 12$ holds, the accuracy of (8) is satisfying. This approximation is often called Gaussian approximation, and its feasibility in the scenario of MIMO-ISI channels has been proven in the available literature [6, 8, 9].

3.2. Dependence between Effective Noise Samples. Due to more or less common sources of randomness, the elements of $\mathbf{V}_m^k$ are in general statistically dependent on each other. However, it is so far unclear whether this dependence is strong or weak. In the following, we will carry out some numerical measurements to obtain a deeper insight into this issue. Many previous works [6–10] show that $p(\mathbf{V}_m^k)$ can be well approximated by a joint Gaussian distribution, as long as the product $N_T(L+1)$ is large enough. Besides, it is well known that two jointly Gaussian distributed variables are independent if they are uncorrelated, and their dependence structure is completely defined by the correlation coefficient. Therefore, by measuring the correlation between the elements of $\mathbf{V}_m^k$, we will be able to get a rough impression on the dependence between the elements of $\mathbf{V}_m^k$.

First, we define that

$$\mathbf{v} = [v_1, v_2, \ldots, v_Q]^T \doteq \mathrm{vec}\left\{\mathbf{V}_m^k\right\}, \tag{10}$$

with $Q \doteq N_R(L+1)$. $\mathrm{vec}\{\cdot\}$ denotes the column stacking operator and $(\cdot)^T$ denotes the matrix/vector transpose operator. Since for a block-fading channel the statistics of $\mathbf{V}_m^k$ do not change with m and k, the subscript m and the superscript k are omitted in $\mathbf{v}$. Next, we define the magnitude of the correlation coefficient between two effective noise samples as

$$\varphi_{i,j} \doteq \frac{\left|\mathbf{E}\left\{\left(v_i - \mu_{v_i}\right)\left(v_j - \mu_{v_j}\right)^*\right\}\right|}{\sigma_{v_i}\sigma_{v_j}}, \tag{11}$$

where $(\cdot)^*$ denotes complex conjugate. Since $\varphi_{i,j}$ is in fact a function of the random channel taps, we further define

$$\phi_{i,j} \doteq \mathrm{E}\left\{\varphi_{i,j}\right\}, \tag{12}$$

where the expectation is taken over random realizations of channel taps. Last, we collect $\phi_{i,j}$ into a matrix

$$\Phi \doteq \begin{bmatrix} \phi_{1,1} & \phi_{1,2} & \cdots & \phi_{1,Q} \\ \phi_{2,1} & \phi_{2,2} & \cdots & \phi_{2,Q} \\ \cdots & \cdots & \ddots & \cdots \\ \phi_{Q,1} & \phi_{Q,2} & \cdots & \phi_{Q,Q} \end{bmatrix}. \tag{13}$$

Clearly, the entries on the main diagonal of Φ will always be 1, because these entries are the magnitudes of autocorrelation coefficients. For entries not on the main diagonal of Φ, their values reflect the strongness of correlation between effective noise samples and sequentially the strongness of dependence between effective noise samples.

Figure 2 demonstrates the measured values of Φ in a BPSK system with independent Rayleigh fading channel taps and an equal delay power profile. Observing Figure 2, we see that the values of $\phi_{i,j}(i \neq j)$ are small, which means that the correlation between the elements of $\mathbf{V}_m^k$ is actually very weak. As a matter of fact, the correlation between effective noise samples drops steadily as the product $N_T(L+1)$ increases [14]. This observation delivers a good message: it may be feasible to partially or even fully neglect the mutual dependence between the effective noise samples, for the sake of complexity reduction. Certainly, the detailed dependence structure of effective noise samples will be different from Figure 2 if one uses another type of channel delay power profile. However, the contour of Figure 2 holds in general.

4. Factor Graph and Message Passing

Before specific algorithm derivation, we briefly revisit the concept of factor graphs and message passing.

4.1. Factor Graphs and Factorization. A factor graph is a type of bipartite graph which visualizes the factorization of certain global functions object to maximization or minimization. To easily understand it, let us consider a simple example. Suppose that we have a BPSK symbol x with three observations:

$$y_1 = x + n_1, \qquad y_2 = x + n_2, \qquad y_3 = x + n_3, \tag{14}$$

where n_1, n_2, and n_3 are additive noise terms. Assuming that no a priori information is available for x, an optimal detector tries to maximize the global likelihood function according to

$$\hat{x} = \arg \max_{\tilde{x} \in \{\pm 1\}} \{p(y_1, y_2, y_3 \mid \tilde{x})\}. \tag{15}$$

If n_1, n_2, and n_3 are mutually independent, we may factorize the above global likelihood function into a product of local likelihood functions:

$$p(y_1, y_2, y_3 \mid x) = p(y_1 \mid x)p(y_2 \mid x)p(y_3 \mid x), \tag{16}$$

which can be visualized by the factor graph given in Figure 3, where a circle represents a symbol node and a square box represents an observation node.

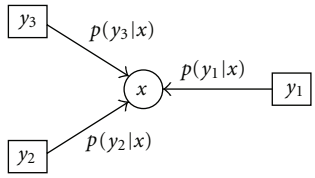
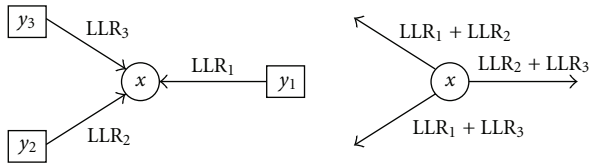
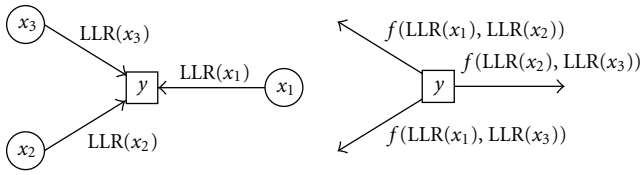


Figure 3: A symbol node connected with three observation nodes.



(a) Message exchange at a binary symbol node



(b) Message exchange at an observation node

Figure 4: Message exchange at different nodes.

4.2. Iterative Message Passing Algorithms. Given a factor graph, the task of variable estimation can be accomplished by combining and exchanging the messages (knowledge) from various sources over this probabilistic network. Such an algorithm is often called an iterative message passing algorithm. For message passing over factor graphs, only extrinsic information should be exchanged and propagated. Although different type of nodes often apply different type of message processing operations, this rule must be carefully followed.

4.3. Message Exchange at Symbol Nodes. For binary variables, it is often convenient to use log-likelihood ratios (LLRs). Define that

$$\mathrm{LLR}_i \doteq \ln \frac{p(y_i \mid x = +1)}{p(y_i \mid x = -1)}, \tag{17}$$

the message exchanging at a BPSK symbol node proceeds as Figure 4(a). The underlying principle is that LLR messages from independent observations are additive. In practice, $\mathrm{LLR}_{\mathrm{sum}} \doteq \sum_i \mathrm{LLR}_i$ is first calculated, then each new message is obtained as $(\mathrm{LLR}_{\mathrm{sum}} - \mathrm{LLR}_i)$. Consequently, the complexity of this operation is always proportional to the amount of edges diverging from this symbol node.

4.4. Message Exchange at Observation Nodes. Considering an observation node connected with three BPSK symbols, the message exchange proceeds as illustrated in Figure 4(b), where $f(\cdot)$ denotes a certain message combining function, often called a message update rule. Different from the situation at symbol nodes, here message combining can no longer be accomplished by a simple linear addition. As a matter of fact, $f(\cdot)$ is the major source of complexity in a graph-based detection algorithm, and hence will be the object of simplification in the remaining part of this paper.

5. Joint Gaussian Detector

According to Section 3, the elements of $\mathbf{V}_m^k$ are roughly Gaussian distributed, and they are in general dependent on each other, although weakly. Hence, a straightforward way to calculate $p(\mathbf{V}_m^k)$ is to approximate the elements of $\mathbf{V}_m^k$ as jointly Gaussian distributed. This approach is usually termed joint Gaussian approximation (JGA), and the algorithm based on this approach is called joint Gaussian detector (JGD), which has been known for years [6–10]. In this section, we will give a clean mathematical derivation of the JGD (For the sake of simple mathematical expression, BPSK mapping is assumed in the rest of the paper.).

5.1. Joint Gaussian Approximation. Using the symbol-oriented channel model (5), the joint Gaussian approximation can be written as

$$p\left(\mathbf{V}_m^k\right) \approx \frac{1}{\pi^Q |\boldsymbol{\Sigma}|^Q} \exp\left(-(\mathbf{v}-\boldsymbol{\mu})^H \boldsymbol{\Sigma}^{-1} (\mathbf{v}-\boldsymbol{\mu})\right) \tag{18}$$

with

$$\begin{aligned} \mathbf{v} &\doteq \mathrm{vec}\left\{\mathbf{V}_m^k\right\}, \qquad \boldsymbol{\mu} \doteq \mathrm{E}\{\mathbf{v}\}, \\ \boldsymbol{\Sigma} &\doteq \mathrm{E}\left\{(\mathbf{v}-\boldsymbol{\mu})(\mathbf{v}-\boldsymbol{\mu})^H\right\}, \qquad Q \doteq N_{\mathrm{R}}(L+1). \end{aligned} \tag{19}$$

Note that $\mathbf{v}$ is a $Q \times 1$ column vector. Therefore, the order of the covariance matrix $\boldsymbol{\Sigma}$ is $Q = N_{\mathrm{R}}(L+1)$. In the literature, however, this covariance matrix usually has an order $N_{\mathrm{R}}K$, where K is the burst length, due to using an output-oriented channel model. The concept of sliding windows is introduced in [6] in order to reduce this order from $N_{\mathrm{R}}K$ to $N_{\mathrm{R}}(L+1)$. Nevertheless, with the symbol-oriented channel model, it is clarified that there is in fact no reason for the order of $\boldsymbol{\Sigma}$ to be related to the burst length.

5.2. Factor Graph with Joint Gaussian Approximation. Applying the joint Gaussian approximation, we admit the mutual dependence between the elements of $\mathbf{V}_m^k$, and hence the PDF $p(\mathbf{V}_m^k)$ as well as the global likelihood function $p(\mathbf{Y}[k]|x_m[k])$ will not be factorizable at all. We also notice that the observation matrices for neighboring data symbols, namely, $\mathbf{Y}[k], \mathbf{Y}[k+1], \ldots, \mathbf{Y}[k+L]$, partially overlap with each other. For these two reasons, the factor graph of a MIMO-ISI channel will look like Figure 5, where $\mathbf{Y}$ denotes the matrix which collects all channel outputs within the current data burst. No factorization exists and also no cycles are present.

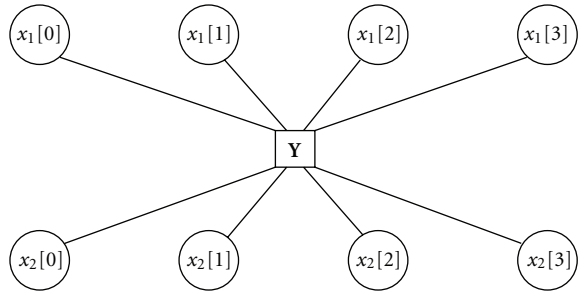


Figure 5: Factor graph of a MIMO-ISI channel with joint Gaussian approximation, $N_T = 2$.

5.3. Message Update Rule at Observation Node. Revisiting (7) and applying (18), the message from an observation node to a symbol node can be calculated as

$$\begin{aligned} \mathrm{LLR}(x_m[k]) &= \ln \frac{p(\mathbf{Y}[k] \mid x_m[k] = +1)}{p(\mathbf{Y}[k] \mid x_m[k] = -1)} \\ &= -(\mathbf{v}_1 - \boldsymbol{\mu})^H \boldsymbol{\Sigma}^{-1} (\mathbf{v}_1 - \boldsymbol{\mu}) \\ &\quad + (\mathbf{v}_2 - \boldsymbol{\mu})^H \boldsymbol{\Sigma}^{-1} (\mathbf{v}_2 - \boldsymbol{\mu}) \end{aligned} \tag{20}$$

with

$$\mathbf{v}_1 = \mathrm{vec}\{\mathbf{Y}[k] - \mathbf{H}_m\}, \qquad \mathbf{v}_2 = \mathrm{vec}\{\mathbf{Y}[k] + \mathbf{H}_m\}. \tag{21}$$

$\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, covering the statistical properties of the effective noise matrix $\mathbf{V}_m^k$, are calculated according to (19), utilizing the incoming LLR messages from all relevant symbol nodes. Due to limited space, we would like to refer interested readers to [6] for a detailed description of this calculation.

5.4. Computational Complexity. The computational complexity of (20) mainly comes from the inversion of the covariance matrix $\boldsymbol{\Sigma}$. Noting that (20) needs to be calculated for N_T data symbols per time index and matrix inversion is an operation with complexity cubic in the matrix order, we have

$$\mathcal{O}(\mathrm{JGD}) \propto N_T {N_R}^3 (L+1)^3. \tag{22}$$

This complexity is much lower than that of the BCJR algorithm, but still is a considerable problem whenever the system possesses many Rx antennas or the channel is severely delay-spread.

6. Grouped Joint Gaussian Detector

In this section, we introduce a grouped joint Gaussian approximation (GJGA) of $p(\mathbf{V}_m^k)$, which brings a significant complexity reduction w.r.t. the joint Gaussian approximation.

6.1. Grouped Joint Gaussian Approximation. From Figure 2 we see that the average magnitude of correlation coefficient between $v_{n_1,m}^{k,i}$ and $v_{n_2,m}^{k,i}$ is constant for all $n_1 \neq n_2$, while the average magnitude of correlation coefficient between $v_{n,m}^{k,i}$ and $v_{n,m}^{k,j}$ drops steadily as the distance $(i\text{-}j)$ increments. This observation inspires us for a new approximation of $p(\mathbf{V}_m^k)$ (Initial work has been presented in [15].). As illustrated in the following expression:

$$\mathbf{V}_m^k = \begin{bmatrix} v_{1,m}^{k,0} & v_{1,m}^{k,1} & \cdots & v_{1,m}^{k,L} \\ v_{2,m}^{k,0} & v_{2,m}^{k,1} & \cdots & v_{2,m}^{k,L} \\ \vdots & \vdots & \ddots & \vdots \\ v_{N_R,m}^{k,0} & v_{N_R,m}^{k,1} & \cdots & v_{N_R,m}^{k,L} \end{bmatrix}. \tag{23}$$

we assume that the columns of $\mathbf{V}_m^k$ are linearly independent from each other while the elements in each column are jointly Gaussian distributed. Mathematically, this approximation can be written as

$$p\left(\mathbf{V}_m^k\right) \approx \prod_{l=0}^{L} p\left(\mathbf{v}_m^{k,l}\right) \tag{24}$$

with

$$\begin{aligned} \mathbf{v}_m^{k,l} &\doteq \left[v_{1,m}^{k,l}, v_{2,m}^{k,l}, \ldots, v_{N_R,m}^{k,l}\right]^T, \\ p\left(\mathbf{v}_m^{k,l}\right) &\propto \exp\left(-\left(\mathbf{v}_m^{k,l} - \boldsymbol{\mu}\right)^H \boldsymbol{\Sigma}^{-1} \left(\mathbf{v}_m^{k,l} - \boldsymbol{\mu}\right)\right), \end{aligned} \tag{25}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are the mean vector and the covariance matrix of $\mathbf{v}_m^{k,l}$, respectively. Note that the order of $\boldsymbol{\Sigma}$ is now only N_R. In the following, we refer to the receiver algorithm based on this approximation as grouped joint Gaussian detector (GJGD).

6.2. Factor Graph with Grouped Joint Gaussian Approximation. Applying the grouped joint Gaussian approximation, we achieve the following factorization:

$$p(\mathbf{Y}[k] \mid x_m[k]) \approx \prod_{l=0}^{L} p(\mathbf{y}[k+l] \mid x_m[k]) \tag{26}$$

with

$$\mathbf{y}[k+l] \doteq \left[y_1[k+l], y_2[k+l], \ldots, y_{N_R}[k+l]\right]^T. \tag{27}$$

The resulting factor graph will look like Figure 6. Now the observation matrix $\mathbf{Y}[k]$ is split into observation vectors $\mathbf{y}[k+l]$. Compared to the factor graph with the JGA, the factor graph with the GJGA becomes more complicated, that is, there are more edges diverging from each symbol node. However, the corresponding detection complexity actually becomes much lower, as explained in Section 6.4.

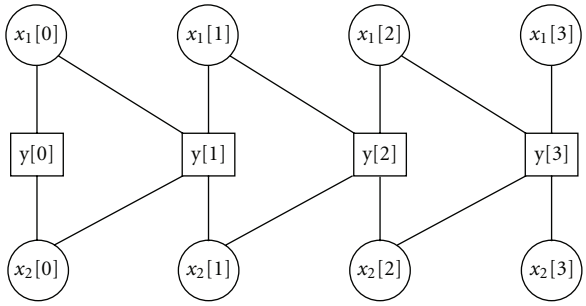


Figure 6: Factor graph of a MIMO-ISI channel with grouped joint Gaussian approximation, $N_T = 2, L = 1$.

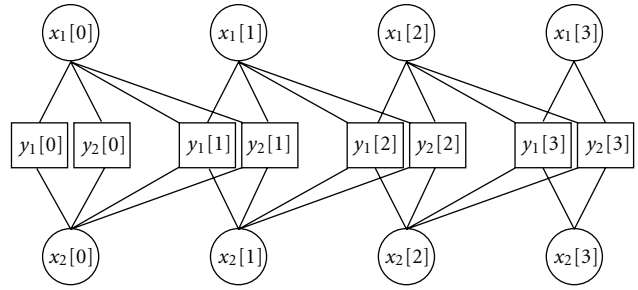


Figure 7: Factor graph of a MIMO-ISI channel with independent Gaussian approximation, $N_T = N_R = 2, L = 1$.

6.3. Message Update Rule at Observation Nodes. With the new approximation, the message updating rule at an observation node can be written as

$$\begin{aligned}\mathrm{LLR}(x_m[k]) &= \ln \frac{p(\mathbf{y}[k+l] \mid x_m[k] = +1)}{p(\mathbf{y}[k+l] \mid x_m[k] = -1)} \\ &= -(\mathbf{v}_1 - \boldsymbol{\mu})^H \boldsymbol{\Sigma}^{-1} (\mathbf{v}_1 - \boldsymbol{\mu}) \\ &\quad + (\mathbf{v}_2 - \boldsymbol{\mu})^H \boldsymbol{\Sigma}^{-1} (\mathbf{v}_2 - \boldsymbol{\mu})\end{aligned} \tag{28}$$

with

$$\begin{aligned}\mathbf{v}_1 &\doteq \mathbf{y}[k+l] - \mathbf{h}_m^l, \\ \mathbf{v}_2 &\doteq \mathbf{y}[k+l] + \mathbf{h}_m^l, \\ \mathbf{h}_m^l &\doteq \left[h_{1,m}^l, h_{2,m}^l, \ldots, h_{N_R,m}^l\right]^T.\end{aligned} \tag{29}$$

The statistical properties $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ can be calculated by utilizing the incoming LLR messages from all relevant symbol nodes. Due to limited space, we would like to refer interested readers to [15] for more details on this topic.

6.4. Computational Complexity. By checking (26) and (28), and noting that the covariance matrix $\boldsymbol{\Sigma}$ is now only of order N_R, we have

$$\mathcal{O}(\mathrm{GJGD}) \propto N_T N_R{}^3 (L+1). \tag{30}$$

Comparing (30) with (22), it is clear that the computational complexity of the GJGD is much lower than that of the JGD, particularly for MIMO systems with severe delay spread. Nevertheless, a cubic term is still present due to matrix inversion.

7. Graph-Based Iterative Gaussian Detector

In this section, we introduce an independent Gaussian approximation (IGA) which completely eliminates matrix inversion and a graph-based iterative Gaussian detector (GIGD) based on that (Initial work has been presented in [14].).

7.1. Independent Gaussian Approximation. In Section 3.2, we mentioned that the cross-correlation between effective noise samples drops steadily as the product $N_T(L+1)$ increases. Therefore, if $N_T(L+1)$ is sufficiently large, we might completely neglect the mutual dependence, that is, to approximate all effective noise samples to be independently Gaussian distributed, as illustrated in the following:

$$\mathbf{V}_m^k = \begin{bmatrix} \nu_{1,m}^{k,0} & \nu_{1,m}^{k,1} & \cdots & \nu_{1,m}^{k,L} \\ \nu_{2,m}^{k,0} & \nu_{2,m}^{k,1} & \cdots & \nu_{2,m}^{k,L} \\ \vdots & \vdots & \ddots & \vdots \\ \nu_{N_R,m}^{k,0} & \nu_{N_R,m}^{k,1} & \cdots & \nu_{N_R,m}^{k,L} \end{bmatrix}. \tag{31}$$

Mathematically, we may write this approximation as

$$p\left(\mathbf{V}_m^k\right) \approx \prod_{n=1}^{N_R} \prod_{l=0}^{L} p\left(\nu_{n,m}^{k,l}\right) \tag{32}$$

with

$$p\left(\nu_{n,m}^{k,l}\right) \approx \frac{1}{\pi\sigma_\nu^2} \exp\left(-\frac{\left|\nu_{n,m}^{k,l} - \mu_\nu\right|^2}{\sigma_\nu^2}\right), \tag{33}$$

where μ_ν and σ_ν^2 are defined as

$$\mu_\nu \doteq \mathrm{E}\left\{\nu_{n,m}^{k,l}\right\}, \qquad \sigma_\nu^2 \doteq \mathrm{E}\left\{\left|\nu_{n,m}^{k,l} - \mu_\nu\right|^2\right\}. \tag{34}$$

7.2. Factor Graph with Independent Gaussian Approximation. Revisiting (7) and applying (32), we achieve the following factorization:

$$p(\mathbf{Y}[k] \mid x_m[k]) \approx \prod_{n=1}^{N_R} \prod_{l=0}^{L} p(y_n[k+l] \mid x_m[k]). \tag{35}$$

The resulting factor graph will look like Figure 7. Now all observations are separately represented in the factor graph, and there are even more edges diverging from each symbol node. However, the corresponding detection complexity is again much lower than that of the GJGD.

7.3. Message Update Rule at Observation Nodes. Combining (4) with (33), the message updating rule at an observation node can be written as

$$\begin{aligned}\mathrm{LLR}(x_m[k]) &= \ln \frac{p(y_n[k+l] \mid x_m[k]=+1)}{p(y_n[k+l] \mid x_m[k]=-1)} \\ &= -\frac{\left| y_n[k+l]-\mu_v-h_{n,m}^l \right|^2}{\sigma_v^2} \\ &\quad + \frac{\left| y_n[k+l]-\mu_v+h_{n,m}^l \right|^2}{\sigma_v^2} \\ &= 4\frac{\mathrm{Re}\left\{h_{n,m}^l (y_n[k+l]-\mu_v)^*\right\}}{\sigma_v^2},\end{aligned} \tag{36}$$

with μ_v and σ_v^2 as defined in (34) and the way of calculating them described in the following.

Revisiting Figure 7, we see that each observation node is connected with $G \doteq N_{\mathrm{T}}(L+1)$ symbol nodes. Replacing complicated indices n, m, k, and l by a single index i, we may simplify the relationship between an observation and its associated data symbols as

$$\begin{aligned}y &= \sum_{i=1}^{G} h_i x_i + w \\ &= h_j x_j + \sum_{i=1, i\neq j}^{G} h_i x_i + w \\ &= h_j x_j + v_j,\end{aligned} \tag{37}$$

with $v_j = \sum_{i=1, i\neq j}^{G} h_i x_i + w$ denoting the effective noise sample w.r.t. x_j. Since all data symbols are mutually independent, the following statement is straightforward:

$$\begin{aligned}\mu_{v_j} &= \sum_{i=1, i\neq j}^{G} h_i \cdot \mu_{x_i}, \\ \sigma_{v_j}^2 &= \sum_{i=1, i\neq j}^{G} |h_i|^2 \cdot \sigma_{x_i}^2 + \sigma_w^2,\end{aligned} \tag{38}$$

where μ_{x_i} and $\sigma_{x_i}^2$ are calculated by utilizing the incoming LLR message from the symbol node:

$$\mu_{x_i} = \frac{e^{\mathrm{LLR}(x_i)}-1}{e^{\mathrm{LLR}(x_i)}+1}, \qquad \sigma_{x_i}^2 = 1-\mu_{x_i}^2. \tag{39}$$

Note that the principle of extrinsic information is implicitly applied in this message updating operation.

7.4. Computational Complexity. The computational load of the GIGD comes from the message updating at the symbol nodes and the observation nodes. Revisiting Figure 7, we find that there are N_{T} symbol nodes per time index, each connected with $N_{\mathrm{R}}(L+1)$ edges. Since the complexity of message exchange at a symbol node is always proportional to the amount of associated edges (c.f. Section 4.3), we have

$$\mathcal{O}(\text{operation at symbol nodes}) \propto N_{\mathrm{T}} N_{\mathrm{R}}(L+1). \tag{40}$$

In each iteration, an observation node needs to calculate the LLR values of $G = N_{\mathrm{T}}(L+1)$ data symbols associated with it. In practice, this task is accomplished in two steps. In step one, μ_{x_i} and $\sigma_{x_i}^2$ are first calculated for $i = 1, 2, \ldots, G$, according to (39). Afterwards, the products $h_i\mu_{x_i}$ and $|h_i|^2\sigma_{x_i}^2$ as well as the summations $\sum_{i=1}^{G} h_i\mu_{x_i}$ and $(\sum_{i=1}^{G} |h_i|^2\sigma_{x_i}^2 + \sigma_w^2)$ are calculated and stored. Obviously, the complexity of this step is proportional to G. In step two, the following calculation:

$$\mu_{v_j} = \sum_{i=1}^{G} h_i\mu_{x_i} - h_j\mu_{x_j}, \tag{41}$$

$$\sigma_{v_j}^2 = \left(\sum_{i=1}^{G} |h_i|^2\sigma_{x_i}^2 + \sigma_w^2\right) - \left|h_j\right|^2\sigma_{x_j}^2, \tag{42}$$

is performed and then $\mathrm{LLR}(x_j)$, $j = 1, 2, \ldots, G$, is obtained according to (36). Since the two sums in (41) and (42) have already been stored in step one, the complexity of step two is proportional to $G = N_{\mathrm{T}}(L+1)$ as well. Given this explanation and noting that there are N_{R} observation nodes per time index, we may conclude that

$$\mathcal{O}(\text{operation at observation nodes}) \propto N_{\mathrm{T}} N_{\mathrm{R}}(L+1). \tag{43}$$

We may recognize that $N_{\mathrm{T}}N_{\mathrm{R}}(L+1)$ actually gives the number of channel taps. In reality, however, the discrete-time channel model often has a sparse ISI structure, that is, many channel taps are quasi zero. In this case, the edges associated with zero taps can safely be removed from the factor graph (c.f. Figure 7). Given this knowledge, and combining (40) and (43), we obtain the following expression:

$$\begin{aligned}\mathcal{O}(\mathrm{GIGD}) &\propto \text{number of nonzero channel taps} \\ &\leq N_{\mathrm{T}} N_{\mathrm{R}}(L+1).\end{aligned} \tag{44}$$

Due to the complete elimination of matrix inversion, the complexity of the GIGD is truly linear. Besides, the GIGD is very attractive for sparse ISI channels, where the maximum delay spread is large while many zero taps are present. Note that neither the JGD nor the GJGD is able to benefit from the sparse ISI channel structure in such a straightforward manner, because of multivariate Gaussian approximations.

8. Performance in Uncoded Systems

In previous sections, we have introduced three low-complexity Gaussian detection algorithms, namely, joint Gaussian detector (JGD), grouped joint Gaussian detector (GJGD), and graph-based iterative Gaussian detector (GIGD). In this section, we provide numerical results from Monte Carlo simulations to assess and compare the performance of these three algorithms in uncoded systems, and ultimately illustrate the merits and demerits of the GIGD algorithm.

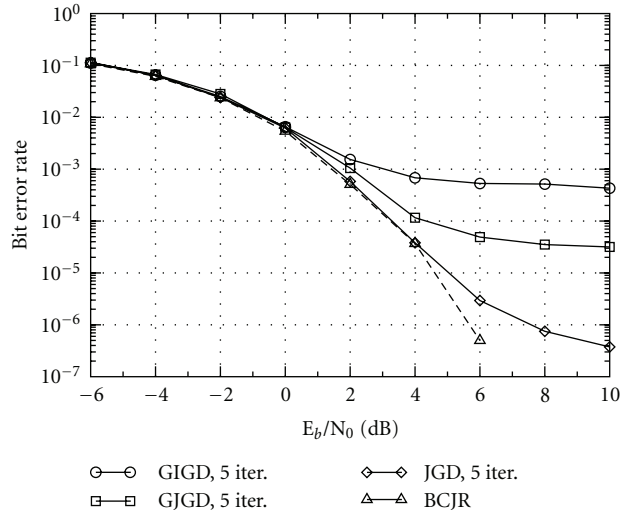


FIGURE 8: Performance of the three Gaussian detectors in an uncoded system, $N_T = N_R = 4$, $L = 4$, and $K = 400$.

8.1. Simulation Setup. Each burst from each Tx antenna contains 400 data symbols. After one burst is transmitted, each Tx antenna ceases transmission for an interval of L symbol durations to avoid interburst interference, where L denotes the effective channel memory length. All Tx and Rx antennas are assumed to be perfectly synchronized. For simplicity, the signal mapping scheme is always BPSK. The channel coefficients $h^l_{n,m}$ of every subchannel are normalized to form an equal delay power profile with an average sum power of one, that is, $\mathrm{E}\{|h^l_{n,m}|^2\} = 1/(L+1)$ with $\sum_{l=0}^{L} \mathrm{E}\{|h^l_{n,m}|^2\} = 1$. For a fair comparison, for all three Gaussian detection algorithms, 5 iterations are performed, that is, the operations of message updating and message exchanging are repeated 5 times.

8.2. Theoretical Performance Bound. Observing the architecture of the JGD, the GJGD, and the GIGD, these three algorithms clearly fall into the class of symbol-by-symbol detectors, as they all try to maximize the global likelihood function w.r.t. individual symbols. Therefore, the symbol-by-symbol MAP detector provides a lower bound for the achievable BER performance in uncoded systems. Here, we use the BCJR algorithm [3] to implement the symbol-by-symbol MAP detector. Certainly, given the BCJR algorithm, no receiver iterations are necessary for uncoded transmission.

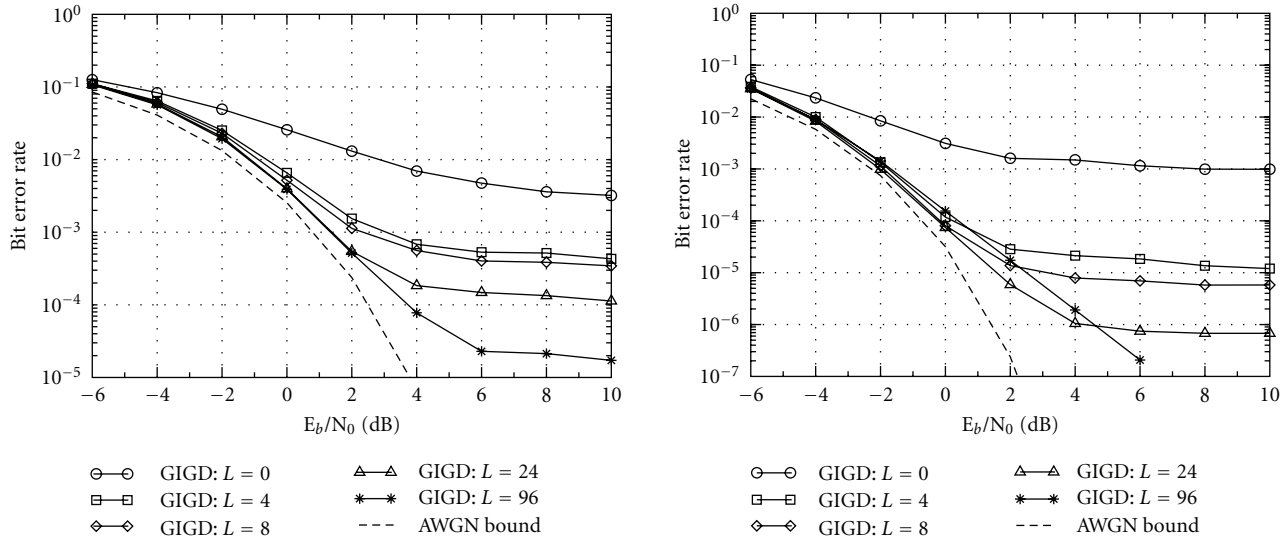
8.3. Performance Comparison. Figure 8 displays the BER performances of the three Gaussian detection algorithms. As can be seen, the JGD algorithm achieves a BER performance very close to that of the BCJR algorithm. It shows a trivial error floor at high SNRs due to the inaccuracy of (18) and feeding back intrinsic information as a priori information. (In an uncoded system, the factor graph for the JGD is cycle-free, c.f. Figure 5. Therefore, a self-feedback is enforced at all symbol nodes in order to implement an iterative detection. The JGD algorithm for uncoded systems in fact falls into the class of probabilistic data association (PDA) algorithms [16]. Nevertheless, it is not necessary and also not proper to do so in a coded system, since the existence of code nodes enables rigorous extrinsic information exchange.) Compared to the JGD, the GJGD algorithm shows a performance loss of approximately 1 dB at BER $= 10^{-4}$. Due to the further inaccuracy introduced by (24), the error floor of the GJGD is higher than that of the JGD and is no longer trivial. The performance of the GIGD algorithm is undesirable in this scenario. It shows a significant error floor at BER $\approx 5 \times 10^{-4}$ due to the coarseness of the approximation given in (32).

8.4. Complexity Comparison. As a matter of fact, the introduced three Gaussian detection algorithms do not really differ in the necessary number of iterations. Though applying different type of approximations, these algorithms never change the amount of channel outputs ($y_n[k]$) that a symbol node can extract information from. Consequently, the speed of information aggregation does not change for these three algorithms, and the required number of iterations for a satisfactory BER performance basically stays constant for a fixed system setup. Given a reasonable burst length, 5 iterations are already good enough, empirically.

For the current system setup, the covariance matrices to be inverted are of order $N_R(L+1) = 20$ in the JGD algorithm. The covariance matrices are only of order $N_R = 4$ in the GJGD algorithm. Finally, matrix inversion is completely eliminated in the GIGD algorithm. Revisiting (22), (30), and (44), we will find that the complexity of the GJGD is about 25 times lower than that of the JGD, and the complexity of GIGD is about 16 times lower than that of the GJGD. In total, a complexity reduction of factor 400 is achieved by the GIGD algorithm w.r.t. the JGD algorithm. As such a complexity reduction is rather attractive, it is worthwhile to study the error floor behavior of the GIGD algorithm.

8.5. Error Floor of the GIGD. The error floor of the GIGD algorithm is mainly caused by approximating the elements of the effective noise matrix to be mutually independent. As mentioned in Section 3.2, the average correlation coefficient between effective noise samples drops when the product $N_T(L+1)$ increases. Therefore, we may expect the error floor of the GIGD to drop when the channel memory length becomes larger or when the system deploys more antennas. To verify our conjecture, we again utilize Monte Carlo simulations.

Figure 9 demonstrates the behavior of the GIGD under different channel memory lengths. Since the complexity of the BCJR algorithm and the JGD algorithm both become prohibitive for severely delay spread MIMO channels, we use the BER bound of an AWGN channel as an asymptotic performance bound if L approaches infinity. As predicted, the error floor drops as the channel memory length increases and/or the number of antennas increases. This observation reveals two issues. First, the independent Gaussian approximation (32) benefits from a large amount of channel taps. Second, despite its extremely low complexity achieved by



(a) $N_T = N_R = 4$ (b) $N_T = N_R = 8$

Figure 9: Performance of the GIGD in uncoded systems, $K = 400$, 5 iterations.

making a very coarse approximation, the GIGD is able to exploit the diversity provided by additional channel taps or receive antennas. The cross-over at $N_T = N_R = 8$ and $L = 96$ is mainly caused by the zero-padding burst structure. Both at the beginning and the end of the burst, the channel outputs are composed of few data symbols and a lot of zeros, which degrades the accuracy of the independent Gaussian approximation. This effect is significant at $L = 96$, given the burst length is $K = 400$. By applying a tail-biting burst structure or a cyclic prefix, this problem can be well eliminated, and the resulting performance will be very close to the AWGN bound.

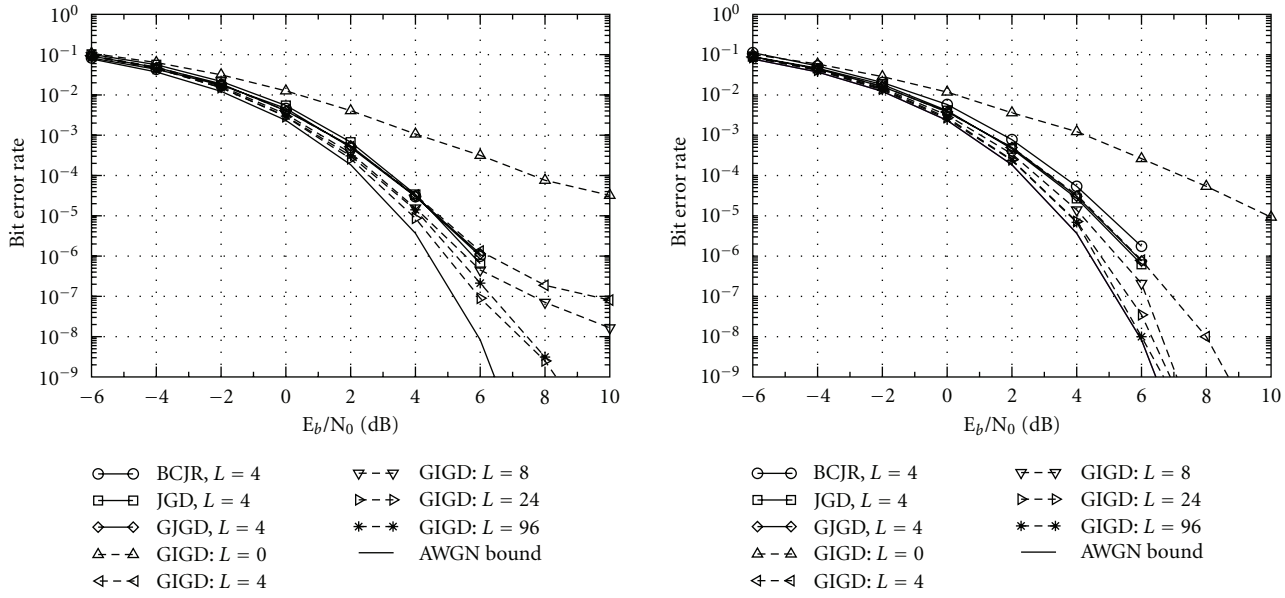
The above results suggest that the GIGD algorithm is very attractive for large systems with severe delay spread. Nevertheless, the GIGD causes a significant error floor when $N_T(L + 1)$ is not sufficiently large. The question that remains is if this error floor can be eliminated by means of channel coding.

9. Performance in Coded Systems

In this section, we check the BER performance of the three Gaussian detectors in coded systems.

9.1. Simulation Setup. For simplicity and for an easy derivation of performance bounds, we adopt repetition encoding with scrambling. The scrambling pattern is fixed, that is, every second bit of a code word is flipped. In case of short data bursts, scrambling is very helpful for the three Gaussian detectors, since they assume that all data symbols come with zero mean. Random interleaving is applied after scrambling in order to make neighboring data symbols as independent as possible. No matter which coding rate is used, the number of symbols per burst per antenna is always $K = 400$. Due to the presence of channel decoding, local iterations in the graph of Gaussian detection are no longer desirable, particularly for the case of JGD. Hence, each receiver iteration contains the following sequential operations: message updating at observation nodes, message updating at symbol nodes, channel decoding, and message updating at symbol nodes. As the use of different Gaussian approximations does not really change the speed of information aggregation at symbol nodes, in the following we will always apply a fixed number of iterations for comparing the performance of using different Gaussian approximations.

9.2. Performance Comparison. Figure 10(a) illustrates the performance of the three Gaussian detectors in a rate 1/2 repetition encoded system. Surprisingly, all three Gaussian detectors as well as the BCJR algorithm show nearly the same performance at $L = 4$, regardless of their huge complexity difference. A purely theoretical analysis of this phenomenon appears difficult. An empirical answer is that the strongest detector for an uncoded system is not necessarily the best one for a coded system. From the JGD to the GJGD, and from the GJGD to the GIGD, more and more coarse approximations are made, which makes the detector outputs less and less accurate. However, this also makes the detector outputs less and less correlated, which is beneficial to the following channel decoder. From Figure 10(a), it seems that the effect of less accuracy is partially compensated by the effect of less correlation. Figure 10(b) further supports our supposition on this issue. In a rate 1/4 repetition encoded system, the performance of the BCJR algorithm is even worse than that of the three Gaussian detectors at $L = 4$. When the coding rate drops, the strong correlation of the outputs of the BCJR algorithm noticeably degrades the system performance, while the three Gaussian detectors



(a) $R = 1/2$ (b) $R = 1/4$

Figure 10: Performance comparison in repetition encoded systems, $N_T = N_R = 4$, $K = 400$, 5 iterations.

stay robust. Among the four algorithms, the GIGD has a decisively lower complexity, meanwhile its BER performance is not worse than that of any other. Therefore, it is the most attractive solution.

9.3. Error Floor of the GIGD. Figures 10(a) and 10(b) also demonstrate the BER performance of the GIGD in repetition encoded systems with various channel memory lengths. Since a repetition code does not provide any coding gain, the AWGN bound still holds. At $R = 1/2$, error floors are still present, but are no longer significant. At $R = 1/4$, error floors nearly disappear, even for $L = 0$, that is, flat-fading channels. The reason of the cross-over at $L = 96$ and $R = 1/2$ is still the zero-padding burst structure. Nevertheless, this effect is well mitigated by the rate 1/4 repetition code. So far we may recognize that repetition encoding is really helpful in mitigating the estimation errors caused by the independent Gaussian approximation, and meanwhile the approximation errors do not present a problem to the convergence property of the repetition decoder. The asymptotic AWGN bound is quasi-approached at $N_T = N_R = 8$, $L = 96$. Note that with this system setup, it is practically impossible to run the BCJR algorithm and it is computationally prohibitive to run the JGD. For systems with short memory lengths, repetition encoding is truly helpful for the GIGD. Necessary to be mentioned, the AWGN bound is only achievable for systems with very large channel memory lengths, since only then the channel instant power tends to be constant. By checking the performance of GIGD with small L values, we may recognize that these curves should also be quasi-bound approaching. Therefore, in repetition encoded systems, the GIGD is applicable for systems with moderate number of antennas and short channel memory lengths as well.

10. Conclusions and Future Work

In this paper, we revisited and slightly revised the joint Gaussian detection (JGD) algorithm, derived the grouped joint Gaussian detection (GJGD) algorithm, and proposed the graph-based iterative Gaussian detection (GIGD) algorithm. A mathematical derivation as well as a detailed performance analysis is provided. From the JGD to the GJGD and from the GJGD to the GIGD, the computational complexity dramatically decreases. The GIGD algorithm has a linear complexity and provides a promising performance for MIMO channels with severe delay spread. In [17], the incorporation of the GIGD algorithm with soft channel estimation has been studied.

The adopted channel model within this paper is very specific, in the sense that it presents the biggest challenge for conventional linear equalizers. Using such a channel model effectively exhibits the high potential of the proposed low-complexity Gaussian detection algorithms, particularly GIGD. Nevertheless, from an engineering standpoint, it deserves to be an interesting topic to test the performance of Gaussian detection with more realistic channel models. Repetition coding is considered within this paper for the sake of easy analysis as well as its strength in mitigating estimation errors due to approximation. Future work should also be targeted at more advanced code structures, particularly concatenations of a repetition code and a sparse graph code, for example, an LDPC code.

Acknowledgments

The authors would like to thank Shan Jiang and Ying Yu for their effort on this topic during their master theses work.

This work has been supported by the German Research Foundation (DFG) under Contract nos. HO 2226/8-1 and HO 2226/10-1.

References

[1] G. D. Forney Jr., "Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference," *IEEE Transactions on Information Theory*, vol. 18, no. 3, pp. 363–378, 1972.

[2] G. Ungerboeck, "Adaptive maximum-likelihood receiver for carrier-modulated data-transmission systems," *IEEE Transactions on Communications*, vol. 22, no. 5, pp. 624–636, 1974.

[3] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, vol. 20, no. 2, pp. 284–287, 1974.

[4] A. Bahai, B. Saltzberg, and M. Ergen, *Multi Carrier Digital Communications: Theory and Applications of OFDM*, Springer, New York, NY, USA, 2004.

[5] X. Wang and H. Vincent Poor, "Iterative (Turbo) soft interference cancellation and decoding for coded CDMA," *IEEE Transactions on Communications*, vol. 47, no. 7, pp. 1046–1061, 1999.

[6] S. Liu and Z. Tian, "Near-optimum soft decision equalization for frequency selective MIMO channels," *IEEE Transactions on Signal Processing*, vol. 52, no. 3, pp. 721–733, 2004.

[7] Y. Jia, C. Andrieu, R. J. Piechocki, and M. Sandell, "Gaussian approximation based mixture reduction for near optimum detection in MIMO systems," *IEEE Communications Letters*, vol. 9, no. 11, pp. 997–999, 2005.

[8] X. Yuan, K. Wu, and L. Ping, "The jointly Gaussian approach to iterative detection in MIMO systems," in *Proceedings of the IEEE International Conference on Communications (ICC '06)*, Istanbul, Turkey, September 2006.

[9] P. H. Tan and L. K. Rasmussen, "Asymptotically optimal nonlinear MMSE multiuser detection based on multivariate Gaussian approximation," *IEEE Transactions on Communications*, vol. 54, no. 8, pp. 1427–1438, 2006.

[10] Y. Jia, C. Andrieu, R. J. Piechocki, and M. Sandell, "Gaussian approximation based mixture reduction for joint channel estimation and detection in MIMO systems," *IEEE Transactions on Wireless Communications*, vol. 6, no. 7, pp. 2384–2389, 2007.

[11] D. Divsalar, M. K. Simon, and D. Raphaeli, "Improved parallel interference cancellation for CDMA," *IEEE Transactions on Communications*, vol. 46, no. 2, pp. 258–268, 1998.

[12] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, 2001.

[13] H.-A. Loeliger, "An introduction to factor graphs," *IEEE Signal Processing Magazine*, vol. 21, no. 1, pp. 28–41, 2004.

[14] T. Wo and P. A. Hoeher, "A simple iterative Gaussian detector for severely delay-spread MIMO channels," in *Proceedings of the IEEE International Conference on Communications (ICC '07)*, pp. 24–28, Glasgow, Scotland, June 2007.

[15] T. Wo, J. C. Fricke, and P. A. Hoeher, "A graph-based iterative Gaussian detector for frequency-selective MIMO channels," in *Proceedings of the IEEE Information Theory Workshop (ITW '06)*, Chengdu, China, October 2006.

[16] J. Ch. Fricke, M. Sandell, J. Mietzner, and P. A. Hoeher, "Impact of the Gaussian approximation on the performance of the probabilistic data association MIMO decoder," *EURASIP Journal on Wireless Communications and Networking*, vol. 2005, no. 5, pp. 796–800, 2005.

[17] T. Wo, C. Liu, and P. A. Hoeher, "Graph-based soft channel and data estimation for MIMO systems with asymmetric LDPC codes," in *Proceedings of the IEEE International Conference on Communications (ICC '08)*, pp. 620–624, Beijing, China, May 2008.

Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2010, Article ID 282493, 8 pages
doi:10.1155/2010/282493

Research Article

Iterative Signal Processing for Blind Code Phase Acquisition of CDMA 1x Signals for Radio Spectrum Monitoring

Ron Kerr and John Lodge

Satellite Communications and Radio Propagation, Communications Research Centre Canada, 3701 Carling Avenue, Box 11490, Station H, Ottawa, ON, Canada K2H 8S2

Correspondence should be addressed to Ron Kerr, ron.kerr@crc.gc.ca

Received 10 February 2010; Revised 14 June 2010; Accepted 2 August 2010

Academic Editor: Peter Hoeher

This paper addresses the problem of recovering the code phase of the composite spreading sequence for a CDMA 1x signal transmitted from a handset, without the benefit of a priori information from the system. The spreading code is required for the radio spectrum monitoring system for signal detection and measurements rather than for communications. The structure of the CDMA 1x signal is exploited by processing sequential pairs of received samples to form a single soft sample for each pair. The approach models the combination of the long-code generator and the two short-code generators, along with the pair-wise processing, by a single linear system over GF(2), with the initial states of the long- and short-code generators forming the input vector. Consequently, a vector of the pair-wise soft samples can be treated as a noisy received codeword that is decoded using iterative soft-in decoding techniques. If the decoder yields the correct candidate "codeword," the original states of the code generators can be computed. This approach does not require direct access to the transmitted spreading sequence but can be applied to the data modulated signal. Simulation results provide performance estimates of the method with noise, Rayleigh fading, and co-channel interference.

1. Introduction

Code division multiple access (CDMA) has become a popular choice for personal communications systems. With the multiple-access scheme multiple users can simultaneously share the same frequency band by using high-rate spreading codes to disperse their transmitted power over the fully-allotted frequency band, thereby keeping the power spectral density for each user at a relatively low level. CDMA represents a new set of challenges for spectrum regulators that need to monitor radio traffic volumes and check license compliance as part of their responsibility to efficiently manage the frequency spectrum in a region. To recover one of the transmitted signals, the receiver generates a local replica of the transmitter's spreading code and uses it to "despread" the desired signal. If the spreading code for the desired signal is suitably distinct from the spreading codes for the other signals present in the band, the desired signal can be received with relatively little interference. Thus, it is important that each user is assigned a different spreading code. In CDMA 1x, all users generate their own spreading codes using the same linear feedback shift registers (LFSR). The requirement of assigning distinct spreading codes is achieved by assigning each user a different starting point in the long spreading code. Thus, each user's code is a distinct time-shifted version of the shared code, and synchronizing to any given user's code is essentially determining the time reference, or equivalently code phase, for that user.

As the network assigns the starting point in the sequence, a receiving base station knows, within a small uncertainty region, the code phase of all of the users assigned to it. A blind acquisition technique is not required for the base station as it has the users' code phase information. The work described in this paper is motivated by the application of spectrum monitoring, where it is necessary to perform measurements on the spectrum usage and to locate transmitters. This is a difficult problem as the mobiles' signals use the same frequency bandwidth and the equipment must operate

without information from the cellular networks. A method to achieve the desired goals is to blindly acquire the code phase of a user in the area to aid in signal detection, signal parameter measurements, direction-finding, and interference cancelation in order to detect other users.

An introduction to acquisitions techniques used for direct sequence spread spectrum (DSSS) is presented in [1]. Three types of acquisition schemes are serial search, parallel search, and sequential estimation. For the sake of discussion, consider the case where there is a spreading sequence which has no underlying structure that can be utilized to acquire the code phase, the sequence has a long period and the receiver has no information about the expected code phase. In a serial search strategy, the receiver would have to test each code phase serially until a correlation with the received signal indicates that the sequences are aligned. The expected number of required tests is approximately half the sequence length [2]. In parallel search techniques, more correlators are used in the receiver to perform multiple tests simultaneously. This approach can reduce the acquisition time compared with the serial approach but with an increase in complexity. In the case of very long code periods, the serial and parallel techniques quickly become impractical due to their expected number of correlations required. It should be noted that there are long spreading sequences that are made up of several shorter component sequences, such as PN ranging codes, which are discussed in [3]. The structure of the shorter component codes can be used to rapidly acquire the code phase of the long spreading code. This type of spreading sequence is not considered here. In this paper, the CDMA 1x cellular standard [4] is used to illustrate the blind acquisition method. The spreading sequence is made up of a long pseudonoise (PN) sequence with period $2^{42} - 1$ and two short PN sequences with period of 2^{15}. The serial and parallel approaches are computationally impractical for acquiring the sequence quickly due to the expected number of correlations required.

The third class of search strategies for acquisition is sequential estimation. Here, the detector uses information on the chip sequence to estimate the state of the shift register. In [5], hard chip estimates were made and loaded into the spreading generator. The method did not use any of the redundancy in the chip sequence. In [6], orthogonal parity check equations and threshold decoding to improve the chip decisions that were then loaded into the spreading generator were utilized. The n chips from a spreading sequence from an LFSR was recognized as a truncated codeword of a maximum length code in [7]. Majority logic decoding was used on a number of independent estimates of the bits in the spreading generator's state.

The above methods used hard decisions on the chips in the sequence. Soft decisions were introduced in [2] for majority logic decoding and the RARASE algorithm [8]. In [9], low-density parity check (LDPC) decoding was used to find the state of an LFSR spreading generator. In [10], the method presented used recursive soft sequential estimation (RSSE) and a soft-in soft-out decoder to update the reliabilities associated with the chips in the sequence. A soft-chip register that stored the current reliabilities was used. The soft-in-soft-out (SISO) decoder was used to update the current chip reliability with information from the channel and the soft-chip register. A differential recursive soft sequential estimator (DRSSE) algorithm was introduced in [11] and eliminated the need for an accurate carrier frequency estimate required in the RSSE algorithm. In [12–14], iterative message passing algorithms were considered for finding the state of spreading sequence generators.

An approach for acquisition of DS-CDMA signals is to treat the spreading code acquisition problem as a decoding problem. In many cases, the spreading code used in the CDMA system is generated by a linear system. The structure of the linear system defines a linear code. In the above references, the problem of acquiring the phase of the spreading sequence using decoding methods was considered. However, there was an assumption that the receiver had access to the CDMA signal modulated only by a spreading sequence (i.e., the chip decisions) and not by data. It is felt that the DRSSE algorithm [11] would not be affected by the data, but the other algorithms would be adversely affected in the presence of data modulation, especially if the spreading factor (i.e., number of bits per chip) is less than the observation interval. The presence of data modulation prevents access to the chip values from the channel and thus the algorithms requiring chip decisions will not work for all cases.

The algorithm that will be presented is capable of acquiring the spreading code when data modulation is present on the signal. In the CDMA 1x system, the effect of the data modulation can be eliminated on the single soft sample that results from each pair of sequential samples. As a result, the algorithm is independent of the data modulated on the signal and can acquire during any phase of the communication. Another benefit of processing sequential pairs of channel samples is the algorithm becomes robust against unknown carrier frequency offsets. In the case of CDMA 1x, the knowledge of the frame structure can be used to quickly eliminate a large number of incorrect spreading generator states without attempting the despreading of the signal.

In this paper, we will present an algorithm that uses iterative soft input decoding to acquire the code phase of a CDMA 1x spread signal without any system information available. This allows the receiver to regenerate the spreading sequence required for despreading of the signal. An application of interest is for use in a spectrum monitoring and signal detection. Once the spreading sequence is found, the signal can be despread for use in monitoring or direction finding.

The outline for the remainder of this paper is as follows. In Section 2, the development of the equations and decoding algorithm will be presented. Performance results will be presented in Section 3, and Section 4 will provide concluding remarks.

2. System Model

In several cellular standards [4, 15, 16], a complex-valued spreading sequence is formed from two bipolar sequences as shown in Figure 1, where c_1 and c_2 are bipolar sequences, which take on values of ± 1. The real component of the

complex sequence (i.e., Real(C_n)) is one of the bipolar sequences. The imaginary component is formed by the product of the first sequence, a decimation by two of the second sequence, and an alternating sequence. The resulting complex sequence C can be described by

$$C_n = c_{1,n}(1 + j(-1)^n c_{2,2\lfloor n/2 \rfloor}), \tag{1}$$

where C_n is the nth value of the complex spreading sequence C, $c_{m,n}$ is the nth value of the mth bipolar sequence, $j = \sqrt{-1}$, and $\lfloor \cdot \rfloor$ is the floor function where the function returns the nearest integer below the argument.

In acquiring the spreading sequence of an unknown transmitter, it is assumed that the structure of the spreading generators that generate the c_1 and c_2 sequences and the procedures for initializing generators are known. For the purposes of the presentation it is assumed that the spreading factor is even, a minor modification of the technique would be required if the spreading factor was odd. In the following, the signal is sampled at one sample per chip and chip timing is perfect. However, generalization to an oversampled signal with imperfect timing is straightforward.

In this section, the method of processing the signal so that the resulting signal can be considered to be a linear code depending on the initial state of the spreading generator is shown. Consider two adjacent complex samples C_n and C_{n+1} of the sequence defined in (1), where n is an even number. The samples are defined by

$$\begin{aligned} C_n &= c_{1,n} + jc_{1,n}c_{2,n}, \\ C_{n+1} &= c_{1,n+1} - jc_{1,n+1}c_{2,n}. \end{aligned} \tag{2}$$

Note that the contribution to the sequence from c_2 is the same for both chips due to the decimation process. The data associated with the nth chip is denoted by $D_n = d_{1,n} + jd_{2,n}$ where the data bits d_1 and d_2 are bipolar valued. The spreading is a complex spreading process, so the two corresponding samples after spreading are given by

$$\begin{aligned} r_n &= (d_{1,n}c_{1,n} - d_{2,n}c_{1,n}c_{2,n}) \\ &\quad + j(d_{1,n}c_{1,n}c_{2,n} + d_{2,n}c_{1,n}), \\ r_{n+1} &= (d_{1,n+1}c_{1,n+1} + d_{2,n+1}c_{1,n+1}c_{2,n}) \\ &\quad + j(d_{1,n+1}c_{1,n+1}c_{2,n} - d_{2,n+1}c_{1,n+1}). \end{aligned} \tag{3}$$

The product of an even index (n) sample r_n and the conjugate of the next sample (i.e., $r_n r_{n+1}^*$) will be denoted by y_n and is given by

$$\begin{aligned} y_n = 2c_{1,n}c_{1,n+1}c_{2,n}((d_{1,n}d_{2,n+1} - d_{1,n+1}d_{2,n}) \\ + j(d_{1,n}d_{1,n+1} + d_{2,n}d_{2,n+1})). \end{aligned} \tag{4}$$

The assumption that the spreading factor used for the data is an even number and is aligned with the alternating sequence results in the property that $d_{1,n} = d_{1,n+1}$ and $d_{2,n} = d_{2,n+1}$ when n is an even number. With this property, (4) reduces to

$$y_n = 4jc_{1,n}c_{1,n+1}c_{2,n}. \tag{5}$$

Thus, from (5), the value of y_n is imaginary and proportional to the product of three bipolar sequences and does not depend on the data being transmitted on the channel. The individual sequences can be formed by linear equations on the original state of the spreading generators. Since the product of bipolar elements corresponds to addition in the Galois Field GF(2), this means that the value of y_n can be expressed as a linear binary equation on the original state of the spreading generators. A truncated sequence composed of

$$\mathbf{y^b} = y_n, y_{n+2}, y_{n+4}, \ldots, y_{n+m} \tag{6}$$

can then be generated by a system of m linear equations based on the original state of the spreading generator. The linear equations can be formed into a binary matrix equation where $\mathbf{G}$ is the generator matrix and $\mathbf{x}$ is the state of the spreading generator, given by

$$\mathbf{y} = \mathbf{G}\mathbf{x}, \tag{7}$$

where $\mathbf{y}$ is the binary image of the bipolar sequence $\mathbf{y^b}$ (i.e., "4j" is mapped to 0 and "$-$4j" is mapped to 1), and the arithmetic in (7) is over GF(2). Let the dimension of the state of the shift register be k. If m is greater than k, then (7) is equivalent to the encoding of an (m, k) code, with $\mathbf{y}$, $\mathbf{G}$, and $\mathbf{x}$ being the m-dimensional codeword, the generator matrix for the code and the k-dimensional information bits, respectively. As the truncated sequence can be seen to a codeword of a linear code, a parity matrix $\mathbf{H}$ can be found such that

$$\mathbf{HG} = \mathbf{0} \tag{8}$$

over GF(2).

A decoder algorithm can utilize the constraints specified by the code and the noisy received version of the codeword to obtain an estimate of the codeword $\mathbf{y}$. Once a codeword estimate is found, then (7) can be used to solve for the $\mathbf{x}$. An efficient method is to compute a pseudoinverse matrix, $\mathbf{P}^{\#}$ which satisfies the following equation:

$$\mathbf{x} = \mathbf{P}^{\#}\mathbf{y}. \tag{9}$$

The $\mathbf{P}^{\#}$ is constant for a given generator matrix and thus can be computed and stored for a given choice of block size. Thus, $\mathbf{x}$ can be solved with a matrix-vector multiplication given a codeword estimate.

2.1. CDMA 1x Information. A block diagram of the spreading generator for CDMA 1x is shown in Figure 2 [4]. The complex spreading code is generated by two bipolar sequences. The two sequences are generated by three independent linear feedback shift registers (LFSR) sequences denoted as the I and Q channel short-code generators and the long-code generator.

In the CDMA 1x system, the state of the shift registers is a function of the system time [4]. A channel mask is used on the long shift register and adds contributions from the various delays within the shift register to form the output bit. The mask is used to shift the phase of the spreading sequence

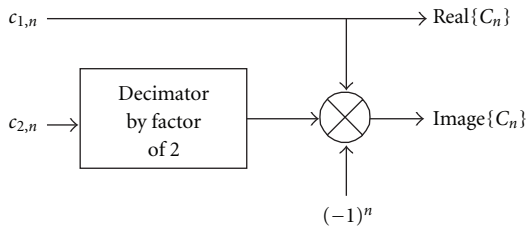


Figure 1: Block diagram of a method to form a complex spreading sequence from two bipolar sequences [4].

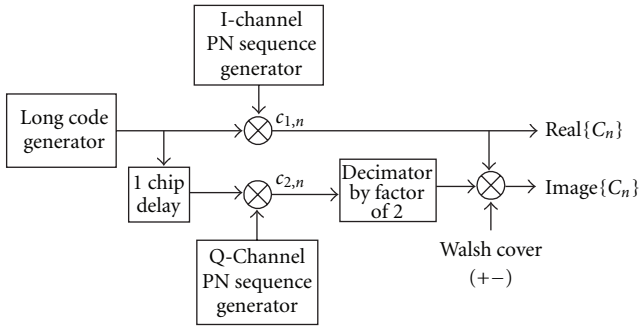


Figure 2: Block diagram of CDMA 1x spreading sequence generation.

of the common state of the shift register in the system to a unique phase for each user. The combining of delayed versions of the sequence within the register forms another phase of the same sequence. As a result of this, a state for the original long shift register (without the mask) will produce the same output sequence as the original state of the shift register with the mask. Thus, it suffices to solve for the state of the shift register without the mask since it will produce the desired spreading sequence. To form the generator matrix for the system, the constant multiplier is removed from (5) and converted to work with binary equations as previously described. A logic diagram showing the equivalent binary process is shown in Figure 3. Given

$$\begin{aligned} y_n &= c_{1,2n} \oplus c_{1,2n+1} \oplus c_{2,2n} \\ &= \mathbf{b}_0\mathbf{x}_{0,2n+1} \oplus \mathbf{b_1}\mathbf{x}_{1,2n} \oplus \mathbf{b}_0\mathbf{x}_{0,2n+2} \oplus \mathbf{b}_1\mathbf{x}_{1,2n+1} \\ &\quad \oplus \mathbf{b}_0\mathbf{x}_{0,2n} \oplus \mathbf{b}_2\mathbf{x}_{2,2n} \\ &= \mathbf{b}_0\left(\mathbf{x}_{0,2n} \oplus \mathbf{x}_{0,2n+1} \oplus \mathbf{x}_{0,2n+2}\right) \\ &\quad \oplus \mathbf{b}_1\left(\mathbf{x}_{1,2n} \oplus \mathbf{x}_{1,2n+1}\right) \oplus \mathbf{b}_2\mathbf{x}_{2,2n} \\ &= \mathbf{b}_0\left(\mathbf{A}_0^{2n} \oplus \mathbf{A}_0^{2n+1} \oplus \mathbf{A}_0^{2n+2}\right)\mathbf{x}_{0,0} \\ &\quad \oplus \mathbf{b}_1\left(\mathbf{A}_1^{2n} \oplus \mathbf{A}_1^{2n+1}\right)\mathbf{x}_{1,0} \oplus \mathbf{b}_2\mathbf{A}_2^{2n}x_{2,0}, \end{aligned} \tag{10}$$

where $\mathbf{x}_{k,n}$ is the state vector for the kth LFSR at time n, $\mathbf{b}_k$ is the observation vector so that $\mathbf{b}_k\mathbf{x}_{k,n}$ produces the output at time n for the kth LFSR, $\mathbf{A}_k$ is the transition matrix for the kth LFSR (i.e., $\mathbf{x}_{n+1} = \mathbf{A}_k\mathbf{x}_{k,n}$), and $\oplus$ is addition in GF(2). A system state vector can be formed by concatenating the individual LFSR state vectors, that is,

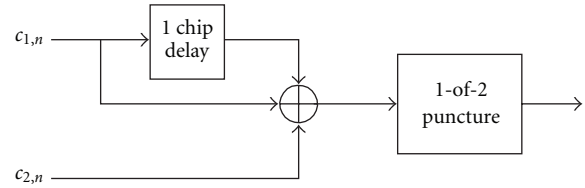


Figure 3: Logic diagram for processing the result from the sequential processing of the complex sequence.

$\mathbf{x_n} = [x_{0,n} \quad x_{1,n} \quad x_{2,n}]^T$. The matrix equation that provides the $\mathbf{y}$ is in the form

$$\mathbf{y} = \mathbf{G}\mathbf{x_0}, \tag{11}$$

where $\mathbf{x_0}$ is the initial state of the spreading generator at a time "0", $\mathbf{y} = [y_0, y_1, \ldots, y_m]^T$, and the nth row of $\mathbf{G}$, denoted as g_n, is formed by arranging the terms in (10) to correspond to the system state vector, that is,

$$g_n = \left[\mathbf{b}_0\left(\mathbf{A}_0^{2n} \oplus \mathbf{A}_0^{2n+1} \oplus \mathbf{A}_0^{2n+2}\right) \quad \mathbf{b}_1\left(\mathbf{A}_1^{2n} \oplus \mathbf{A}_1^{2n+1}\right) \quad \mathbf{b}_2\mathbf{A}_2^{2n}\right]. \tag{12}$$

All of the matrices and vectors in (12) are dependent on the LFSRs used in the system. For the CDMA 1x system, the state of the spreading generator is the sum of the dimensions of the individual shift registers, that is, 72 (i.e., 42 + 15 + 15). Thus, the dimension of $\mathbf{G}$ is a $m \times 72$ matrix, $\mathbf{x_0}$ and $\mathbf{y}$ are 72×1 and $m \times 1$, respectively. The vector $\mathbf{y}$ can be seen as a codeword for a $(m, 72)$ code defined by the generator matrix $\mathbf{G}$ with $\mathbf{x_0}$ being treated as the information bits associated with the codeword. The corresponding parity check matrix, $\mathbf{H}$, has dimension $(m - 72) \times m$ and the solution matrix $\mathbf{P}^{\#}$, which is used to solve for the state of the shift registers given 72 contiguous values from the vector $\mathbf{y}$, is a 72×72 matrix.

In the CDMA 1x system, the I and Q short shift registers have a known state at the beginning of the frame [4]. The shift registers are clocked simultaneously for 32767 cycles and then a fill zero is added to form a frame of 32768 chips [4]. The development of the equations did not take into consideration the extra fill bit and thus the parity check matrix will not be valid for the $\mathbf{y}$ vector if the fill bit falls within the samples used to create $\mathbf{y}$. The decoding algorithm will not be able to converge to a valid codeword when the parity check matrix is not valid. However as practical block sizes used for decoding are much smaller than the frame size, the number of starting positions within the frame where the equations are not valid is small. For example, if a block size of 1024 is used for the decoder, the receiver needs 2048 samples to form the $\mathbf{y}$ vector. Thus, there are only 2048 starting positions where the fill bit would be contained in the vector, which leaves 30720 positions within the frame which will have valid equations.

In order to check if the solved state of the shift register is valid, one method is to load a local spreading generator and run the generator to produce the spreading sequence and then correlate this sequence with the received sequence. However, there is a less complex method that can be utilized

that is based on the initialization of the short spreading registers at the beginning of each frame. As the states of the short shift registers are known at the beginning of the frame and the registers are clocked together, the states of both shift registers are known for each offset (in chips) from the start of the frame. Another property that results from this procedure is there is a one-to-one correspondence of the states in the I and Q short shift registers. In other words, there are valid pairs of states. This property can be used to determine if the solved state of the spreading generators is likely to be correct. To use this method, the decoder forms a candidate codeword (described in the next section), and then solves for states of the short shift registers, x_1 and x_2. As an example, consider the case where the solved state x_1 corresponds to an offset of k chips from the start of the frame for the I short sequence generator. The state for the Q short sequence generator for the kth offset from the start of the frame is compared with the solved state x_2. If the states are not equal, then at least one of the solved states is incorrect which implies that the current candidate codeword is not valid. If the states are equal, then it is highly likely that the solved state is correct.

2.2. Decoding Algorithm. The decoding algorithm used to find an estimate of the codeword is the Vector SISO [17]. The algorithm is a soft-in soft-out decoding algorithm. Versions of the algorithm for binary codes have been described in [17–19]. For this application, the algorithm was modified to return hard decisions only. For use with CDMA 1x detection, the decision rule to determine if the estimated codeword is likely to be correct was modified to use the correlations in the I and Q short shift registers as described above.

For the sake of completeness, a brief outline of the decoding algorithm is provided here. For more detail regarding the decoding algorithm, the reader is referred to [17] or [18].

Consider general linear codes for which codewords can be generated from $\mathbf{y} = \mathbf{G}\mathbf{x}$, where $\mathbf{x}$ is a column k-vector of information symbols, $\mathbf{y}$ is a column n-vector of coded bits, and $\mathbf{G}$ is the $n \times k$ generator matrix. A parity check matrix, $\mathbf{H}$, is any $(n-k)\times n$ matrix of rank $(n-k)$ for which $\mathbf{HG} = \mathbf{0}$.

A set of symbol positions (i.e., indices in the codeword vector) are said to be linearly independent if the corresponding columns of a parity check matrix are linearly independent.

A parity check matrix is referred to as being in pseudosystematic form relative to a set of $n-k$ symbol positions if by moving the corresponding $n-k$ columns, and appropriately ordering them, the matrix can be put in the form $\mathbf{H} = [\mathbf{I}\ \ \mathbf{H}']$ where $\mathbf{I}$ denotes the identity matrix. If bit values are given for the $n-k$ positions corresponding to the nonidentity portion of the pseudosystematic parity check matrix, it is easy to form the entire codeword by selecting the remaining k bits to satisfy the parity constraints (e.g., by forming the corresponding pseudo-systematic generator matrix). Forming the codewords in this manner is referred to here as "recoding."

We can obtain $\mathbf{d}$ from $\mathbf{y}$ by mapping 0-valued elements in $\mathbf{y}$ to 1-valued elements in $\mathbf{d}$, and 1-valued elements in $\mathbf{y}$ to -1-valued elements in $\mathbf{d}$. There are 2^k possible vector values that can be assumed by $\mathbf{d}$, and when enumeration is required the jth possibility is denoted d_j with mth element $d_{m,j}$.

The general problem can be stated as follows. Given that one of the 2^k possible $\mathbf{d}$ vectors has been transmitted (each assumed here to be equally likely), and $\mathbf{r}$ the corresponding vector of noisy samples received over an antipodal additive white Gaussian noise channel, approximate the *a posteriori* log likelihood ratio for each bit in the coded vector. For this application we are only concerned with the corresponding hard decisions.

The required approximate values can be computed using the max-log-APP algorithm (sometimes referred to as the max-log-MAP algorithm) [20]. To briefly explain the max-log-APP algorithm, assume that it is possible to efficiently find the maximum likelihood (ML) codeword, j, under the constraint that the transmitted bit at time k is a 1, as well as the ML codeword, j' under the constraint that the transmitted bit at time k is a -1.

Consider the difference between the summed metrics

$$\frac{1}{2}\sum_{m=1}^{n} d_{m,j} r_m - \frac{1}{2}\sum_{m=1}^{n} d_{m,j'} r_m = r_k + \sum_{\substack{m \neq k \\ d_{m,j} \neq d_{m,j'}}} d_{m,j} r_m \tag{13}$$

$$= llr_k^i + llr_k^{\hat{e}}. \tag{14}$$

Note that the composite information given by the right-hand side of the equation only involves the bit positions for which the two codewords differ. The first term of the composite information is the intrinsic information (i.e., the information about the kth bit's value from the kth channel sample), while the second term provides an approximation to the extrinsic information (i.e., the information about the kth bit's value from the structure of the code and the other channel samples).

The steps followed by the basic SISO algorithm are summarized below.

(1) Form an initial "reliability" vector by taking the absolute values of the elements of r.

(2) This is the first step in the iteration loop. Select the $n-k$ linearly independent bit positions that are the least reliable in the sense that they correspond to the smallest elements in the reliability vector. We will refer to these as the "least reliable bits" (LRBs). Similarly, the remaining k bit positions are referred to as the "most reliable bits" (MRBs).

(3) Using row reduction techniques, put $\mathbf{H}$ into pseudo-systematic form such that the identity portion of the parity check matrix corresponds to the LRBs. Let $\mathbf{d}'$ denote the vector of best decisions found to the given point in the algorithm, where "best" means the codeword with the best metric. For the first iteration, $\mathbf{d}'$ will be the codeword obtained by performing hard decisions on $\mathbf{y}$ for the MRBs, and then using the pseudo-systematic parity check matrix selecting the LRBs to satisfy the parity constraints.

(4) Compute the extrinsic information for the MRBs. For the MRBs, a reasonable approximation to the

kth extrinsic information is to take the metric difference between the best decision codeword, and the recoding solution with all of the MRBs remaining the same except for the kth one. Thus, the extrinsic information from (14) becomes

$$llr_k^{\hat{e}} = d_{m,j} * \sum_{\substack{m \neq k \\ d_{m,j} \neq d_{m,j'}}} d_{m,j} r_m, \tag{15}$$

where each of the m locations corresponds to an LRB and multiplication by $d_{k,j}$ accounts for the possibility that the kth bit may not be $a + 1$ in the vector of best decisions.

(5) Compute the MRB metric differences by summing the extrinsic information and the intrinsic information, and compare with the signs of the MRBs in the current best codeword. If any of the signs differ, then a new best path has been found. If the signs differ in several locations, the new best path is the one for which the sign is changed for the bit corresponding to the largest magnitude of composite information (with sign differing from the current best decision). If there are no sign differences or if the maximum allowable number of iterations has been reached, go to Step (7). Otherwise, form the new best path by first changing the sign of the appropriate MRB and then computing the LRBs using recoding.

(6) Form the new reliability vector by performing element-by-element multiplication between the new best decision vector and $\mathbf{y}$. Note that the reliability vector is now a signed vector with the "smallest" ones being negative (i.e., the intrinsic information and the "best" decision differ in sign). Go to Step (2).

(7) Check if the best codeword is likely to be correct by comparing the states of the short shift registers. If they form a valid pair of states or if the maximum allowable number of iterations has been reached, the algorithm terminates. Otherwise, bias the algorithm away from the current solution by multiplying the current solution by a small scale factor, subtracting it from the received vector, and return to Step (2).

3. Simulation Results

This section presents the results from Monte-Carlo simulations for both the additive white Gaussian noise (AWGN) channel and a quasistatic Rayleigh fading channel. The performance plots present the codeword error rate (CER), where the codeword is from a $(m, 72)$ code. In the simulations, we chose the size of m to be equal to 512, 1024, and 2048. (Note that there are two chip samples from the channel to produce one codeword sample.)

The simulation has perfect chip timing to the desired user and the simulation operates at 1 sample/chip. The CDMA 1x signal is spread at 1.2288 Mchips/s. When multiple users are present, the users are chip synchronous with the desired user. The desired user has a frequency offset of 3000 Hz from 0 Hz in the simulations. The algorithm is robust against a frequency offset and the offset was included in the simulations to show the performance with an imperfect carrier frequency estimate. The data signal before spreading includes the pilot channel on the in-phase channel and with random data multiplied by a Walsh code (1 1 1 1 -1 -1 -1 -1) with a relative gain to the pilot of 0.8.

The detector used the biased version of the Vector SISO [18]. The maximum number of bias modifications was set to 20 and a scale factor of 0.5 was used. The codeword was modified and decoding continued, if after solving for the initial state of the shift registers a valid pair for the short shift registers did not occur. A maximum of 50 iterations were allowed for each decoding. In the simulations, a minimum of 10000 codewords were simulated. Once the minimum number of codewords were simulated, the simulation stopped when a minimum of 200 codeword errors were observed.

Equation (9) is used on a candidate codeword (truncated sequence) to solve for the 72 bits necessary to define the state of the shift registers. An error occurs when the decision bits from the decoder do not match the initial state of the shift register.

In Figure 4, the CER performance is presented for the 1 and 2 user cases on the AWGN channel. For the 2 user case, the interfering user has random frequency and phase offsets relative to the desired user and the power is set to -0.9 dB (i.e., less than 1 dB difference) relative to the desired user. The codeword is marked in error if it does not agree with the state of the desired user's signal. As seen in Figure 4, by increasing the block length of the code from 512 to 2048, improvements in performance of 1.1 dB and 2.3 dB for the 1 and 2 user cases, respectively, are obtained at a CER of 10^{-2}. There is a degradation in performance for the case when there is an interfering user. For the two user case, the performance of the decoder is degraded from the 1 user case by 3 dB, 2.2 dB, and 1.9 dB for block sizes of 512, 1024, and 2048, respectively. The performance of a hard decision decoder that makes hard decisions on the sequence of bits and solves for the state of the spreading generator is provided for comparison. The performance is for the single user on the AWGN channel. The decoder has no coding gain when using 72 bits. The (512,72) code using the soft-decision decoder has a gain in performance of 8.2 dB at a PER of 10^{-2}. The gain is a result of both the coding gain and using soft-decision information for the bits.

As CDMA systems usually work on an E_c/N_0 below 0 dB, it should be noted that the results presented in Figure 4 are still of interest for this application. The application is for detecting users and monitoring spectrum usage. As our application is not involved in communications with the mobiles, there is no strict acquisition time requirement. For this application very poor CERs are tolerable because multiple attempts to recover the code phase are possible. If i attempts are made, the probability of successful acquisition is $1 - \text{CER}^i$. Consequently, values of CER much worse than 0.1 can still be useful in this application.

The performance with Rayleigh fading was considered as the Rayleigh fading channel is common in cellular communications. The channel model that was used was

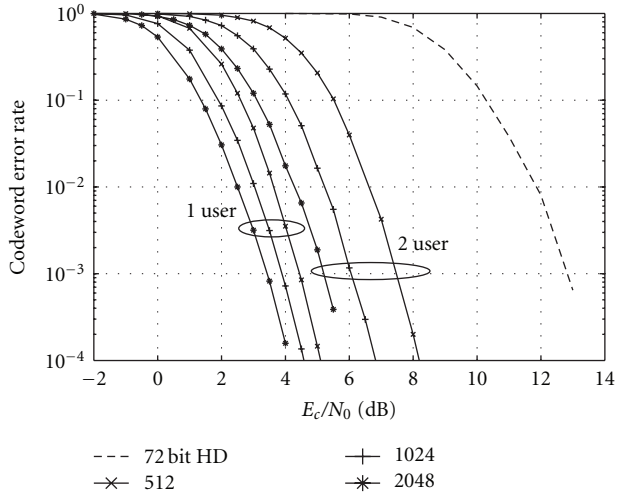


Figure 4: Codeword error rate performance results for codeword lengths of 512, 1024, and 2048 for 1 and 2 users on the AWGN channel. Results for the single user case when 72 hard decisions on the processed sequential pairs of channel samples are made and then used to solve for the spreading generator state are shown for comparison.

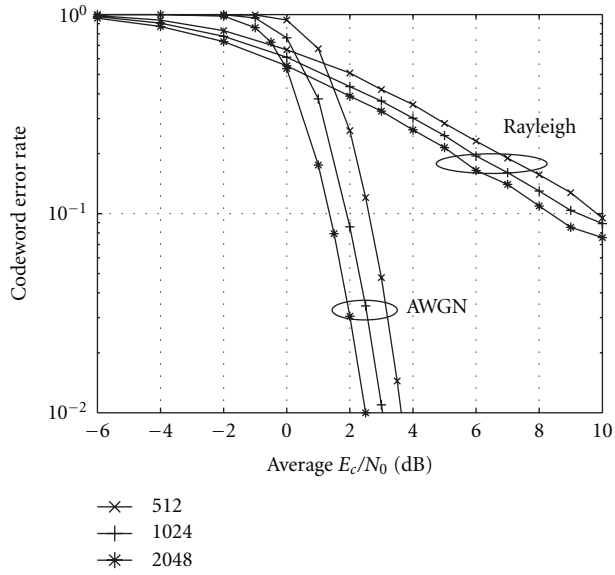


Figure 5: Codeword error rate performance results for codeword lengths of 512, 1024, and 2048 for a single user on a quasi-static Rayleigh fading channel. Single user performance on the AWGN channel is included for comparison.

quasistatic Rayleigh fading, where the user's signal was scaled by a Rayleigh variate for each codeword. The Rayleigh variate was generated by scaling the square root of the sum of the squares of two independent Gaussian random variates with zero-mean and a variance of σ. A scale factor was chosen such that the mean of the Rayleigh variates was 1.0 and the resulting variance was $4/\pi - 1$.

For the Rayleigh fading channel simulations with two users, the detected codeword was considered in error only

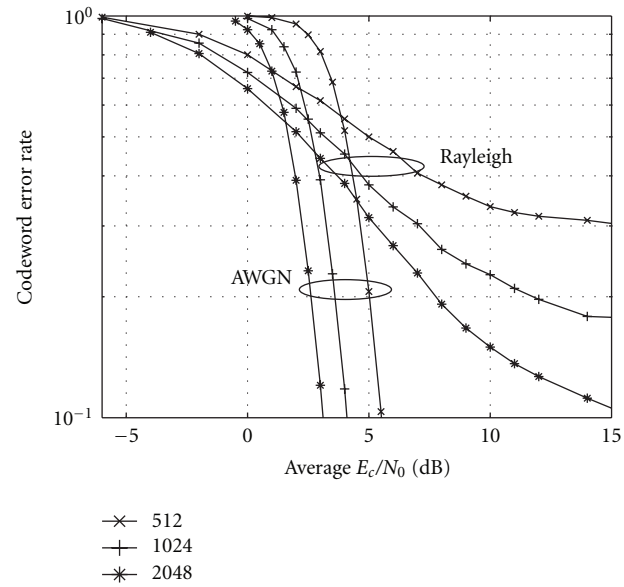


Figure 6: Codeword error rate performance results for codeword lengths of 512, 1024, and 2048 for two users on a quasi-static Rayleigh fading channel. The case with two users on the AWGN channel is included for comparison. In the AWGN case, the interfering user's power is set to 0.9 dB less than the desired user.

if it did not match either of the users. In other words, the detector was successful if it detected either one of the users' codewords. This measurement was chosen to correspond to typical results for a spectrum monitoring application as we are interested in detecting any user's signal. We have assumed the signal with the highest power would be the most likely signal detected. In the AWGN case, the interfering user's signal power was set slightly below that of the desired user's signal. In the Rayleigh fading case, the users' signals were set to have an equal power prior to fading, however, after fading either signal could have the highest receive power and considered the "desired" signal.

In Figure 5, the codeword error performance results of the algorithm are shown for the cases when the decoder uses block sizes of 512, 1024, and 2048. As seen in the figures, the detector has a better performance in quasi-static Rayleigh fading at lower E_c/N_0 values. For example, if we consider a detection rate of 1 in 5 trials (e.g., a CER of 0.8) then there are gains of approximately 2 dB for the block sizes tested.

The case for two equal-powered signals is presented in Figure 6. In the low E_c/N_0 region, this case has a higher gain over the two-user AWGN case when comparing the single user cases. The independent fading of the two users is the dominating factor in this result. If due to the fading the interfering signal power is reduced, the algorithm has a better chance to detect the nonfaded signal.

The method presented here is a practical method for blind acquisition of CDMA 1x signals. Speed tests were run on a Pentium IV processor with a clock speed of 3.8 GHz. The results are for 1 sample/chip operation with perfect chip timing and a signal present. The decoding parameters used during the speed tests were that the decoder was allowed 1

iteration, the maximum number of bias modifications was 10 and the extrinsic scale factor was 0.2. The average decoder speeds for block sizes of 512, 1024, and 2048 were 27.2, 9.8, and 3.6 decoding/second, respectively, for an $E_c/N_0 = 0$ dB. The speeds increase with an increase in E_c/N_0 as fewer average iterations are required.

4. Conclusion

A technique was presented for recovering the code phase of the composite spreading sequence for a CDMA 1x signal transmitted from a handset, without the benefit of any *a priori* information from the system. Key steps in the technique are pair-wise processing to eliminate sensitivity to carrier frequency and phase offsets, followed by soft-in iterative decoding to solve for the state of an equivalent spreading code generator. A novel approach, based upon computing and checking the states of the two short-code generators, for determining if the resulting state is likely to be correct was presented.

The utility of the technique was demonstrated using computer simulation, in both AWGN and Rayleigh fading environments. It was shown that code phase recovery is possible even at very low signal-to-noise ratios and in the presence of significant co-channel interference.

The simulations assumed perfect chip timing synchronization. If chip timing is unknown, as is usually the case, the system can try each of four evenly spaced hypotheses (with a spacing of a quarter of a chip period). This method was found to be robust in low-SNR environments. While the results are not described in this paper, this technique has been used to successfully recover the code phase for off-air signals.

A suggested area of future work is an investigation on the effect of providing reliability information to the decoder that takes into account the non-Gaussian noise terms associated with y_n in (5). Another area of investigation is an investigation of various methods to reduce the effect of multiple-access interference on the detection algorithm.

References

[1] M. Simon, J. Omura, R. Scholtz, and B. Levitt, *Spread Spectrum Communications*, vol. 3, Computer Science Press, Rockville, Md, USA, 1985.

[2] G. Stuber, J. W. Mark, and I. F. Blake, "Sequence acquisition using bit estimation techniques," *Information Sciences*, vol. 32, no. 3, pp. 217–229, 1984.

[3] J. L. Massey, G. Boscagli, and E. Vassallo, "Regenerative pseudo-noise (PN) ranging sequences for deep-space missions," *International Journal of Satellite Communications and Networking*, vol. 25, no. 3, pp. 284–304, 2007.

[4] 3GPP2 C.S0002-C, Version 2, "Physical layer standard for cdma2000 spread spectrum systems rev C," July 2004.

[5] R. B. Ward, "Acquisition of pseudo-noise signals by sequential estimation," *IEEE Transactions on Communications*, vol. COM-13, pp. 475–483, 1965.

[6] H. Pearce and M. Ristenblatt, "The threshold decoding estimator for synchronization with binary linear recursive sequences," in *Proceedings of the IEEE International Conference on Communications (ICC '71)*, pp. 43–30, Montreal, Canada, June 1971.

[7] C. Kilgus, "Pseudonoise code acquisition using majority logic decoding," *IEEE Transactions on Communications*, vol. 21, no. 6, pp. 772–774, 1973.

[8] R. B. Ward and K. P. Yiu, "Acquisition of pseudonoise signals by recursion-aided sequential estimation," *IEEE Transactions on Communications*, vol. 25, no. 8, pp. 784–794, 1977.

[9] P. Guinand and J. Lodge, "Iterative decoding of truncated simplex codes," in *Proceedings of the 2nd Biennial Symposium on Communications*, Kingston, Canada, June 2002.

[10] L.-L. Yang and L. Hanzo, "Iterative soft sequential estimation assisted acquisition of m-sequences," *Electronics Letters*, vol. 38, no. 24, pp. 1550–1551, 2002.

[11] L.-L. Yang and L. Hanzo, "Differential acquisition of m-sequences using recursive soft sequential estimation," *IEEE Transactions on Wireless Communications*, vol. 4, no. 1, pp. 128–136, 2005.

[12] M. Zhu and K. M. Chugg, "Iterative message passing techniques for rapid code acquisition," in *Proceedings of the IEEE Military Communications Conference (MILCOM '03)*, pp. 434–439, Boston, Mass, USA, October 2003.

[13] K. M. Chugg and M. Zhu, "A new approach to rapid PN code acquisition using iterative message passing techniques," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 5, pp. 884–897, 2005.

[14] F. Principe, K. M. Chugg, and M. Luise, "Rapid acquisition of gold codes and related sequences using iterative message passing on redundant graphical models," in *Proceedings of the Military Communications Conference (MILCOM '06)*, October 2006.

[15] 3G TS 25.213 V4.3.0 (2002–2006), "3rd generation partnership project; technical specification group radio access network; spreading and modulation (FDD) release 4," June 2002.

[16] 3GPP2 C.S.0024-A, "High rate packet data air interface specification version 3.0," September 2006.

[17] J. Lodge and R. Kerr, "Vector soft-in-soft-out decoding of linear block codes," in *Proceedings of the 22nd Biennial Symposium on Communications*, pp. 373–375, Kingston, Canada, May 2004.

[18] R. Kerr and J. Lodge, "Near ML performance for linear block codes using an iterative vector SISO decoder," in *Proceedings of the 4th International Symposium on Turbo Codes and Related Topics*, Munich, Germany, April 2006.

[19] A.-R. Abdul-Shakoor, R. Kerr, J. Lodge, and V. Szwarc, "An FPGA implementation of a soft-in soft-out decoder for block codes," in *Proceedings of the 24th Biennial Symposium on Communications (BSC '08)*, pp. 226–230, Kingston, Canada, June 2008.

[20] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," in *Proceedings of the IEEE International Conference on Communications (ICC '95)*, pp. 1009–1013, June 1995.

Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2010, Article ID 492079, 9 pages
doi:10.1155/2010/492079

Research Article

MIMO Self-Encoded Spread Spectrum with Iterative Detection over Rayleigh Fading Channels

Shichuan Ma, Lim Nguyen, Won Mee Jang, and Yaoqing (Lamar) Yang

Department of Computer and Electronics Engineering, University of Nebraska-Lincoln, Omaha, NE 68182, USA

Correspondence should be addressed to Shichuan Ma, sma@huskers.unl.edu

Received 15 February 2010; Revised 15 July 2010; Accepted 16 August 2010

Academic Editor: Christian Schlegel

Self-encoded spread spectrum (SESS) is a novel communication technique that derives its spreading code from the randomness of the source stream rather than using conventional pseudorandom noise (PN) code. In this paper, we propose to incorporate SESS in multiple-input multiple-output (MIMO) systems as a means to combat against fading effects in wireless channels. Orthogonal space-time block-coded MIMO technique is employed to achieve spatial diversity, and the inherent temporal diversity in SESS modulation is exploited with iterative detection. Simulation results demonstrate that MIMO-SESS can effectively mitigate the channel fading effect such that the system can achieve a bit error rate of 10^{-4} with very low signal-to-noise ratio, from 3.3 dB for a 2×2 antenna configuration to just less than 0 dB for a 4×2 configuration under Rayleigh fading. The performance improvement for the 2×2 case is as much as 6.7 dB when compared to an MIMO PN-coded spread spectrum system.

1. Introduction

Self-encoded spread spectrum (SESS) and multiple access communications have been proposed and shown to have a number of unique features [1, 2]. By deriving its spreading sequences from the randomness of the source stream, SESS provides a feasible implementation of random-coded spread spectrum and potentially enhances the transmission security. Since the use of PN code generators is obviated, SESS can simplify multirate transmissions with variable processing gains in multimedia applications [3]. In previous works, we have shown that the modulation memory in SESS can yield signal gain in AWGN channels [4]. We have also demonstrated that the inherent temporal diversity in SESS modulation can be exploited with iterative detection to significantly improve the system performance over time-varying fading channels [5].

It is also well known that PN-coded spread spectrum can be incorporated with multiple-input multiple-output (MIMO) techniques to achieve multipath and spatial diversities [6–8]. In delay-spread or frequency-selective fading, the system performance can be improved with a Rake receiver that can resolve and combine the multi-path signal components by matching the Rake fingers with the delayed PN codes to achieve multi-path diversity. By deploying multiple antennas at both transmitter and receiver, MIMO architectures are capable of mitigating channel fading by taking advantage of the spatial diversity and enhancing system capacity by employing spatial multiplexing [9–11].

In this paper, we propose a novel approach to combat against fading in wireless channels by incorporating SESS in MIMO system. Our motivation is based on the observation that SESS not only can achieve multi-path diversity like PN-coded spread spectrum, but it can also provide both inherent temporal diversity and signal gain. Our work combines orthogonal space-time block-coded MIMO technique and iterative detection to obtain spatial and temporal diversities, respectively. We determine the bit error rate (BER) performance of MIMO-SESS system over Rayleigh fading channels and show that the fading effects can be completely mitigated by exploiting diversities in both space and time domains. The proposed scheme in this paper employs only one spreading sequence which we refer to as the *single-code* scheme. Recently, we have reported a *multicode* Alamouti scheme that can enhance the system throughput [12] with multiple spreading sequences, but at the expense of a BER

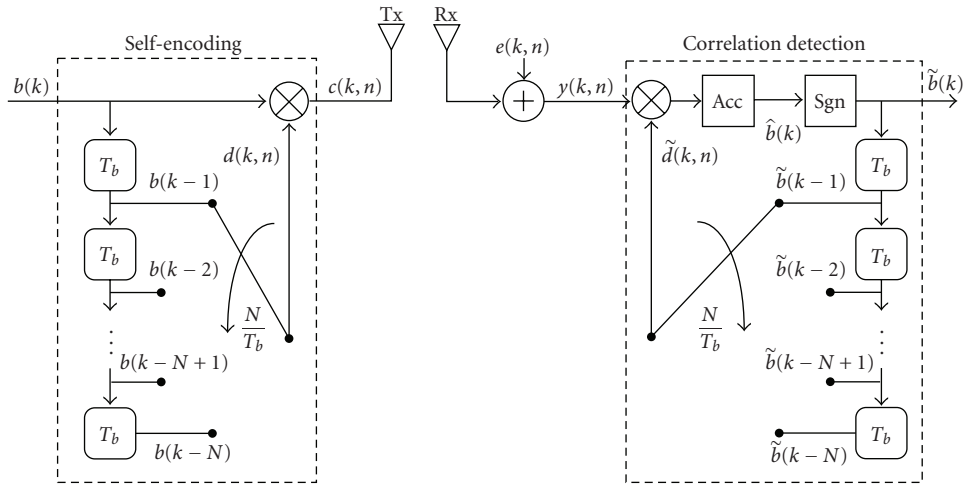


Figure 1: Block diagram of SESS system.

degradation. The orthogonal space-time block codes and multiple iterations in this paper thus could be incorporated into the multi-code scheme in order to improve the BER performance.

The rest of this paper is organized as follows. In Section 2, we briefly describe the original SESS system. The proposed MIMO-SESS with iterative detection is described in Section 3. Section 4 presents the simulation results and compares the proposed system to an MIMO PN-coded spread spectrum (MIMO-PNSS) system. Finally, Section 5 concludes this paper.

Notations. The superscripts $*$ and T represent the conjugate and the transpose operations, respectively. $|\cdot|$, $E\{\cdot\}$, and $\mathrm{diag}(\cdot)$ denote the absolute value of a scalar, the expectation operation, and the diagonal vector of a matrix, respectively. Matrix (vector) is represented by capital (small) bold letter, with $a(k, n)$ representing the nth element of vector $\mathbf{a}(k)$.

2. Background

In this section, we briefly review the concept of self-encoding spread spectrum. The readers are referred to the original description in [1] for more details.

2.1. Transmitter. A block diagram of SESS system is shown in Figure 1, where each rounded corner block represents one delay register, N denotes the length of spreading sequence, and T_b is the bit duration.

The source information b is assumed to be bipolar values of ± 1. The bits are first spread by the self-encoded spreading sequence $\mathbf{d}(k)$ of length N at a chip rate of N/T_b. This sequence is constructed from the source information stored in the delay registers that are updated every T_b. For example, the spreading sequence for the kth bit, $b(k)$, is given as

$$\mathbf{d}(k) = [b(k-1)b(k-2)\cdots b(k-N)]^T. \tag{1}$$

Thus, with a random input data stream, the sequence is also random and time varying from one bit to another. We assume that the delay registers have been seeded by a randomly selected sequence (which has also been acquired at the receiver to initialize the despreading process).

The spread spectrum chips are then transmitted as

$$\mathbf{c}(k) = b(k)\mathbf{d}(k). \tag{2}$$

Clearly, SESS signal has a modulation memory depth equal to N:

$$c(k, n) = b(k)d(k, n) = b(k)b(k-n). \tag{3}$$

2.2. Receiver. We assume that the chips are transmitted over a wireless channel and each bit experiences independent Rayleigh fading that is constant over the bit duration. The channel coefficient for the kth bit interval is represented by $\alpha(k)$. The received signal $y(k, n)$ can be expressed as

$$y(k, n) = \alpha(k)c(k, n) + e(k, n), \tag{4}$$

where the additive Gaussian noise $e(k, n)$ has zero mean and variance $NN_0/2$. Note that the noise is sampled at the chip rate and is broadband; its variance is thus the narrow-band noise variance $N_0/2$ multiplied by the spreading factor N.

The soft correlation estimate of the kth bit is computed from

$$\begin{aligned}\hat{b}(k) &= \frac{1}{N}\sum_{n=1}^{N} y(k, n)\tilde{d}(k, n) \\ &= \frac{1}{N}\sum_{n=1}^{N} \alpha(k)c(k, n)\tilde{d}(k, n) + u(k),\end{aligned} \tag{5}$$

where

$$u(k) = \frac{1}{N}\sum_{n=1}^{N} e(k, n)\tilde{d}(k, n) \tag{6}$$

is the narrow-band Gaussian noise component of the correlation output and $\widetilde{d}(k,n)$ is the nth chip of the despreading sequence $\widetilde{\mathbf{d}}(k)$ generated by the delay registers. We assume that the receiver delay registers have been initialized by the same seed sequence at the transmitter. The content of the delay registers, $\widetilde{\mathbf{d}}(k)$, are then updated by the hard decision of the correlation estimate, $\widetilde{b}(k) = \text{sgn}[\widehat{b}(k)]$. Since the hard decision may be erroneous, the reconstructed despreading sequence could be different from the self-encoded spreading sequence at the transmitter. A bit error therefore will propagate through the delay registers for the next N bits and cause self-interference that attenuates the strength of the despread signal $\widehat{b}(k)$. The correlation estimate given in (5) can be written as

$$\widehat{b}(k) = \frac{1}{N}\sum_{n=1}^{N}\alpha(k)c(k,n)\widetilde{b}(k-n) + u(k). \tag{7}$$

Substituting (3) into (7), we have

$$\widehat{b}(k) = \alpha(k)\left[\frac{1}{N}\sum_{n=1}^{N}b(k-n)\widetilde{b}(k-n)\right]b(k) + u(k). \tag{8}$$

Thus, the hard decision is performed on the soft estimate from the correlator output, a noisy and fading signal that is also subjected to self interference (expressed by the term inside the square bracket in (8)):

$$\widetilde{b}(k) = \text{sgn}\left[\widehat{b}(k)\right] = \begin{cases} 1, & \widehat{b}(k) > 0, \\ -1, & \widehat{b}(k) < 0. \end{cases} \tag{9}$$

3. MIMO-SESS with Iterative Detection

In this section, we seek to improve the performance of SESS system in fading channels by employing multiple antennas to take advantage of spatial diversity and by utilizing iterative detection to achieve temporal diversity. Figure 2 shows a block diagram of the proposed MIMO-SESS system.

At the transmitter, the information bits first undergo spread spectrum modulation according to the self encoding procedure in Section 2.1. The spread spectrum chips are further encoded using orthogonal space-time block codes (OSTBCs) and then transmitted over an $N_t \times N_r$ MIMO fading channel, where N_t antennas and N_r antennas are deployed at the transmitter and receiver, respectively. We assume that the channel between each transmit/receive antenna pair is subjected to Rayleigh fading that remains constant over T_b but is independent from bit to bit [5, 13]. Furthermore, the channels for different transmit/receive antenna pairs experience independent fading. The signals from the received antennas are combined linearly to generate the output that is then detected with an iterative algorithm for bit recovery.

3.1. MIMO Transmission. In the above MIMO-SESS system, the (inner) MIMO encoding is independent from (outer) self-encoding as can be seen from Figure 2. The decoupling nature of this approach simplifies decoding at the receiver and lets us employ, in principle, any MIMO techniques for signal transmission, such as beamforming for spatial filtering [14] and for signal quality improvement [15–17], space-time block coding for spatial diversity [18–20], and layered space-time architecture for spatial multiplexing [21–23]. In this paper, we adopt OSTBC which provides full diversity gain and allows linear maximum-likelihood (ML) decoding [24].

Numerous OSTBCs have been reported in the literature. In this work, we employ the full-rate code $\mathcal{G}_2$ for the two-transmit-antenna case, the half-rate code $\mathcal{G}_3$ for the three-transmit-antenna case, and the half-rate code $\mathcal{G}_4$ for the four-transmit-antenna case. The full-rate code $\mathcal{G}_2$ was proposed in [25] and is known as the Alamouti scheme. The half-rate codes $\mathcal{G}_3$ and $\mathcal{G}_4$ were developed in [24]. The code matrices have been reproduced as follows:

$$\mathcal{G}_2 = \begin{pmatrix} s_1 & s_2 \\ -s_2^* & s_1^* \end{pmatrix}, \tag{10}$$

$$\mathcal{G}_3 = \begin{pmatrix} s_1 & s_2 & s_3 \\ -s_2 & s_1 & -s_4 \\ -s_3 & s_4 & s_1 \\ -s_4 & -s_3 & s_2 \\ s_1^* & s_2^* & s_3^* \\ -s_2^* & s_1^* & -s_4^* \\ -s_3^* & s_4^* & s_1^* \\ -s_4^* & -s_3^* & s_2^* \end{pmatrix}, \tag{11}$$

$$\mathcal{G}_4 = \begin{pmatrix} s_1 & s_2 & s_3 & s_4 \\ -s_2 & s_1 & -s_4 & s_3 \\ -s_3 & s_4 & s_1 & -s_2 \\ -s_4 & -s_3 & s_2 & s_1 \\ s_1^* & s_2^* & s_3^* & s_4^* \\ -s_2^* & s_1^* & -s_4^* & s_3^* \\ -s_3^* & s_4^* & s_1^* & -s_2^* \\ -s_4^* & -s_3^* & s_2^* & s_1^* \end{pmatrix}. \tag{12}$$

In (10), (11), and (12), each row includes the symbols to be transmitted in one time slot via N_t transmit antennas, and each column consists of the symbols in one block for each antenna. The block sizes here are 2, 8, and 8 for the two-, three-, and four-antenna cases, respectively.

In order to enable linear maximum likelihood decoding of OSTBC, the channel should be stable in one block. Since we have assumed that the channel fading remains constant over a bit duration, this implies that the spreading factor N should be a multiple of the block size. Notice that the total transmit power of MIMO system should be the same as single-antenna system. This means that the equivalent MIMO power per transmit antenna must be normalized by N_t.

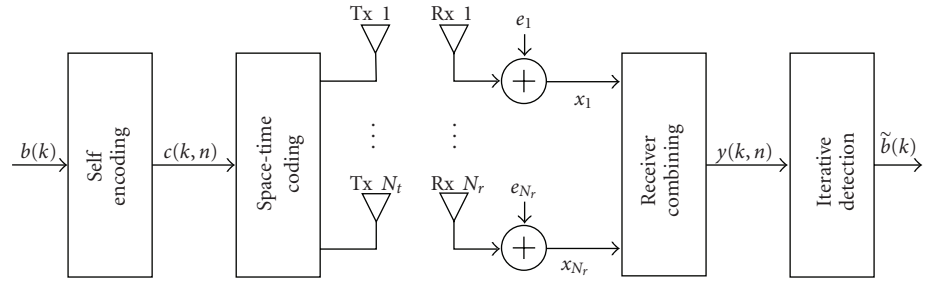


Figure 2: Block diagram of the proposed MIMO-SESS system.

3.2. Space-Time Processing Example. In the following, we will use the Alamouti scheme for two transmit antennas as an example to describe space-time block coding and decoding procedures.

At the transmitter, the spread spectrum chips are divided into two streams according to $\mathcal{G}_2$. A block of two consecutive chips, $c(k, 2i)$ and $c(k, 2i + 1)$, are transmitted by sending $c(k, 2i)$ and $-c^*(k, 2i+1)$ to the first antenna, and $c(k, 2i+1)$ and $c^*(k, 2i)$ to the second antenna. Here $i \in \{0, 1, \ldots, N/2-1\}$ is the block index for the chips of the kth bit. We note that in this case the power per antenna is one-half of the total power. The signals are then transmitted over an MIMO fading channel.

At the receiver, the signals from the antennas are first combined to achieve spatial diversity. For the mth receive antenna, the signals within the ith block are given as

$$\begin{aligned} x_m(k, 2i) &= h_{1,m}(k)c(k, 2i) \\ &\quad + h_{2,m}(k)c(k, 2i + 1) + e_m(k, 2i), \end{aligned} \tag{13}$$

$$\begin{aligned} x_m(k, 2i + 1) &= -h_{1,m}(k)c^*(k, 2i + 1) \\ &\quad + h_{2,m}(k)c^*(k, 2i) + e_m(k, 2i + 1), \end{aligned} \tag{14}$$

where $h_{1,m}(k)$ and $h_{2,m}(k)$ denote the complex channel impulse response coefficients (for the kth bit) between the mth receive antenna and the first—and second—transmit antennas, respectively. e_m is the Gaussian noise with zero mean and variance $NN_0/2$.

Again we assume that the delay registers in the receiver have been initially synchronized with the transmitter [26], and that perfect channel knowledge is available. Under these assumptions, diversity combining is carried out over two consecutive chip intervals per [11, 25]

$$\begin{aligned} y(k, 2i) &= \sum_{m=1}^{N_r} \Big(h_{1,m}^*(k)x_m(k, 2i) \\ &\qquad + h_{2,m}(k)x_m^*(k, 2i + 1)) + w(k, 2i) \\ &= \alpha(k)c(k, 2i) + w(k, 2i), \end{aligned} \tag{15}$$

$$\begin{aligned} y(k, 2i + 1) &= \sum_{m=1}^{N_r} \Big(h_{2,m}^*(k)x_m(k, 2i) \\ &\qquad - h_{1,m}(k)x_m^*(k, 2i + 1)) + w(k, 2i + 1) \\ &= \alpha(k)c(k, 2i + 1) + w(k, 2i + 1), \end{aligned} \tag{16}$$

where

$$\alpha(k) = \sum_{m=1}^{N_r} \Big(|h_{1,m}(k)|^2 + |h_{2,m}(k)|^2 \Big), \tag{17}$$

$$w(k, n) = \begin{cases} \displaystyle\sum_{m=1}^{N_r} \Big(h_{1,m}^*(k)e_m(k, n) + h_{2,m}(k)e_m^*(k, n + 1) \Big), \\ \qquad\qquad \text{if } n \text{ is even}, \\ \displaystyle\sum_{m=1}^{N_r} \Big(h_{2,m}^*(k)e_m(k, n - 1) - h_{1,m}(k)e_m^*(k, n) \Big), \\ \qquad\qquad \text{if } n \text{ is odd}. \end{cases} \tag{18}$$

The combined signals given in (15) and (16) can be expressed as

$$y(k, n) = \alpha(k)c(k, n) + w(k, n), \tag{19}$$

and further in vector form

$$\mathbf{y}(k) = \alpha(k)\mathbf{c}(k) + \mathbf{w}(k). \tag{20}$$

Similar signal models and analysis can be developed for the three- and four-transmit-antenna cases. In general for an $N_t \times N_r$ MIMO system, (19) and (20) remain valid with

$$\alpha(k) = \sum_{n=1}^{N_t} \sum_{m=1}^{N_r} |h_{n,m}(k)|^2, \tag{21}$$

albeit with a more complicated noise term $\mathbf{w}(k)$.

3.3. Iterative Detection. Since SESS signal has modulation memory, its optimum detection is the maximum likelihood sequence detector or the Viterbi detector. However, the complexity of the optimum detection grows exponentially as

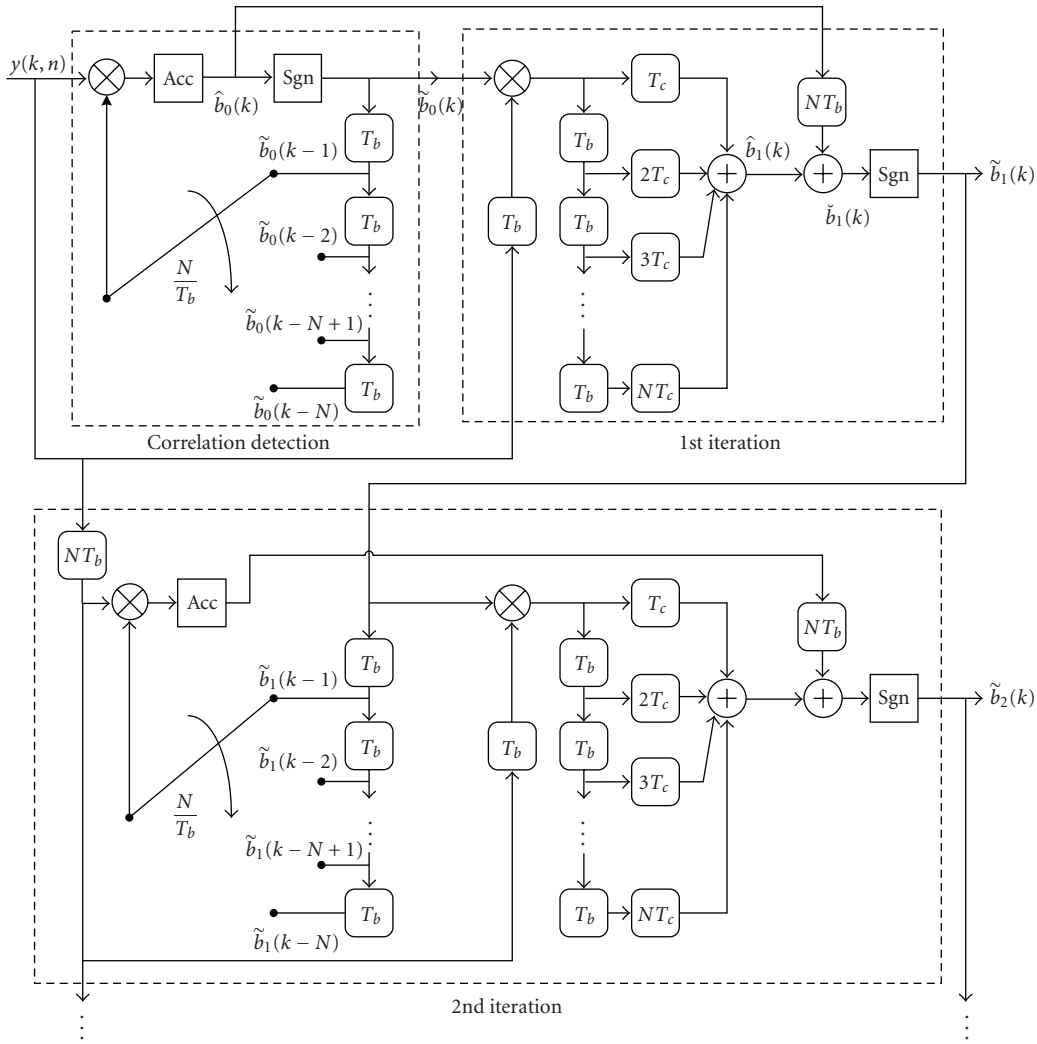


Figure 3: Block diagram of iterative detection.

2^N, making it impractical to implement unless N is rather small. On the other hand, iterative detection is suboptimum but linear in N, and can converge iteratively toward the optimum performance.

The iterative detection for SESS is described by the detailed block diagram illustrated in Figure 3. Notice that correlation detection provides the initial estimate $\tilde{b}_0(k)$ for the iterative detection. For simplicity we will ignore self interference in the following signal analysis, so that $\sum_{n=1}^{N} b(k-n)\tilde{b}(k-n) = N$. This in fact is justified if the BER is sufficiently low. According to the description in Section 2, the soft correlation output or estimate of the kth bit is then given as

$$\hat{b}_0(k) = \alpha(k)b(k) + v_1(k), \tag{22}$$

where $v_1(k)$ is the noise term.

The hard decision of the correlation output $\tilde{b}_0(k) = \text{sgn}[\hat{b}_0(k)]$ is then used to construct the despreading sequences.

Now assuming that we have obtained the correlation detection of $b(k)$, then based on (3) and (19) the next N^2 chips of bits $b(k+1), b(k+2), \ldots, b(k+N)$ can be written in a square matrix $\mathbf{P}(k)$ as shown by

$$\mathbf{P}(k) = \begin{bmatrix} \alpha(k+1)b(k+1)b(k) & \alpha(k+1)b(k+1)b(k-1) & \cdots & \alpha(k+1)b(k+1)b(k-N+1) \\ \alpha(k+2)b(k+2)b(k+1) & \alpha(k+2)b(k+2)b(k) & \cdots & \alpha(k+2)b(k+2)b(k-N) \\ \vdots & \vdots & \ddots & \vdots \\ \alpha(k+N)b(k+N)b(k+N-1) & \alpha(k+N)b(k+N)b(k+N-2) & \cdots & \alpha(k+N)b(k+N)b(k) \end{bmatrix}. \tag{23}$$

Each element of this matrix represents a chip as given in (19) and the jth row includes the chips of bit $b(k+j)$. Note that the noise term has been omitted due to the page limitation.

Observe that the kth bit, $b(k)$, are also present in the diagonal elements of the matrix $\mathbf{P}(k)$. This means that the information of the kth bit is also included in the next N bits. Thus, the modulation memory in SESS signals could be utilized to enhance the detection of the kth bit. The iterative detection employs the hard decisions of these next N bits, $[\tilde{b}_0(k+1)\ \ \tilde{b}_0(k+2)\ \ \cdots\ \ \tilde{b}_0(k+N)]$, in order to accumulate the time-diversity soft estimate of $b(k)$ as

$$\begin{aligned}\hat{b}_1(k) &= \frac{1}{N}\,\mathrm{diag}(\mathbf{P}(k))\left[\tilde{b}_0(k+1)\tilde{b}_0(k+2)\cdots\tilde{b}_0(k+N)\right]^T \\ &\quad + v_2(k) \\ &= \frac{1}{N}\sum_{n=1}^{N}\alpha(k+n)b(k) + v_2(k).\end{aligned} \tag{24}$$

where $v_2(k)$ is a noise term.

The summation of the correlation estimate, $\hat{b}_0(k)$, delayed by NT_b, and the time-diversity estimate, $\hat{b}_1(k)$, provides the soft estimate for the hard decision of the first iteration. That is,

$$\begin{aligned}\breve{b}_1(k) &= \hat{b}_0(k) + \hat{b}_1(k) \\ &= \left(\alpha(k) + \frac{1}{N}\sum_{n=1}^{N}\alpha(k+n)\right)b(k) + v(k),\end{aligned} \tag{25}$$

where $v(k) = v_1(k) + v_2(k)$.

The output of the first iteration $\tilde{b}_1(k)$ is then obtained by the hard decision of $\breve{b}_1(k)$.

This first iterative bit decision $\tilde{b}_1(k)$ is an improvement over the initial estimate $\tilde{b}_0(k)$ because it is based on the combined soft estimates which yield not only a signal (to noise) gain of 3 dB but also the time diversity gain from SESS modulation memory. The improved bit decision can be further fed back again to reconstruct the de-spreading sequences for the correlation detection and also to reaccumulate the soft bit estimate from temporal diversity combining in the manner described by (24).

The block diagram for this second iteration is illustrated in the lower part in Figure 3. It is clear from the block diagram that procedure can be repeated iteratively. We note in addition that each iteration introduces a time delay of NT_b. Furthermore, the detector structure is partly similar to a Rake receiver that employs N fingers to exploit the N-fold temporal diversity in SESS signals.

4. Simulation Results

In this section, the performance of the proposed MIMO-SESS system has been determined by simulations and is compared with a conventional MIMO-PNSS system. The length of the spreading sequence is set to $N = 64$ unless noted otherwise. For each scenario and bit signal-to-noise ratio E_b/N_o (SNR), 100 runs of 100,000 bits have been simulated to obtain the average bit error rate.

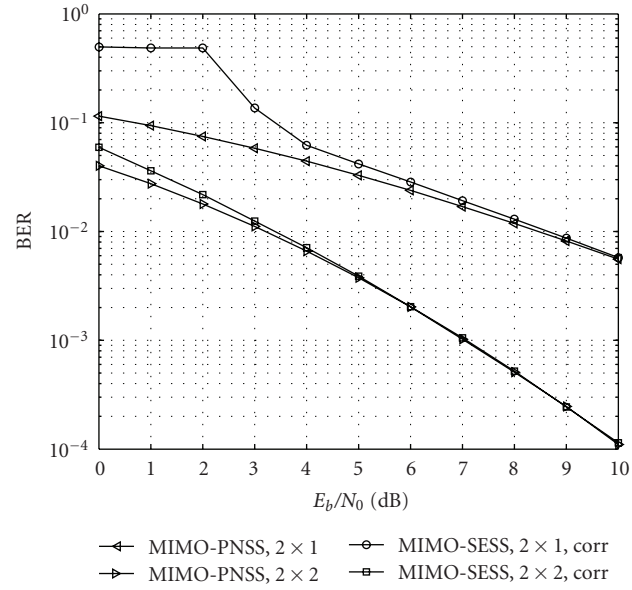


Figure 4: Comparison between an MIMO-PNSS system and the proposed MIMO-SESS system without iterative detection, two transmit antennas using full-rate code $\mathcal{G}_2$, $N = 64$.

Figure 4 compares two-transmit-antenna MIMO-SESS system without iterative detection to MIMO-PNSS system. Because iterative detection is not applied in this scenario, no temporal diversity gain is achieved. This means that the same performance can be expected with or without self-encoding. However, the MIMO-SESS system degrades at low SNR due to the effects of error propagation [1]. By employing MIMO technique, error propagation can be efficiently alleviated, as shown by the 2×2 scenario.

The results for the two-transmit-antenna MIMO-SESS system with iterative detection are plotted in Figure 5, which clearly shows that the BER performance has been significantly improved with iterative detection. With only one iteration, there is about 6.7 dB gain for the 2×2 scenario at 10^{-4} BER. This performance improvement can be attributed to the temporal diversity introduced by self-encoding, in addition to the signal gain from combining the soft estimates. Moreover, the improvement with the second iteration over the first iteration is quite marginal (less than 0.1 dB at 10^{-4} BER). This demonstrates that the algorithm is very efficient and converges extremely fast (to within one iteration in this case). Thus, in practice, the second iteration may not be necessary nor desirable given the diminishingly small gain and the increased computational complexity and delays.

Figures 7 and 6 show the performance of the MIMO-SESS system with three- and four-transmit-antenna configurations, respectively. Clearly, the spatial diversity gains have increased compared to the two-transmit-antenna case

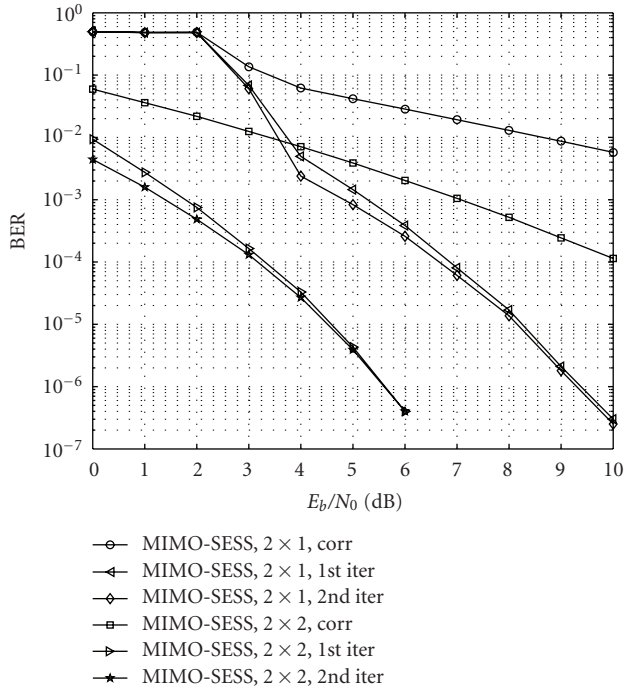


Figure 5: Performance of the proposed MIMO-SESS system with iterative detection, two transmit antennas using full-rate code $\mathcal{G}_2$, $N = 64$.

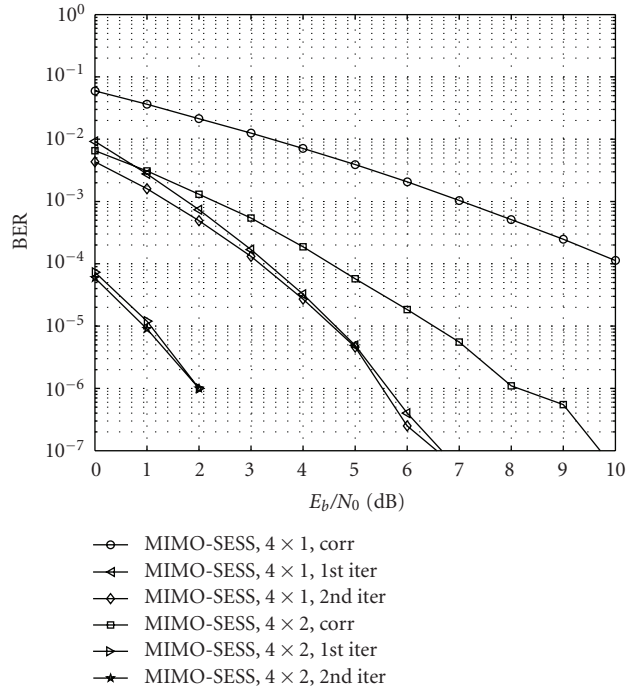


Figure 6: Performance of the proposed MIMO-SESS system with iterative detection, four transmit antennas using half-rate code $\mathcal{G}_4$, $N = 64$.

as evident from the steeper slopes of the BER curves. This can also be quantified from the plots by observing, for example, that the overall gain at 10^{-4} BER has decreased from 6.7 dB for the 2×2 scenario to about 5.3 dB and 4.5 dB for the 3×2 and 4×2 configurations, respectively. Again, the performance gain with the iterative detection is significant for all antenna configurations. In particular, the 4×2 configuration requires an E_b/N_o of just less than 0 dB at 10^{-4} BER. It should be noted that these results also reflect additional coding gains (compared to the results in Figure 5) since the space-time codes are half rates.

Finally, Figure 8 shows the effects of varying the spreading lengths on the example 2×2 MIMO-SESS system. The performance of a BPSK system under AWGN and a 2×2 MIMO-PNSS system under Rayleigh fading has also been plotted for comparison. The plots show that the performance improves with the spreading length (especially at low SNR) because the temporal diversity increases with N (more Rake fingers) as would be expected. When compared to the BPSK system under AWGN, the performance gain is nearly 5 dB at 10^{-4} BER, demonstrating that the fading effect on the BER has been completely mitigated with MIMO-SESS. The results also show that this performance can be achieved with a relatively small value of $N = 64$.

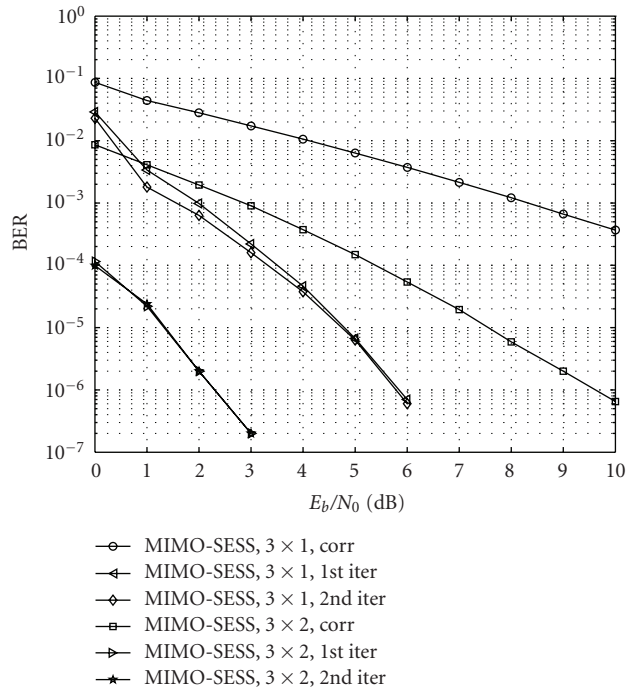


Figure 7: Performance of the proposed MIMO-SESS system with iterative detection, three transmit antennas using half-rate code $\mathcal{G}_3$, $N = 64$.

5. Conclusion

In this paper, we have described a novel MIMO-SESS technique with iterative detection as a means to provide temporal and spatial diversities for wireless communications over fading channels. The proposed scheme combines SESS with MIMO space-time block coding in a decoupling manner

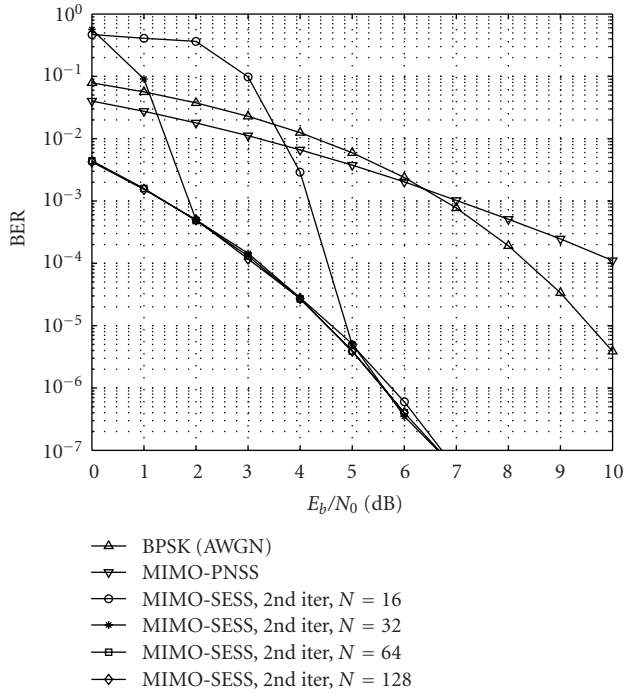


Figure 8: Performance of the proposed 2×2 MIMO-SESS system with iterative detection, $N = 16, 32, 64, 128$.

such that the number of antennas is not constrained by SESS structure. Our work has examined example OSTBC with two to four transmit antennas and one to two receive antennas in fast Rayleigh fading channels. The results show that SESS with iterative detection can significantly improve the overall BER performance, by as much as 6.7 dB gain over MIMO-PNSS for a 2×2 antenna configuration. The linear iterative detection is based on soft estimates and has been found to converge quickly after only one iteration, making it very efficient for a practical implementation.

Our work has shown that significant diversity gain can be exploited from SESS-modulated signals such that the proposed system can completely mitigate the fading effect and achieve a BER performance of only 3.3 dB SNR at 10^{-4} BER for the 2×2, to just less than 0 dB for the 4×2. The performance will be poorer if the channel fading is slow and the symbols experience block fading because the temporal diversity gain with SESS will be reduced, especially if the fading block length exceeds the spreading length. In general, the diversity loss due to slow or block fading could be compensated for and recovered with interleaving.

The approach in this paper could be extended to multi-code MIMO-SESS for BER improvement while enhancing the system throughput. Similarly, we anticipate that the spectral efficiency of the system can be further improved by incorporating QAM with SESS and by performing self encoding across multiple QAM symbols. We whould also like to point out that the decoupling nature of the proposed MIMO-SESS technique suggests the intriguing possibility of a coupled approach that may be advantageous. The design of such an encoding and decoding scheme (possibly together with error correction capability) for MIMO-SESS and its performance in fading channels could be an interesting and challenging problem.

Acknowledgments

The authors would like to express their appreciation for the valuable comments from the editor as well as the anonymous reviewers of their manuscript. This work was funded in part by Contract award FA9550-08-1-0393 from the U.S. Air Force Office of Scientific Research. They wish to thank Dr. J. Sjogren for his support of this study.

References

[1] L. Nguyen, "Self-encoded spread spectrum communications," in *Proceedings of the IEEE Military Communications Conference (MILCOM '99)*, vol. 1, pp. 182–186, October 1999.

[2] L. Nguyen, "Self-encoded spread spectrum and multiple access communications," in *Proceedings of the 6th IEEE International Symposium on Spread Spectrum Techniques and Applications*, vol. 2, pp. 394–398, Parsippany, NJ, USA, September 2000.

[3] L. Chi, Y. H. Jung, W. M. Jang, and L. Nguyen, "Self-encoded spread spectrum with iterative detection in multi-rate multimedia communication systems," in *Proceedings of the 6th International Conference on Digital Content, Multimedia Technology and its Applications (IDC '10)*, Seoul, South Korea, August 2010.

[4] P. Duraisamy and L. Nguyen, "Coded-sequence self-encoded spread spectrum communications," in *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM '09)*, pp. 1–5, Honolulu, Hawaii, USA, 2009.

[5] Y. S. Kim, W. M. Jang, Y. Kong, and L. Nguyen, "Chip-interleaved self-encoded multiple access with iterative detection in fading channels," *Journal of Communications and Networks*, vol. 9, no. 1, pp. 50–55, 2007.

[6] A. Tehrani, R. Negi, and J. Cioffi, "Space-time coding over a code division multiple access system," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC '99)*, New Orleans, La, USA, September 1999.

[7] H. Huang, H. Viswanathan, and G. J. Foschini, "Achieving high data rates in CDMA systems using BLAST techniques," in *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM '99)*, vol. 5, pp. 2316–2320, Rio de Janeiro, Brazil, December 1999.

[8] B. Hochwald, T. L. Marzetta, and C. B. Papadias, "A transmitter diversity scheme for wideband CDMA systems based on space-time spreading," *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 1, pp. 48–60, 2001.

[9] D. Gesbert, M. Shafi, D.-S. Shiu, P. J. Smith, and A. Naguib, "From theory to practice: an overview of MIMO space-time coded wireless systems," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 3, pp. 281–302, 2003.

[10] A. J. Paulraj, D. A. Gore, R. U. Nabar, and H. Bölcskei, "An overview of MIMO communications—a key to gigabit wireless," *Proceedings of the IEEE*, vol. 92, no. 2, pp. 198–217, 2004.

[11] A. Paulraj, R. Nabar, and D. Gore, *Introduction to Space-Time Wireless Communications*, Cambridge University Press, Cambridge, UK, 2003.

[12] S. Ma, L. Nguyen, W. M. Jang, and Y. Yang, "Performance enhancement in MIMO self-encoded spread spectrum systems by using multiple codes," in *Proceedings of the 33rd IEEE Sarnoff Symposium*, Princeton, NJ, USA, April 2010.

[13] J. G. Proakis, *Digital Communications*, McGraw–Hill, New York, NY, USA, 4th edition, 1989.

[14] 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.

[15] D. J. Love, R. W. Heath Jr., and T. Strohmer, "Grassmannian beamforming for multiple-input multiple-output wireless systems," *IEEE Transactions on Information Theory*, vol. 49, no. 10, pp. 2735–2747, 2003.

[16] B. C. Lim, W. A. Krzymień, and C. Schlegel, "Transmit antenna selection for sum rate maximization in transmit zero-forcing beamforming," in *Proceedings of the 10th IEEE Singapore International Conference on Communications Systems (ICCS '06)*, October 2006.

[17] Y. Ma, A. Leith, and R. Schober, "Predictive feedback for transmit beamforming with delayed feedback and channel estimation errors," in *Proceedings of the IEEE International Conference on Communications (ICCS '08)*, pp. 4678–4682, Beijing, China, May 2008.

[18] V. Tarokh, H. Jafarkhani, and A. R. Calderbank, "Space-time block codes from orthogonal designs," *IEEE Transactions on Information Theory*, vol. 45, no. 5, pp. 1456–1467, 1999.

[19] G. Ganesan and P. Stoica, "Space-time block codes: a maximum SNR approach," *IEEE Transactions on Information Theory*, vol. 47, no. 4, pp. 1650–1656, 2001.

[20] W. Su, X.-G. Xia, and K. J. R. Liu, "A systematic design of high-rate complex orthogonal space-time block codes," *IEEE Communications Letters*, vol. 8, no. 6, pp. 380–382, 2004.

[21] G. J. Foschini, "Layered space-time architecture for wireless communication in a fading environment when using multi-element antennas," *Bell Labs Technical Journal*, vol. 1, no. 2, pp. 41–59, 1996.

[22] D. J. Love and R. W. Heath Jr., "Limited feedback precoding for spatial multiplexing systems," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM '03)*, vol. 4, pp. 1–5, San Francisco, Calif, USA, December 2003.

[23] A. S. Khrwat, B. S. Sharif, C. C. Tsimenidis, and S. Boussakta, "Channel prediction for precoded spatial multiplexing multiple-input multiple-output systems in time-varying fading channels," *IET Signal Processing*, vol. 3, no. 6, pp. 459–466, 2009.

[24] V. Tarokh, H. Jafarkhani, and A. R. Calderbank, "Space-time block codes for wireless communications: performance results," *IEEE Transaction on Information Theory*, vol. 45, no. 5, pp. 1456–1467, 1999.

[25] S. M. Alamouti, "A simple transmit diversity technique for wireless communications," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 8, pp. 1451–1458, 1998.

[26] K. Hua, L. Nguyen, and W. M. Jang, "Synchronisation of self-encoded spread spectrum system," *Electronics Letters*, vol. 44, no. 12, pp. 749–751, 2008.

Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2010, Article ID 432495, 17 pages
doi:10.1155/2010/432495

Research Article

The Manifestation of Stopping Sets and Absorbing Sets as Deviations on the Computation Trees of LDPC Codes

Eric Psota[1] and Lance C. Pérez[2]

[1] *University of Nebraska-Lincoln, 329 SEC, Lincoln, NE 68588-0511, USA*
[2] *University of Nebraska-Lincoln, 243N SEC, Lincoln, NE 68588-0511, USA*

Correspondence should be addressed to Eric Psota, epsota24@huskers.unl.edu

Received 16 March 2010; Accepted 10 June 2010

Academic Editor: Christian Schlegel

The error mechanisms of iterative message-passing decoders for low-density parity-check codes are studied. A tutorial review is given of the various graphical structures, including trapping sets, stopping sets, and absorbing sets that are frequently used to characterize the errors observed in simulations of iterative decoding of low-density parity-check codes. The connections between trapping sets and deviations on computation trees are explored in depth using the notion of *problematic* trapping sets in order to bridge the experimental and analytic approaches to these error mechanisms. A new iterative algorithm for finding low-weight problematic trapping sets is presented and shown to be capable of identifying many trapping sets that are frequently observed during iterative decoding of low-density parity-check codes on the additive white Gaussian noise channel. Finally, a new method is given for characterizing the weight of deviations that result from problematic trapping sets.

1. Introduction

Prior to 1993, channel codes were typically designed with the goal of maximizing the minimum distance of the code [1, 2]. The combination of a code with a large minimum distance and a decoder that minimizes the probability of codeword error often resulted in good asymptotic performance. The discovery of turbo codes [3] and the rediscovery of low-density parity-check (LDPC) codes [4] revealed that codes with relatively poor minimum distance properties could achieve near-capacity performance at bit error rates of $P_b < 10^{-6}$. This resulted in a reduced emphasis on maximizing minimum distance when design codes for use on the additive white Gaussian noise (AWGN) channel. As with many other classes of codes, there are no practical bounds for the decoders used for turbo codes and LDPC codes and simulations are required to accurately determine the performance at practical operating points.

With the discovery of turbo codes and the various subsequent iterative decoders, the phenomenon of the error floor has become prominent in practical code design. The term *error floor* refers to the situation where the error rate at the output of the decoder suddenly starts to decrease at a slower rate as a function of increasing signal-to-noise ratio (SNR); that is, the performance curve flattens out. The error floors that occur with iterative decoding of turbo codes and LDPC codes are problematic in practical systems because it is difficult to predict the specific operating point at which they occur, and thus design engineers risk using codes that may have unknown error floors that limit the performance of the system. In the case of iterative decoding of turbo codes, it has been shown that the error floor is usually the result of the overall turbo code having low weight codewords that begin to limit the performance of the code after some SNR is reached [5]. The minimum distance of a turbo code can be increased, and hence the likelihood of an error floor sufficiently mitigated, through the use of various interleaver designs [6, 7].

Low-density parity-check codes with iterative decoding are also known to exhibit error floors [8, 9], albeit at much lower bit error rates than the error floors of turbo codes of similar block length. In many cases, because of the exceptionally low bit error rates at which the error floors of LDPC codes appear, it is not practical to use Monte

Carlo simulations to demonstrate the existence of these error floors. The inability to run conventional computer simulations down to the error floor combined with the lack of practical upper bounds for LDPC codes has inhibited the deployment of LDPC codes in high-throughput applications that require near error-free performance with bit error rates of $P_b < 10^{-15}$.

LDPC codes are most commonly decoded using iterative message-passing decoders such as the min-sum decoder [10] and the sum-product decoder [11] due to their excellent performance and low implementation complexity. Many attempts have been made to estimate the performance of these decoders by characterizing their error mechanisms. Three of the most well-known error mechanisms are stopping sets [12], trapping sets [13], and absorbing sets [8]. Unfortunately, none of these mechanisms leads to strict upper bounds on the performance of iterative decoding over the AWGN channel, and thus they are of limited use in determining error floors. Wiberg showed that deviations on the computation trees of LDPC codes can be used to compute tight upper bounds on the performance of LDPCs with iterative decoders [10]; however, it is computationally intractable to do so after even a small number of decoder iterations. A practical method for determining the error floor of LDPCs with iterative decoding has yet to be discovered.

This paper attempts to make progress on this problem by integrating the precise, but computational intractable, work of Wiberg with the experimental studies of the error mechanisms observed when iteratively decoding LDPCs. The paper begins with a tutorial review of the existing methods for analyzing the performance of iterative message-passing decoders. Then, the notion of a *problematic* trapping set is introduced and its relationship to deviations is examined in detail, with the goal of determining what makes deviations either more or less problematic during iterative message-passing decoding. Finally, an iterative method is given for finding problematic trapping sets using the weights of deviations on the computation trees.

2. Background

The following model for channel coding is used throughout this paper. First, a vector $\mathbf{u} \in \mathbb{F}_2^K$ of K information bits is generated by a binary source. The binary source is assumed to be memoryless, which is often the result of source coding (data compression), and therefore all information sequences in $\mathbb{F}_2^K$ are equally probable. A binary $K \times N$ generator matrix G may be used by the channel encoder to map the information bits $\mathbf{u}$ to a length N codeword $\mathbf{c} \in \mathbb{F}_2^N$, via the mapping $\mathbf{c} = \mathbf{u}G$. Here, it is assumed that the matrix G is full-rank, and thus the rate of the code is $R = K/N$.

Before a codeword $\mathbf{c} \in C$ is transmitted over the channel, it is modulated via the transformation

$$x_i = m(c_i) = 2c_i - 1, \tag{1}$$

for all $i = 0, \ldots, N-1$. The received signal vector $\mathbf{y} \in \mathbb{R}^N$ is given by

$$\mathbf{y} = \mathbf{x} + \mathbf{n}, \tag{2}$$

where $\mathbf{n} \in \mathbb{R}^N$ is the Gaussian noise vector. The log-likelihood ratio (LLR) vector, often used for soft-decision decoding, is given by

$$\lambda_i = \frac{P_{Y|X}(y_i \mid -1)}{P_{Y|X}(y_i \mid 1)}, \tag{3}$$

for all $i = 1, \ldots, N$. This reduces to $\lambda_i = (-2/\sigma^2)y_i$ when $\mathbf{n}$ is a vector of AWGN. An estimate $\hat{\mathbf{c}}$ of the transmitted codeword $\mathbf{c}$ is derived from the received vector $\mathbf{y}$ at the channel decoder. Finally, the information bits $\hat{\mathbf{u}}$ extracted from $\hat{\mathbf{c}}$ are passed to the sink.

As mentioned earlier, each of the information sequences $\mathbf{u} \in \mathbb{F}_2^K$ is equiprobable. Since there is a one-to-one mapping between information sequences and codewords, all codewords in the code C are equiprobable as well. Therefore, $P(\mathbf{c}_i) = P(\mathbf{c}_j)$ for all $\mathbf{c}_i, \mathbf{c}_j \in C$, where $P(\mathbf{c}_i)$ is the probability that codeword $\mathbf{c}_i$ is transmitted. When considering the performance of linear codes, equiprobable codewords allow for the assumption that the all-zeros codeword was transmitted. The all-zeros codeword assumption is used throughout this paper.

From the generator matrix G, it is possible to derive an $(N-K) \times N$ parity-check matrix H for the code. A *parity-check matrix* of a code C is any matrix H, such that $H\mathbf{c}^T = \mathbf{0}$ for all $\mathbf{c} \in C$. LDPC codes are often defined by their parity-check matrix H. In particular, LDPC codes are a class of codes with sparse parity-check matrices. A *sparse* parity-check matrix is any binary matrix that contains more binary 0s than binary 1s. A (d_V, d_F)-*regular* LDPC code is one that has a fixed number d_V of binary 1s in each column of the parity-check matrix and some fixed number d_F of binary 1s in each row of the parity-check matrix. An example of a (2,3)-regular LDPC code of length $N = 6$ and dimension $K = 3$ is given by the parity-check matrix

$$H_{(2,3)} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}. \tag{4}$$

The parity-check matrix of a length N, dimension K code must contain at least $(N-K)$ rows, since the kernel of G has dimension $(N-K)$. However, it is possible for the parity-check matrix to contain more than $(N-K)$ rows. Therefore, the number of rows in the parity-check matrix is denoted by M, where $M \geq (N-K)$.

A *Tanner graph* is a bipartite graphical representation of a low-density parity-check matrix. To construct a Tanner graph from a parity-check matrix, each column i in the parity-check matrix is assigned to a corresponding variable node v_i in the Tanner graph, and each row j is assigned to a corresponding check node f_j in the Tanner graph. The set of all variable nodes is V, and the set of all check nodes is F. There is an edge $e_{i,j}$ between variable node v_i and check node f_j in the Tanner graph if and only if the entry in H at the intersection of the jth row and ith column is a binary 1. The Tanner graph $T = (V \bigcup F, E)$ is thus defined by the set of variable nodes V, the set of check nodes F, and the set of

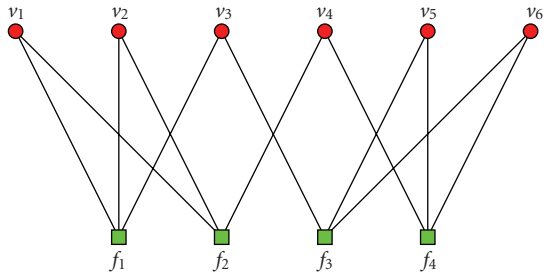


Figure 1: Tanner graph of a $(2,3)$-regular LDPC code.

edges E. The Tanner graph corresponding to the parity-check matrix $H_{(2,3)}$ (given by (4)) is shown in Figure 1.

Note that in the Tanner graphs of irregular LDPC codes variable nodes and check nodes do not all have the same number of incident edges. The number of check nodes that a specific variable node v_i is connected to is denoted by d_{v_i}, and the number of variable nodes that a specific check node f_i is connected to is denoted d_{f_i}.

2.1. Iterative Message-Passing Decoding. The min-sum (MS) and sum-product (SP) decoders are low-complexity, suboptimal iterative decoders that can be used to decode low-density parity-check codes. Given a particular parity-check matrix, the MS decoder operates by passing messages between the check nodes and the variable nodes along the edges of the Tanner graph of the code.

Before introducing the decoders, some additional notation is necessary. The set of neighbors of check node f_i in the Tanner graph is denoted $N(f_i) = \{v_j \mid h_{i,j} = 1\}$, and similarly the set of neighbors of variable node v_i in the Tanner graph is denoted $N(v_i) = \{f_j \mid h_{j,i} = 1\}$. To denote the set of neighbors of check node f_i excluding variable node v_j, the notation $N(f_i) \setminus v_j$ is used. Similarly, the set of neighbors of variable node v_j excluding check node f_i is denoted $N(v_j) \setminus f_i$. During decoding, messages are passed between neighboring check nodes and variable nodes along the edges of the Tanner graph. Messages from check node f_i to variable node $v_j \in N(f_i)$ are denoted by $m_{f_i \to v_j}$, and messages from variable node v_i to check node $f_j \in N(v_i)$ are denoted $m_{v_i \to f_j}$. Given the transmitted codeword $\mathbf{x}$, the channel output $\mathbf{y}$ available at the receiver, and a maximum number of iterations $\ell_{\max}$, the steps for MS and SP decoding are given in the following algorithm

Algorithm 1 (Min-Sum/Sum-Product Decoding).

Step 1 (Initialization). Set the number of iterations to $\ell = 0$. For all messages $m_{f_i \to v_j}$, set

$$m_{v_i \to f_j} = \lambda_i = \frac{P_{Y|X}(y_i \mid -1)}{P_{Y|X}(y_i \mid 1)} = \frac{-2}{\sigma^2} y_i. \tag{5}$$

Step 2 (Check Node Update). Set $\ell = \ell + 1$. For all messages $m_{v_i \to f_j}$, set

Min-Sum:

$$m_{f_i \to v_j} = \left(\prod_{v_k \in N(f_i) \setminus v_j} \operatorname{sgn}\left(m_{v_k \to f_i}\right) \right) \left(\min_{v_k \in N(f_i) \setminus v_j} \mid m_{v_k \to f_i} \right). \tag{6}$$

Sum-Product:

$$m_{f_i \to v_j} = 2 \cdot \tanh^{-1} \left(\prod_{v_k \in N(f_i) \setminus v_j} \tanh\left(\frac{m_{v_k \to f_i}}{2} \right) \right). \tag{7}$$

Step 3 (Variable Node Update). For all messages $m_{v_i \to f_j}$, set

$$m_{v_i \to f_j} = \lambda_i + \sum_{f_k \in N(v_i) \setminus f_j} m_{f_k \to v_i}. \tag{8}$$

Step 4 (Check Stop Criteria). For all m_{v_i}, set

$$m_{v_i} = \lambda_i + \sum_{f_k \in N(v_i)} m_{f_k \to v_i}. \tag{9}$$

For all $\hat{c}_i$, set

$$\hat{c}_i = \begin{cases} 0 & \text{if } m_{v_i} > 0, \\ 1 & \text{if } m_{v_i} < 0, \end{cases} \tag{10}$$

with $P(\hat{c}_i = 0 \mid m_{v_i} = 0) = P(\hat{c}_i = 1 \mid m_{v_i} = 0) = 0.5$.

If $H\hat{\mathbf{c}}^T = \mathbf{0}$ or $\ell \geq \ell_{\max}$, stop decoding, else return to Step 2.

One of the primary strengths of the min-sum and sum-product decoders is the relatively small number of operations performed during each iteration. During each iteration, the messages $m_{v_i \to f_j}$ and $m_{f_j \to v_i}$ must be computed for each binary 1 in the parity-check matrix. For a (d_V, d_F)-regular LDPC code, there are $(N \times d_V) = (M \times d_F)$ binary 1s in the parity-check matrix. When the degree of the nodes and the number of iterations is fixed, the complexity of MS decoding scales linearly with the length N of the code.

In practice, the min-sum and sum-product decoders do not always output a codeword. It has been shown that when the MS decoder does not output a codeword after a large number (>200) of iterations has been performed, the output often cycles in a repeating sequence of two or more noncodeword outputs [14]. In Sections 2.2 through Section 2.4, three different characterizations are given for the noncodeword outputs of iterative message-passing decoders.

2.2. Stopping Sets. The notion of stopping sets was first introduced by Forney et al. [15] in 2001. Two years later, a formal definition of stopping sets was given by Di et al. [12]. They demonstrated that the bit and word error probabilities of iteratively decoded LDPC codes on the binary erasure channel (BEC) can be determined exactly from the stopping sets of the parity-check matrix.

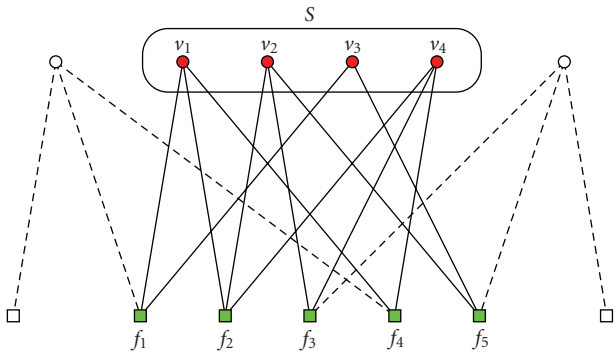


Figure 2: Example of a stopping set in the Tanner graph of an LDPC code.

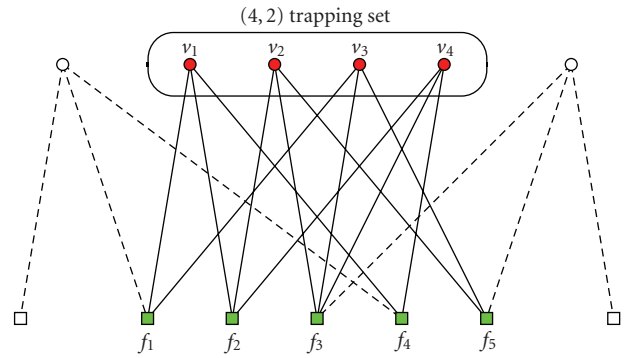


Figure 3: Example of a $(4, 2)$ trapping set in the Tanner graph of an LDPC code.

Definition 1 (stopping sets [12]). A stopping set $\mathcal{S}$ is a subset of the set of variable nodes V, such that any check node connected to a variable node contained in $\mathcal{S}$ is connected to at least two variable nodes in $\mathcal{S}$.

A small example of a stopping set is given in Figure 2. Consider the subset $\mathcal{S} = \{v_1, v_2, v_3, v_4\}$ of the set of variable nodes V. There are five check nodes $\{f_1, f_2, f_3, f_4, f_5\}$ connected to the set $\mathcal{S}$, and each of them is connected to $\mathcal{S}$ at least two times. Note that only f_2 is connected to the set $\mathcal{S}$ an odd number of times; If each of the check nodes is connected to $\mathcal{S}$ an even number of times, $\mathcal{S}$ corresponds to a codeword support set where all bits in $\mathcal{S}$ can be flipped without changing the overall parity of any of the check nodes.

The intuition behind stopping sets begins with an understanding of iterative message-passing decoders. Information given to a specific variable node from a neighboring check node is derived from all other variable nodes connected to that check node. Consider two variable nodes $v_i, v_j \in N(f_k)$, where both variable nodes contain an erasure. In this case, each of the sets $N(f_k) \setminus v_i$ and $N(f_k) \setminus v_j$ contains at least one erasure, thus making it impossible for the check node f_k to determine the parity of either set. For this reason, none of the check nodes connected to a stopping set is capable of resolving erasures, if each variable node contained in the stopping set begins with an erasure from the channel.

Work relating linear programming (LP) pseudocodewords to stopping sets for the binary erasure channel [15], and both the binary symmetric channel (BSC) and the additive white Gaussian noise channel [16], has revealed a relationship between linear programming pseudocodewords and the size of stopping sets. Although stopping sets have a strong relationship with LP pseudocodewords, the performance of neither the MS decoder or the SP decoder on the BSC and AWGN channels can be predicted using stopping sets alone.

2.3. Trapping Sets. Trapping sets, also referred to as near-codewords, were first introduced by MacKay and Postol [13] to provide an explanation for the weaknesses of algebraically constructed low-density parity-check codes. They define trapping sets as follows.

Definition 2 (trapping sets [13]). Consider a length N code with parity-check matrix H, and let $\mathcal{T} \subseteq \{1, \ldots, N\}$ be a set containing $|\mathcal{T}| = t$ coordinates. Consider a length-N binary vector $\mathbf{y}$ with 1s in the coordinates of $\mathcal{T}$ and 0s elsewhere. If the syndrome $\mathbf{s} = H\mathbf{y}$ has Hamming weight w_t, the set $\mathcal{T}$ is referred to as a (t, w_t) trapping set.

Consider the trapping set shown in Figure 3, where the set $\mathcal{T} = \{1, 2, 3, 4\}$ corresponds with a set of variable nodes $\{v_1, v_2, v_3, v_4\}$ in the Tanner graph of the parity-check matrix H. There are four variable nodes in the set, so $t = 4$, and if all variable nodes are set to a binary 1, only check nodes f_2 and f_3 are connected to an odd number of binary 1s, so the syndrome $\mathbf{s}$ has Hamming weight equal to 2. Therefore, according to Definition 2, this set of variable nodes defines a $(4, 2)$ trapping set.

It is important to note that any set of variable nodes can be considered a trapping set defined by some set of parameters, and the significance of trapping sets varies greatly depending on the parameters (t, w_t). In much the same way that low-weight codewords are problematic to decoding, erroneous channel information is more likely to affect the majority of variable nodes in a trapping set which has low-weight t. Richardson [17] shows that trapping sets with small weight t and a small number of unsatisfied check nodes w_t are more likely to cause errors. When a trapping set has small w_t, the extrinsic information being passed into $\mathcal{T}$ can not overcome the intrinsic information reinforced within $\mathcal{T}$.

In [17], trapping sets are examined for different decoders on the binary erasure channel, binary symmetric channel, and the additive white Gaussian noise channel. Whereas stopping sets can be used to precisely determine the probability of error on the BEC, trapping sets appear to cause errors on the AWGN channel. Richardson [17] uses the parameters and multiplicity of various problematic trapping sets to estimate the error floor of LDPC codes at bit error rates where simulations are not feasible. Unfortunately, the somewhat vague definition of problematic trapping sets makes it difficult to use them for performance analysis.

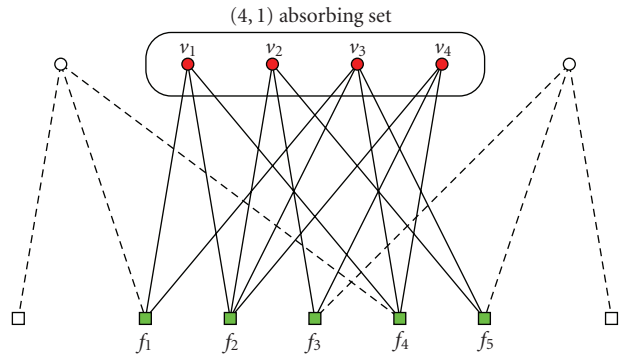


Figure 4: Example of a $(4, 1)$ absorbing set in the Tanner graph of an LDPC code.

2.4. Absorbing Sets. In an attempt to clarify the ambiguity of problematic trapping sets, Zhang et al. introduced the notion of absorbing sets [8]. They define absorbing sets as follows.

Definition 3 (Absorbing Sets [8]). Let $\mathcal{A} \subseteq V$ be a set containing $|\mathcal{A}| = a$ variable nodes. Also, let $O(\mathcal{A}) \subseteq F$ be a set of check nodes such that $|O(\mathcal{A})| = w_a$, and each check node in the set $O(\mathcal{A})$ has an odd number of edges connected to $\mathcal{A}$. If each variable node in $\mathcal{A}$ is connected to strictly more check nodes in $F \setminus O(\mathcal{A})$ than in $O(\mathcal{A})$, the set $\mathcal{A}$ is referred to as a (a, w_a) absorbing set. A *fully absorbing set* also satisfies the condition that each variable node in V is connected to more check nodes in $F \setminus O(\mathcal{A})$ than in $O(\mathcal{A})$.

Note that an (a, w_a) absorbing set is also an (a, w_a) trapping set, but the converse is not always true. Figure 4 shows an example of a $(4, 1)$ absorbing set. The set of variable nodes in this absorbing set is $\mathcal{A} = \{v_1, v_2, v_3, v_4\}$, and the set of unsatisfied check nodes is $O(\mathcal{A}) = \{f_4\}$. The variable node v_2 is not connected to f_4, and the variable nodes v_1, v_3, and v_4 are each connected to f_4 and at least two other check nodes in F. Therefore, each of the variable nodes in $\mathcal{A}$ is connected to more satisfied check nodes than unsatisfied check nodes. Also, note that the $(4, 2)$ trapping set in Figure 3 is not an absorbing set because variable nodes v_2 and v_4 are connected to two unsatisfied check nodes and only one satisfied check node.

Simulations show that the majority of errors encountered in the error floor region during sum-product decoding of the IEEE 802.3 an low-density parity-check code could be attributed to absorbing sets [8, 9]. Although absorbing sets appear to be useful for estimating the performance of iterative message-passing decoding, they do not lead to strict upper bounds. For upper bounds, it is possible to use the concept of deviations on the computation tree.

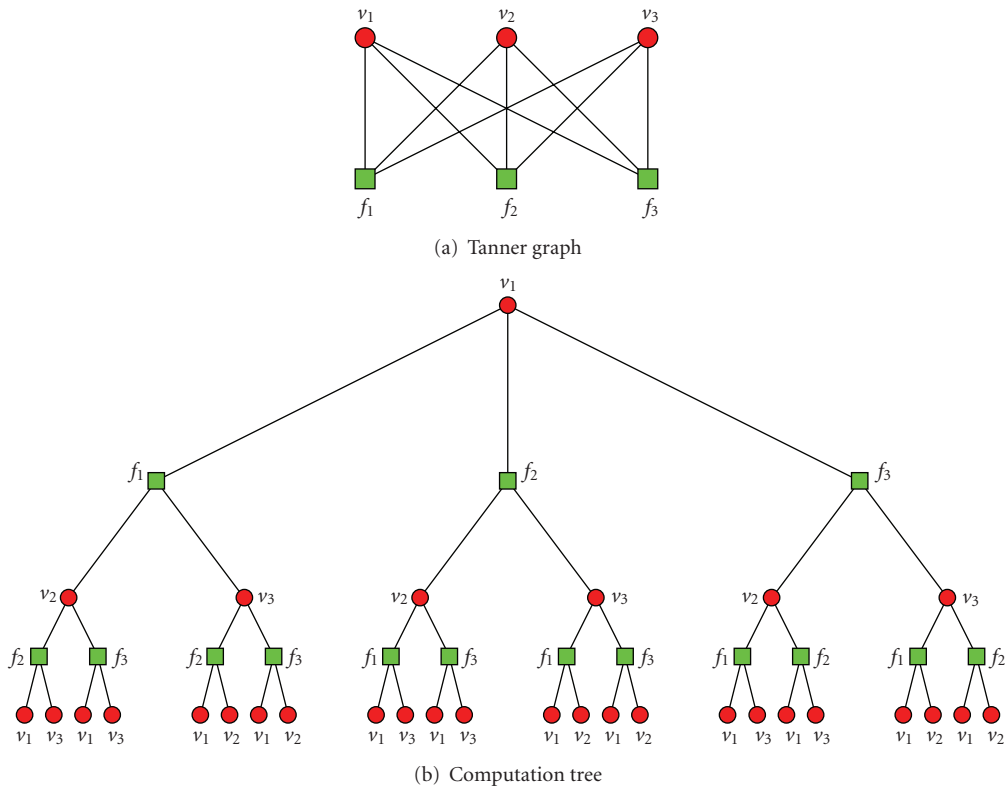
2.5. Computation Trees and Deviations. In his 1996 dissertation, Wiberg [10] presented groundbreaking analytical results with respect to iterative decoding of low-density parity-check codes. He provided extensive analysis of both the MS and SP decoders by introducing a model of iterative decoding known as the computation tree. Wiberg showed that the MS decoder minimizes the probability of word error when decoding a code whose Tanner graph is a tree, while for the same type of code the SP decoder minimizes the probability of bit error.

In addition to introducing computation trees, Wiberg also introduced the concept of deviations. Wiberg proved that deviations on the computation tree with negative cost are required in order for errors to occur during MS and SP decoding. Because of the importance of computation trees and deviations in understanding finite tree-based decoding, they are examined in detail in this section.

Consider a low-density parity-check code represented by a Tanner graph $T = (V \bigcup F, E)$. A computation tree rooted at variable node v_i after ℓ iterations is denoted $R_{v_i}^{(\ell)}$. In order to construct a computation tree from the Tanner graph, a variable node v_i is placed at the top level (root) of a descending tree. To construct the next level in the tree directly below v_i, each of v_i's neighbors in $N(v_i)$ is added to this level and connected to v_i. This process continues level-by-level, where nodes in the previous level are used to determine nodes on next level, while maintaining that each node in the computation tree has the same set of neighbors as its corresponding node in the Tanner graph. For example, if variable node v_j on the last completed level is connected to check node f_k on the level above it, then all check nodes in $N(v_j) \setminus f_k$ must appear on the next level and be connected to v_j, thereby ensuring that v_j is connected to exactly one copy of each check node in $N(v_j)$.

Figure 5 gives an example of a Tanner graph, and its corresponding computation tree rooted at v_1 after two iterations. Nodes at the bottom level of the computation tree are referred to as *leaf nodes*. Notice that the leaf nodes are the only nodes in the computation tree that are not connected to a copy of each of their neighbors in the original Tanner graph.

Computation trees are precise models for analyzing the performance and behavior of min-sum and sum-product decoding for a finite set of iterations. Each of these decoders can be precisely modeled after ℓ iterations by constructing N different computation trees that contain $2\ell+1$ levels of nodes including the root node. The N computation trees are each rooted at a different variable node from the original Tanner graph. Then, for every variable node v_i in each computation tree, the LLR cost γ_i is assigned to that variable node. At this point, MS or SP decoding operations can be performed from the leaf nodes up to the root node. The final cost at each of the root nodes determines the binary estimate of the transmitted codeword computed by the decoder. Because the MS and SP decoders are optimal on Tanner graphs that are trees, the MS and SP decoders are optimal on each of the computation trees derived from the Tanner graph. MS chooses the least cost valid configuration on the tree, where a *valid configuration* refers to any assignment of binary numbers to the variable nodes such that each check node is adjacent to an even number of variable nodes assigned to a binary 1. The SP decoder, on the other hand, chooses the value at the root node that has the highest probability over all valid configurations.



(a) Tanner graph

(b) Computation tree

Figure 5: Computation tree of a simple repetition code after $\ell = 2$ iterations.

Although the computation tree model is precise, after a small number of iterations it becomes impractical to analyze the performance of specific codes by considering all valid configurations on the computation tree. The number of valid configurations on the computation tree can be computed by treating the computation tree as a Tanner graph. In order to define a Tanner graph given the computation tree, treat all check nodes and variable nodes in the computation tree separately. For example, if multiple copies of variable node v_1 are distributed throughout the computation tree, each copy is treated as a distinct variable node. After regarding each variable node in the computation tree as distinct, one can show that each check node on the computation tree corresponds to a linearly independent parity-check equation. If there are $|R_{v_i}^{(\ell)}(V)|$ variable nodes and $|R_{v_i}^{(\ell)}(F)|$ check nodes on a computation tree rooted at variable node v_i after ℓ iterations, then there are a total of $2^{|R_{v_i}^{(\ell)}(V)|-|R_{v_i}^{(\ell)}(F)|}$ valid configurations on the tree. On a (d_V, d_F)-regular LDPC code, the number of variable nodes after ℓ iterations is given by

$$\left|R_{v_i}^{(\ell)}(V)\right| = 1 + \sum_{i=0}^{\ell-1} d_V(d_F-1)\left(((d_V-1)(d_F-1))^i\right), \tag{11}$$

and the number of check nodes is given by

$$\left|R_{v_i}^{(\ell)}(F)\right| = \sum_{i=0}^{\ell-1} d_V\left(((d_V-1)(d_F-1))^i\right). \tag{12}$$

Table 1: The number of nodes and valid configurations on the computation tree of a $(3,6)$-regular LDPC code.

Iterations	Variable Nodes	Check Nodes	Configurations
1	16	3	8192
2	166	33	$\approx 10^{40}$
3	1666	333	$\approx 10^{401}$

To illustrate the growth rate in the number of valid configurations on the computation tree, consider an LDPC code where each variable node has degree $d_V = 3$ and each check node has degree $d_F = 6$. These commonly used code parameters result in what are known as a $(3,6)$-regular LDPC codes. Table 1 shows the number of variable nodes given by

$$\left|R_{v_i}^{(\ell)}(V)\right| = 1 + \sum_{i=0}^{\ell-1} 15\left(10^i\right), \tag{13}$$

the number of checks nodes given by

$$\left|R_{v_i}^{(\ell)}(F)\right| = \sum_{i=0}^{\ell-1} 3\cdot 10^i, \tag{14}$$

and the corresponding number of valid configurations on the computation tree after 1, 2, and 3 iterations. Note that the growth rate is not affected by the block length of the code.

Table 1 illustrates the computational complexity associated with considering each valid configuration on the

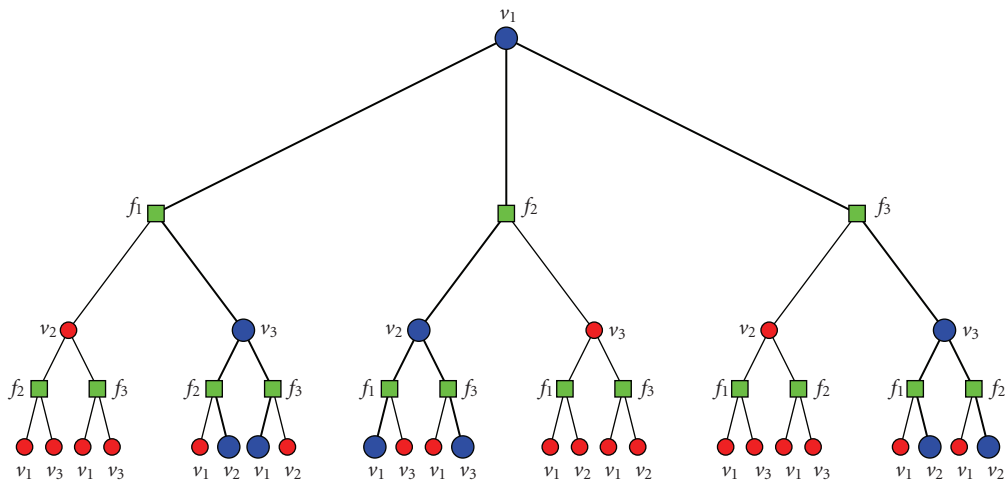


Figure 6: Example of a deviation on the computation tree.

computation tree. In light of this, Wiberg [10] derived a simplified bound on the performance of MS decoding operating on a particular computation tree. In order to obtain this bound, Wiberg introduced the concept of deviations on the computation tree.

Definition 4 (Deviation [10]). A *deviation* is any set of variable nodes on the computation tree satisfying the following three conditions.

(1) Each check node in the computation tree is adjacent to either two or zero variable nodes in the deviation set.

(2) A deviation set contains the root node of the computation tree.

(3) No proper and nonempty subset of variable nodes in the deviation from a valid configuration on the computation tree.

Figure 6 shows an example of a deviation on the computation tree given in Figure 5(b). The larger blue variable nodes are contained in the deviation, whereas the smaller red nodes are not.

Wiberg uses the set of deviations on the computation tree to derive an upper bound on the performance of the min-sum decoder. It is necessary, but not sufficient, for at least one deviation δ in the set of all deviations Δ to have negative cost in order for an error to occur at the root node. The cost of the deviation, denoted by $G(\delta)$, can be found by summing the LLR cost of each of the nodes in the support of the deviation. The cost of a deviation is given by

$$G(\delta) = \sum_{v_i \in \delta} \gamma_i, \tag{15}$$

where copies of $v_i \in \delta$ are counted as many times as they appear in the deviation. A necessary, but not sufficient, condition for an error to occur on the computation tree rooted at variable node v_i is given by [10]

$$\min_{\delta \in \Delta} G(\delta) < 0. \tag{16}$$

Using this condition, a bound can be derived on the probability that the minimum-cost configuration on the computation tree contains a binary 1 at the root node. This bound is

$$\begin{aligned} P(v_i = 1) &\leq P\left(\min_{\delta \in \Delta} G(\delta) < 0\right), \\ &\leq \bigcup_{\delta \in \Delta} P(G(\delta) < 0), \end{aligned} \tag{17}$$

which can be further loosened to

$$P(v_i = 1) \leq \sum_{\delta \in \Delta} P(G(\delta) < 0), \tag{18}$$

by using the union bound.

Wiberg [10] shows that the bound given by (18) can be used to predict the performance of min-sum decoding of infinite-length codes after a specific number of iterations. Wiberg begins by assuming that the computation trees have no repeated nodes. This assumption simplifies the weight enumerators of the deviations for regular LDPC codes. Wiberg also shows that (18) can be used to bound MS decoder performance when there are multiple copies of each variable node in the tree. Thus, in theory, Wiberg's deviation bound can be used to bound the performance of MS decoding of finite length codes. The following proposition shows that the number of deviations grows exponentially with d_V, thus making it computationally intractable to enumerate the deviations even after a small number of iterations.

Proposition 1. *Let $R_{v_i}^{(\ell)}$ be the computation tree of a (d_V, d_F)-regular LDPC code, rooted at variable node v_i after ℓ iterations. Then, the number of deviations that exist on $R_{v_i}^{(\ell)}$ is*

$$(d_F - 1)^{\sum_{i=1}^{\ell} d_V (d_V - 1)^{i-1}}. \tag{19}$$

Table 2: Number of deviations at iterations 1–5 for a $(3,6)$-regular LDPC code.

Iterations	# of deviations
1	125
2	1,953,125
3	4.7684×10^{14}
4	2.8422×10^{31}
5	1.0097×10^{65}

Proof. By the definition of a deviation, we must assign the root node v_i to a binary 1. Each of the d_V check nodes immediately below v_i must assign exactly one of the their $(d_F - 1)$ child variable nodes to a binary 1. Thus, there are a total of $(d_F - 1)^{d_V}$ deviations after one iteration. In addition, there are exactly d_V leaf nodes in the support of each deviation after one iteration.

Each of the previous d_V leaf nodes gets connected to $(d_V - 1)$ check nodes after two iterations. Each of these check nodes assigns one of their $(d_F - 1)$ child variable nodes to a binary 1. Therefore, for each deviation after one iteration there are $(d_F - 1)^{d_V(d_V-1)}$ different deviations after two iterations. This brings the total number of deviations to $(d_F - 1)^{d_V}(d_F - 1)^{d_V(d_V-1)} = (d_F - 1)^{(d_V)^2}$ after two iterations. The total number of leaf nodes in the support of the deviation after two iterations is $d_V(d_V - 1)$.

Following this pattern, the $d_V(d_V - 1)$ variable nodes in support of the deviation after two iterations branchs out to $d_V(d_V - 1)^2$ check nodes. There are $(d_F - 1)^{d_V(d_V-1)^2}$ ways of assigning the leaf nodes to the support of the previous deviation. This brings the total number of deviations to $(d_F - 1)^{(d_V)^2}(d_F - 1)^{d_V(d_V-1)^2} = (d_F - 1)^{(d_V)^3-(d_V)^2+d_V}$ after three iterations.

After ℓ iterations, the $d_V(d_V - 1)^{\ell-2}$ old leaf nodes in the support of the deviation branch out to $d_V(d_V - 1)^{\ell-1}$ new leaf nodes in the support of the deviation. There are $(d_F - 1)^{d_V(d_V-1)^{\ell-1}}$ ways of assigning the support to the previous deviation, and the total number of deviations after ℓ iterations is

$$\prod_{i=1}^{\ell}(d_F - 1)^{d_V(d_V-1)^{i-1}} = (d_F - 1)^{\sum_{i=1}^{\ell} d_V(d_V-1)^{i-1}}. \quad (20)$$

□

The number of deviations on the computation tree of a $(3,6)$-regular low-density parity-check code is given in Table 2 for iterations 1 through 5. Even after only a small number of iterations, it becomes impractical to enumerate each of the deviations in order to compute the upper bound on the probability of bit error of the root variable node of the computation tree.

Using computation trees, Wiberg provided a precise model of the behavior of the min-sum and sum-product decoders. Unfortunately, the size of the computation trees and the number of configurations on them grows too large for practical analysis. Deviations provide a simplified approach to the analysis of computation trees, but the number of deviations also grows exponentially with the number of iterations.

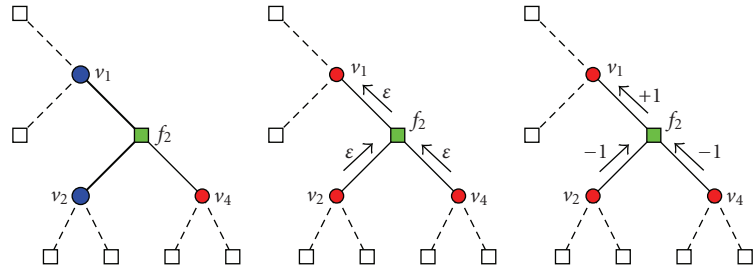
3. Stopping Sets, Absorbing Sets, and Resulting Deviations

Deviations can be used to define a necessary condition for an error to occur during iterative decoding. The condition simply states that there must be at least one deviation with cost less than zero, assuming that the all-zeros codeword was sent. However, this condition says nothing about which deviations are more or less likely to cause errors. What is known is that at high SNRs low-weight, deviations are much more likely to cause errors than high-weight deviations. Thus it is reasonable to expect that low-weight stopping sets and low-weight deviations coincide over the BEC channel, since low-weight stopping sets are precisely the cause of errors for iterative decoding over the BEC [15]. For the same reason, one can expect that trapping sets, or more specifically absorbing sets, coincide with low-weight deviations over the AWGN channel, since they have been frequently observed to cause errors at high SNR during iterative decoding of LDPC codes over the AWGN channel [8, 17]. Connections between stopping/absorbing sets and deviations and their effect on decoding performance are examined in this section.

3.1. Stopping Sets as Deviations. Stopping sets consist of a subset $\mathcal{S}$ of the variable nodes, such that each check node connected to an element in $\mathcal{S}$ is connected at least twice. According to the definition of a deviation given in Definition 4, each check node connected to the variable nodes in a deviation is connected exactly two times. These two properties can be used to study the relationship between stopping sets and their corresponding deviations.

First, consider a computation tree where each of the variable nodes begins with an assignment of a binary 0. Then, assign all copies of variable nodes in $\mathcal{S}$ to a binary 1. If $\mathcal{S}$ does not correspond with a codeword in C, the resulting configuration on the computation tree will not be a valid configuration. For example, consider check node f_2 in Figure 2. Each time, check node f_2 appears in the computation tree, it will be connected to three variable nodes assigned to a binary 1, including the parent variable node and two child variable nodes. If one of the child variable nodes of f_2 along with all of its descendants is set to a binary 0, check node f_2 will be satisfied. If this is done for every unsatisfied check node in the computation tree, a deviation is created that contains only variable nodes in $\mathcal{S}$. Thus, this method allows one to create a deviation from a stopping set.

Using the method previously described, a deviation can be constructed using only the variable nodes contained in a stopping set. This is illustrated in Figure 7(a), where a portion of the deviation defined by the set $\mathcal{S}$ from Figure 2 is given. Since only one of the child variable nodes can be included in the deviation, it is sufficient to randomly include v_2 and exclude v_4, since both nodes are included in $\mathcal{S}$. The effect of this deviation is now examined over the BEC and



(a) Deviation (b) Costs on the BEC (c) Costs on the BSC

FIGURE 7: A small portion of a deviation on a computation tree illustrating the impact of stopping set deviations on the BEC and BSC channels.

the BSC. Stopping sets are known to cause errors over the BEC. The reason for this is illustrated by Figure 7(b). When each variable node in $\mathcal{S}$ is received as an erasure ε, it only takes one erasure below f_2 to cause an erasure message to be sent from f_2 to v_1. However, even though two erasures are connected below f_2 in Figure 7(b), the same erasure message is sent up to v_1. This example illustrates that a deviation containing all erasures is sufficient for an erasure output at the root node. Thus there is a strong connection between stopping sets and deviations over the BEC, since both cause errors using iterative decoding and deviations can be created from stopping sets.

The reason stopping sets will not cause errors as frequently over the BSC is illustrated in Figure 7(c). A deviation containing only the variable nodes in a stopping set exists on a computation tree rooted at any one of the variable nodes in $\mathcal{S}$ regardless of the channel. However, the impact of deviations is not the same across all channels. For example, if two messages of -1 (representing binary 1s) are being sent from v_2 and v_4, the message sent from f_2 up to v_1 is a $+1$ (representing a binary 0). This example shows that while a deviation created from a stopping set is sufficient to cause erasure outputs from the decoder over the BEC, this same deviation may not cause errors over the BSC.

From Figure 7, it is clear that deviations created from stopping sets have a different effect on iterative decoding over the binary erasure channel and the binary symmetric channel. While it is not as clear how deviations created from stopping sets will effect decoding over the AWGN channel, some similarities can be drawn between the BEC and the AWGN. In terms of channel LLR costs, an erasure over the BEC behaves like a LLR cost of zero over the AWGN. A real-valued LLR interpretation of the BEC channel can be created using real-valued costs of $-1.0, +1.0$, and 0.0 to represent a binary 1, 0, and an erasure ε, respectively. Binary information is transmitted as $x = -1.0$ and $x = +1.0$ over the AWGN channel, and the probability $P(y = +1.0 \mid x = -1.0) = P(y = -1.0 \mid x = +1.0) < P(y = 0.0 \mid x = -1.0) = P(y = 0.0 \mid x = +1.0)$, regardless of the channel SNR. Thus, it is reasonable to suspect that the AWGN channel behaves more like the BEC than the BSC, especially at high channel SNR.

3.2. Absorbing Sets as Deviations. Absorbing sets project to deviations on the computation tree in a different way than stopping sets. Since each check node connected to a stopping set $\mathcal{S}$ is connected at least twice, a deviation can easily be defined using only nodes in $\mathcal{S}$ on any computation tree rooted at a node in $\mathcal{S}$. Unlike stopping sets, absorbing sets can have only a single connection to a check node. When an absorbing set has a single connection to a check node, it is not possible to form a deviation on any computation tree using only the variable nodes in $\mathcal{A}$, unless there is a stopping set $\mathcal{S} \subseteq \mathcal{A}$. If $\mathcal{S} \subseteq \mathcal{A}$, a deviation can be formed on the computation tree by simply avoiding variable nodes in $\mathcal{A}$ that are not in $\mathcal{S}$.

For an absorbing set $\mathcal{A}$ that does not contain a stopping set, it is of interest to know how the absorbing set manifests itself as a deviation on the computation tree rooted at one of the variable nodes in $\mathcal{A}$. This manifestation takes the form of a deviation with as many variable nodes in $\mathcal{A}$ as possible. Because it is known that each variable node is connected to strictly more satisfied check nodes than unsatisfied check nodes, it is possible to compute a bound on the number of variable nodes in a deviation δ that are contained in $\mathcal{A}$ for regular LDPC codes.

Consider an absorbing set $\mathcal{A}$ on the Tanner graph of a (d_V, d_F)-regular low-density parity-check code. Let each variable node $v_i \in \mathcal{A}$ be connected to at least $d_{\mathcal{A}} > d_V/2$ satisfied check nodes. A computation tree rooted at a variable node $v_i \in \mathcal{A}$ after ℓ iterations is given by $R^{\ell}_{v_i}$. A deviation on this computation tree can be constructed by selecting the nodes in the deviation level-by-level. The deviation construction begins by including the root node v_i at level $\ell = 0$. At the next level, one variable node connected to each of the check nodes in $N(v_i)$ must be included in δ. When possible, variable nodes in $\mathcal{A}$ will always be included in δ. Therefore, after $\ell = 1$ there are at least $1 + d_{\mathcal{A}}$ variable nodes in δ that are also in $\mathcal{A}$, and at most $d_V - d_{\mathcal{A}}$ variable nodes in δ that are not in $\mathcal{A}$. Continuing to level $\ell = 2$ in the computation tree, each of the $d_{\mathcal{A}}$ variable nodes in $\mathcal{A}$ at level $\ell = 1$ in δ connects to $d_{\mathcal{A}} - 1$ new variable nodes in $\mathcal{A}$ and $d_V - d_{\mathcal{A}}$ new variable nodes not in $\mathcal{A}$. Each of the $d_V - d_{\mathcal{A}}$ variable nodes at level $\ell = 1$ in δ that are not in $\mathcal{A}$ connects to $d_V - 1$ variable nodes that are not it $\mathcal{A}$. After $\ell = 2$, the number of variable nodes in δ that are also in $\mathcal{A}$ is

$$|\delta_{\mathcal{A}}| \geq 1 + d_{\mathcal{A}} \sum_{i=1}^{\ell} (d_{\mathcal{A}} - 1)^{i-1}. \tag{21}$$

Similarly, the total number of variable nodes in δ is

$$|\delta| = 1 + d_V \sum_{i=1}^{\ell} (d_V - 1)^{i-1}. \tag{22}$$

Therefore, the number of variable nodes in δ that are not in $\mathcal{A}$ is $|\delta| - |\delta_{\mathcal{A}}|$.

It is clear from (21) and (22) that the lower bound on the portion of variable nodes in $\mathcal{A}$ within the deviation δ given by $|\delta_{\mathcal{A}}|/|\delta|$ approaches zero as ℓ approaches infinity. Thus, the bound does not appear to be an accurate method for calculating the true portion. The bound given by (21) is computed under the worst-case scenario that, after a deviation reaches a check node with only one connection to the set $\mathcal{A}$, the descendants of that check node contained in the deviation do not contain any more variable nodes in $\mathcal{A}$. On a connected Tanner graph with no nodes of degree one and $d_{\mathcal{A}} \neq d_V$, this assumption is most likely never true, since nodes in $\mathcal{A}$ will eventually (with increasing ℓ) be included in the descendants of the failed check node, and consequently the variable nodes will also be included in any deviation which is a manifestation of $\mathcal{A}$.

For finite-length, (d_V, d_F)-regular low-density parity check codes, it is possible to determine the exact number of variable nodes $|\delta_{\mathcal{A}}|$ after a given number of iterations ℓ. In order to find $|\delta_{\mathcal{A}}|$, each variable node $v_i \in \mathcal{A}$ is assigned a LLR cost of $\lambda_i = 0.0$ and each variable node $v_j \in V \setminus \mathcal{A}$ is assigned a LLR cost of $\lambda_j = +1.0$. Then, using the resulting LLR cost vector, MS decoding is performed for ℓ iterations. The final cost m_i for any variable node $v_i \in \mathcal{A}$, is the number of nodes in the minimum-cost deviation on the computation tree rooted at v_i after ℓ iterations. Deviations for (d_V, d_F)-regular LDPC codes contain a fixed number of variable nodes determined by (22), and the only way to reduce the cost of the deviation is to include variable nodes from the set $\mathcal{A}$. Thus, the minimum-cost deviation on the computation tree of a (d_V, d_F)-regular LDPC code will include the maximum number of variable nodes in $\mathcal{A}$ that is possible, and the MS decoder will output the cost of this deviation. It is important to note that the cost of the deviation returned by the MS decoder corresponds to the number of variable nodes in the deviation that are not in $\mathcal{A}$. Therefore, in order to determine the number of variable nodes in the deviation that are in $\mathcal{A}$, it is necessary to subtract this MS decoder cost from the result given by (22).

Example 1. Consider the length $N = 20$, dimension $K = 10$, $(3, 6)$-regular low-density parity-check code defined by the parity check matrix

$$H = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0
\end{bmatrix}. \tag{23}$$

A $(3, 1)$ fully absorbing set is defined by the set of variable nodes $\mathcal{A}_1 = \{v_3, v_7, v_{16}\}$, where columns 1 through 20 of H corresponds to the set of variable nodes $\{v_0, v_1, \ldots, v_{19}\}$. After 50 iterations, each of the deviations on each of the N computation trees contains $|\delta| \approx 3.378 \times 10^{15}$ variable nodes, as computed by (22). After setting the LLR costs for variable nodes in $\mathcal{A}_1$ to $\lambda = 0.0$, and all other LLR cost to $\lambda = +1.0$, the output of the MS decoder after 50 iterations for variable node v_7 is a cost of $m_{v_7} = 3.044 \times 10^{14}$. Therefore, the minimum-weight deviation contains $3.073 \times 10^{15} = 3.378 \times 10^{15} - 3.044 \times 10^{14}$ variable nodes also contained in the absorbing set $\mathcal{A}_1$. Thus, a deviation can be formed on the computation tree with 91% of its variable nodes coming from $\mathcal{A}_1$. In contrast, the bound given by (21) for $(3, 6)$-regular LDPC codes after 50 iterations with $d_{\mathcal{A}} = 2$ only guarantees that 101 variable nodes from $\mathcal{A}_1$ will be included in the minimum weight deviation. This huge disparity between the bound and the actual result given by MS decoding reveals the effects of the assumptions used to derive (21). It is worth noting that the proportion of variable nodes in $\mathcal{A}_1$ in the minimum weight deviation after 51 and 52 iterations were also 91%, so this proportion appears to stabilize after a sufficient number of iterations.

Using this same method on the $(3, 3)$ fully absorbing set $\mathcal{A}_2 = \{v_0, v_7, v_{18}\}$, it is found that the minimum weight deviation after 50 iterations contains 2.696×10^{15} copies of the variable nodes in $\mathcal{A}_2$, equivalent to 78% of the total number of variable nodes in the deviation. By comparing the properties of the minimum-weight deviations resulting from $\mathcal{A}_1$ and $\mathcal{A}_2$, one might expect that $\mathcal{A}_1$ is more likely to cause errors than $\mathcal{A}_2$. This is because 91% of the cost of the deviation resulting from $\mathcal{A}_1$ is determined by the variable nodes in $\mathcal{A}_1$, compared to only 78% for $\mathcal{A}_2$. Simulation results in Section 4 show that $\mathcal{A}_1$ causes errors much more frequently than $\mathcal{A}_2$.

for $i_{\text{fixed}} = 1, \ldots, N$
  Set $m_{\min} = \infty$.
  Set $\chi = \{i_{\text{fixed}}\}$.
  while $m_{\min} > 0.0$
    for $i = 1, \ldots, N$
    -Set $\chi = \chi \bigcup \{i\}$.
    -Set $\lambda_k = 0.0$ for all $k \in \chi$.
    -Set $\lambda_k = 1.0$ for all $k \in V \setminus \chi$.
    -Perform MS Decoding for ℓ iterations.
      if $\min_{j=1,\ldots,N} m_{v_j} < m_{\min}$
        -Set $m_{\min} = \min_{j=1,\ldots,N} m_{v_j}$.
        -Set $j_{\min} = \arg\min_{j=1,\ldots,N} m_{v_j}$.
      end
      -Set $\chi = \chi \bigcap (\{V \setminus \{i\}\} \bigcup \{i_{\text{fixed}}\})$.
      end
    -Set $\chi = \chi \bigcup \{j_{\min}\}$.
    -Create a binary vector $\mathbf{v}$ with $v_k = 1$ if $k \in \chi$, and $v_k = 0$ if $k \in V \setminus \chi$.
    -Compute the integar syndrome $\mathbf{s}_{\text{int}} = H\mathbf{v}^T$.
    -Compute the binary syndrome $\mathbf{s}_{\text{bin}} = H\mathbf{v}^T$ with Hamming weight w_s.
    -Compute the integar vector $\mathbf{z} = H^T \mathbf{s}_{\text{bin}}$
    if $\min_{k=1,\ldots,M} s_{\text{int},k} \geq 2$
      χ is a $(|\chi|, w_s)$ Stopping Set.
    end
    χ is a $(|\chi|, w_s)$ Trapping Set.
    if $z_k < \left\lfloor \frac{d_{v_k}}{2} \right\rfloor$ for all $k \in \chi$
      χ is a $(|\chi|, w_s)$ Absorbing Set.
    end
    if $z_k < \left\lfloor \frac{d_{v_k}}{2} \right\rfloor$ for all $k = 1, \ldots, N$
      χ is a $(|\chi|, w_s)$ Fully Absorbing Set.
    end
  end
end

Algorithm 1: Iterative problematic trapping set finder.

4. Finding Problematic Trapping Sets

Any set of nodes can be interpreted as a trapping set, including stopping sets, absorbing sets, and fully absorbing sets. This is because trapping sets are only defined by the number of variable nodes in the set and the number of failed check nodes. In order to simplify analysis, the trapping sets studied in this paper are restricted to the study of problematic trapping sets.

Definition 5 (Problematic Trapping Set). A *problematic trapping set* is a trapping set such that the number of failed check nodes connected to the trapping set is less than or equal to the number variable nodes contained in the trapping set.

Because trapping sets with small weight and a small number of failed check nodes are often the cause of errors at high SNR [17], it is unlikely that the restriction to problematic trapping sets will eliminate error patterns of interest. In Section 3, it was shown that MS decoding can be used to determine the proportion of variable nodes that are both inside and outside an absorbing set. The same idea is used in this section to find problematic trapping sets using MS decoding. The iterative method given by Algorithm 1 operates by forcing the trapping set to contain one variable node, and then adding more variable nodes one-by-one that decrease the cost of the minimum-cost deviation the most.

Once Algorithm 1 reaches cost $m_{\min} = 0.0$, it is possible to discover more problematic trapping sets by removing nodes one-by-one from the set $\mathcal{X}$ that result in the smallest increase in the cost $m_{\min}$. In order to examine the efficacy of Algorithm 1, the length $N = 20$, dimension $K = 10$, $(3, 6)$-regular low-density parity-check code given in Example 1 was used. This code was chosen because the Hamming weight of all of its codewords could be enumerated, and thus the minimum distance of the code could be easily computed.

Table 3: Stopping/trapping/absorbing sets of a length $N = 20$, dimension $K = 10$, $(3,6)$-regular LDPC code with weight of 3 or less found using Algorithm 1, and the number of times they were observed after 1000 iterations of SP decoding at SNR = 8.0 dB.

Set	Size	Dev. %	Stop.	Abs.	Full Abs.	Observed
$\{v_0, v_{17}\}$	$(2,2)$	74%		X		16
$\{v_1, v_{19}\}$	$(2,2)$	73%		X		7
$\{v_2, v_{14}\}$	$(2,2)$	71%		X		4
$\{v_3, v_{16}\}$	$(2,2)$	72%		X		7
$\{v_4, v_{17}\}$	$(2,2)$	72%		X		28
$\{v_6, v_{15}\}$	$(2,2)$	76%		X		0
$\{v_7, v_{17}\}$	$(2,2)$	73%		X		6
$\{v_8, v_{11}\}$	$(2,2)$	75%		X		21
$\{v_9, v_{14}\}$	$(2,2)$	73%		X		19
$\{v_{10}, v_{18}\}$	$(2,2)$	73%		X		4
$\{v_{11}, v_{19}\}$	$(2,2)$	76%		X		28
$\{v_6, v_{12}\}$	$(2,2)$	76%		X		0
$\{v_6, v_{12}, v_{15}\}$	$(3,1)$	100%	X	X	X	908
$\{v_0, v_{10}, v_{17}\}$	$(3,1)$	91%		X	X	9
$\{v_1, v_4, v_{19}\}$	$(3,1)$	90%		X	X	20
$\{v_2, v_5, v_{14}\}$	$(3,1)$	89%		X	X	6
$\{v_3, v_7, v_{16}\}$	$(3,1)$	91%		X	X	26
$\{v_4, v_{14}, v_{17}\}$	$(3,1)$	90%		X	X	35
$\{v_6, v_7, v_{17}\}$	$(3,1)$	92%		X	X	0
$\{v_8, v_{11}, v_{18}\}$	$(3,1)$	91%		X	X	28
$\{v_9, v_{14}, v_{19}\}$	$(3,1)$	90%		X	X	20
$\{v_7, v_{10}, v_{18}\}$	$(3,1)$	91%		X	X	16
$\{v_{11}, v_{15}, v_{19}\}$	$(3,1)$	91%		X	X	17
$\{v_0, v_7, v_{18}\}$	$(3,3)$	80%		X		0
$\{v_1, v_{14}, v_{19}\}$	$(3,3)$	64%				0
$\{v_2, v_5, v_{19}\}$	$(3,3)$	79%		X		0
$\{v_0, v_4, v_{14}\}$	$(3,3)$	79%		X		0
$\{v_5, v_6, v_{15}\}$	$(3,3)$	86%				0
$\{v_7, v_{12}, v_{15}\}$	$(3,3)$	86%				0
$\{v_2, v_9, v_{12}\}$	$(3,3)$	78%		X		0
$\{v_6, v_{11}, v_{12}\}$	$(3,3)$	85%				0
$\{v_{12}, v_{13}, v_{15}\}$	$(3,3)$	85%				0
$\{v_{13}, v_{14}, v_{19}\}$	$(3,3)$	57%		X		0
$\{v_7, v_{17}, v_{18}\}$	$(3,3)$	85%				0
Other						275

Applying Alogorithm 1 to this code resulted in a total of 37 trapping sets with parameters shown in Table 3. Note that the proportion of nodes in each set that are included in the minimum weight deviation is given by "Dev. %". This code was found to have minimum distance equal to 4, so only stopping/trapping/absorbing sets of weight less than or equal to 3 are tabulated.

In order to determine how effective Algorithm 1 is at locating problematic trapping sets, the same length $N = 20$, dimension $K = 10$, $(3,6)$-regular low-density parity-check code was simulated using sum-product decoding over the additive white Gaussian noise channel with an SNR of $E_b/N_0 = 8.0$ dB. A total of 1500 noncodeword outputs with weight less than or equal to 3 were observed during SP decoding after 1000 iterations. It is important to note that the noncodewords outputs were not simply the last quantized output given after 1000 iterations. The output of SP decoding typically changes after each iteration when it does not converge to a codeword. For this reason, the noncodeword outputs were computed by averaging the cost m_{v_i} for each variable node v_i from $i = 1, \ldots, N$ over the last 200 iterations to compute a final output cost. This is similar to the method used in [14] for characterizing the changing outputs of the MS decoder.

Table 3 shows the number of observed output errors, and compares them to the problematic trapping sets found using Algorithm 1. Approximately 82% of the observed errors corresponded to one of the problematic trapping sets found using Algorithm 1. Also, the number of times a particular problematic trapping set is observed indicates how

problematic the set is to the SP decoder. The single most problematic set, resulting in over 60% of the output errors, was the $(3, 1)$ set that satisfies the definitions of a stopping set, absorbing set, and fully absorbing set. Errors falling into the "Other" category were highly variable, and no specific output pattern in this set accounted for more than 6 of the total observed errors.

The problematic trapping sets with the highest proportion of nodes within their corresponding deviation were the $(3, 1)$ trapping sets. Not surprisingly, the $(3, 1)$ stopping set has the highest proportion of variable nodes in its deviation. While the proportions were noticeably different between different-sized sets, the difference was minimal within sets of the same size. Furthermore, it is difficult to make any connections between proportions within sets of the same size and their corresponding probability of causing an error. One possible reason for this might be the overlap between the different problematic trapping sets, and between the problematic trapping sets and codewords. For example, the reason that the $(2, 2)$ fully absorbing set $\{v_6, v_{12}\}$ did not appear in the simulations might be because two of its variable nodes overlap with the exceptionally problematic $(3, 1)$ stopping set, and thus any significant channel noise received by variable nodes v_6 and v_{12} may be highly likely to cause the $(3, 1)$ stopping set to be output by the decoder.

It is worth noting that the average value of received information within the absorbing sets of weight less than or equal to three was $y = 0.102115$. Recall that a binary 0 is modulated to $x = -1.0$, so the mean value of the noise within the absorbing sets was +1.102115. This cost further justifies the earlier assertion that the AWGN channel behaves more like the BEC channel at high SNR than the BSC channel, since an erasure over the BEC can be interpreted as noise of +1.0 and a bit flip over the BSC can be interpreted as noise of +2.0.

Algorithm 1 was able to find 82% of the most problematic errors with weight less than $d_{\min}$. In order to test the algorithm on an LDPC code with longer block length, a length $N = 200$, $K = 100$, $(3, 6)$-regular LDPC code was used. The resulting output of Algorithm 1 was 1019 absorbing sets, 941 fully absorbing sets, and 1 stopping set, and the sizes of the sets ranged from 3 to 9. Only sets containing less than 10 variable nodes were considered problematic, since the code is known to contain a codeword of weight 10. Simulations were performed using SP decoding at SNR = 5.0 dB, at which the bit error rate of the code is $P_b = 4.8 \times 10^{-9}$. Overall, 40 noncodeword errors were observed, of which 20 were error patterns of weight less than 10. Of these, all were absorbing sets and 19 were fully absorbing sets. Unlike the results given for the length $N = 20$ code, only two of the 20 absorbing sets was found by Algorithm 1. However, the two that were found were the two smallest absorbing sets, including $(5, 3)$ and $(6, 4)$ fully absorbing sets.

As expected, the number of trapping sets grows very large when increasing the size and dimension of the code. Algorithm 1 is capable of locating the majority of problematic trapping sets for a small length $N = 20$ LDPC code, but for the larger length $N = 200$ LDPC code it was only able to identify the two smallest sets observed during simulations. This is likely due to the fact that the code had not yet reached its error floor, as evidenced by the fact that half of the observed error patterns had weight greater than or equal to the weight of a known codeword. Because error floors occur at such low bit error rates for large LDPC codes, it is difficult to observe problematic trapping sets using simulations. Thus, the effectiveness of Algorithm 1 at identifying problematic trapping sets remains unknown for large codes with error floors beyond the reach of simulations.

5. The Weight of Deviations Induced by Problematic Trapping Sets

In [17], Richardson characterizes trapping sets by their size and the number of associated failed check nodes. To find the impact of the trapping sets with respect to probability of error, Richardson uses simulations that force the noise in the trapping set and push the received information away from modulated 0s and towards modulated 1s. The result is an estimate of the probability of error caused by trapping sets at high SNR. In this section, a new method is used to examine the probability of error associated with trapping sets. Instead of using simulations to estimate the probability of error, deviations induced by the trapping set are created to analyze the probability of error. Since bounds on the probability of error can be derived from deviations, if one could prove that minimum-weight deviations were induced by problematic trappings sets and then computed the weights of the deviations, it may be possible determine the probability of error associated with trapping sets without having to rely on simulations.

It may seem surprising that almost all problematic trapping sets listed in Table 3 result in a deviation where the variable nodes within the trappings set make up the majority of the variable nodes within that deviation. For example, consider the fully absorbing set given by the nodes $\{v_{11}, v_{19}\}$. While there are only two variable nodes contained in the absorbing set, they make up 76% of the variable nodes in a deviation that exists on the computation tree rooted at either variable node within the set. This implies that there might be a way of cleverly designing deviations which contain a disproportionately large number of certain variable nodes. A deviation rooted at variable node v_{19} after 4 iterations is shown in Figure 8. This deviation was designed to include more copies of variable nodes v_{11} and v_{19} than other variable nodes. Overall, the number of copies of each variable node in the deviation is $\#v_{11} = 16$, $\#v_{19} = 13$, $\#v_6 = 9$, $\#v_{12} = 5$, and $\#v_{15} = 3$. Although the overall configuration contains 5 different variable nodes, almost 2/3 of them are copies of v_{11} or v_{19}. Now, consider the subgraph of the Tanner graph defined by these 5 variable nodes, shown in Figure 9. This subgraph defines a $(5, 1)$ stopping set. It was shown in Section 3.1 that, because it is a stopping set, the deviation in Figure 8 could continue to grow indefinitely without the need for variable nodes outside the stopping set.

To construct the deviation in Figure 8, decisions were made at check nodes f_0 and f_6. Those decisions are expressed in the directed bipartite graph shown in Figure 10. The number of decisions for each check node is equivalent to the

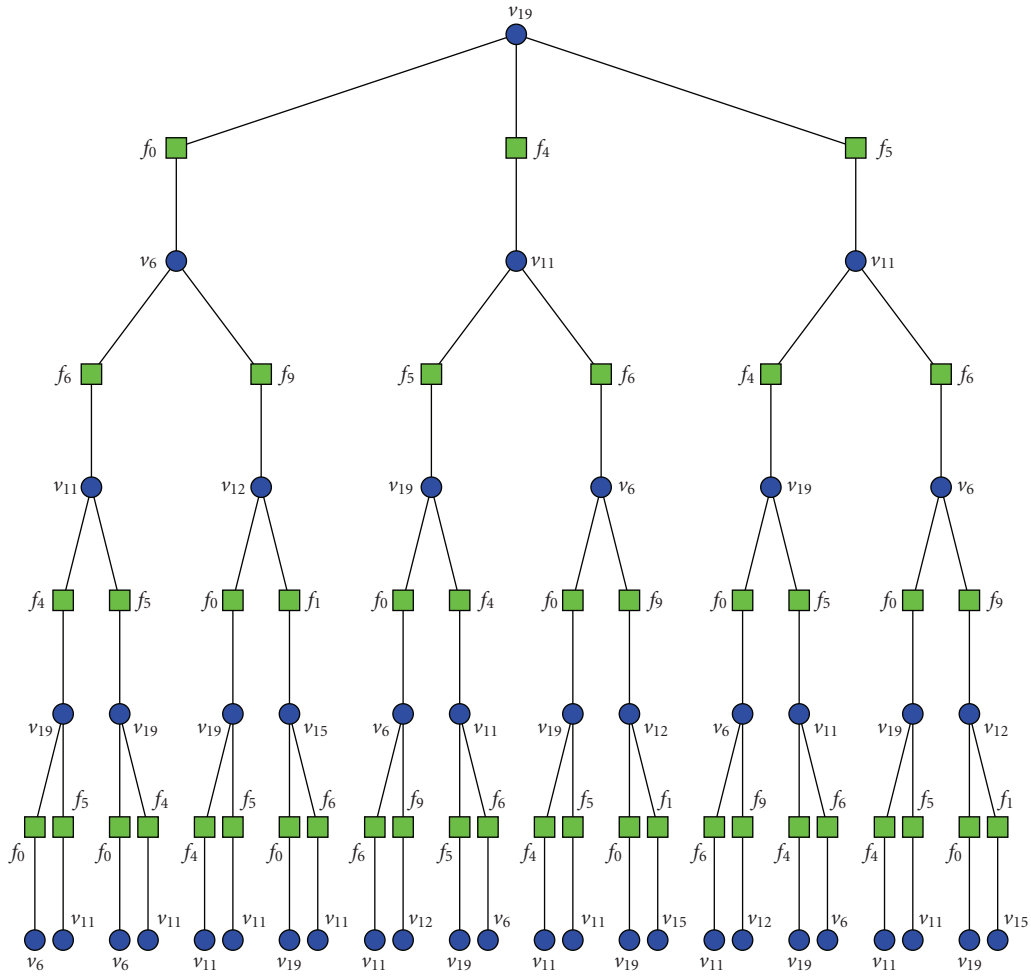


Figure 8: Deviation designed to maximize the number of copies of v_{11} and v_{19}.

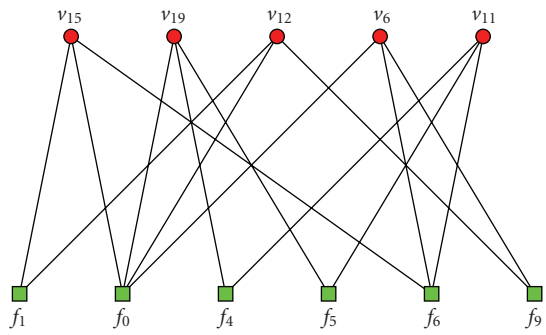


Figure 9: Subgraph of the Tanner graph.

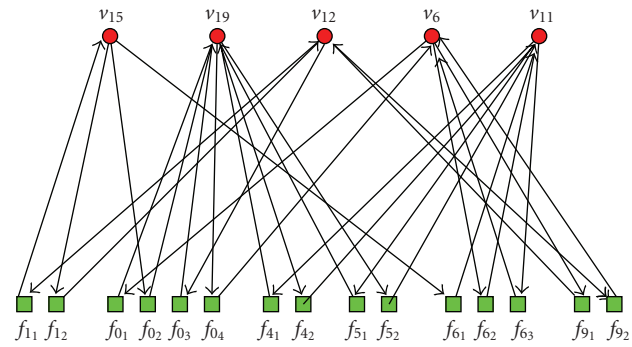


Figure 10: Subgraph of the Tanner graph with repeated check nodes and directed edges.

number of edges incident to the check node. Check nodes f_1, f_4, f_5, and f_9 each have two copies in the directed graph to preserve the fact that they have bidirectional edges. However, check nodes f_0 and f_6 do not simply have bidirectional edges connecting them to each of their incident variable nodes, as demonstrated by the deviation in Figure 8. This comes from the fact that check nodes with degree higher than 2 are still only connected to 2 variable nodes within the deviation. Thus, at each check node with degree greater than 2, a decision is made as to which variable nodes it includes in the deviation.

Using the adjacency matrix of the directed bipartite graph, it is possible to compute the number of copies, or the multiplicity, of each node at each level in the deviation.

Table 4: The multiplicity of variable nodes contained in the deviation created using the directed graph given in Figure 10.

i	v_6	v_{11}	v_{12}	v_{15}	v_{19}
0	0	0	0	0	1
2	1	2	0	0	0
4	2	1	1	0	2
6	2	2	2	1	5
8	4	11	2	2	5
20	326	468	158	78	506
60	3.4×10^8	4.9×10^8	1.7×10^8	8.5×10^7	5.3×10^8
100	3.6×10^{14}	5.2×10^{14}	1.8×10^{14}	8.9×10^{13}	5.5×10^{14}
Total	7.1×10^{14}	1.0×10^{15}	3.6×10^{14}	1.8×10^{14}	1.1×10^{15}

The method for computing the multiplicity of nodes in the deviation is similar to the method given in [18] for computing the multiplicity of nodes on computation trees. For the directed Tanner graph given in Figure 10, the adjacency matrix is given by

$$A = \begin{bmatrix} & & 0&0&0&1&0&0&0&0&0&0&0&0&1&0&1 \\ & & 0&0&0&0&0&0&0&1&0&1&1&1&0&0&0 \\ & \mathbf{0} & 0&0&0&0&0&1&0&0&0&0&0&0&0&1&0 \\ & & 0&0&0&0&1&0&0&0&0&0&0&0&0&0&0 \\ & & 1&1&1&0&0&0&1&0&1&0&0&0&0&0&0 \\ 1&0&0&0&0 & & & & & & & & & & \\ 0&0&0&1&0 & & & & & & & & & & \\ 0&0&1&0&0 & & & & & & & & & & \\ 0&0&0&0&1 & & & & & & & & & & \\ 0&0&1&0&0 & & & & & & & & & & \\ 0&0&0&1&0 & & & & & & & & & & \\ 0&1&0&0&0 & & & & & & & & & & \\ 0&0&0&0&1 & & & \mathbf{0} & & & & & & & \\ 0&1&0&0&0 & & & & & & & & & & \\ 0&0&0&0&1 & & & & & & & & & & \\ 0&0&0&1&0 & & & & & & & & & & \\ 1&0&0&0&0 & & & & & & & & & & \\ 0&1&0&0&0 & & & & & & & & & & \\ 1&0&0&0&0 & & & & & & & & & & \\ 0&0&1&0&0 & & & & & & & & & & \end{bmatrix}. \tag{24}$$

The columns and rows of A correspond, in the same order, to the nodes v_6, v_{11}, v_{12}, v_{15}, v_{19}, f_{0_1}, f_{0_2}, f_{0_3}, f_{0_4}, f_{1_1}, f_{1_2}, f_{4_1}, f_{4_2}, f_{5_1}, f_{5_2}, f_{6_1}, f_{6_2}, f_{6_3}, f_{9_1}, and f_{9_2}. Beginning with a vector $\mathbf{m}_0$ which has a 1 in the position of the root node and 0s elsewhere, the number of nodes on each level i of the deviation can be calculated recursively using

$$\mathbf{m}_i = \begin{cases} A\mathbf{m}_{i-1}, & \text{if } i \text{ is even}, \\ A\mathbf{m}_{i-1} - B\mathbf{m}_{i-2}, & \text{if } i \text{ is odd}, \end{cases} \tag{25}$$

where B is used to subtract the parent check nodes at each check node level in the deviation. The matrix B for the directed Tanner graph in Figure 10 is given by

$$B = \begin{bmatrix} & & & & & & & & & & & & & & \\ & \mathbf{0} & & & & & & & & & & & & & \\ & & & & & & & & & & & & & & \\ & & & & & & & & & & & & & & \\ & & 0&0&0&1 & & & & & & & & & \\ & & 0&0&0&0 & & & & & & & & & \\ & & 0&0&0&0 & & & & & & & & & \\ & & 1&1&1&0 & & & & & & & & & \\ & & & & 0&1 & & & & & & & & & \\ & & & & 1&0 & & & & & & & & & \\ & & & & & & 0&1 & & & & & & & \\ & & & & & & 1&0 & & & & & & & \\ & & & & & & & & 0&1 & & & & & \\ & & & & & & & & 1&0 & & & & & \\ & & & & & & & & & & 0&0&0 & & \\ & & & & & & & & & & 0&0&1 & & \\ & & & & & & & & & & 1&1&0 & & \\ & & & & & & & & & & & & & 0&1 \\ & & & & & & & & & & & & & 1&0 \end{bmatrix}. \tag{26}$$

Using the recursive formula given by (25), the number of variable nodes at different levels in the computation tree is given in Table 4. The deviation at level $i = 100$ exists on the computation tree rooted at variable node v_{19} after $\ell = 50$ iterations. This deviation contains a total of 3.38×10^{15} variable nodes, over 63% of which are copies of v_{11} and v_{19}. Using a similar method to that given in [19], the effective weight of the deviation can be calculated. First, let a_{v_i} be the number of copies of variable node v_i within the deviation, and let b be the total number of variable nodes in the deviation. The cost of the deviation is modeled by the normal distribution

$$\mathcal{N}\left(\sum_{v_i \in \mathcal{S}} \frac{a_{v_i}}{b}, \sum_{v_i \in \mathcal{S}} \left(\frac{a_{v_i}}{b}\sigma\right)^2\right), \tag{27}$$

where σ^2 is the variance of the AWGN noise. This random variable can be rescaled resulting in the distribution

$$\mathcal{N}\left(\left(\frac{\sum_{v_i \in \mathcal{S}} a_{v_i}}{\sqrt{\sum_{v_i \in \mathcal{S}} \left(a_{v_i}\right)^2}}\right)^2, \left(\frac{\sum_{v_i \in \mathcal{S}} a_{v_i}}{\sqrt{\sum_{v_i \in \mathcal{S}} \left(a_{v_i}\right)^2}}\right)^2 \sigma^2\right), \tag{28}$$

where the mean

$$\left(\frac{\sum_{v_i \in \mathcal{S}} a_{v_i}}{\sqrt{\sum_{v_i \in \mathcal{S}} \left(a_{v_i}\right)^2}}\right)^2, \tag{29}$$

is the weight of the deviation. Note that if all a_{v_i} were equal, the weight of the set $\mathcal{S}$ would be equal to the number of nodes it contains, which is consistent with the notion of Hamming weight when $\mathcal{S}$ is equal to a codeword. From Table 4, the weight of the deviation created from the directed Tanner graph in Figure 10 after $\ell = 50$ iterations is 3.8784. This is less than the Hamming weight of the minimum distance codeword in the code, which has weight 4.0. Thus, the minimum weight of deviations on the computation tree rooted at v_{19} is probably less than the minimum distance of the code.

6. Conclusion

Practical methods for predicting and understanding the performance of low-density parity-check codes with iterative decoders are needed in order to avoid the use of codes with error floors. Trapping sets, which include absorbing sets and stopping sets, provide insight into the error mechanisms of iterative decoders but are too imprecise to be used to make design decisions with respect to error floors. Deviations on computation trees are precise and can be used to compute strict upper bounds on the performance of MS and SP decoding, but computing these bounds quickly becomes computationally intractable. The paper examined the connections between trapping sets and their corresponding deviations through the notion of problematic trapping sets in an attempt to find a practical and precise method for predicting the performance of LDPC codes with iterative decoding.

It was shown that the variable nodes in a stopping set can be used to define a deviation, while trapping sets and absorbing sets only define a deviation if a subset of their variable nodes forms a stopping set. When trapping sets and absorbing sets do not include a stopping set, it is necessary to include additional variable nodes in order to construct a corresponding deviation. The number and proportion of variable nodes outside the set that are needed to construct the deviation can be found experimentally using a modification of the MS decoder. This modified MS algorithm leads to an iterative method for identifying low-weight problematic trapping sets in an LDPC code. Simulation results demonstrate that this method is capable of finding many of the low-weight trapping sets that determine the performance of LDPC codes at moderate SNRs. The efficacy of this algorithm is limited by computational constraints.

Finally, an analytical approach for determining the weight of deviations induced by trapping sets on the computation tree was introduced. This approach involves determining the minimum-weight stopping set that contains a given trapping set, and then determining a directed Tanner graph from the stopping set that favors certain variable nodes within the trapping set. It was then shown that the effective weight of the deviation can be found using a recursive method for computing the multiplicity of variable nodes within the deviation, In one example, it was proven that a deviation exists on the computation tree with weight less than the Hamming weight of the code. This result shows that trapping sets probably result in a necessary condition for an error to occur during iterative decoding, and in certain cases, this condition is satisfied with probability higher than the probability of an ML codeword error.

Acknowledgments

This paper was funded in part by AFOSR Contract FA9550-06-1-0375 and Department of Education Grant no. P200A070344.

References

[1] I. Reed and G. Solomon, "Polynomial codes over certain finite fields," *SIAM Journal on Applied Mathematics*, vol. 8, pp. 300–304, 1960.

[2] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. 13, pp. 260–269, 1967.

[3] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit errorcorrecting coding and decoding," in *Proceedings of the IEEE International Conference on Communications*, pp. 1064–1070, Geneva, Switzerland, 1993.

[4] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes," *Electronics Letters*, vol. 32, no. 18, pp. 1645–1646, 1996.

[5] L. C. Pérez, J. Seghers, and D. J. Costello Jr., "A distance spectrum interpretation of turbo codes," *IEEE Transactions on Information Theory*, vol. 42, no. 6, pp. 1698–1709, 1996.

[6] O. Y. Takeshita and D. J. Costello Jr., "New deterministic interleaver designs for turbo codes," *IEEE Transactions on Information Theory*, vol. 46, no. 6, pp. 1988–2006, 2000.

[7] J. Sun and O. Y. Takeshita, "Interleavers for turbo codes using permutation polynomials over integer rings," *IEEE Transactions on Information Theory*, vol. 51, no. 1, pp. 101–119, 2005.

[8] Z. Zhang, L. Dolecek, B. Nikolić, V. Anantharam, and M. J. Wainwright, "Design of ldpc decoders for improved low error rate performance: quantization and algorithm choices," *IEEE Transactions on Communications*, vol. 57, no. 11, pp. 1–12, 2009.

[9] C. Schlegel and S. Zhang, "On the dynamics of the error floor behavior in (regular) ldpc codes," submitted to *IEEE Transactions on Information Theory*.

[10] N. Wiberg, *Codes and decoding on general graphs*, Ph.D. thesis, Linköping University, Linköping, Sweden, 1996.

[11] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, San Mateo, Calif, USA, 1988.

[12] C. Di, D. Proietti, I. E. Telatar, T. J. Richardson, and R. L. Urbanke, "Finite-length analysis of low-density parity-check codes on the binary erasure channel," *IEEE Transactions on Information Theory*, vol. 48, no. 6, pp. 1570–1579, 2002.

[13] D. J. C. MacKay and M. S. Postol, "Weaknesses of Margulis and Ramanujan-Margulis low-density parity-check codes," *Electronic Notes in Theoretical Computer Science*, vol. 74, pp. 99–106, 2003.

[14] N. Axvig, D. Dreher, K. Morrison, E. Psota, L. C. Pérez, and J. L. Walker, "Average min-sum decoding of LDPC codes," in *Proceedings of the 5th International Symposium on Turbo Codes and Related Topics (TURBOCODING '08)*, pp. 356–361, September 2008.

[15] G. D. Forney Jr., R. Koetter, F. R. Kschischang, and A. Reznik, "On the effective weights of pseudocodewords for codes defined on graphs with cycles," in *Codes, Systems, and Graphical Models (Minneapolis, MN, 1999)*, vol. 123 of *IMA Volumes in Mathematics and Its Applications*, pp. 101–112, Springer, New York, NY, USA, 2001.

[16] C. Kelley, D. Sridhara, J. Xu, and J. Rosenthal, "Pseudocodeword weights and stopping sets," in *Proceedings of the IEEE International Symposium on Information Theory*, p. 150, Chicago, Ill, USA, June-July 2004.

[17] T. Richardson, "Error floors of LDPC codes," in *Proceedings of the 41st Allerton Conference on Communications, Control, and Computing*, Monticello, Ill, USA, October 2003.

[18] B. J. Frey, R. Koetter, and A. Vardy, "Signal-space characterization of iterative decoding," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 766–781, 2001.

[19] E. Psota and L. C. Pérez, "LDPC decoding and code design on extrinsic trees," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT '09)*, pp. 2161–2165, June-July 2009.