Research Article

Optimization of the Forcing Term for the Solution of Two-Point Boundary Value Problems

Gianni Arioli

*Dipartimento di Matematica "F. Brioschi", Modellistica e Calcolo Scientifico (MOX), Politecnico di Milano,
Via Bonardi 9, 20133 Milano, Italy*

Correspondence should be addressed to Gianni Arioli; gianni.arioli@polimi.it

Received 28 August 2012; Accepted 1 January 2013

Academic Editor: Alfredo Peris

Copyright © 2013 Gianni Arioli. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

We present a new numerical method for the computation of the forcing term of minimal norm such that a two-point boundary value problem admits a solution. The method relies on the following steps. The forcing term is written as a (truncated) Chebyshev series, whose coefficients are free parameters. A technique derived from automatic differentiation is used to solve the initial value problem, so that the final value is obtained as a series of polynomials whose coefficients depend explicitly on (the coefficients of) the forcing term. Then the minimization problem becomes purely algebraic and can be solved by standard methods of constrained optimization, for example, with Lagrange multipliers. We provide an application of this algorithm to the planar restricted three body problem in order to study the planning of low-thrust transfer orbits.

1. Introduction

We consider a two-point boundary value problem for the nonautonomous dynamical system in $\mathbb{R}^N$:

$$\begin{aligned} y'(t) &= f(y(t)) + \varphi(t), \\ y(-1) &= y_0, \\ y(1) &= y_1, \end{aligned} \quad (1)$$

where $f \in C^1(\mathbb{R}^N, \mathbb{R}^N)$ and $\varphi \in C([-1, 1], \mathbb{R}^N)$. We introduce a new numerical method for the computation of φ of minimal norm, among the functions representable with a truncated Chebyshev series, such that (1) admits a solution. In order to evaluate the performance of the algorithm, we apply it to a well-known problem, that is, the study of low-thrust orbits between the Earth and the Moon. We refer to [1] and references there for a discussion of the astrodynamical problem.

Our new approach is derived from the method introduced in [2, 3] to consider the dependence of parameters of the solution of hyperbolic equations. Such method heavily relies on a computer representation of algebras of functions,

in such a way that the computer can use such functions in a straightforward and transparent way as if they were floating point numbers, reducing to a minimum the computations to be done explicitly by hand. The underlying ideas of this method stem from automatic differentiation algorithms, see [4–8] and recently have been used in the field of computer-assisted proofs, see [9, 10] for some examples, but its potentiality in optimization and control appears to have been overlooked. In Section 2 we describe the approach in a generic setting; in Section 3 we describe the specific-planar-restricted three body problem that we chose as an example; in Section 4 we present the results that we obtained with the three body problem; in Section 5 we discuss the features of the method and compare the Taylor and the Chebyshev representation.

2. The Generic Problem

Consider an initial value problem in $\mathbb{R}^n$

$$\begin{aligned} y'_c(t) &= f(t, y(t)) + \varphi_c(t), \\ y_c(-1) &= y_0, \end{aligned} \quad (2)$$

where f is an analytic function, $c = \{c_{ij}\}$ is an $n \times N$ real matrix and the forcing term $\varphi_c : [-1, 1] \rightarrow \mathbb{R}^n$ depends on c as follows:

$$(\varphi_c(t))_i = \sum_{j=0}^{N-1} c_{ij} T_j(t), \quad (3)$$

where $T_j(t)$ is the j th Chebyshev polynomial and $(\varphi_c(t))_i$ denotes the i th component of $\varphi_c(t)$. Our first step is to compute explicitly the dependence of $y_c(t)$ on c , for $t \in (-1, 1]$.

We choose a numeric method to solve the IVP; in order to focus on the issue of the parameter dependence, we choose the simplest method, for example, an explicit first-order Euler scheme with constant time step, and we consider the case $n = N = 1$. We show later that it is straightforward to adapt the method to other schemes, for example, a Runge-Kutta scheme with variable time step, and to the case $n, N > 1$.

Choose $K \in \mathbb{N}$, let $h = 2/K$ be the time step, set $t_k = kh - 1$ and $Y_k(c) = y_c(t_k)$ so that the Euler scheme reads

$$Y_{k+1}(c) = Y_k(c) + hf(t_k, Y_k(c)) + h\varphi_c(t_k) \quad (4)$$

and $Y_K(c) = y_c(1)$. Now write

$$Y_k(c) = \sum_{j=0}^{+\infty} y_{k,j} U_j(c), \quad (5)$$

where $\{U_j(c)\}_{j \in \mathbb{N}}$ is a basis of some space of functions X of our choice. In this paper we consider both the Taylor expansion for the set of analytic functions in a disc (i.e., $U_j(c) = c^j$) and the Chebyshev expansion for the set of continuous functions in $[-1, 1]$ (i.e., $U_j(c) = \cos(j \arccos c)$). Clearly, the method can be generalized to other expansions, for example, Fourier series or Legendre polynomials. Now we choose an order M and we approximate $Y_k(c)$ with the truncated expansion $\sum_{j=0}^M y_{k,j} U_j(c)$. Our aim is to compute the coefficients $y_{k,j}$ using an automatic algorithm. Since $y_0(c) = y_0$, then we have $y_{0,0} = y_0$ and $y_{0,j} = 0$, when $j \geq 1$.

The core of the method consists in a generalization of the Taylor Models approach; for a detailed description of the technique with different expansions see [2, 3]. Here, we only recall the basic ideas. First, we observe that it is straightforward to implement on a computer an arithmetic of Taylor or Chebyshev polynomials of one variable. Using object-oriented programming and operator overloading, it is possible to define a class `Fun` which represents a Taylor or Chebyshev expansion and a set of methods which perform the basic operations and the computation of the elementary functions. More precisely, an object `Fun` is represented as the list of the coefficients. It is straightforward to implement a procedure that, given a scalar α and two objects `Fun` T_1, T_2 , computes the object `Fun` corresponding to $\alpha(T_1 * T_2)$, where $*$ is either the addition or the multiplication. Then, given a polynomial $p(x)$, it is possible to compute the object `Fun` corresponding to $p(T_1)$, and since all analytic functions $f(x)$ can be approximated with polynomials, it is also possible to compute the object `Fun` that better approximates $f(T_1)$.

A more algebraic-oriented way of expressing this idea is the following: let X be the space of functions spanned by $\{U_j(c)\}_{j \in \{0, \dots, M\}}$ and let $T : X \rightarrow \mathbb{R}^{M+1}$ be the (invertible) map that extracts the coefficients. Our approach consists in lifting the basic arithmetics of X to $\mathbb{R}^{M+1}$. More precisely, we compute operators

$$\begin{aligned} \{\oplus, \otimes\} : \mathbb{R}^{M+1} \times \mathbb{R}^{M+1} &\longrightarrow \mathbb{R}^{M+1}, \\ \odot : \mathbb{R} \times \mathbb{R}^{M+1} &\longrightarrow \mathbb{R}^{M+1}, \end{aligned} \quad (6)$$

such that $T(f + g) = T(f) \oplus T(g)$, $T(fg) = T(f) \otimes T(g)$, and $T(\alpha g) = \alpha \odot T(g)$. Once this has been done, we can make extensive use of object-oriented programming and operator overloading in order to make this lifting completely transparent to the user.

The extension of this method to the case $n > 1$ is straightforward. The case $N > 1$ requires some additional work, since we have to consider a multivariable Taylor or Chebyshev expansion:

$$Y_k(c) = \sum_{j_1, \dots, j_N=0}^{+\infty} y_{k,j_1, \dots, j_N} U_{j_1}(c_1) \cdots U_{j_N}(c_N). \quad (7)$$

The sum of two of such expansions is done componentwise. The multiplication is less straightforward, but a very efficient algorithm for the case of the Taylor expansion has been introduced in [8], and it is not too hard to extend it to the Chebyshev (or some other) expansion. We refer to our programs for details.

With the class `Fun` consisting of the objects (7) and the functions for addition and multiplication in place, we can use a standard implementation of the algorithm (4), observing that, given Y_k , the methods of the class will take care of the computation of $f(t_k, Y_k)$, while $\varphi_c(t_k)$ is explicitly given by (3). Extending this approach to better integration schemes, say Runge-Kutta or multistep, is straightforward: it suffices to implement the chosen algorithm using objects `Fun` and their methods. By implementing and running the integration scheme of choice with the class `Fun`, we obtain all the coefficients $\{y_{k,j_1, \dots, j_N}\}$ and therefore the polynomials $Y_k(c)$ for all k . Now consider the polynomial $Y_K(c)$: solving the boundary value problem (1) consists in computing $c \in \mathbb{R}^{nN}$ such that $Y_K(c) = y_1$. This amounts to solving n algebraic equations in N unknowns. If we wish $\|\varphi_c\|$ to be small (where $\|\cdot\|$ is some norm of our choice), we recall that when $N > 1$ the system $Y_K(c) = y_1$ is underdetermined, therefore we can compute an explicit formula $\Phi(c) = \|\varphi_c\|$ and exploit the additional degrees of freedom to set the following constrained minimization problem: find $\bar{c} \in \mathbb{R}^{nN}$ such that

$$\Phi(\bar{c}) = \min_{Y_K(c)=y_1} \Phi(c). \quad (8)$$

This result can be achieved by standard constrained minimization algorithms, for example, Lagrange multipliers. Note that this method can be easily extended to solve other kinds of conditions at $t = 1$: let $m \leq n$ and $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ some

polynomial function. We can apply the same approach to find $\bar{c} \in \mathbb{R}^{nN}$ such that

$$\Phi(\bar{c}) = \min_{f(Y_K(c))=0} \Phi(c). \quad (9)$$

To sum up, once the class Fun and some algorithm for solving initial value problems for ODE's has been implemented, boundary value problems with optimization of the forcing term are reduced to computing the constrained minimum of a polynomial, the constraint being represented by a polynomial equation or system of equations. We remark that, in principle, such polynomial constrained minimization problems can be solved exactly (up to floating point errors), but this does not imply that the boundary condition is solved exactly for the original problem, since we are truncating the expansion (7) and we are neglecting the truncation error. One needs to check *a posteriori*, for example, by solving (2) with standard floating point numbers and with the forcing term provided by (3), with $c = \bar{c}$ and computing the error $\varepsilon = |y(1) - y_1|$ or $\varepsilon = |f(y(1))|$.

3. An Application: Low-Thrust Orbits in the Planar RTBP

In order to model the dynamics of a spaceship travelling from an orbit around the Earth to an orbit around the Moon, we consider the forced planar RTBP, that is the system:

$$\begin{aligned} \ddot{x} + 2\dot{y} &= \Omega_x + f(t) \frac{\dot{x}}{\sqrt{\dot{x}^2 + \dot{y}^2}}, \\ \ddot{y} - 2\dot{x} &= \Omega_y + f(t) \frac{\dot{y}}{\sqrt{\dot{x}^2 + \dot{y}^2}}, \end{aligned} \quad (10)$$

where

$$\begin{aligned} \Omega(x, y) &= \frac{x^2}{2} + \frac{y^2}{2} + \frac{1-\mu}{\sqrt{(x+\mu)^2 + y^2}} \\ &\quad + \frac{\mu}{\sqrt{(x+\mu-1)^2 + y^2}} + C. \end{aligned} \quad (11)$$

We choose $\mu \simeq 0.0121506683$, that is, the reduced mass of the Earth-Moon system, and $C = -1.600172454916536$, so that $\Omega(L_1) = 0$, L_1 being the Lagrangian point between the primaries. The forcing term is given by a scalar function $f(t)$ to be determined times a unit vector directed as the velocity of the spaceship. In other words, we assume that the thrust is always parallel to the velocity. We show how to apply the method described in the previous section to the problem of optimizing the travel from an orbit 167 km above the Earth to an orbit 100 km above the Moon.

We choose a connecting orbit passing close to L_1 . Note that the Jacobi function

$$J(x, y, \dot{x}, \dot{y}) = \dot{x}^2 + \dot{y}^2 - 2\Omega(x, y) \quad (12)$$

is approximately -60 on an orbit 167 km above the Earth and approximately -1.6 on an orbit 100 km above the Moon.

FIGURE 1: A complete orbit.

TABLE 1: Result enforcing R and V .

R	V	ε_T	$\ f_T\ $	ε_C	$\ f_C\ $
0.4	0.9	$1.03 \cdot 10^{-7}$	0.3012	$1.27 \cdot 10^{-7}$	0.3012
0.4	0.85	$2.32 \cdot 10^{-4}$	0.4215	$3.45 \cdot 10^{-4}$	0.4235
0.45	0.9	NA	NA	$7.66 \cdot 10^{-5}$	0.2104
0.43	0.9	$6.58 \cdot 10^{-10}$	0.1540	$1.39 \cdot 10^{-7}$	0.1540

Furthermore, in order to “cross” the region around the Lagrangian point L_1 , it is necessary to raise it above 0. We build the connecting orbit as follows. We start from the Lyapunov orbit around L_1 with $J = 0.1$ and approximate periodic orbits around the Moon and the Earth. Then we compute approximate connections from the Lyapunov orbit to the orbit around the Earth (backwards in time) and to the orbit around the Moon (forward in time). Choosing the Lyapunov orbit as a starting point is useful, if not necessary, because of its high instability: it is much harder to aim at the Lyapunov orbit than at a stable orbit around a primary. These orbits are obtained by applying a constant low thrust (i.e., $f(t) = c$, with small c), chosen by simple trial and error.

Once these rough orbits are computed, we apply the technique described above to compute an accurate correction so that the trajectory satisfies some required boundary conditions and at the same time the thrust is minimal. More precisely, we write (10) as a system of 4 first-order equations by setting $v = \dot{x}$ and $w = \dot{y}$, we write the modulus of the forcing term as

$$f(t) = c + \sum_{j=0}^{N-1} c_j T_j(t), \quad (13)$$

we choose some initial value on the trajectory at the arbitrary time $t = 0$ and we solve the equation with a fourth-order Runge-Kutta scheme up to some time $t = T$, using objects Fun as scalars, the coefficients c_j being the independent variables of the objects Fun. Referring to Section 2, we have $n = 4$ (the dimension of the system), we choose $N = 4$ and $M = 10$. We rescale the time in the interval $[-1, 1]$ and apply the method described in Section 2 to obtain $(x(T), y(T), v(T), w(T))$ as a function of $\{c_j\}$, expressed either as a Taylor series or a Chebyshev series.

TABLE 2: Result enforcing R , V and tangential trajectory.

R	V	ε_T	$\varepsilon_T^\perp$	$\ f_T\ $	ε_C	$\varepsilon_C^\perp$	$\ f_C\ $
0.42	0.87	$3.04 \cdot 10^{-4}$	$8.48 \cdot 10^{-5}$	0.3627	$1.53 \cdot 10^{-4}$	$8.26 \cdot 10^{-7}$	0.3653
0.42	0.9	$3.18 \cdot 10^{-6}$	$6.00 \cdot 10^{-7}$	0.2540	$4.02 \cdot 10^{-6}$	$9.75 \cdot 10^{-6}$	0.2541
0.44	0.82	$4.18 \cdot 10^{-9}$	$3.37 \cdot 10^{-10}$	0.2044	$2.48 \cdot 10^{-8}$	$2.34 \cdot 10^{-8}$	0.2044
0.46	0.82	$7.32 \cdot 10^{-8}$	$1.43 \cdot 10^{-7}$	0.1439	$3.77 \cdot 10^{-6}$	$2.22 \cdot 10^{-6}$	0.1439

Then we choose the condition to be satisfied at $t = T$. We tested the following: find $\{c_j\}$ such that $(x(T), y(T), v(T), w(T))$ is the same as in the case $f(t) = c$, but $\|f\|$ is minimal; find $\{c_j\}$ such that the distance of the spaceship to the Earth or the Moon is assigned together with the modulus of the final velocity, that is, $(x(T) - \mu)^2 + y(T) - R^2 = 0$ and $(v(T))^2 + w(T)^2 - V^2 = 0$ and $\|f\|$ is minimal; find $\{c_j\}$ such that the distance of the spaceship to the Earth or the Moon is assigned together with the modulus of the final velocity, with the additional requirement that the final velocity is orthogonal to the line connecting the satellite to a primary. All these problem belong to the class of problem described in the previous section, and the polynomial constrained minimization problems have been solved by the Lagrange multipliers method.

4. Results

Figure 1 represents an example of an orbit computed by using the technique. Tables 1 and 2 collect some results obtained for a trajectory backward in time starting close to a Lyapunov orbit around L_1 , more precisely at the point $(x(0), y(0), v(0), w(0)) = (0.73356888, -0.017228233, 0.25064405, 0.17220602)$ and ending close to an orbit around the Earth at time $T = -1.7$. The trajectory with a constant thrust $c = 0.2$ ends at $R = \sqrt{(x(T) - \mu)^2 + y(T)} = 0.421852$ and $V = \sqrt{(v(T))^2 + w(T)^2} = 0.876662$.

For both tables we computed the thrust of minimal norm necessary to end the trajectory with the values given in the columns R and V . The results of Table 2 represent the solution with the additional constraint that the final velocity is orthogonal to the segment from the spaceship to the Earth, corresponding to an approximate circular orbit. We tested the accuracy of the computation as explained at the end of Section 2, that is, we repeated the computation of the orbit with standard floating point numbers, using the explicit expression of the forcing term, and we used the result of such computation to verify that the constraints were satisfied. The values ε_T and ε_C represent the mean square of the relative errors on R and V , obtained with the Taylor and the Chebyshev computation, respectively, while $\varepsilon_T^\perp$ and $\varepsilon_C^\perp$ is $|(x(T) - \mu)v(T) + y(T)w(T)|$, that is, the error on the orthogonality requirement. The columns $\|f_T\|$ and $\|f_C\|$ represent the norm of the forcing terms. It is clear from the tables that the method can achieve very accurate results, and it is also clear that the norm of the forcing term obtained with the Taylor and the Chebyshev computation is (almost) the same.

5. Conclusions

We discuss and compare the features of the two expansion that we tested: Taylor and Chebyshev. Generally speaking (see [2, 3]), the main advantages of the Taylor expansion consist in three features: the algorithms are simpler and faster, they provide directly the derivatives with respect to the parameters and furthermore, and, since one does not need to know the radius of convergence *a priori*, one can apply them and compute *a posteriori* the interval of validity of the computation. The Taylor expansion has also two main disadvantages: it does not provide uniform errors in the interval where the computation is performed and the radius of convergence may be bounded by poles in the complex plane. The Chebyshev expansion has opposite features: the error is uniform in the interval and the region of convergence is an ellipse as thin as necessary, therefore there are no problems caused by poles with nontrivial imaginary parts. On the other hand, the algorithm for the multiplication is much more complicated and much slower, it does not provide any direct computation of the derivatives and finally, since the Chebyshev polynomials are defined in $[-1, 1]$, one needs to choose *a priori* (via a suitable translation/rescaling) the interval where the parameter ranges, finding out only *a posteriori* if the approximation is acceptable. In the application considered in [2, 3], the Chebyshev expansion was a clear winner, due to the fact that, when considering the numerical approximation of shock waves and when trying to improve the resolution, the problem of poles becomes the main issue. Here, our judgment is the opposite: since the solutions of the ode's that we are studying are analytic functions, the issue of poles limiting the radius of convergence of Taylor series becomes negligible, while the disadvantages of the Chebyshev expansion, in particular the slower algorithms and the fact that the domain has to be chosen *a priori*, become very relevant. Additionally, we do not have evidence of a better performance in terms of accuracy. It is easy to find examples when either expansion performs better than the other one, but on average the accuracy is similar.

References

- [1] G. Mingotti, F. Topputo, and F. Bernelli-Zazzera, "Numerical methods to design low-energy, low-thrust sun-perturbed transfers to the moon," in *Proceedings of the 49th Israel Annual Conference on Aerospace Sciences*, Tel Aviv-Haifa, Israel, March 2009.

- [2] G. Arioli and M. Gamba, “An algorithm for the study of parameter dependence for hyperbolic systems,” MOX-Report 05/2011, <http://mox.polimi.it/it/progetti/pubblicazioni/quaderni/05-2011.pdf>.
- [3] G. Arioli and M. Gamba, “Automatic computation of Chebyshev polynomials for the study of parameter dependence for hyperbolic systems,” MOX-Report 07/2011, <http://mox.polimi.it/it/progetti/pubblicazioni/quaderni/07-2011.pdf>.
- [4] K. Makino and M. Berz, “Higher order verified inclusions of multidimensional systems by Taylor models,” in *Proceedings of the 3rd World Congress of Nonlinear Analysts*, part 5, Catania, Italy, July 2000.
- [5] K. Makino and M. Berz, “Higher order verified inclusions of multidimensional systems by Taylor models,” *Nonlinear Analysis: Theory, Methods and Applications*, vol. 47, no. 5, pp. 3503–3514, 2001.
- [6] M. Berz and K. Makino, “Taylor models and other validated functional inclusion methods,” *International Journal of Pure and Applied Mathematics*, vol. 6, no. 3, pp. 239–316, 2003.
- [7] M. Berz and K. Makino, “Taylor models and other validated functional inclusion methods,” *International Journal of Pure and Applied Mathematics*, vol. 4, no. 4, pp. 379–456, 2003.
- [8] M. Berz and K. Makino, “Higher order multivariate automatic differentiation, validated computation of remainder bounds,” *Transactions on Mathematics*, vol. 3, no. 1, pp. 37–44, 2004.
- [9] G. Arioli and H. Koch, “Computer-assisted methods for the study of stationary solutions in dissipative systems, applied to the Kuramoto-Sivashinski equation,” *Archive for Rational Mechanics and Analysis*, vol. 197, no. 3, pp. 1033–1051, 2010.
- [10] G. Arioli and H. Koch, “Integration of dissipative partial differential equations: a case study,” *SIAM Journal on Applied Dynamical Systems*, vol. 9, no. 3, pp. 1119–1133, 2010.