Efficient path tracking methods

Andrew Sommese, Daniel Bates, Jonathan Hauenstein

Publication Date

28-02-2024

Numer Algor (2011) 58:451–459
DOI 10.1007/s11075-011-9463-8

ORIGINAL PAPER

Efficient path tracking methods

Daniel J. Bates · Jonathan D. Hauenstein · Andrew J. Sommese

Received: 21 April 2010 / Accepted: 28 March 2011 /
Published online: 20 April 2011

Abstract Path tracking is the fundamental computational tool in homotopy continuation and is therefore key in most algorithms in the emerging field of numerical algebraic geometry. Though the basic notions of predictor-corrector methods have been known for years, there is still much to be considered, particularly in the specialized algebraic setting of solving polynomial systems. In this article, the effects of the choice of predictor method on the performance of a tracker is analyzed, and details for using Runge-Kutta methods in conjunction with adaptive precision are provided. These methods have

D. J. Bates was supported by the Institute for Mathematics and its Applications (IMA), Colorado State University, and NSF DMS-0914674.

J. D. Hauenstein was supported by the Fields Institute, Texas A&M University, the Duncan Chair of the University of Notre Dame, and NSF grant DMS-0712910.

A. J. Sommese was supported by the Duncan Chair of the University of Notre Dame and NSF grant DMS-0712910.

D. J. Bates (✉)
Department of Mathematics, Colorado State University, Fort Collins, CO 80523, USA
e-mail: bates@math.colostate.edu
URL: http://www.math.colostate.edu/~bates

J. D. Hauenstein
Department of Mathematics, Texas A&M University, College Station,
Mailstop 3368, TX 77843, USA
e-mail: jhauenst@math.tamu.edu
URL: www.math.tamu.edu/~jhauenst

A. J. Sommese
Department of Applied and Computational Mathematics and Statistics,
University of Notre Dame, Notre Dame, IN 46556, USA
e-mail: sommese@nd.edu
URL: www.nd.edu/~sommese

been implemented in the Bertini software package, and several examples are described.

Keywords Path tracking · Homotopy continuation · Numerical algebraic geometry · Polynomial systems · Ordinary differential equations · Euler's method · Runge-Kutta methods · Precision · Adaptive precision

AMS 2000 Subject Classifications 65H10 · 65H20 · 65E05 · 65L06

1 Introduction

Homotopy continuation is one way to approximate the zero-dimensional solutions of a polynomial system $f : \mathbb{C}^N \to \mathbb{C}^N$. The idea is to cast $f(z)$ as a member of a family of polynomial systems of the same size, one of which, say $g(z)$ has known solutions. There are several common ways of doing so [15]. We may then build a homotopy, $H : \mathbb{C}^N \times \mathbb{C} \to \mathbb{C}^N$, connecting $f(z)$ (at $t = 0$) to $g(z)$ (at $t = 1$).

In particular, let $H(z, t)$ be a system of N polynomials with $(z, t) \in \mathbb{C}^N \times \mathbb{C}$ such that given any element x^* of the finite set $\mathcal{F}$ of nonsingular isolated solutions of $H(z, 1) = 0$, the connected component $\mathcal{C}$ of the set

$$\left\{(z, t) \in \mathbb{C}^N \times (0, 1] \,|\, H(z, t) = 0\right\} \tag{1}$$

that contains x^* is the graph of a differentiable map $t \to x(t)$ with $x(1) = x^*$. We say that $H(z, t)$ is a good homotopy with respect to $\mathcal{F}$ and that $x(t)$ is the solution path starting at $x^* \in \mathcal{F}$.

To approximate the solutions of $f(z)$, the task is to numerically follow each of these solution paths from the known solutions $\mathcal{F}$ at $t = 1$ to those we desire at $t = 0$. Predictor/corrector methods, which make use of ordinary differential equation solvers for prediction and Newton's method for corrections, are a common choice. More details about homotopy continuation and the construction of homotopies may be found in [1, 10, 11, 15].

In this article, we consider the consequences of various choices of a differential equations solver and explain how to incorporate adaptive precision methodologies into Runge-Kutta methods. One of the two main contributions of this article is that it provides a means for applying a combined adaptive precision/adaptive steplength technique to higher-order predictor methods. With fixed precision, numerical methods such as homotopy continuation will fail in the proximity of a singularity (a point (z', t') at which the Jacobian of the polynomial system is singular). This is due to the loss of accuracy caused by the ill-conditioning of the Jacobian matrix at (z', t'). For the standard homotopies [10, 15] the occurrence of singularities along a path for a particular problem is a probability zero event. Since the number of digits used in computer arithmetic is limited, even for t not near zero, paths can come close to singularities to

cause numerical difficulty. Adaptive precision methods go a significant way towards countering the numerical difficulties of these almost singularities.

Unnoticed path crossing and path failure are the two most common adverse effects of using a fixed level of precision. In particular, singularities occur when two paths cross. In the neighborhood of such a singularity, the accuracy of predictor and corrector methods drops, so it sometimes happens that the tracker will inadvertently correct to the wrong path, which can result in an incorrect set of endpoints at $t = 0$. Similarly, near a singularity, inaccurate predictions can cause a path tracker to repeatedly cut the steplength until the steplength drops below the threshold at which the tracker declares a path failure. The use of adaptive precision during path tracking mitigates these adverse effects of path tracking caused by the use of fixed precision. This article provides the first known set of heuristics for applying adaptive precision methods in the setting of higher-order predictor methods. This method indicates how best to choose both precision levels and steplengths to provide security and efficiency.

Though one familiar with numerical techniques for solving differential equations would expect higher order methods to be more efficient, Euler/Newton methods appear to be usual methods in the algebraic predictor/corrector setting. The results in this article indicate that there is indeed value in using higher-order methods such as the fifth-order Runge-Kutta-Fehlberg method (RKF45). This article underlines the need to bring the experience of the numerical analysis community to bear on the implementation of the algorithms of the numerical algebraic geometry community.

In Section 2, we recall Euler's method as well as higher-order methods. In Section 3, we indicate how to adapt precision on the fly while using higher-order methods and also how error control is different with Runge-Kutta methods than with the standard Euler/Newton scheme. Finally, computational evidence supporting the value of higher-order methods in this situation is provided in Section 4.

2 Overview of ODE methods

Runge-Kutta methods are well-suited for use as the prediction method for homotopy continuation path tracking. They only require an initial value to start the method, do not require the evaluation of higher-order derivatives, but can still have local truncation errors of any order. Runge-Kutta methods are classified based on both the number of function evaluations and the order of the local truncation error. For example, Euler's method is the first order Runge-Kutta method requiring one function evaluation, while the classical fourth order Runge-Kutta method (RK4) requires four function evaluations.

One way to monitor the local truncation error is to evaluate the method using a stepsize s and then evaluate the method twice each with stepsize $\frac{s}{2}$. An error estimate is obtained by comparing the two approximations.

Another way to monitor the local truncation error is to use embedded Runge-Kutta methods. Embedded Runge-Kutta methods, developed by

Fehlberg [7, 8], evaluate multiple Runge-Kutta approximations simultaneously. The fifth-order Runge-Kutta-Fehlberg method (RKF45) utilizes six function evaluations to compute both a fourth- and a fifth-order approximation. The difference between the two approximations provides an estimate for the local truncation error. RKF45 is compared with several other embedded Runge-Kutta methods in Section 4.

See [9, 14] for more information.

3 Adaptive stepsize and precision with higher-order methods

The adaptive precision tracking methods of [3, 4] describe rules for changing precision based on the local behavior of the path being followed. The rules provided in [3] for changing precision are based on a careful analysis of the Newton correction scheme. This analysis is extended to the Euler prediction step in [4]. Heuristics for changing the stepsize and precision together to approximate the optimal settings are also provided in that article. The following summarizes the methods of [3, 4] and extends this analysis to predictor methods with error estimates.

3.1 Summary of the adaptive precision methods

The adaptive precision methods of [3, 4] require the enforcement of three rules to maintain accuracy. Following the notation of those articles, let P denote the number of digits of precision and $u = 10^{-P}$ be the unit roundoff error. Let $10^{-\tau}$ be the accuracy to track the path and N be the maximum number of Newton iterations per step. Let $\|\cdot\|$ be a vector norm and its induced submultiplicative matrix norm and $\|d\|$ be the most recent error approximation (e.g., local truncation error approximation or Newton residual).

For a continuously differentiable function $F(z) : \mathbb{C}^n \to \mathbb{C}^n$, let $J(z)$ denote its Jacobian matrix, i.e., the matrix of partial derivatives. Let $\psi(z, u)$ and $\phi(z, u)$ account for the errors in evaluating $F(z)$ and $J(z)$, respectively, and suppose that they are of the form $\psi = \Psi u$ and $\phi = \Phi u$. The values Ψ and Φ can be approximated using methods presented in [3]. Let $\mathcal{E}$ account for the growth in errors for solving a system of linear equations. Extra digits, called safety digits and denoted σ_1 and σ_2, are used to account for underestimations of the values required.

The first rule asserts that the error perturbed Jacobian matrix J must be nonsingular, namely

$$P > \sigma_1 + \log_{10}[\|J^{-1}\|\mathcal{E}(\|J\| + \Phi)]. \tag{2}$$

With an error approximation $\|d\|$, the second rule asserts that the corrector must be able to converge using $(N - i)$ Newton iterations. By letting $D = \log_{10}\left[\|J^{-1}\|((2 + \mathcal{E})\|J\| + \mathcal{E}\Phi) + 1\right]$, the second rule is

$$P > \sigma_1 + D + (\tau + \log_{10}\|d\|)/(N - i). \tag{3}$$

The final rule asserts that the final accuracy of the corrector must be smaller than the required tolerance, namely

$$P > \sigma_2 + \tau + \log_{10}(\|J^{-1}\|\Psi + \|z\|). \tag{4}$$

In [4], (3) is used to relate stepsize and precision. To do this, the Euler prediction is written as the initial Newton iteration creating a residual from this initial iteration that is proportional to the stepsize s, i.e., $\|d\| = a|s|$. By writing $|s| = 10^{-\xi}$, (3) reduces to

$$P + \xi/N > \sigma_1 + D + (\tau + \log_{10} a)/N, \tag{5}$$

which can be satisfied by either increasing the precision P or decreasing stepsize by increasing ξ. The values of P and ξ are set to attempt to minimize the cost per unit advance along the path.

3.2 Outline of the algorithm

The adaptive precision rules presented in [3, 4] and summarized in the previous section extend to prediction methods that provide error estimates. Suppose that the prediction method computes a local error estimate of order p. That is, using a stepsize s, the local error estimate $\|d\|$ is approximately proportional to $|s|^{p+1}$.

As discussed in [4], (2) is superseded by (3) upon knowing a local error estimate. Hence, (3) is applied for $i = 0$ using the local error estimate $\|d\|$ provided by the prediction method. If this error estimate $\|d\|$ is smaller than the required tolerance, we accept the prediction without any Newton correction steps. Otherwise, Newton iterations are used and (3) is applied for $i > 0$ using the Newton residual of the last iteration as the local error estimate. To validate the accuracy of the prediction, Newton iterations are also applied if there are M consecutive steps for which the prediction error is smaller than the required tolerance. In Bertini, M has a default value of 5.

Equation (3) also is used to relate stepsize and precision. Letting $|s| = 10^{-\xi}$ and $\|d\| = a|s|^{p+1}$, (3) becomes

$$P + (p+1)\xi/N > \sigma_1 + D + (\tau + \log_{10} a)/N \tag{6}$$

for the prediction (i.e., $i = 0$). The values of P and ξ are set to attempt to minimize the cost per unit advance along the path.

4 Computational evidence for using higher-order methods

Adaptive precision tracking using higher-order predictor methods is implemented in the software package Bertini [2]. The embedded Runge-Kutta predictor methods available in Bertini are presented in Table 1.

Table 1 Summary of embedded Runge-Kutta methods implemented in Bertini

Name	Local error order	Approximation order	# of evaluations
Heun-Euler (HE12) [9, Section 8.3]	1	2	2
Norsett (RKN34) [6]	3	4	5
Fehlberg (RKF45) [9, Section 8.3]	4	5	6
Cash-Karp (RKCK45) [5]	4	5	6
Dormand-Prince (RKDP56) [13]	5	6	8
Verner (RKV67) [16]	6	7	10

The non-parallel examples presented here were run on a 2.4 GHz Opteron 250 processor with 64-bit Linux. The parallel examples were run on a cluster consisting of a manager that uses one core of a Xeon 5410 processor and 8 computing nodes each containing two 2.33 GHz quad-core Xeon 5410 processors running 64-bit Linux, i.e., one manager and 64 workers.

4.1 Comparing the methods

Section 5.5 of [3] and Section 3.2 of [4] describe solving a polynomial system arising from the inverse kinematics problem for a general size-revolute serial-link robot [12]. Using the same settings, Table 2 compares the methods of [3] and [4], using the minimum fixed precision (96 bits) with various predictor methods, and the adaptive precision method of this paper. The column labeled "nfe/path" presents the average number of function evaluations per path.

For each function evaluation, there is an associated set of linear equations to solve. Function evaluation together with solving the associated system of linear equations is by far the most expensive part of homotopy continuation. Table 3 shows the different precision used for the various predictor methods.

Table 2 Comparison of the average time of 10 runs of the IPP system, in seconds

ODE method	96 bit fixed precision	Method of [3]	New method	nfe/path	Method of [4]	
					time	nfe/path
Euler	184.01	38.54			32.73	1,296
HE12	80.12		17.97	550		
RKN34	62.21		16.96	454		
RKF45	55.01		14.48	445		
RKCK45	49.13		14.13	403		
RKDP56	56.50		16.41	461		
RKV67	66.46		16.64	620		

Table 3 Precision comparison for the IPP system

ODE method	nfe/path					
	Double	64 bit	96 bit	128 bit	160 bit	Total
Euler	1,178.5	34.4	77.2	5.9	0	1,296
HE12	457.7	19.1	64.4	8.8	0	550
RKN34	353.0	26.8	66.3	7.8	0.1	454
RKF45	367.3	16.5	43.9	15.9	1.4	445
RKCK45	333.7	19.4	40.2	9.4	0.3	403
RKDP56	356.7	43.5	46.5	12.1	2.2	461
RKV67	509.1	50.2	46.5	11.1	3.1	620

4.2 A family of systems

An example of a family of systems where every path leads to a nonsingular solution is the family of economics problems derived from [11, Section 7]. The original presentation of the polynomial system is

$$G_n(x_1,\ldots,x_n)=\begin{bmatrix} x_{n-1}x_n-1\\ (x_{n-2}+x_1x_{n-1})x_n-1\\ (x_{n-3}+x_1x_{n-2}+x_2x_{n-1})x_n-1\\ \vdots\\ (x_1+x_1x_2+\cdots+x_{n-2}x_{n-1})x_n-1\\ x_1+x_2+\cdots+x_{n-1}+1 \end{bmatrix}=0. \tag{7}$$

This system is reduced in [11] by noting that the first equation implies that $x_{n-1}\neq 0$ so that $x_n=\frac{1}{x_{n-1}}$. Upon substitution and clearing denominators for the remaining equations, we obtain the polynomial system

$$F_n(x_1,\ldots,x_{n-1})=\begin{bmatrix} x_{n-2}+x_1x_{n-1}-x_{n-1}\\ x_{n-3}+x_1x_{n-2}+x_2x_{n-1}-x_{n-1}\\ \vdots\\ x_1+x_1x_2+\cdots+x_{n-2}x_{n-1}-x_{n-1}\\ x_1+x_2+\cdots+x_{n-1}+1 \end{bmatrix}=0. \tag{8}$$

Table 4 Comparison for solving F_n using serial processing

n	Paths	Euler	HE12	RKN34	RKF45	RKCK45	RKDP56	RKV67
10	256	12.2s	2.5s	1.7s	1.7s	1.4s	1.8s	2.1s
11	512	21.5s	4.6s	3.2s	3.2s	2.6s	3.5s	4.6s
12	1,024	56.7s	11.8s	8.5s	8.7s	6.9s	9.2s	11.0s
13	2,048	3m3s	32.0s	21.4s	21.6s	16.9s	24.1s	28.9s
14	4,096	9m3s	1m32s	1m9s	1m12s	54.1s	1m21s	1m35s
15	8,192	15m41s	2m49s	2m4s	2m10s	1m39s	2m22s	2m46s
16	16,384	41m51s	7m3s	5m7s	5m25s	4m15s	6m2s	7m12s
17	32,768	1h37m35s	16m46s	12m50s	14m11s	10m58s	15m49s	19m33s
18	65,536	4h6m40s	42m3s	30m52s	34m19s	25m21s	36m48s	52m6s

Table 5 Comparison for solving nine-point path synthesis problem, in minutes

ODE method	Time
Euler	28.46
HE12	8.48
RKN34	5.93
RKF45	6.32
RKCK45	5.73
RKDP56	7.02
RKV67	8.77

For all n, F_n has total degree 2^{n-2}, which is equal to the number of nonsingular solutions.

The system F_n for $n = 10, \ldots, 18$ was solved using various prediction methods with a tracking tolerance of 10^{-7} and a final endgame convergence tolerance of 10^{-10}. The results are summarized in Table 4.

4.3 Solving a large system

Section 3.3 of [4] describes solving a polynomial system arising from the nine-point path synthesis problem [17] using a homotopy that utilizes its 2-homogeneous structure and two-fold symmetry. This system was solved using various prediction methods with a tracking tolerance of 10^{-7} and a final endgame convergence tolerance of 10^{-10} in parallel. The results are summarized in Table 5.

5 Conclusions

This article introduces the use of adaptive precision Runge-Kutta methods in homotopy continuation. Such higher-order methods are not new in their own right, but the addition of adaptive precision techniques to Runge-Kutta methods is new. This article also shows that higher order methods *are* worth using in this setting.

This article will certainly not be the final article on efficiency in homotopy continuation; indeed, much remains to be studied, e.g., multistep methods are not covered in this article. Finally, some of the ideas of this article, particularly regarding higher-order methods, carry over to the case of homotopy continuation for general nonlinear analytic functions.

References

1. Allgower, E.L., Georg, K.: Numerical continuation methods, An introduction. Springer Series in Computational Mathematics, vol. 13. Springer, Berlin (1990)
2. Bates, D.J., Hauenstein, J.D., Sommese, A.J., Wampler, C.W.: Bertini: Software for Numerical Algebraic Geometry. Available at http://www.nd.edu/~sommese/bertini
3. Bates, D.J., Hauenstein, J.D., Sommese, A.J., Wampler, C.W.: Adaptive multiprecision path tracking. SIAM J. Numer. Anal., **46**, 722–746 (2008)

4. Bates, D.J., Hauenstein, J.D., Sommese, A.J., Wampler, C.W.: Stepsize control for adaptive multiprecision path tracking. Contemp. Math. **496**, 21–31 (2009)
5. Cash, J.R., Karp, A.H.: A variable order Runge-Kutta method for initial value problems with rapidly varying right-hand sides. ACM Trans. Math. Software, **16**(3), 201–222 (1990)
6. Enright, W.H., Jackson, K.R., Nørsett, S.P., Thomsen, P.G.: Interpolants for Runge-Kutta formulas. ACM Trans. Math. Software, **12**(3), 193–218 (1986)
7. Fehlberg, E.: Klassische Runge-Kutta-Formeln fünfter und siebenter Ordnung mit Schrittweiten-Kontrolle. Computing (Arch. Elektron. Rechnen), **4**, 93–106 (1969)
8. Fehlberg, E.: Klassische Runge-Kutta-Formeln vierter und niedrigerer Ordnung mit Schrittweiten-Kontrolle und ihre Anwendung auf Wärmeleitungsprobleme. Computing (Arch. Elektron. Rechnen), **6**, 65–71 (1970)
9. Kincaid, D., Cheney, W.: Numerical Analysis: Mathematics of Scientific Computing, 3rd edn. Brooks/Cole Publishing Co., Pacific Grove, CA (2002)
10. Li, T.Y.: Numerical solution of polynomial systems by homotopy continuation methods. In: Cucker, F. (ed.) Handbook of Numerical Analysis, Volume XI, Special Volume: Foundations of Computational Mathematics, North-Holland, pp. 209–304 (2003)
11. Morgan, A.P.: Solving polynomial systems using continuation for engineering and scientific problems. Prentice Hall Inc., Englewood Cliffs, NJ (1987) Reprinted as Classics in Applied Mathematics (2009) 57, SIAM
12. Morgan, A.P., Sommese, A.J.: Computing all solutions to polynomial systems using homotopy continuation. Appl. Math. Comput. **24**(2), 115–138 (1987) Errata: Appl. Math. Comput., **51**, 209 (1992)
13. Prince, P.J., Dormand, J.R.: High order embedded Runge-Kutta formulae. J. Comput. Appl. Math. **7**(1), 67–75 1981
14. Shampine, L.F.: Numerical Solution of Ordinary Differential Equations. Chapman & Hall, New York (1994)
15. Sommese, A.J., Wampler, C.W.: The Numerical Solution of Systems of Polynomials Arising in Engineering and Science. World Scientific, Singapore (2005)
16. Verner, J.H.: Explicit Runge-Kutta methods with estimates of the local truncation error. SIAM J. Numer. Anal. **15**(4), 772–790 (1978)
17. Wampler, C.W., Morgan, A., Sommese, A.J.: Complete solution of the nine-point path synthesis problem for four-bar linkages. ASME J. Mech. Des. **114**(1), 153–159 (1992)