SIAM J. SCI. COMPUT.
Vol. 42, No. 6, pp. A3610–A3637
© 2020 Society for Industrial and Applied Mathematics

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

# A ROBUST NUMERICAL PATH TRACKING ALGORITHM FOR POLYNOMIAL HOMOTOPY CONTINUATION*

SIMON TELEN†, MARC VAN BAREL†, AND JAN VERSCHELDE‡

**Abstract.** We propose a new algorithm for numerical path tracking in polynomial homotopy continuation. The algorithm is "robust" in the sense that it is designed to prevent path jumping, and in many cases it can be used in (only) double precision arithmetic. It is based on an adaptive stepsize predictor that uses Padé techniques to detect local difficulties for function approximation and danger for path jumping. We show the potential of the new path tracking algorithm through several numerical examples and compare it with existing implementations.

**Key words.** homotopy continuation, Padé approximant, polynomial systems, a priori stepsize control, Fabry ratio theorem, power series

**AMS subject classifications.** 65H04, 65H10, 65H20, 41A21

**DOI.** 10.1137/19M1288036

## 1. Introduction.

Homotopy continuation is an important tool in numerical algebraic geometry. It is used for, among other things, isolated polynomial root finding and the numerical decomposition of algebraic varieties into irreducible components. For introductory texts on numerical algebraic geometry and homotopy continuation, we refer the reader to [1, 35, 46, 47, 48] and references therein. The reader who is unfamiliar with algebraic varieties may also consult, e.g., [15] for an excellent introduction.

Let $X$ be an affine variety of dimension $n$ with coordinate ring $R = \mathbb{C}[X]$ (this is the ring of polynomial functions on $X$; see [15, Chapter 5, section 4]), and let $h_i, i = 1, \ldots, n$, be elements of $R[t] = \mathbb{C}[X \times \mathbb{C}] = \mathbb{C}[X] \otimes_{\mathbb{C}} \mathbb{C}[t]$. The $h_i$ define the map

$$H : X \times \mathbb{C} \to \mathbb{C}^n$$

given by $H(x,t) = (h_i(x,t))_{i=1}^n$. Such a map $H$ should be thought of as a family of morphisms $X \to \mathbb{C}^n$ parametrized by $t$, which defines a *homotopy* with continuation parameter $t$. This gives the *solution variety*

$$Z = H^{-1}(0) = \{(x,t) \in X \times \mathbb{C} \mid h_i(x,t) = 0, i = 1, \ldots, n\} \subset X \times \mathbb{C}.$$

We will limit ourselves to the cases $X = \mathbb{C}^n, R = \mathbb{C}[x_1, \ldots, x_n]$ and $X = (\mathbb{C} \setminus \{0\})^n, R = \mathbb{C}[x_1^{\pm 1}, \ldots, x_n^{\pm 1}]$. We will refer to the second case as the *toric case,*

*Submitted to the journal's Methods and Algorithms for Scientific Computing section September 18, 2019; accepted for publication (in revised form) September 3, 2020; published electronically November 9, 2020.

https://doi.org/10.1137/19M1288036

**Funding:** The work of the second author was supported by the Research Council KU Leuven under the C1-project (Numerical Linear Algebra and Polynomial Computations), by the Fund for Scientific Research-Flanders (Belgium) through grant G.0828.14N (Multivariate Polynomial and Rational Interpolation and Approximation), and by EOS Project 30468160. The work of the third author was supported by the National Science Foundation under grant DMS 1854513.

†Department of Computer Science, KU Leuven, Leuven (Heverlee), B-3001, Belgium (simon.telen@kuleuven.be, Marc.VanBarel@cs.kuleuven.be).

‡Department of Mathematics, Statistics, and Computer Science, University of Illinois at Chicago, Chicago, IL 60607 USA (janv@uic.edu).



Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

and $(\mathbb{C} \setminus \{0\})^n$ is called the *algebraic torus*. In both cases, we will use the coordinates $x = (x_1, \ldots, x_n)$ on $X$. Note that for every fixed parameter value $t^* \in \mathbb{C}$, $H_{t^*} : X \to \mathbb{C}^n : x \mapsto H(x, t^*)$ represents a system of $n$ (Laurent) polynomial equations in $n$ variables with solutions $H_{t^*}^{-1}(0) \subset X$. Typically, for some parameter value $t_0 \in \mathbb{C}$, $H_{t_0}$ is a *start system* with known, isolated, and regular solutions, and for some other $t_1 \neq t_0$, $H_{t_1}$ represents a *target system* that we are interested in. Consider a point $(z_0, t_0) \in Z$. The task of a homotopy continuation algorithm is to track the point $(z_0, t_0) \in Z$ to a point $(z_1, t_1) \in Z$ along a continuous path

$$\{(x(s), \Gamma(s)), s \in [0, 1)\} \subset Z,$$

with $\Gamma : [0, 1] \to \mathbb{C}$ and $x(s) \in X, s \in [0, 1)$ such that $\Gamma(0) = t_0, x(0) = z_0, \Gamma(1) = t_1, x(1) = z_1$. We will mainly restrict ourselves to paths of the form $\{(x(t), t), t \in [0, 1)\}$ (i.e., $\Gamma(s) = s$), but other $\Gamma$ will be useful for constructing illustrative examples. The reason for excluding the point $s = 1$ from some of the intervals in these definitions is that continuous paths in $Z$ might "escape" from $X \times \mathbb{C}$ when the parameter $t$ approaches the target value $t_1$. For example, solutions may move to infinity or out of the algebraic torus. This kind of behavior, together with singular points on the path (e.g., *path crossing*), may cause trouble for numerical path tracking (we will make this more precise in section 2). Many tools have been developed for dealing with such situations [28, 40, 41, 44]. In this paper, we do not focus on these kinds of difficulties. Existing techniques can be incorporated into the algorithms we present.

In typical constructions, such as linear homotopies for polynomial system solving, $H$ is randomized such that the paths that need to be tracked do not contain singular points with probability one [48]. This implies, for example, that all paths are disjoint. However, there might be singularities very near the path in the parameter space. In this situation, the coordinates in $X$ along the path may become very large, which causes scaling problems,[1] or two different paths may be very near to each other for some parameter values. The latter phenomenon causes *path jumping*, which is considered one of the main problems for numerical path trackers. Path jumping occurs when, along the way, the solution that is being tracked "jumps" from one path to another. The typical reason for this is that starting from a point in $H_{t^*}^{-1}(0)$, the *predictor* step in the path tracking algorithm returns a point in $X \times \{t^* + \Delta t\}$ which, according to the *corrector* step, is a numerical approximation of a point in $H_{t^*+\Delta t}^{-1}(0)$ which is on a different path than the one being tracked. It is clear that path jumping is more likely to occur in the case where two or more paths come near each other. Ideally, a numerical path tracker should take small steps $\Delta t$ in such "difficult" regions and larger steps where there is no risk of path jumping. There have been many efforts to design such *adaptive stepsize* path trackers [22, 33, 45, 53]. However, the state-of-the-art homotopy software packages, such as PHCpack [60], Bertini [5], and HomotopyContinuation.jl [11], still suffer from path jumping, as we will show in our experiments. A typical way to adjust the stepsize is by an a posteriori step control. This is represented schematically (in a simplified way) by Figure 1.

In the figure, $0 < \beta < 1$ is a real constant, the $\| \cdot \|$ should be interpreted as a relative measure of the backward error, and $\tilde{z}$ is the predicted solution which is refined to $z_{t^*+\Delta t}$ by the corrector. If tol $\leq \epsilon$, then the corrector stage is not needed. If tol $= \infty$, then the first feedback loop never happens. Such extreme choices for tol are not recommended. With well-chosen values for tol and $\epsilon$, the second feedback

[1] Scaling problems caused by large coordinates can be resolved in homogeneous coordinates, after a projective transformation [39].

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

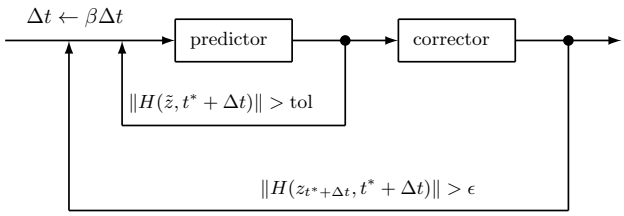


FIG. 1. *Two feedback loops in a predictor-corrector method for a posteriori step control.*

loop never occurs, as Newton's method converges to the required accuracy of $\epsilon$ in just a couple of steps. This type of feedback loop is implemented in, e.g., PHCpack [60] and Bertini [4]. Recently, Timme developed a new adaptive stepsize algorithm that is implemented in HomotopyContinuation.jl (v1.1) [53]. In this algorithm, the first $\Delta t$ that enters the loop in Figure 1 is computed such that it is an (estimate for an) upper bound for all "feasible stepsizes," and the corrected stepsize in the case of rejection is computed in a novel way. For details, see [53].

Certified path trackers have been developed to prevent path jumping [8, 13, 58, 64], but they require more computational effort. Moreover, the certification assumes that the coefficients of the input systems are exact rational numbers, as stated in [8].

In this paper, we propose an adaptive stepsize path tracking algorithm that is robust yet efficient. As opposed to standard methods, we use a priori step control: we compute the appropriate stepsize *before* taking the step. We use Padé approximants [3] of the solution curve $x(t)$ in the predictor step, not only to generate a next approximate solution, but also to detect nearby singularities in the parameter space. In the case of type $(L, 1)$ approximants (see section 3 for a definition), this is a direct application of Fabry's ratio theorem (Theorem 3.3). The Padé approximants are computed from the series expansion of $x(t)$. We use the iterative, symbolic-numeric algorithm from [10] to compute this series expansion. For an appropriate starting value $x^{(0)}(t) \in \mathbb{C}[[t]]$, we prove "second order convergence" of this iteration in the sense that $x(t) - x^{(k)}(t) = 0 \mod \langle t^{2k} \rangle$, where $x^{(k)}(t) \in \mathbb{C}[[t]]$ is the approximate series solution after the $k$th iteration, and $\langle \cdot \rangle$ denotes the ideal generated in the power series ring $\mathbb{C}[[t]]$ (see Proposition A.2). We use information contained in the Padé approximant to determine a trust region for the predictor and use this as a first criterion to compute the adaptive stepsize. A second criterion is based on an estimate for the distance to the most nearby path and a standard approximation error estimate for the Padé approximant.

We note that Padé approximants have been used before in path tracking algorithms [22, 45]. In these articles, their use has been limited to type $(2, 1)$ Padé approximants (see Definition 3.1) and they have not been used as nearby singularity detectors. In [30], Padé approximants are used in the context of symbolic deformation methods. Padé approximants are applied to solve nonlinear systems arising in power systems [56]. In [57], conceptual differences with continuation methods are discussed. Recent practical comparisons between this holomorphic embedding based continuation method and polynomial homotopy continuation can be found in [63].

The paper is organized as follows. In the next section, we describe numerical path tracking algorithms for smooth paths in general and give some examples. In section 3 we discuss fractional power series solutions and Padé approximants. Our path tracking algorithm is described in section 4 and implemented in version 2.4.72 of PHCpack,

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

which is available on github (https://github.com/janverschelde/PHCpack). We show the algorithm's effectiveness through several numerical experiments in section 5. We make comparisons with the built-in path tracking routines in (previous versions of) PHCpack [60], Bertini [5], and HomotopyContinuation.jl [11]. Appendix A contains a description of an algorithm introduced in [10] for computing power series solutions and a new proof of convergence.

**2. Tracking smooth paths.** Let $H(x,t) : X \times \mathbb{C} \to \mathbb{C}^n$ be as in the introduction where $X$ is either $\mathbb{C}^n$ or $(\mathbb{C} \setminus \{0\})^n$. We denote $Z = H^{-1}(0)$, and we assume that $\dim(Z) = 1$. To avoid ambiguities, we will denote $t$ for the coordinate on $\mathbb{C}$ in $X \times \mathbb{C}$ and $t^* \in \mathbb{C}$ for points in $\mathbb{C}$. We define the projection map $\pi : Z \to \mathbb{C} : (x,t) \mapsto t$. By [48, Theorem 7.1.1] $\pi$ is a ramified cover of $\mathbb{C}$ with ramification locus $S$ consisting of a finite set of points in $\mathbb{C}$, such that the fiber $\pi^{-1}(t^*)$ consists of a fixed number $\deg \pi = \delta \in \mathbb{N}$ of points in $Z$ if and only if $t^* \in \mathbb{C} \setminus S$. Let

$$J_H(x,t) = \left( \frac{\partial h_i}{\partial x_j} \right)_{i,j=1,\ldots,n}$$

be the Jacobian matrix of $H$ with respect to the $x_j$.

DEFINITION 2.1. *Let $H, Z$ be defined as above. Let $\Gamma : [0,1] \to \mathbb{C}$, and let $P = \{(x(s), \Gamma(s)), s \in [0,1)\} \subset Z$ be a continuous path in $Z$. We say that $P$ is* smooth *if $J_H(x,t) \in \mathrm{GL}(n,\mathbb{C})$ for all $(x,t) \in P$.*

If $P = \{(x(s), \Gamma(s)) \mid s \in [0,1)\} \subset Z$ is continuous with $\Gamma([0,1)) \cap S = \emptyset$, then $P \subset \pi^{-1}(\mathbb{C} \setminus S)$ is smooth. In this case, $\Gamma$ is called a *smooth parameter path*. In more down-to-earth terms, $\Gamma$ is smooth if $\{\Gamma(s), s \in [0,1)\} \subset \mathbb{C}$ contains only parameter values $t^*$ for which $H_{t^*}$ represents a (Laurent) polynomial system with the expected number of regular solutions.

*Example* 2.2. Consider the homotopy taken from [33] defined by

$$H(x,t) = x^2 - (t - 1/2)^2 - p^2, \tag{2.1}$$

where $p \in \mathbb{R}$ is a parameter which we take to be 0.1 in this example. It is clear that a generic fiber $\pi^{-1}(t^*)$ consists of the two points

$$\pm\sqrt{(t^* - 1/2)^2 + p^2},$$

and the ramification locus is $S = \{1/2 \pm p\sqrt{-1}\}$. Note that $J_H = \frac{\partial H}{\partial x}$ is equal to zero at $\pi^{-1}(t^*)$ for $t^* \in S$. We consider three different parameter paths:

$$\begin{aligned} &\Gamma_1 : s \mapsto s, \\ &\Gamma_2 : s \mapsto s - 4ps(s-1)\sqrt{-1}, \\ &\Gamma_3 : s \mapsto s + 0.2\sin(\pi s)\sqrt{-1}. \end{aligned}$$

In Figure 2 these paths are drawn in the complex plane. The background color at $t^* \in \mathbb{C}$ in this figure corresponds to the absolute value of $J_H$ evaluated at a point in $\pi^{-1}(t^*)$: dark (blue) regions correspond to a small value, as opposed to light (yellow) regions.

For each $\Gamma_i$, we track two different paths in $Z$ for $s \in [0,1]$ starting at $(z_0^{(1)}, 0) = (\sqrt{1/4 + p^2}, 0)$ and $(z_0^{(2)}, 0) = (-\sqrt{1/4 + p^2}, 0)$, respectively. The result is shown in Figure 3.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

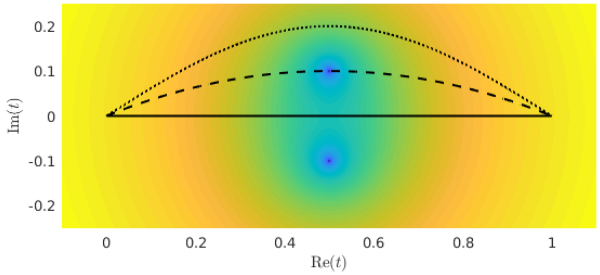


FIG. 2. *The image of* $[0,1]$ *under* $\Gamma_1$ *(full line),* $\Gamma_2$ *(dashed line), and* $\Gamma_3$ *(dotted line) as defined in Example* 2.2.

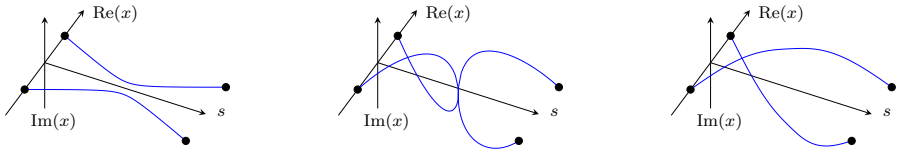


FIG. 3. *Solution curves with respect to* $s$ *using, from left to right,* $\Gamma_1, \Gamma_2$, *and* $\Gamma_3$.

Denote the corresponding paths on $Z$ by $P_j^{(i)} = \{(x^{(i)}(s), \Gamma_j(s)), s \in [0,1]\}$ where $x^{(i)}(0) = z_0^{(i)}$. Since $\Gamma_1$ and $\Gamma_3$ do not hit any singular points in the parameter space (Figure 2), the corresponding paths $P_j^{(i)}$ are disjoint and smooth. The paths corresponding to $\Gamma_2$, on the other hand, cross a singularity. They intersect at $s = 1/2$, as can be seen from Figure 3. We conclude that $\Gamma_2$ is not smooth.

An important application of smooth path tracking is the solution of systems of polynomial equations. The typical setup is the following. Define

$$F : X \to \mathbb{C}^n : x \mapsto (f_1(x), \ldots, f_n(x))$$

with $f_i \in R$. We want to compute $F^{-1}(0)$, that is, all points $x \in X$ such that $f_i(x) = 0, i = 1, \ldots, n$. The homotopy approach to this problem is to construct $H : X \times \mathbb{C} \to \mathbb{C}^n$ such that $H_1 : x \mapsto H(x,1)$ satisfies $Z_1 = H_1^{-1}(0) = F^{-1}(0)$ (the *target system* is equivalent to $F$), and the *start system* $G = H_0 : x \mapsto H(x,0)$ is such that $Z_0 = G^{-1}(0)$ is easy to compute and contains the expected number $\delta$ of regular solutions. The number $\delta$ is equal to, for example, the Bézout number in the case of total degree homotopies, or the mixed volume of the Newton polytopes in the case of polyhedral homotopies [48, 27, 62]. Moreover, $H$ has the additional property that $\Gamma : [0,1) \to \mathbb{C} : s \mapsto s$ is a smooth parameter path. We denote

$$Z_0 = G^{-1}(0) = \{z_0^{(1)}, \ldots, z_0^{(\delta)}\},$$

and by smoothness of $\Gamma$, we have that

$$Z_{t^*} = H_{t^*}^{-1}(0) = \{z_{t^*}^{(1)}, \ldots, z_{t^*}^{(\delta)}\}$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

consists of $\delta$ distinct points in $X$ for $t^* \in [0,1)$, and the paths $\{(z_{t^*}^{(i)}, t^*), t^* \in [0,1)\}$ are smooth and disjoint. Depending on the given system $F$, $Z_1$ may consist of fewer than $\delta$ points, or it might even consist of infinitely many points. Two or more paths may approach the same point as $t^* \to 1$, or paths may diverge to infinity. As stated in the introduction, several *end games* have been developed to deal with these kinds of situations [28, 40, 41, 44]. Here we will focus on the path tracking before the paths enter the end game operating region. We assume, for simplicity that this region is $[t_{\mathrm{EG}}, 1]$ for $t_{\mathrm{EG}}$ a parameter value "near" 1. Algorithm 2.1 is a simple template algorithm for smooth path tracking. With a slight abuse of notation, we use $z_{t^*}^{(i)}$ for both actual points on the path and "satisfactory" numerical approximations of the $z_{t^*}^{(i)}$.

**Algorithm 2.1.** Template path tracking algorithm with a priori step control.

```
1: procedure Track(H, Z_0)
2: Z_1 ← ∅
3: for z_0^(i) ∈ Z_0 do
4:    t* ← 0
5:    while t* < t_EG do
6:       (z̃, Δt) ← predict(H, z_{t*}^(i), t*)
7:       z_{t*+Δt}^(i) ← correct(H, z̃, t* + Δt)
8:       t* ← t* + Δt
9:    end while
10:   z_1^(i) ← end game(H, z_{t*}^(i), t*)
11:   Z_1 ← Z_1 ∪ {z_1^(i)}
12: end for
13: return Z_1
14: end procedure
```

The algorithm uses several auxiliary procedures. The *predictor* (line 6) computes a point $\tilde{z} \in X$ and a stepsize $\Delta t$ such that $\tilde{z}$ is an approximation for $z_{t^*+\Delta t}^{(i)}$. Some existing predictors use an Euler step (tangent predictor) or higher order integrating techniques, such as RK4.[2] Intuitively, the computed stepsize $\Delta t$ should be small in "difficult" regions. Algorithms that take this into account are called *adaptive stepsize* algorithms. The main contribution of this paper is the adaptive stepsize predictor algorithm which we present in detail in section 4. Our predictor computes an appropriate stepsize *before* the step is taken (a priori step control). The *corrector* step (line 7) then refines $\tilde{z}$ to a satisfactory numerical approximation of $z_{t^*+\Delta t}^{(i)}$. Typically, satisfactory means that the relative backward error of $z_{t^*+\Delta t}^{(i)}$ is of size $\pm$ the unit roundoff. The end game procedure in line 10 finishes the path tracking by performing an appropriate end game.

**3. Padé approximants.** Let $R = \mathbb{C}[x_1^{\pm 1}, \ldots, x_n^{\pm 1}]$ be the ring of Laurent polynomials in $n$ variables, and let $X = (\mathbb{C} \setminus \{0\})^n$ be the $n$-dimensional algebraic torus. Let $\mathbb{C}[[t]]$ be the ring of formal power series in the variable $t$, and let $\mathfrak{m} = \langle t \rangle$ be its maximal ideal. We denote by $\mathbb{C}[t]_{\le d} \simeq \mathbb{C}[[t]]/\mathfrak{m}^{d+1}$ the $\mathbb{C}$-vector space of polynomials of degree at most $d$. For $f, g \in \mathbb{C}[[t]]$, the notation $f = g + O(t^{d+1})$ means that $f - g \in \mathfrak{m}^{d+1}$. The field of fractions of $\mathbb{C}[t]$ is denoted $\mathbb{C}(t)$.

[2] Some higher order predictors need several previous points on the path in order to compute $\tilde{z}$. The predictor we present in this algorithm uses only the last computed point; hence we have the notation in Algorithm 2.1.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

We consider a homotopy given by $H(x,t) : X \times \mathbb{C} \to \mathbb{C}^n$. A *series solution at* $t^* \in \mathbb{C}$ of $H(x,t)$ is a parametrization of the form

$$x_j(s) = a_j s^{\omega_j} \left(1 + \sum_{\ell=1}^{\infty} a_{j\ell} s^{\ell}\right), \quad j = 1, \ldots, n, \quad \text{and} \quad t(s) = t^* + s^m, \tag{3.1}$$

with $m \in \mathbb{N} \setminus \{0\}$, $\omega = (\omega_1, \ldots, \omega_n) \in \mathbb{Z}^n$, $a = (a_1, \ldots, a_n) \in (\mathbb{C} \setminus \{0\})^n, a_{j\ell} \in \mathbb{C}$, and such that $H(x(s), t(s)) = H(x_1(s), \ldots, x_n(s), t(s)) \equiv 0$, and there is a real $\epsilon > 0$ such that the series $x_j(s)$ converge for $0 < |s| \leq \epsilon$. Such a series representation can be found for all irreducible components of $Z = H^{-1}(0)$ not contained in the hyperplane $\{t = t^*\}$ (see, for instance, [28, 37, 38, 41]). Substituting $s = (t - t^*)^{1/m}$ in (3.1) gives a set of *Puiseux series* $x_j(t)$ locally capturing the behavior of the solutions around $t = t^*$. If $J_H(x, t^*) \in \mathrm{GL}(n, \mathbb{C})$ for some $(x, t^*) \in Z_{t^*}$, the corresponding series solution is a set of Taylor series $x_j(t) \in \mathbb{C}[[t]], j = 1, \ldots, n$. This is the case when $t^*$ belongs to a smooth parameter path, as defined in section 2. From such a set of power series, we would like to estimate the location of nearby parameter values at which some $x_j(t)$ is singular. For this we will use Padé approximants. Note that it follows from (3.1) that the types of singularities we can encounter are branch points ($m > 1$) or poles ($\omega_j < 0, m = 1$).

Motivated by this discussion, in this section we discuss Padé approximants and the way they behave in the presence of poles and branch points. An extensive treatment of Padé approximants can be found in [3]. We will limit ourselves to the definition and the properties that are relevant to the heuristics of our algorithm.

The following definition uses some notation of [3].

DEFINITION 3.1 (Padé approximant). *Let* $x(t) = \sum_{\ell=0}^{\infty} c_\ell t^\ell \in \mathbb{C}[[t]]$. *The* type $(L, M)$ Padé approximant *of* $x(t)$ *is*

$$[L/M]_x = \frac{p(t)}{q(t)} \in \mathbb{C}(t)$$

*such that* $p(t) \in \mathbb{C}[t]_{\leq L}$, *and* $q(t) \in \mathbb{C}[t]_{\leq M}$ *is a unit in* $\mathbb{C}[[t]]$, *with*

$$[L/M]_x - x \in \mathfrak{m}^k \tag{3.2}$$

*for* $k$ *maximal.*

Informally, Padé approximants are rational functions agreeing with the Maclaurin series of a function $x$ up to a degree that is as large as possible. They are generalizations of truncated Maclaurin series, which are type $(L, 0)$ Padé approximants. Just as Maclaurin expansions are specific instances of Taylor expansions, it is straightforward to define Padé approximants around points $t = t^*$ in the complex plane different from 0. Without loss of generality, we consider only approximants around $t^* = 0$, since the general case reduces to this case after a simple change of coordinates. The type $(L, M)$ Padé approximant is known to exist and is unique. Multiplying the condition (3.2) by $q$ yields

$$p(t) - x(t)q(t) \in \mathfrak{m}^k \quad \text{or, equivalently,} \quad p(t) = x(t)q(t) + O(t^k) \tag{3.3}$$

for $k$ maximal. Writing $p(t) = a_0 + a_1 t + \cdots + a_L t^L$, $q(t) = b_0 + b_1 t + \cdots + b_M t^M$ and equating terms of the same degree, this gives $k$ linear conditions on the $a_i$, $b_i$, which can always be satisfied for $k \leq M + L + 1$. So for the linearized condition (3.3), $k$ is at

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

least $M+L+1$. Computing the $a_i$ and $b_i$ in practice is a nontrivial task. Difficulties are, for instance, degenerate situations where $\deg(p) < L$ or $\deg(q) < M$ and the presence of so-called *Froissart doublets* (spurious pole-zero pairs [54, Chapter 27]). Some of the issues are discussed in [3, Chapter 2] and [7, 25, 29]. In [25], a robust algorithm is proposed for computing Padé approximants. We will use this algorithm to compute Padé approximants from the coefficients $c_i$ in our algorithm, presented in section 4. The algorithm we use to compute the $c_i$ is discussed in the next section.

What is important for our purpose is that a Padé approximant can be used to detect singularities of $x(t)$ of the types we are interested in (poles and branch points) close to $t^* = 0$, even for relatively small $L$ and $M$. The idea is to compute Padé approximants of the coordinate functions $x_j(t)$ from local information on the path (the series coefficients $c_\ell$) and use them as a *radar* for detecting difficulties near the path. We are now going to motivate this. Since we intend to use Padé approximants to detect only *nearby* singularities, a natural first class to consider is that of type $(L, 1)$ approximants. We allow the approximant to have only one singularity, and we hope that it chooses to place this singularity near the actual nearest singularity to capture the nearby nonanalytic behavior. Here is a powerful result due to Beardon [6].

THEOREM 3.2. *Let $x_j(t)$ be analytic in $\{t^* \in \mathbb{C} \mid |t^*| \leq r\}$. An infinite subsequence of $\{[L/1]_{x_j}\}_{L=0}^{\infty}$ converges to $x_j(t)$ uniformly in $\{t^* \in \mathbb{C} \mid |t^*| \leq r\}$.*

*Proof.* We refer the reader to [6] or [3, Theorem 6.1.1] for a proof. ∎

This applies in our case as follows. Suppose that $(a, 0) \in X \times \mathbb{C}$ is a regular point of the variety $Z = H^{-1}(0)$, and the irreducible component of $Z$ containing $(a, 0)$ is not contained in $\{(x, t^*) \in X \times \mathbb{C} \mid t^* = 0\}$. Then there is a holomorphic function $x : \mathbb{C} \to X$ such that $x(0) = a$ and $H(x(t^*), t^*) = 0$ for $t^*$ in some nonempty open neighborhood of 0 (see, for instance, Theorem A.3.2 in [48]). That is, if $a$ is a regular solution of $H_0$, then the corresponding power series solution (3.1) consists of $n$ Taylor series $x_j(t)$. The function $x(t)$ can be continued analytically in a disk with radius $r$ if no singularities lie within a distance $r$ from the origin. Theorem 3.2 makes the following statement precise. For large enough degrees $L$ of the numerator of the Padé approximant, the $[L/1]_{x_j}$ are expected to approximate the coordinate functions $x_j(t)$ in a disk centered at the origin with radius $\pm$ the distance to the most nearby singularity. The fact that for sufficiently large $L$, the pole of $[L/1]_{x_j}$ is expected to give an indication of the *distance* to the nearest singularity (also if it is a branch point) can be seen as follows. Write $x_j(t) = \sum_{\ell=0}^{\infty} c_\ell t^\ell$ for the Maclaurin expansion of the coordinate function $x_j(t)$. Then a simple computation shows that if $c_L \neq 0$,

$$[L/1]_{x_j} = c_0 + c_1 t + \cdots + c_{L-1} t^{L-1} + \frac{c_L t^L}{1 - c_{L+1} t / c_L}.$$

Hence the pole of $[L/1]_{x_j}$ is $c_L/c_{L+1}$ (or it is $\infty$ if $c_{L+1} = 0$). For large $L$, the modulus $|c_L/c_{L+1}|$ can be considered an approximation of the limit

$$\lim_{L \to \infty} \left| \frac{c_L}{c_{L+1}} \right|$$

if this limit exists. Also, if this limit exists, it is a well-known expression for the convergence radius of the power series $x_j(t) = \sum_{\ell=0}^{\infty} c_\ell t^\ell$, which is the distance to the nearest singularity. Since the main application we have in mind is polynomial system solving, in which the homotopy is usually "randomized," in practice this limit

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

exists, and for reasonably small $L$, $|c_L/c_{L+1}|$ is a satisfactory approximation of the convergence radius of the power series. Theorem 3.2 suggests that more is true: it can be expected that the ratio $c_L/c_{L+1}$ is a reasonable estimate for the actual location of the most nearby singularity. This is Fabry's ratio theorem [20]; see also [9, 18, 51].

THEOREM 3.3. *If the coefficients of the power series* $x_j(t) = \sum_{\ell=0}^{\infty} c_\ell t^\ell$ *satisfy* $\lim_{L\to\infty} c_L/c_{L+1} = t_s$, *then* $t = t_s$ *is a singular point of the sum of this series. The point* $t = t_s$ *belongs to the boundary of the circle of convergence of the series.*

*Proof.* See [20]. ∎

We now briefly discuss the behavior of type $(L, M)$ Padé approximants in the presence of poles and branch points, and we end the section with two illustrative examples.

**3.1. Padé approximants and nearby poles.** Since Padé approximants are rational functions, it is reasonable to expect that they can capture this kind of behavior quite well. The following theorem, due to De Montessus [16], gives strong evidence of this intuition.

THEOREM 3.4. *Suppose* $x_j(t)$ *is meromorphic in the disk* $\{t^* \in \mathbb{C} \mid |t^*| \leq r\}$, *with* $\mu$ *distinct poles* $z_1, \ldots, z_\mu \in \mathbb{C}$ *in the punctured disk* $\{t^* \in \mathbb{C} \setminus \{0\} \mid |t^*| < r\}$. *Furthermore, suppose that* $m_i$ *is the multiplicity of the pole* $z_i$ *and* $\sum_{i=1}^{\mu} m_i = M$. *Then* $\lim_{L\to\infty}[L/M]_{x_j} = x_j$ *on any compact subset of* $\{t^* \in \mathbb{C} \mid |t^*| \leq r, t^* \neq z_i, i = 1, \ldots, \mu\}$.

*Proof.* This is Theorem 6.2.2 in [3]. ∎

Loosely speaking, this tells us that the poles of $[L/M]_{x_j}$, for large enough $L$, will converge to the $M$ most nearby poles of $x_j(t)$ (counting multiplicities) if these are the only singularities encountered in the disk $\{t^* \in \mathbb{C} \mid |t^*| \leq r\}$. For the $[L/1]_{x_j}$ approximant, this means that convergence may be expected beyond the nearest singularity if this is a simple pole, and the pole of $[L/1]_{x_j}$ will approximate the actual nearby pole. This may be considered as a practical approach to analytic continuation [55]. Padé approximants also give answers to the inverse problem: the asymptotic behavior of the poles of $[L/M]_{x_j}$ as $L \to \infty$ can be used to describe meromorphic continuations of the function $x_j(t)$. We do not give any details here; the interested reader is referred to [24, 50, 59].

**3.2. Padé approximants and nearby branch points.** Many singularities encountered in polynomial homotopy continuation are not poles but branch points. This situation is more subtle since the Padé approximant, being a rational function, cannot have branch points. For an intuitive description of the behavior of Padé approximants for functions with multivalued continuations, the reader may consult [3, section 2.2]. The conclusion is that the poles and zeros of $[L/M]_{x_j}$ are expected to delineate a "natural" branch cut. The authors of [3] also describe some ways to estimate the location and winding number of branch points using Padé approximants. We should also mention that there are convergence results in the presence of branch points which involve potential theory. We refer the reader to [49] for some important results for convergence of sequences of Padé approximants with $L, M \to \infty$, $L/M \to 1$ (so-called *near-diagonal* sequences). These results are beyond the scope of this paper, mainly because we will limit ourselves to *near-polynomial* approximants: we allow only a small number of poles (often we even take $M = 1$), and we will estimate nearby singularities directly from $[L/M]_{x_j}$. This is an unusual choice, since near-diagonal approximants tend to show better behavior for the approximation of algebraic

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

functions (see, e.g., [42, section 6.2]). The reason for this choice will become clear from the example in section 3.3.2.

We will show in experiments that in this way, even for small $L$, we can predict at least the order of magnitude of the distance to the nearest branch point, which is enough to sound an alarm when this distance gets small, and often we can do much better.

The reason for limiting ourselves to a small number of parameters $L+M$ and for not trying to compute a very accurate location of the nearest branch point and its winding number is, of course, efficiency. Moreover, for the purpose of this paper, a local approximation of the coordinate functions and a rough estimate of the nearest singularity suffice. The above-mentioned techniques for computing more information about nearby branch points may be powerful for approximation of algebraic curves in compact regions of the complex plane and for computing monodromy groups. We leave this for future research.

**3.3. Examples.** Our first example shows the potential of using Padé approximants for locating nearby singularities in the parameter space. The second one motivates the choice of type $(L,1)$ approximants over near-diagonal approximants.

**3.3.1. Padé approximants for a family of hyperbolas.** We consider again the homotopy (2.1) from Example 2.2. Let us first take $p = 0.19$ and consider the smooth parameter path $\Gamma_3$. It is clear that the singularity $z_+ = 1/2 + p\sqrt{-1} \in S$ is the closest singularity to nearly every point in $\Gamma_3([0,1])$. As $s$ moves closer to $1/2$, it moves closer to $z_+$. To show how this causes difficulties for the local approximation using Padé approximants, we have performed the following experiment. For several points $t^*$ on the parameter path $\Gamma_3([0,1])$, we have plotted the contour in $\mathbb{C}$ where the absolute value of the difference between $x(t) = \sqrt{(t-1/2)^2+p^2}$ and its type $(6,1)$ Padé approximation around $t^*$ equals $10^{-4}$. The result is shown in Figure 4. It is clear that the local approximation can be "trusted" in a much larger region if the singularity is far away.

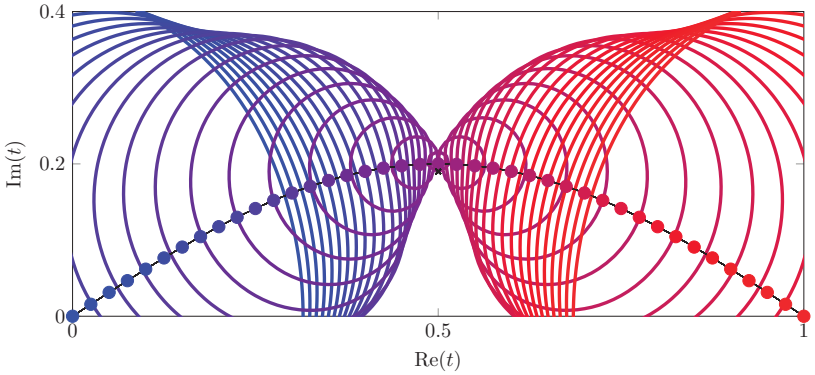


FIG. 4. *Contours of the approximation error as described in section* 3.3.1. *The color of a contour is the color of the corresponding dot on the parameter path. The singularity $z_+$ is shown as a small black cross.*

We now investigate the behavior of the pole of $[L/1]_x$ as we move along the path. We consider the four cases defined by $p = 0.15, 0.19$ and $L = 2, 6$. The results are shown in Figure 5.

The figure shows that as we move closer to $\Gamma(0.5)$ on the path, the pole of the

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

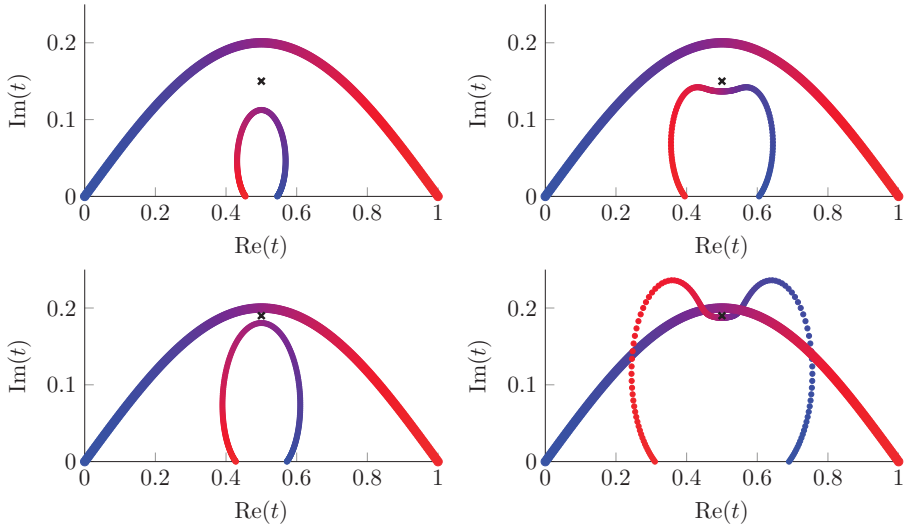


FIG. 5. *The path* $\Gamma_3([0,1])$ *and the corresponding path described by the pole of the type* $(L,1)$ *Padé approximant (associated points on the two paths have been given the same color) for* $p = 0.15$ *(first row),* $p = 0.19$ *(second row),* $L = 2$ *(left column),* $L = 6$ *(right column).*

Padé approximant moves closer to the actual branch point. What is important is that in the trouble region of the path ($s$ close to 0.5), the pole of $[L/1]_x$ is fairly close to $z_+$. It gives, at the least, an indication of the order of magnitude of the distance to $z_+$. Another way to see this is to note that on a point of the path near $z_+$, the $(L,1)$ Padé approximant is not so much influenced by the presence of $z_-$. For instance, at $t^* = 0$, the pole is real because $z_+$ and $z_-$ are complex conjugates, and they are located at the same distance from $\Gamma_3(0)$. For $t^*$ near $\Gamma_3(0.5)$, the pole has a relatively large positive imaginary part. A comparison of the first row to the second row in the figure shows that this effect gets stronger when a singularity moves closer to the path. Comparing the left column to the right column, we see that the approximation of $z_+$ gets better as $L$ increases, which is to be expected. If we use $\Gamma_1$ instead of $\Gamma_3$, for whatever $p$, the branch points $z_+$ and $z_-$ will have the same distance to each point of the path. The result is that the $(L,1)$ Padé approximant will have poles on the real line. For $L = 4$, $p = 0.001$, $t^* \in [0,1]$, the pole is contained in the real interval $[0.4997, 0.5003]$, so the local difficulties are detected. However, in this specific situation, it is more natural to use type $(L,2)$ approximants. The result for $L = 6$, $p = 0.05$ is shown in Figure 6.

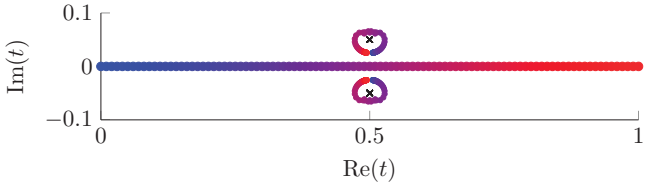


FIG. 6. *The path* $\Gamma_1([0,1])$ *and the corresponding paths described by the poles of the type* $(6,2)$ *Padé approximant (associated points on the two paths have been given the same color) for* $p = 0.05$.

We note that in a randomized homotopy, it is not to be expected that at a general point of the path two poles are equally important. As we move along the path, the

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

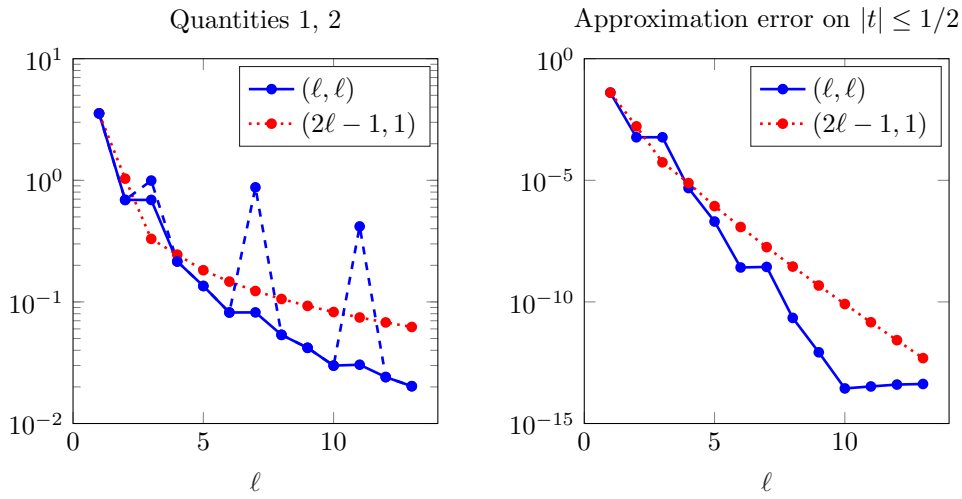


FIG. 7. *Results of the experiment in section* 3.3.2.

most important singularity may change, and the type $(L, 1)$ approximant can be expected to relocate its pole accordingly.

**3.3.2. Near-diagonal versus near-polynomial approximants.** Consider the algebraic function $x(t) = \sqrt{(t + 1.01)(t^2 - t + 37/4)}$ with branch points

$$S = \{-1.01, 1/2 + 3\sqrt{-1}, 1/2 - 3\sqrt{-1}\}.$$

For $\ell = 1, \ldots, 13$, we compute both type $(\ell, \ell)$ and type $(2\ell - 1, 1)$ Padé approximants (around $t = 0$) of $x(t)$ using a MATLAB implementation of the algorithm in [25]. For all these approximants we compute the following:

1. the minimum of the distances of the poles of the Padé approximant to the branch point $-1.01$;
2. the difference between the smallest modulus of the poles of the Padé approximant and the modulus of the nearest branch point, which is 1.01; and
3. an estimate for the approximation error (the infinity norm of a discretized approximation) of $x(t)$ on the disk $|t| \leq 1/2$ in the complex plane.

Results are shown in Figure 7. The right part of the figure shows that the diagonal approximants behave better for function approximation. However, for small $\ell$, the near-polynomial approximants are competitive.

For the type $(2\ell - 1, 1)$ approximant, the first two quantities coincide since the pole is real. For the $(\ell, \ell)$ case, the first quantity is a lower bound for the second one. This is illustrated by the difference between the dashed line and the full blue line in Figure 7. What happens is the following. One of the poles of the type $(\ell, \ell)$ approximant approximates the branch point $-1.01$, but some other pole indicates that there could be a branch point with smaller modulus. This is illustrated in Figure 8 for $\ell = 3, 4$ (for $\ell = 4$, one of the poles of the $(\ell, \ell)$ approximant lies close to that of the $(2\ell - 1, 1)$ approximant, and the corresponding dot is nearly invisible). The pole of the type $(3, 3)$ approximant that is closest to the origin actually comes from a Froissart doublet, which was not detected using the default settings in the algorithm of [25]. As a consequence, this spurious pole tells us that a singularity is nearby such that only a small step can be taken (see section 4.1.3), while the actual branch point

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

is quite far away. Detecting such Froissart doublets is often tricky. Since we will use only low orders, the approximation quality of the $(L, 1)$ approximant suffices for our purpose. Moreover, this example shows that they are more robust for estimating the distance to the nearest singularity. We will use this type of approximant in our default settings.

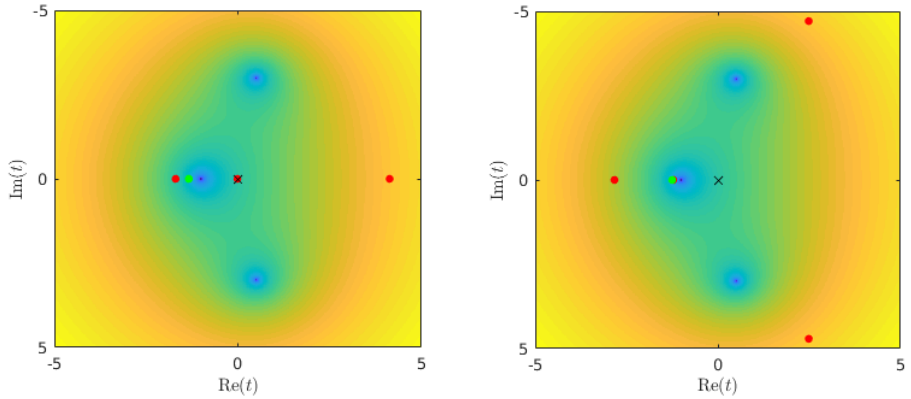


FIG. 8. *Poles of the type* $(\ell, \ell)$ *approximant (red dots) and pole of the type* $(2\ell-1, 1)$ *approximant (green dot) for* $\ell = 3, 4$ *(left and right respectively). The origin is indicated with a black cross. The background color corresponds to* $|x(t)|$*. (See online version for color.)*

**4. A robust algorithm for tracking smooth paths.** In this section we show how the results of the previous sections lead to a smooth path tracking algorithm. More specifically, we propose a new adaptive stepsize predictor for homotopy path tracking. We will use $\Gamma(s) = s$ and assume that this is a smooth parameter path for simplicity, but the generalization to different parameter paths is straightforward. The aim of this section is to motivate the heuristics and to present and analyze the algorithm. In the next section we will show some convincing experiments.

We will use Padé approximants for the prediction. The stepsize computation is based on two criteria. That is, we compute two candidate stepsizes $\{\Delta t_1, \Delta t_2\}$ based on two different estimates of the largest "safe" stepsize. The eventual value of $\Delta t$ that is returned by the predictor (line 6 in Algorithm 2.1) is $\min\{\Delta t_1, \Delta t_2, t_{\text{EG}} - t^*\}$. For the first criterion we estimate the distance to the nearest point of a different path in $X \times \{t^*\}$. This estimate is only accurate if we are actually in a difficult region. Comparing this to an estimate for the Padé approximation error, we compute $\Delta t_1$ such that the predicted point $\tilde{z}$ is much closer to the correct path than it is to the nearest different path. The value of $\Delta t_2$ is an estimate for the radius of the "trust region" of the Padé approximant, which is influenced by nearby singularities in the parameter space (see section 3). We discuss these two criteria in detail in the first subsection. In the second subsection we present the algorithm. In the last subsection we present a complexity analysis of our algorithm.

**4.1. Adaptive stepsize: Two criteria.** The values of $\Delta t_1$ and $\Delta t_2$ are computed from an estimate of the distance to the nearest different path, the approximation error of the Padé approximant for small stepsizes, and an estimate for some global "trust radius" of the Padé approximants. We discuss these estimates first and then turn to the computation of $\Delta t_1$ and $\Delta t_2$ from these estimates.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

**4.1.1. Distance to the nearest path.** We will use $\|\cdot\|$ to denote the Euclidean 2-norm for vectors and use the induced operator norm for matrices. Consider the homotopy $H : X \times \mathbb{C} \to \mathbb{C}^n$. Suppose that for some $t^* \in [0, 1)$, we have $H(z_{t^*}^{(1)}, t^*) = H(z_{t^*}^{(2)}, t^*) = 0$, so $z_{t^*}^{(1)} \neq z_{t^*}^{(2)} \in Z_{t^*}$ lie on two different solution paths. We assume that $z_{t^*}^{(1)}$ is close to $z_{t^*}^{(2)}$. Denote $\Delta z = z_{t^*}^{(2)} - z_{t^*}^{(1)} \in \mathbb{C}^n$, and think of $\Delta z$ as a column vector. Our goal here is to estimate $\|\Delta z\|$. Neglecting higher order terms, we get

$$H(z_{t^*}^{(2)}, t^*) \approx H(z_{t^*}^{(1)}, t^*) + J_H(z_{t^*}^{(1)}, t^*)\Delta z + \frac{v}{2}, \quad v = \begin{bmatrix} \langle \mathcal{H}_1(z_{t^*}^{(1)}, t^*)\Delta z, \Delta z\rangle \\ \vdots \\ \langle \mathcal{H}_n(z_{t^*}^{(1)}, t^*)\Delta z, \Delta z\rangle \end{bmatrix}, \tag{4.1}$$

where $(\mathcal{H}_i(x,t))_{j,k} = \frac{\partial^2 h_i}{\partial x_j \partial x_k}, 1 \leq j, k \leq n$, are the Hessian matrices of the individual equations, and $\langle \cdot, \cdot \rangle$ is the usual inner product in $\mathbb{C}^n$. To simplify the notation, we denote $\mathcal{H}_i = \mathcal{H}_i(z_{t^*}^{(1)}, t^*)$ and $J_H = J_H(z_{t^*}^{(1)}, t^*)$. The Hessian matrices are Hermitian, so they have a unitary diagonalization $\mathcal{H}_i = V_i \Lambda_i V_i^H$, where $\cdot^H$ is the Hermitian transpose, and the $V_i$ are unitary matrices with eigenvectors of $\mathcal{H}_i$ in their columns. We may write $\Delta z = V_i w_i$ for some coefficient vector $w_i$ such that $\|w_i\| = \|\Delta z\|$. We have $\langle \mathcal{H}_i \Delta z, \Delta z\rangle = \langle \Lambda_i w_i, w_i\rangle$. Let $\sigma_{k,\ell} = \sigma_\ell(\mathcal{H}_k)$ be the $\ell$th singular value of $\mathcal{H}_k$. The absolute values of the diagonal entries of $\Lambda_i$ are exactly the singular values, so that $|\langle \mathcal{H}_i \Delta z, \Delta z\rangle| \leq \sigma_{i,1} \|w_i\|^2 = \sigma_{i,1} \|\Delta z\|^2$. It follows easily that $\|v\| \leq \sqrt{\sigma_{1,1}^2 + \cdots + \sigma_{n,1}^2}\, \|\Delta z\|^2$. Since $\|J_H \Delta z\| \geq \sigma_n(J_H) \|\Delta z\|$ and by (4.1) we have $\|J_H \Delta z\| \approx \|v\|/2$, it follows that

$$\|\Delta z\| \gtrsim 2\sigma_n(J_H)(\sigma_{1,1}^2 + \cdots + \sigma_{n,1}^2)^{-\frac{1}{2}}. \tag{4.2}$$

Intuitively, the "more regular" the Jacobian, the larger $\|\Delta z\|$, and the more curvature, the smaller $\|\Delta z\|$. Motivated by (4.2), we state the following definition.

DEFINITION 4.1. *For $z_{t^*}^{(i)} \in Z_{t^*}, t^* \in [0, 1)$, set $J_H = J_H(z_{t^*}^{(i)}, t^*)$ and $\sigma_{k,\ell} = \sigma_\ell(\mathcal{H}_k(z_{t^*}^{(i)}, t^*))$, and define $\eta_{i,t^*} = 2\sigma_n(J_H)(\sigma_{1,1}^2 + \cdots + \sigma_{n,1}^2)^{-\frac{1}{2}}$.*

The numbers $\eta_{i,t^*}$ are estimates for the distance to the most nearby different path. To make sure the prediction error $\|x(t^* + \Delta t) - \tilde{x}(t^* + \Delta t)\|$ (where $\tilde{x}(t)$ is the coordinatewise Padé approximant) is highly unlikely to cause path jumping, we will solve $\|x(t^* + \Delta t) - \tilde{x}(t^* + \Delta t)\| = \beta_1 \eta_{i,t^*}$ for a small fraction $0 < \beta_1 \ll 1$ to compute an adaptive stepsize $\Delta t$. We now discuss how to estimate $\|x(t^* + \Delta t) - \tilde{x}(t^* + \Delta t)\|$.

**4.1.2. Approximation error of the Padé approximant.** Without loss of generality, we take the current value of $t$ to be zero and consider Padé approximants around $t^* = 0$ as in section 3. Suppose that we have computed a type $(L, M)$ Padé approximant $[L/M]_{x_j} = p_j(t)/q_j(t)$ of a coordinate function $x_j(t)$ around 0. Given a small real stepsize $\Delta t$, we want to estimate the error

$$|e_j(\Delta t)| = \left| \frac{p_j(\Delta t)}{q_j(\Delta t)} - x_j(\Delta t) \right| = \left| \frac{a_0 + a_1 \Delta t + \cdots + a_L \Delta t^L}{b_0 + b_1 \Delta t + \cdots + b_M \Delta t^M} - x_j(\Delta t) \right|. \tag{4.3}$$

From Definition 3.1 we know that $e_j(t) \in \mathfrak{m}^k$ (where usually $k = L + M + 1$), so (4.3) can be written as $|e_{0,j}\Delta t^k + e_{1,j}\Delta t^{k+1} + \cdots|$ with $e_{0,j} \neq 0$. For small $\Delta t$, the first term is expected to dominate the sum, and so $|e_j(\Delta t)| \approx |e_{0,j}\Delta t^k|$. This estimate is also used in [22] for the case $L = 2, M = 1$, and a similar strategy is common

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

for estimating the error in a power series approximation. An alternative is to use an estimate for the "linearized" error

$$|q_j(\Delta t)e_j(\Delta t)| = |p_j(\Delta t) - x_j(\Delta t)q_j(\Delta t)|, \tag{4.4}$$

which is equal to

$$|(b_0 + b_1\Delta t + \cdots + b_M\Delta t^M)(e_{0,j}\Delta t^k + e_{1,j}\Delta t^{k+1} + \cdots)| \approx |b_0 e_{0,j}\Delta t^k|.$$

Since $q_j(t)$ is a unit in $\mathbb{C}[[t]]$, $b_0 \neq 0$, and we can scale $p_j$ and $q_j$ such that $b_0 = 1$, and the estimates of (4.3) and (4.4) coincide. Taking $b_0 = 1$, we see that the constant $e_{0,j}$ is the coefficient of $t^k$ in $(a_0 + a_1 t + \cdots + a_L t^L) - (1 + b_1 t + \cdots + b_M t^M)(c_0 + c_1 t + \cdots)$, which is easily seen to be

$$e_{0,j} = a_k - (c_k + b_1 c_{k-1} + \cdots + b_M c_{k-M}), \tag{4.5}$$

where $a_k = 0$ if $k > L$ and $c_j = 0$ for $j < 0$. Doing this for all $j$ and assuming that $k$ is the same for all coordinates, we get an estimate

$$\left\| x(\Delta t) - \left( \frac{p_1(\Delta t)}{q_1(\Delta t)}, \ldots, \frac{p_n(\Delta t)}{q_n(\Delta t)} \right) \right\| \approx \|e_0\| \, |\Delta t|^k, \tag{4.6}$$

with $e_0 = (e_{0,1}, \ldots, e_{0,n})$.

**4.1.3. Trust region for the Padé approximant.** As discussed in section 3 and illustrated in section 3.3.1, branch points in the parameter space that are close to the parameter path cause problems for the Padé approximation. If none of the poles of $[L/M]_{x_j}$ are close to a current parameter value on the path, we may be able to take a reasonably large step forward without getting into difficulties. However, since we take $L$ and $M$ to be small, we cannot expect the approximants $[L/M]_{x_j}$ to have already converged in a disk with radius the distance to the nearest singularity; nor can we expect that the poles of $[L/M]_{x_j}$ are very good approximations of the actual singularities. Taking the distance $D$ to the most nearby pole of $[L/M]_{x_j}$ as an estimate for the convergence radius is a very rough estimate in this case. However, we observe that $D$ does give an estimate of the order of magnitude of the region in which $[L/M]_{x_j}$ is a satisfactory approximation. The conclusion is that we do not use $D$ itself but rather $\beta_2 D$, where $0 < \beta_2 < 1$ is a safety factor.

**4.1.4. The candidate stepsizes $\Delta t_1$ and $\Delta t_2$.** We now use the ingredients presented above to compute two candidate stepsizes $\Delta t_1$ and $\Delta t_2$. For $\Delta t_1$, we use the estimate $\eta_{i,t^*}$ for the distance to the nearest path and the estimate $\|e_0\| \, |\Delta t|^k$ for the approximation error of the Padé approximant. The heuristic is that we want the approximation error to be only a small fraction of the estimated distance to the nearest path, so that the predicted point $\tilde{z}$ is much closer to the path being tracked than it is to the nearest different path. That is, we solve $\|e_0\| \, |\Delta t_1|^k = \beta_1 \eta_{i,t^*}$ for $\Delta t_1$, where $\beta_1 > 0$ is a small factor. Since the attraction basins of Newton correction can behave in unexpected ways, it is best to take $\beta_1$ to be fairly small, for instance, $\beta_1 = 0.005$. This gives

$$\Delta t_1 = \sqrt[k]{\beta_1 \eta_{i,t^*} \, \|e_0\|^{-1}}. \tag{4.7}$$

Both the estimates $\eta_{i,t^*}$ and $\|e_0\| \, |\Delta t|^k$ are only accurate in the case when trouble is near (they are based on lowest order approximations). The case when $\|e_0\| \approx 0$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

causes trouble in the formula for $\Delta t_1$, but for too small values of $\|e_0\|$, we may set $\Delta t_1$ simply to one. If the resulting $\Delta t_1$ is large, the only thing this tells us is that we are not on a difficult point on the path with high probability. The second candidate stepsize, $\Delta t_2$, will make sure we do not take a step that is too large in this situation. At the same time, $\Delta t_2$ will be small when singularities in the parameter space are near the current point on the path. Let $D$ be the distance to the nearest pole out of all the poles of the $[L/M]_{x_j}, j = 1, \ldots, n$. We set

$$\Delta t_2 = \beta_2 D, \tag{4.8}$$

where $0 < \beta_2 < 1$ is a safety factor which should not change the order of magnitude, for instance, $\beta_2 = 0.5$.

*Example* 4.2. As mentioned above, the estimate $\eta_{i,t^*}$ for the distance to the nearest different path is only accurate when another path is actually near. If this is not the case, $\Delta t_1$ may be too large, and we need $\Delta t_2$ to make sure the resulting stepsize is still safe. To see that it is not enough to take only $\Delta t_2$ into account, consider the homotopy

$$H(x,t) = (x - (t - (a + b\sqrt{-1}))^2)(x + (t - (a + b\sqrt{-1}))^2), \quad t \in [0,1],$$

with $a, b \in \mathbb{R}$, $0 < a < 1$, and $|b|$ small. The paths corresponding to the two solutions are smooth and can be analytically continued in the entire complex plane: there are no singular points in $x_1(t), x_2(t)$. However, for $t = a + b\sqrt{-1}$ the two solutions coincide. By the assumptions on $a$ and $b$, this value of $t$ lies close to the parameter path $[0,1]$. Intuitively, the singularity of the Jacobian $J_H = \partial H/\partial x$ is canceled by a zero of $\partial H/\partial t$: along the solution paths we have

$$\frac{dx}{dt} = \frac{-\frac{\partial H}{\partial t}}{\frac{\partial H}{\partial x}} = \frac{4(t - (a + b\sqrt{-1}))^3}{2x} = \frac{4(t - (a + b\sqrt{-1}))^3}{\pm 2(t - (a + b\sqrt{-1}))^2} = \pm 2(t - (a + b\sqrt{-1})).$$

For $t = a$, the solutions are $x_1 = -b^2, x_2 = b^2$, so for small $b$, the paths are very close to each other. The type $(1,1)$ Padé approximant will have no poles (or will only have very large ones due to numerical artefacts), so taking only this criterion into account would allow for taking large steps. However, the estimate (4.2) at $t = a$ gives $|\Delta z| \approx 4b^2/2$, which is exactly the distance to the nearest different path.

**4.2. Path tracking algorithm.** We are now ready to present the path tracking algorithm. Since our contribution is in the predictor step (line 6 in Algorithm 2.1), we focus on this part. The predictor algorithm is Algorithm 4.1 below. It is straightforward to embed this predictor algorithm in the template Algorithm 2.1.

We briefly discuss some of the steps in Algorithm 4.1. In step 2, The algorithm of [10] is used. We included a description of the algorithm and a proof of convergence in Appendix A. The point around which we compute the series is $t^*$, the current parameter value on the path. The parameter $w = L + M + 2$ is the number of coefficients needed to compute the Padé approximant of type $(L, M)$ and the approximation error estimate. The starting value of the power series is the constant vector $x^{(0)} = z_{t^*}^{(i)}$, satisfying $H(z_{t^*}^{(i)}, t^*) = 0$ such that convergence of the algorithm in [10] is guaranteed. In step 6, the type $(L, M)$ Padé approximant of the coordinate function $x_j(t)$ is computed using the algorithm of [25]. Algorithm 4.1 has more input parameters than the predictor in the template algorithm. We will usually take $M$ very small (and often 1), motivated by the conclusions of section 3. The value of $L$ is chosen,

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

**Algorithm 4.1.** Predictor algorithm.

1: **procedure** Predict$(H, z_{t^*}^{(i)}, t^*, L, M, \beta_1, \beta_2, t_{\mathrm{EG}})$
2: $\{x_1(t), \ldots, x_n(t)\} \leftarrow$ series solution of $H$ at $t = t^*$ computed up to order $L + M + 2$ such that $x(0) = z_{t^*}^{(i)}$
3: $D \leftarrow \infty$
4: compute $\eta_{i,t^*}$ as in Definition 4.1
5: **for** $j = 1, \ldots, n$ **do**
6: $\quad p_j, q_j \leftarrow$ PadéApprox$(x_j(t), L, M)$
7: $\quad$ compute $e_{0,j}$ using (4.5)
8: $\quad D \leftarrow \min\{D, \min\{|z| \mid q_j(z) = 0\}\}$
9: **end for**
10: $e_0 \leftarrow (e_{0,1}, \ldots, e_{0,n})$
11: $\Delta t_1 \leftarrow \sqrt[k]{\frac{\beta_1 \eta_{i,t^*}}{\|e_0\|}}$
12: $\Delta t_2 \leftarrow \beta_2 D$
13: $\Delta t \leftarrow \min\{\Delta t_1, \Delta t_2, t_{\mathrm{EG}} - t^*\}$
14: $\tilde{z} \leftarrow (p_1(\Delta t)/q_1(\Delta t), \ldots, p_n(\Delta t)/q_n(\Delta t))$
15: **return** $\tilde{z}, \Delta t$
16: **end procedure**

for instance, such that $L + M + 2$ is a power of 2, e.g., $L = 5, M = 1$, because of the quadratic convergence property of the power series algorithm proved in Proposition A.2. Reasonable values for $\beta_1, \beta_2$ are $\beta_1 = 0.005, \beta_2 = 0.5$ as stated before. The parameter $t_{\mathrm{EG}}$ is the beginning of the end game operating region as in section 2.

**4.3. Complexity analysis.** To conclude this section, we analyze the cost of one predictor-corrector step of our adaptive stepsize algorithm as a function of the number of variables $n$. The total cost of the algorithm consists roughly of two main parts: the cost of the numerical linear algebra and the evaluation/differentiation operations.

LEMMA 4.3. *Consider a homotopy in $n$ variables. Let $L + M + 1$ be $O(n)$. The cost of the linear algebra operations of Algorithm* 4.1 *is $O(n^4)$.*

*Proof.* The dominant linear algebra operations in Algorithm 4.1 are the following:
line 2: solving a lower triangular block Toeplitz linear system,
line 4: computing the SVD of the Hessian matrices $\mathcal{H}_i$,
line 6: computing the Padé approximant from the series coefficients,
line 8: computing the roots of the denominators of the Padé approximants.
The lemma will follow from investigating the complexity of each of these computations.

Exploiting the block structure of the linearized representations of the power series, we see that the cost of the linear algebra operations in one Newton step on a series truncated to degree $\ell$ requires $O(n^3) + O(\ell n^2)$ operations [10]. In our application, $\ell = L + M + 1$, which is $O(n)$. Counting on the quadratic convergence of Newton's method, we need $O(\log(n))$ steps, so the linear algebra cost to compute the power series is $O(\log(n)n^3)$, which is $O(n^4)$.

The cost of one SVD decomposition of an $n$-by-$n$ matrix is $O(n^3)$; see. e.g., [17, section 5.4]. The bound (4.7) requires $n$ SVD decompositions, so we obtain $O(n^4)$.

The power series are input to $n$ Padé approximants of degrees $L$ and $M$, bounded by $O(n)$, as solutions of linear systems of size $O(n)$. The total cost of computing the coefficients of the Padé approximants is bounded by $O(n^4)$.

The pole distance requires the computation of the roots of the denominators of the Padé approximants. The denominators have degree $M$, and $M$ is $O(n)$. Computing

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

all eigenvalues of $n$ companion matrices is $O(n^4)$ using classical eigenvalue algorithms, and $O(n^3)$ using a specialized algorithm; see, e.g., [2].

Thus the cost of all linear algebra operations is $O(n^4)$. ∎

LEMMA 4.4. *The cost to differentiate and evaluate the $n$ Hessians $\mathcal{H}_k$ is $2n$ times the cost of computing the Jacobian matrix $J_H$.*

*Proof.* We apply a result from algorithmic differentiation; see [26] and, in particular, [14]. Let $f$ be a function in $n$ variables. If $W_f$ is the cost to evaluate $f$ and its gradient, then the cost of the evaluation of the Hessian matrix of $f$ is $2nW_f$. The lemma follows by application of this result to all polynomials $h_i$ in the homotopy $H$. ∎

LEMMA 4.5. *Let $W_N$ be the cost of evaluation and differentiation for applying Newton's method on $H(x,t)$ at $(z_{t^*}, t^*) \in X \times \mathbb{C}$. The cost of evaluation and differentiation for applying Newton's method on $H(x,t)$ at the series $x(t)$ truncated at degree $O(n)$ is $O(n\log(n))W_N$.*

*Proof.* If we evaluate a polynomial in a power series, then we have to perform as many multiplications of power series as there are multiplications in the evaluation of the polynomial at constant numbers. The overhead cost is therefore the cost to multiply two power series, denoted by $M(n)$ for power series truncated to degree $n$. According to [12], $M(n)$ is $O(n\log(n))$. ∎

The lemmas lead to the following result.

THEOREM 4.6. *The overhead cost of the a priori adaptive step control algorithm (Algorithm 4.1) relative to the a posteriori adaptive step control algorithm for a homotopy in $n$ variables is at most $O(n\log(n))$.*

*Proof.* Consider a predictor-corrector step in an a posteriori step control algorithm. The predictor is typically a fourth order extrapolator and runs in $O(n)$. The corrector applies a couple of Newton steps, which requires $O(n^3)$ linear algebra operations, with evaluation and differentiation cost $W_N$. According to Lemma 4.3, the overhead cost of the linear algebra operations is $O(n)$. By Lemma 4.4, $O(n)$ is also the overhead cost for the computation of the Hessians. The $O(n\log(n))$ is provided by Lemma 4.5. ∎

We comment on the "*at most* $O(n\log(n))$" in Theorem 4.6.

1. *The cost of evaluation/differentiation relative to linear algebra.* For very sparse polynomial systems, the cost of evaluation and differentiation could be independent of the degrees and as low as, for example, $O(n^2)$ or even $O(n)$. In that case, the evaluation and differentiation cost to compute the power series would be $O(n^3\log(n))$ or even as low as $O(n^2\log(n))$. In both cases, the cost of the linear algebra operations would dominate, and the overhead cost drops to $O(n)$.
2. *The value of $\ell = L + M + 1$ versus $n$.* Our analysis was based on the assumption that $\ell$ is $O(n)$. For many polynomial systems arising in applications, the number of variables $n \lesssim 10$. A typical value for $\ell$ is 7, as our default values for $L$ and $M$ are 5 and 1, respectively, so our assumption is valid.
   For cases when $n \gg \ell$, the overhead cost of working with power series then becomes $O(\ell\log(\ell))$, and $\ell$ may even remain fixed to 8. In cases when $n \gg \ell$, the overhead cost drops again to $O(n)$, as the cost of linear algebra operations dominates.

The focus of our cost analysis was on one step of applying our a priori adaptive step control algorithm and not on the total cost of tracking one entire path. This cost

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

TABLE 1
*Results of the experiment of subsection 5.1 for $p = 10^{-k}, k = 1, \ldots, 7$. An "✗" indicates that path jumping occurred.*

| Solver \ $k$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| `brt_DP` | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ |
| `brt_AP` | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ |
| `HC.jl` | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| `phc -p` | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| `phc -u` | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

depends on the number of steps required to track a path. We observe in experiments (see section 5) that using our algorithm allows some paths to be tracked successfully by taking only very few steps, even for high degree problems.

**5. Numerical experiments.** In this section we show some numerical experiments to illustrate the effectiveness of the techniques proposed in this article. The proposed method is implemented in PHCpack (v2.4.72), available at https://github.com/janverschelde/PHCpack, and in `Padé.jl`, an implementation of our algorithm in Julia. In the experiments, our implementations are compared with the state of the art. We will use the following short notation for the different solvers in our experiments:

`brt_DP` Bertini v1.6 using double precision arithmetic (MPTYPE = 0) [5];
`brt_AP` Bertini v1.6 using adaptive precision (MPTYPE = 2) [4];
`HC.jl` HomotopyContinuation.jl v1.1 [11];
`phc -p` the `phc -p` command of PHCpack v2.4.72 [60];
`phc -u` our algorithm, used in PHCpack v2.4.72 via `phc -u`;
`Padé.jl` our algorithm, implemented in Julia.

We use default double precision settings for all solvers except `brt_AP`, for which we use default adaptive precision settings. The experiments in all but the last subsection are performed on an 8 GB RAM machine with an Intel Core 17-6820HQ CPU working at 2.70 GHz. We restrict all solvers to the use of only one core for all the experiments, unless stated otherwise. We will use $\Gamma : [0,1] \mapsto \mathbb{C} : s \mapsto s$, which will be a smooth parameter path as defined in section 2 by the constructions in the experiments. Therefore, the parameter $s$ will not occur in this section, and paths are of the form $\{(x(t), t), t \in [0,1)\} \subset X \times [0,1)$. In all experiments, we use $\beta_1 = 0.005, \beta_2 = 0.5$. To measure the quality of a numerical solution of a system of polynomial equations, we compute its residual as a measure for the relative backward error. We use the definition of [52, section 7] to compute the residual.

**5.1. A family of hyperbolas.** Consider again the homotopy (2.1) from Example 2.2, which represents a family of hyperbolas parametrized by the real parameter $p$. Recall that the ramification locus is $S = \{1/2 + p\sqrt{-1}\}$. We will consider $p \neq 0$ here, such that $[0,1]$ is a smooth parameter path. The smaller $|p|$, the closer the branch points move toward the line segment $[0,1]$. As the value of $p > 0$ decreases, the two solution paths approach each other for parameter values $t^* \approx 0.5$ which causes danger for path jumping. This is confirmed by our experiments. Table 1 shows the results. We used $L = 5, M = 1$ in `phc -u`. The Julia implementation `HC.jl` checks whether the starting solutions are (coincidentally) solutions of the target system. For this reason, with this solver, we track for $t \in [0.1, 1]$.

**5.2. Wilkinson polynomials.** As a second experiment, consider the Wilkinson polynomial $W_d(x) = \prod_{i=1}^{d}(x - i)$ for $d \in \mathbb{N}_{>0}$. When $d > 10$, it is notoriously hard

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

TABLE 2
*Results for the experiment of subsection* 5.2.

| $d$ | `phc -p` | | `HC.jl` | | `brt_DP` | | `brt_AP` | | `phc -u` | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | $e$ | T | $e$ | T | $e$ | T | $e$ | T | $e$ | T | # |
| 10 | 5 | 8.0e-3 | 0 | 2.5e-3 | 0 | 4.5e-2 | 0 | 2.5e-2 | 0 | 4.0e-2 | 23-42 |
| 11 | 7 | 2.9e-2 | 0 | 3.6e-3 | 0 | 1.9e-1 | 0 | 1.4e+0 | 0 | 5.2e-2 | 12-45 |
| 12 | 9 | 3.4e-2 | 0 | 6.7e-3 | 0 | 1.5e-1 | 0 | 2.0e+0 | 0 | 6.9e-2 | 12-50 |
| 13 | 10 | 3.5e-2 | 0 | 4.1e-3 | 0 | 3.2e-1 | 0 | 2.8e+0 | 0 | 1.1e-1 | 35-54 |
| 14 | 11 | 2.4e-2 | 1 | 6.2e-3 | 0 | 4.8e-1 | 0 | 3.8e+0 | 0 | 1.0e-1 | 12-69 |
| 15 | 13 | 1.7e-2 | 1 | 9.0e-3 | 15 | 1.5e-2 | 15 | 1.6e-2 | 0 | 1.2e-1 | 43-63 |
| 16 | 15 | 2.1e-2 | 6 | 6.7e-3 | 16 | 1.6e-2 | 16 | 1.4e-2 | 0 | 1.7e-1 | 12-74 |
| 17 | 16 | 1.6e-2 | 10 | 3.2e-3 | 17 | 1.8e-2 | 17 | 1.3e-2 | 0 | 1.9e-1 | 11-73 |
| 18 | 18 | 6.0e-3 | 11 | 1.4e-2 | 18 | 1.8e-2 | 18 | 1.4e-2 | 0 | 2.4e-1 | 57-81 |
| 19 | 18 | 1.8e-2 | 13 | 7.0e-3 | 19 | 1.8e-2 | 19 | 1.4e-2 | 0 | 2.6e-1 | 12-83 |

to compute the roots of these polynomials numerically when they are presented in the standard monomial basis. For Bertini and HomotopyContinuation.jl, we use the blackbox solvers to find the roots of the $W_d(x)$. In PHCpack, we use

$$H(x,t) = (x^d - 1)(1-t) + \gamma W_d(x)t,$$

with $\gamma$ a random complex number.[3] We use default settings for other solvers and use $L = 5, M = 1$ in our algorithm to solve $W_d(x)$ for $d = 10, \ldots, 19$. The results are reported in Table 2.

The number $e$ is the number of failures, i.e., $d$ minus the number of distinct solutions (up to a certain tolerance) returned by each solver with residual $< 10^{-9}$, and T is the computation time in seconds. The column indexed by "#" gives the minimum and maximum number of steps on a path for our solver. We conclude this section with a brief comparison with certified tracking algorithms. For $W_4(x)$, the algorithm[4] proposed in [8] takes 6261.6 steps for the path starting at $z_0 = -1$ (this is averaged out over five experiments with random, rational $\gamma$). For $W_{15}(x)$, the certified tracking algorithm of [64] (which is specialized for the univariate case) takes on average 790 steps per path.

**5.3. Generic polynomial systems.** In this subsection, we consider random, square polynomial systems and solve them using the different homotopy continuation packages and the algorithm proposed in this paper. We now specify what "random" means. Fix $n$ and $d \in \mathbb{N} \setminus \{0\}$. A *generic polynomial system* of dimension $n$ and degree $d$ is given by $F : \mathbb{C}^n \to \mathbb{C}^n : x \mapsto (f_1(x), \ldots, f_n(x))$, where

$$f_i(x) = \sum_{|q| \le d} c_{i,q} x^q \in R = \mathbb{C}[x_1, \ldots, x_n],$$

with $q = (q_1, \ldots, q_n) \in \mathbb{N}^n$, $|q| = q_1 + \cdots + q_n$, and $c_{i,q}$ are complex numbers whose real and imaginary parts are drawn from a standard normal distribution. The *solutions* of $F$ are the points in the fiber $F^{-1}(0) \subset \mathbb{C}^n$, and by Bézout's theorem, there are $d^n$ such points. In order to find these solutions, we track the paths of the homotopy

$$H(x,t) = G(x)(1-t) + \gamma F(x)t, \quad t \in [0,1],$$

[3] The other solvers use $\Gamma(s) = 1 - s$ by default. This is not important here.

[4] We use a Macaulay2 implementation, available at http://people.math.gatech.edu/~aleykin3/RobustCHT/ to perform these experiments.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

TABLE 3
*Results for the experiment of subsection* 5.3.

| $n$ | $d$ | phc -p | | HC.jl | | brt_DP | | brt_AP | | phc -u | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $e$ | T | $e$ | T | $e$ | T | $e$ | T | $e$ | T | # | $h$ |
| 1 | 20 | 0 | 5.0e+0 | 0 | 1.7e-3 | 0 | 3.1e-2 | 0 | 7.5e-2 | 0 | 4.2e-2 | 6-16 | 0.09 |
| | 50 | 0 | 2.6e-2 | 0 | 6.3e-3 | 0 | 1.3e-1 | 0 | 2.3e+0 | 0 | 2.4e-1 | 5-27 | 0.07 |
| | 100 | 2 | 9.1e-2 | 0 | 1.1e-2 | 49 | 5.3e-1 | 0 | 1.2e+1 | 0 | 8.9e-1 | 4-27 | 0.13 |
| | 200 | 2 | 2.7e-1 | 0 | 3.2e-2 | 97 | 1.6e+0 | 1 | 4.5e+1 | 0 | 2.9e+0 | 5-25 | 0.13 |
| | 300 | 5 | 6.6e-1 | × | × | 221 | 2.8e+0 | 27 | 3.3e+2 | 0 | 8.3e+0 | 4-49 | 0.13 |
| 2 | 10 | 0 | 1.8e-1 | 0 | 1.5e-2 | 0 | 3.8e-1 | 0 | 2.4e+0 | 0 | 2.1e+0 | 8-37 | 0.10 |
| | 20 | 2 | 2.2e+0 | 0 | 8.9e-2 | 0 | 1.4e+1 | 0 | 1.2e+2 | 0 | 2.6e+1 | 8-55 | 0.13 |
| | 30 | 8 | 1.2e+1 | 0 | 3.3e-1 | 0 | 9.9e+1 | 0 | 2.0e+3 | 0 | 1.3e+2 | 8-68 | 0.13 |
| | 40 | 22 | 3.7e+2 | 0 | 9.1e-1 | 68 | 3.5e+2 | 0 | 7.8e+3 | 0 | 4.2e+2 | 6-57 | 0.15 |
| | 50 | 39 | 8.7e+2 | 0 | 2.3e+0 | 12 | 1.4e+3 | 0 | 3.4e+4 | 0 | 1.0e+3 | 7-57 | 0.14 |
| 3 | 5 | 0 | 3.5e-1 | 0 | 3.0e-2 | 0 | 7.0e-1 | 0 | 7.0e-1 | 0 | 4.8e+0 | 9-55 | 0.09 |
| | 9 | 1 | 8.5e+0 | 0 | 2.3e-1 | 0 | 2.1e+1 | 0 | 4.8e+1 | 0 | 9.8e+1 | 8-56 | 0.10 |
| | 13 | 4 | 6.8e+1 | 0 | 1.5e+0 | 0 | 2.3e+2 | 0 | 1.0e+3 | 0 | 8.3e+2 | 8-85 | 0.11 |

where $\gamma$ is a random complex constant, and $G : \mathbb{C}^n \to \mathbb{C}^n : x \mapsto (x_1^d - 1, \ldots, x_n^d - 1)$ represents the *start system* with $d^n$ known, regular solutions. Results are given in Table 3.

In the table, $n$ and $d$ are as in the discussion above, and $e$ is the number of failures (i.e., $d^n$ minus the number of successfully computed solutions, as in subsection 5.2). For `phc -u`, the column indexed by "#" gives the minimum and maximum number of steps on a path, and the column indexed by $h$ gives the ratio of the number of steps for which $\Delta t = \Delta t_1$ is the first candidate stepsize. In this experiment, we took $L = 5, M = 1$, and we set the maximum stepsize to be 0.5. Note that even for this type of generic system, the "difficulty" of the paths (based on the number of steps needed) can vary widely. The case $n = 1, d = 300$ is not supported by `HC.jl`, because only one byte is used to represent the degree. Note that `HC.jl` performs extremely well in all other cases in this experiment in terms of both speed and robustness. The extra comparative experiment in the next subsection will show that, for difficult (nongeneric) paths, our heuristic shows better results (this was also shown in subsections 5.1 and 5.2).

**5.4. Clustered solutions.** Homotopies that cause danger for path jumping are such that for some parameter value $t^*$ on the path, the map $H(x, t^*)$ is a polynomial system with some solutions that are clustered together. Motivated by this, we construct the following experiment. Let $n_c$ be a parameter representing the number of solution clusters, and let CS represent the "cluster size." We consider the set of clusters $\{C_1, \ldots, C_{n_c}\}$ where $C_i = \{z_{i,1}, \ldots, z_{i,\mathrm{CS}}\} \subset \mathbb{C}$ is a set of complex numbers that are "clustered" in the following sense. Take $c_i = e^{\frac{i-1}{n_c} 2\pi\sqrt{-1}}$, and for a real parameter $\alpha$, we define $z_{i,j} = c_i + \alpha u^{1/\mathrm{CS}} e^{\frac{j-1}{\mathrm{CS}} 2\pi\sqrt{-1}}$, where $u$ is the unit roundoff ($\approx 10^{-16}$ in double precision arithmetic). Define the polynomial $E(x) = \prod_{i=1}^{n_c} (\prod_{j=1}^{\mathrm{CS}} (x - z_{i,j}))$. For $\alpha = 1$, we know from classical perturbation theory of univariate polynomials that the roots of $E(x)$ look like the roots of a slightly perturbed version of a polynomial whose $n_c$ roots are the cluster centers, which have multiplicity CS. We will use $\alpha \geq 10$, such that the roots of $E(x)$ are not "numerically singular." Let $d = n_c \mathrm{CS}$. Let $G(x) = x^d - 1$, and let $F(x)$ be a polynomial of degree $d$ with random complex coefficients, with real and imaginary parts drawn from a standard normal distribution. We consider the homotopy

$$H(x,t) = (1-t)(1/2-t)G(x) + \gamma_1 t(1-t)E(x) + \gamma_2 t(1/2-t)F(x), \quad t \in [0,1],$$

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

TABLE 4
*Results for $n_c = 5$.*

| $\alpha$ | Solver \ CS | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 10 | HC.jl | **1.0** | 0.740 | 0.100 | 0.060 | 0.080 |
| | Padé.jl | **1.0** | **0.990** | **0.993** | **0.995** | **0.988** |
| 100 | HC.jl | **1.0** | **1.0** | 0.627 | **0.985** | 0.980 |
| | Padé.jl | **1.0** | **1.0** | **1.0** | **0.985** | **0.996** |
| 1000 | HC.jl | **1.0** | **1.0** | **1.0** | **1.0** | **1.0** |
| | Padé.jl | **1.0** | **1.0** | 0.987 | **1.0** | **1.0** |

TABLE 5
*Results for $n_c = 10$.*

| $\alpha$ | Solver \ CS | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 10 | HC.jl | **1.0** | 0.095 | 0.083 | 0.078 | 0.504 |
| | Padé.jl | **1.0** | **0.995** | **1.0** | **1.0** | **0.990** |
| 100 | HC.jl | **1.0** | 0.530 | 0.673 | 0.982 | **1.0** |
| | Padé.jl | **1.0** | **1.0** | **0.997** | **0.988** | **1.0** |
| 1000 | HC.jl | **1.0** | **0.995** | 0.990 | **1.0** | 0.310 |
| | Padé.jl | **1.0** | **0.995** | **0.997** | **1.0** | **0.992** |

where $\gamma_1$ and $\gamma_2$ are random complex constants. $G(x)$ represents the start system with starting solutions the $d$th roots of unity. By tracking the homotopy $H$, the polynomial $G(x)$ is continuously transformed into the random polynomial $F(x)$, passing through the polynomial $(\gamma_1/4)E(x)$ (for $t^* = 1/2$) with clustered solutions. The *success rate* (SR) of a numerical path tracker for solving this problem is defined as follows. Let $\hat{d}$ be the number of points among the solutions of $F(x)$ that coincide with a point returned by the path tracker up to a certain tolerance (e.g., $10^{-6}$). We set SR $= \hat{d}/d$. For fixed $\alpha, n_c$, CS and track 10 homotopies $H(x,t)$ constructed as above with different random $\gamma_i$ using `HC.jl` and `Padé.jl`. We compute the average success rate for these 10 runs. Results are reported in Tables 4 and 5. For each problem, the best average success rate is highlighted in bold print.

**5.5. Benchmark problems.** Parallel computations were applied for the problems in this section. For two families of structured polynomial systems, our experiments show that no path failures and no path jumpings occur, even when the number of solution paths goes past one million.

**5.5.1. Hardware and software.** The program for the experiments is available in the MPI folder of PHCpack, available in its source code distribution on github, under the current name `mpi2padcon`. The code was executed on two 22-core 2.2 GHz Intel Xeon E5-2699 processors in a CentOS Linux workstation with 256 GB RAM. The number of processes for each run equals 44. The root node manages the distribution of the start solutions and the collection of the end paths. In a static workload assignment, each of the other 43 processes tracks the same number of paths.

**5.5.2. The `katsura-`*n* systems.** The `katsura` family of systems is named after the problem posed by Katsura [31]; see [32] for a description of its relevance to applications. The `katsura-`$n$ problem consists of $n$ quadratic equations and one linear equation. The number of solutions equals $2^n$, the product of the degrees of all polynomials in the system.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

Table 6 summarizes the characteristics and wall clock times on `katsura`-$n$ for $n$ ranging from 12 to 20. While the times with HOM4PS-2.0para [34] are much faster than those in Table 6, Table 3 of [34] reports two and four curve jumpings, respectively, for `katsura`-19 and `katsura`-20. In the runs with the MPI version for our code, no path failures and no path jumpings occurred.

The good results we obtained required the use of homogeneous coordinates. When tracking the paths first in affine coordinates, we observed large values for the coordinates, which forced too small stepsizes, which then resulted in path failures.

Although the defining equations are nice quadrics, the condition numbers of the solutions gradually increase as $n$ grows. For example, for $n = 20$, the largest condition number of the Jacobian matrix was of the order $10^7$, observed for 66 solutions. Table 6 reports the number of real solutions in the column with header #real and the number of solutions with nonzero imaginary part under the header #imag.

Table 6

*Wall clock time on* 44 *processes on the* `katsura` *problem, in a static workload balancing schedule with one manager node and* 43 *worker nodes. Only the workers track solution paths.*

| $n$ | #sols | #real | #imag | Wall clock time (seconds) | |
|---|---|---|---|---|---|
| 12 | 4,096 | 582 | 3,514 | 7.925e+1 | 1m 19s |
| 13 | 8,192 | 900 | 7,292 | 2.081e+2 | 3m 28s |
| 14 | 16,384 | 1,606 | 14,778 | 5.065e+2 | 8m 27s |
| 15 | 32,768 | 2,542 | 30,226 | 1.456e+3 | 24m 16s |
| 16 | 65,536 | 4,440 | 61,096 | 4.156e+3 | 1h 9m 16s |
| 17 | 131,072 | 7,116 | 123,956 | 1.001e+4 | 2h 46m 50s |
| 18 | 262,144 | 12,458 | 249,686 | 2.308e+4 | 6h 24m 15s |
| 19 | 524,288 | 20,210 | 504,078 | 5.696e+4 | 15h 49m 20s |
| 20 | 1,048,576 | 35,206 | 1,013,370 | 1.317e+5 | 36h 34m 11s |

The progression of the wall clock times in Table 6 illustrates that our new path tracking algorithm scales well for increasing dimensions despite the $O(n^4)$ factor in its cost.

**5.5.3. A model of a neural network.** An interesting class of polynomial systems [43] was introduced to the computer algebra community by [21]. The $n$-dimensional system consists of $n$ cubic equations and originated from a model of a neural network. A linear-product root bound provides a sharp root count. Although the permutation symmetry could be exploited, with a symmetric homotopy using the algorithms in [61], this did not happen for the computations summarized in Table 7. Homogeneous coordinates were also applied in the runs. The formulation of the polynomials in [43] depends on one parameter $c$, which was set to 1.1. The number of real solutions is reported in Table 7 in the column with header #real, and the number of solutions with nonzero imaginary part is under the header #imag.

Because every new equation is of degree three and the number of paths triples, the wall clock time increases more than in the previous benchmark. As before, no path failures and no path jumpings occurred.

**6. Conclusion and future work.** We have proposed an adaptive stepsize predictor algorithm for numerical path tracking in polynomial homotopy continuation. The resulting algorithm can be used to solve challenging problems successfully using only double precision arithmetic and is competitive with existing software. An implementation is available in PHCpack (available on github). It is expected that analogous techniques can be used to track paths that contain singular points for $t \in [0, 1)$, to compute monodromy groups, and to design efficient new end games for dealing with

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

TABLE 7

*Wall clock times on 44 processes on polynomial systems modeling a neural network, in a static workload balancing schedule with one manager node and 43 worker nodes. Only the worker nodes track solution paths.*

| $n$ | #sols | #real | #imag | Wall clock time (seconds) | |
|---|---|---|---|---|---|
| 10 | 59,029 | 21 | 59,008 | 3.478e+3 | 57m 58s |
| 11 | 177,125 | 23 | 177,102 | 1.594e+4 | 4h 25m 37s |
| 12 | 531,417 | 25 | 531,392 | 7.202e+4 | 20h 0m 17s |
| 13 | 1,594,297 | 27 | 1,594,270 | 3.030e+5 | 84h 9m 58s |

singular endpoints and solutions at infinity. Another possible direction for future research is investigating whether the methods of this paper can be made certifiable, for instance, by bounding the factors $\beta_1, \beta_2$. One could also choose the parameters $L$ and $M$ based on an analysis of the Padé table at several points on the path. Finally, the use of generalized Padé approximants could speed up the computations [23].

**Appendix A. Computing power series solutions.** In this appendix we discuss the algorithm for computing a power series solution of

$$H(x,t) = (h_1(x,t), \ldots, h_n(x,t))$$

at $t^* = 0$ proposed in [10] and prove a result of convergence. An analogous result for the case $n = 1$ can be found in [36]. We will consider the situation where the series solution has the form (3.1) with parameters satisfying $\omega_i \geq 0$. Furthermore, we assume that the winding number $m$ is known. If this is not the case, $m$ can be computed by using, for instance, monodromy loops. Note that it is sufficient to consider the case where $m = 1$, since if $m$ is known and $m > 1$, we can consider the homotopy $\hat{H}(x,\tau) = (h_1(x,\tau^m), \ldots, h_n(x,\tau^m))$ with power series solution $x_j(s) = a_j s^{\omega_j}\left(1 + \sum_{\ell=1}^{\infty} a_{j\ell} s^{\ell}\right), j = 1, \ldots, n$, and $\tau(s) = s$. Therefore, we can avoid introducing the extra parameter $s$, and the unknown power series solution is given by

$$x_j(t) = a_j t^{\omega_j}\left(1 + \sum_{\ell=1}^{\infty} a_{j\ell} t^{\ell}\right),\ j = 1, \ldots, n. \tag{A.1}$$

We think of $H(x,t)$ as a column vector $[h_1 \ \cdots \ h_n]^\top$ in $R[[t]]^n \simeq R^n[[t]]$, and the Jacobian matrix $J_H(x,t)$ is considered an element of $R[[t]]^{n\times n} \simeq R^{n\times n}[[t]]$. For any $h(x,t) \in R[[t]]^n$, plugging in $y(t) \in \mathbb{C}[[t]]^n$ gives $h(y(t),t) \in \mathbb{C}[[t]]^n$, and the same can be done for $J(x,t) \in R[[t]]^{n\times n}$, which gives $J(y(t),t) \in \mathbb{C}[[t]]^{n\times n}$.

DEFINITION A.1. *Let $\star$ be either $\mathbb{C}^n$ or $\mathbb{C}^{n\times n}$. For $v = \sum_{\ell=0}^{\infty} v_\ell t^\ell \in \star[[t]] \setminus \{0\}$, the* order *of $v$ is*

$$\operatorname{ord}(v) = \min_{\ell}\{v_\ell \neq 0\},$$

*where $v_\ell \in \star, \ell \in \mathbb{N}$. For $w \neq v \in \star[[t]]$ we denote $v = w + O(t^k)$ if $\operatorname{ord}(v-w) \geq k$. For $v = 0$, we define $\operatorname{ord}(v) = \infty$.*

Note that this means that for a vector or matrix $v$ with power series entries, $v = O(t^k)$ if and only if every entry of $v$ is in $\mathfrak{m}^k$, where $\mathfrak{m}$ is the maximal ideal of $\mathbb{C}[[t]]$. With elementwise addition and multiplication in $\mathbb{C}[[t]]^n$ and the usual addition and multiplication in $\mathbb{C}[[t]]^{n\times n}$, it is clear that for $v, w \in \star[[t]]$, $\operatorname{ord}(v) = \operatorname{ord}(-v)$, $\operatorname{ord}(v+w) \geq \min(\operatorname{ord}(v), \operatorname{ord}(w))$, and $\operatorname{ord}(vw) \geq \operatorname{ord}(v) + \operatorname{ord}(w)$. For the product rule,

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

equality holds if $\star = \mathbb{C}^n$. Matrix-vector multiplication $\mathbb{C}[[t]]^{n\times n} \times \mathbb{C}[[t]]^n \to \mathbb{C}[[t]]^n$ is defined in the usual way, and for $M \in \mathbb{C}[[t]]^{n\times n}, v \in \mathbb{C}[[t]]^n$ we have $\mathrm{ord}(Mv) \geq \mathrm{ord}(M) + \mathrm{ord}(v)$.

Given $x^{(0)}(t) = (x_1^{(0)}(t), \ldots, x_n^{(0)}(t)) \in \mathbb{C}[[t]]^n$, fix positive integers $w_k \in \mathbb{N} \setminus \{0\}$, and consider the sequence $\{x^{(k)}(t)\}_{k\geq 0}$ defined by

$$\begin{aligned} \tilde{x}^{(k+1)}(t) &= x^{(k)}(t) - J_H(x^{(k)}(t), t)^{-1} H(x^{(k)}(t), t) = \sum_{\ell=0}^{\infty} b_\ell t^\ell, \\ x^{(k+1)}(t) &= \sum_{\ell=0}^{w_k - 1} b_\ell t^\ell, \end{aligned} \tag{A.2}$$

where we assume that $J_H(x^{(k)}(t), t)$ is a unit in $\mathbb{C}[[t]]^{n\times n}$ for all $k$, and this is equivalent to assuming that $J_H(x^{(k)}(0), 0) \in \mathrm{GL}(n, \mathbb{C})$ for all $k \geq 0$. The iteration is clearly based on the well-known Newton–Raphson iteration for approximating a root of a nonlinear system of equations. The following proposition specifies the statement that the iteration has similar "quadratic" convergence properties. It is related to a multivariate version of Hensel lifting; see, for instance, [19, Exercise 7.26].

PROPOSITION A.2. *Let $H(x, t) : X \times \mathbb{C} \to \mathbb{C}$ be a homotopy with power series solution $x(t) \in \mathbb{C}[[t]]^n$ given by* (A.1)*, and let $\{x^{(k)}(t)\}_{k\geq 0}$ be a sequence generated as in* (A.2)*. If $J_H(x^{(k)}(t), t)$ is a unit in $\mathbb{C}[[t]]^{n\times n}$ for all $k \geq 0$, then*

$$\mathrm{ord}(x^{(k+1)}(t) - x(t)) \geq \min(2\mathrm{ord}(x^{(k)}(t) - x(t)), w_k), \quad k \geq 0.$$

*Proof.* We know that $x(t) = (x_1(t), \ldots, x_n(t))^\top \in \mathbb{C}[[t]]^n$ satisfies $H(x(t), t) = 0$. Take $x^{(k)}(t) \in \mathbb{C}[[t]]^n$ and define $e^{(k)}(t) = x^{(k)}(t) - x(t)$. We have

$$0 = H(x^{(k)}(t) - e^{(k)}(t), t) = H(x^{(k)}(t), t) - J_H(x^{(k)}(t), t)e^{(k)}(t) + O(t^{2\mathrm{ord}(e^{(k)}(t))}). \tag{A.3}$$

By assumption, $J_H(x^{(k)}(t), t)$ is a unit, and thus $\mathrm{ord}(J_H(x^{(k)}(t), t)^{-1}) = 0$. We now multiply (A.3) from the left with $J_H(x^{(k)}(t), t)^{-1}$, and we get (using $e^{(k)}(t) = x^{(k)}(t) - x(t)$)

$$-J_H(x^{(k)}(t), t)^{-1} H(x^{(k)}(t), t) + (x^{(k)}(t) - x(t)) = O(t^{2\mathrm{ord}(e^{(k)}(t))}).$$

It follows that $\tilde{x}^{(k+1)}(t) - x(t) = O(t^{2\mathrm{ord}(e^{(k)}(t))})$. So we find that

$$\mathrm{ord}(e^{(k+1)}(t)) \geq \min(2\mathrm{ord}(e^{(k)}(t)), w_k). \qquad \square$$

It follows that if $e^{(0)}(t)$ has order $\geq 1$, the iteration converges to the solution $x(t)$, and the order of the error doubles in every iteration, as long as the truncation orders $w_k$ allow for it. Also, if $\mathrm{ord}(e^{(0)}(t)) \geq 1$, $H(x^{(0)}(0), 0) = 0$, and thus $\mathrm{ord}(H(x^{(0)}(t), t)) \geq 1$. It follows that the term $-J_H(x^{(k)}(t), t)^{-1} H(x^{(k)}(t), t)$ has order at least 1, and so the constant terms of $x^{(1)}$ and $x^{(0)}$ agree. This holds true for the following iterations as well. We conclude that if $\mathrm{ord}(e^{(0)}(t)) \geq 1$, the assumption that $J_H(x^{(k)}(t), t)$ is a unit for all $k$ translates to the assumption that $x^{(0)}(0) = a$ is a regular solution of the polynomial system defined by $H_0$. If we want to compute a series solution that is accurate up to order $w$, and $\mathrm{ord}(e^{(0)}(t)) = r \geq 1$, we set $w_k = \min(r2^k, w)$ and execute $\lceil \log_2(w/r) \rceil$ steps of the iteration. This can be done by solving a lower triangular block Toeplitz system of linear equations. For details, we refer the reader to [10].

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

**Acknowledgments.** We would like to thank the anonymous referees for their useful suggestions and comments on an earlier version of this paper.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

REFERENCES

[1] E. L. Allgower and K. Georg, *Introduction to Numerical Continuation Methods*, Classics in Applied Mathematics 45, SIAM, Philadelphia, 2003, https://doi.org/10.1137/1.9780898719154.

[2] J. L. Aurentz, T. Mach, R. Vandebril, and D. S. Watkins, *Fast and backward stable computation of roots of polynomials*, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 942–973, https://doi.org/10.1137/140983434.

[3] G. A. Baker, Jr. and P. Graves-Morris, *Padé Approximants*, 2nd ed., Encyclopedia Math. Appl. 59, Cambridge University Press, 1996.

[4] D. J. Bates, J. D. Hauenstein, A. J. Sommese, and C. W. Wampler II, *Adaptive multiprecision path tracking*, SIAM J. Numer. Anal., 46 (2008), pp. 722–746, https://doi.org/10.1137/060658862.

[5] D. J. Bates, A. J. Sommese, J. D. Hauenstein, and C. W. Wampler, *Numerically Solving Polynomial Systems with Bertini*, Software Environments Tools 25, SIAM, Philadelphia, 2013, https://doi.org/10.1137/1.9781611972702.

[6] A. Beardon, *On the location of poles of Padé approximants*, J. Math. Anal. Appl., 21 (1968), pp. 469–474.

[7] B. Beckermann and A. C. Matos, *Algebraic properties of robust Padé approximants*, J. Approx. Theory, 190 (2015), pp. 91–115.

[8] C. Beltrán and A. Leykin, *Robust certified numerical homotopy tracking*, Found. Comput. Math., 13 (2013), pp. 253–295.

[9] L. Bieberbach, *Analytische Fortsetzung*, Ergeb. Math. Grenzgeb. 3, Springer-Verlag, Berlin, 1955.

[10] N. Bliss and J. Verschelde, *The method of Gauss–Newton to compute power series solutions of polynomial homotopies*, Linear Algebra Appl., 542 (2018), pp. 569–588.

[11] P. Breiding and S. Timme, *HomotopyContinuation.JL: A package for homotopy continuation in Julia*, in International Congress on Mathematical Software, Springer, Berlin, 2018, pp. 458–465.

[12] R. P. Brent and H. T. Kung, *Fast algorithms for manipulating formal power series*, J. ACM, 25 (1978), pp. 581–595.

[13] P. Bürgisser and F. Cucker, *Condition: The Geometry of Numerical Algorithms*, Grund. Math. Wiss. 349, Springer-Verlag, Berlin, 2013.

[14] B. Christianson, *Automatic Hessians by reverse accumulation*, IMA J. Numer. Anal., 12 (1992), pp. 135–150.

[15] D. A. Cox, J. Little, and D. O'Shea, *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*, Undergrad. Texts Math., Springer-Verlag, Berlin, 1992.

[16] R. de Montessus, *Sur les fractions continues algébriques*, Bull. Soc. Math. France, 30 (1902), pp. 28–36.

[17] J. W. Demmel, *Applied Numerical Linear Algebra*, SIAM, Philadelphia, 1997, https://doi.org/10.1137/1.9781611971446.

[18] P. Dienes, *The Taylor Series: An Introduction to the Theory of Functions of a Complex Variable*, Dover, New York, 1957.

[19] D. Eisenbud, *Commutative Algebra: With a View Toward Algebraic Geometry*, Grad. Text Math. 150, Springer, New York, 2013.

[20] E. Fabry, *Sur les points singuliers d'une fonction donnée par son développement en série et l'impossibilité du prolongement analytique dans des cas très généraux*, in Annales Scientifiques de l'École Normale Supérieure, 3e série, vol. 13, Elsevier, 1896, pp. 367–399.

[21] K. Gatermann, *Symbolic solution of polynomial equation systems with symmetry*, in Proceedings of ISSAC-90 (Tokyo, Japan, 1990), S. Watanabe and M. Nagata, eds., ACM, New York, 1990, pp. 112–119.

[22] J.-J. Gervais and H. Sadiky, *A continuation method based on a high order predictor and an adaptive steplength control*, ZAMM Z. Angew. Math. Mech., 84 (2004), pp. 551–563.

[23] A. A. Gončar, *On the convergence of generalized Padé approximants of meromorphic functions*, Math. USSR-Sb., 27 (1975), p. 503–514 (in Russian).

[24] A. A. Gončar, *Poles of rows of the Padé table and meromorphic continuation of functions*, Mat. Sb., 157 (1981), pp. 590–613 (in Russian).

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

[25] P. Gonnet, S. Güttel, and L. N. Trefethen, *Robust Padé approximation via SVD*, SIAM Rev., 55 (2013), pp. 101–117, https://doi.org/10.1137/110853236.
[26] A. Griewank and A. Walther, *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*, SIAM, Philadelphia, 2008, https://doi.org/10.1137/1.9780898717761.
[27] B. Huber and B. Sturmfels, *A polyhedral method for solving sparse polynomial systems*, Math. Comp., 64 (1995), pp. 1541–1555.
[28] B. Huber and J. Verschelde, *Polyhedral end games for polynomial continuation*, Numer. Algorithms, 18 (1998), pp. 91–108.
[29] O. Ibryaeva and V. Adukov, *An algorithm for computing a Padé approximant with minimal degree denominator*, J. Comput. Appl. Math., 237 (2013), pp. 529–541.
[30] G. Jeronimo, G. Matera, P. Solernó, and A. Waissbein, *Deformation techniques for sparse systems*, Found. Comput. Math., 9 (2009), pp. 1–50.
[31] S. Katsura, *Users posing problems to PoSSO*, PoSSO Newsletter, no. 2, July 1994.
[32] S. Katsura, *Spin glass problem by the method of integral equation of the effective field*, in New Trends in Magnetism, M. Coutinho-Filho and S. Resende, eds., World Scientific, Singapore, 1990, pp. 110–121.
[33] R. B. Kearfott and Z. Xing, *An interval step control for continuation methods*, SIAM J. Numer. Anal., 31 (1994), pp. 892–914, https://doi.org/10.1137/0731048.
[34] T. Li and C. Tsai, *HOM4PS-2.0para: Parallelization of HOM4PS-2.0 for solving polynomial systems*, Parallel Comput., 35 (2009), pp. 226–238.
[35] T.-Y. Li, *Numerical solution of multivariate polynomial systems by homotopy continuation methods*, Acta Numer., 6 (1997), pp. 399–436.
[36] J. D. Lipson, *Newton's method: A great algebraic algorithm*, in Proceedings of the third ACM Symposium on Symbolic and Algebraic Computation, ACM, New York, 1976, pp. 260–270.
[37] W. D. M. Millan, *A method for determining the solutions of a system of analytic functions in the neighborhood of a branch point*, Math. Ann., 72 (1912), pp. 180–202.
[38] J. Maurer, *Puiseux expansion for space curves*, Manuscripta Math., 32 (1980), pp. 91–100.
[39] A. Morgan, *Solving Polynomial Systems Using Continuation for Engineering and Scientific Problems*, Classics Appl. Math. 57, SIAM, Philadelphia, 2009, https://doi.org/10.1137/1.9780898719031.
[40] A. P. Morgan, A. J. Sommese, and C. W. Wampler, *Computing singular solutions to nonlinear analytic systems*, Numer. Math., 58 (1990), pp. 669–684.
[41] A. P. Morgan, A. J. Sommese, and C. W. Wampler, *A power series method for computing singular solutions to nonlinear analytic systems*, Numer. Math., 63 (1992), pp. 391–409.
[42] Y. Nakatsukasa, O. Sète, and L. N. Trefethen, *The AAA algorithm for rational approximation*, SIAM J. Sci. Comput., 40 (2018), pp. A1494–A1522, https://doi.org/10.1137/16M1106122.
[43] V. W. Noonburg, *A neural network modeled by an adaptive Lotka-Volterra system*, SIAM J. Appl. Math., 49 (1989), pp. 1779–1792, https://doi.org/10.1137/0149109.
[44] K. Piret and J. Verschelde, *Sweeping algebraic curves for singular solutions*, J. Comput. Appl. Math., 234 (2010), pp. 1228–1237.
[45] H. Schwetlick and J. Cleve, *Higher order predictors and adaptive steplength control in path following algorithms*, SIAM J. Numer. Anal., 24 (1987), pp. 1382–1393, https://doi.org/10.1137/0724089.
[46] A. J. Sommese, J. Verschelde, and C. W. Wampler, *Numerical decomposition of the solution sets of polynomial systems into irreducible components*, SIAM J. Numer. Anal., 38 (2001), pp. 2022–2046, https://doi.org/10.1137/S0036142900372549.
[47] A. J. Sommese, J. Verschelde, and C. W. Wampler, *Introduction to numerical algebraic geometry*, in Solving Polynomial Equations, Springer, Berlin, 2005, pp. 301–337.
[48] A. J. Sommese and C. W. Wampler, *The Numerical Solution of Systems of Polynomials Arising in Engineering Science*, World Scientific, Singapore,2005.
[49] H. Stahl, *The convergence of Padé approximants to functions with branch points*, J. Approx. Theory, 91 (1997), pp. 139–204.
[50] S. P. Suetin, *On an inverse problem for the m-th row of the Padé table*, Math. USSR-Sb., 52 (1985), pp. 231–244.
[51] S. P. Suetin, *Padé approximants and efficient analytic continuation of a power series*, Russian Math. Surveys, 57 (2002), pp. 43–141.
[52] S. Telen and M. Van Barel, *A stabilized normal form algorithm for generic systems of polynomial equations*, J. Comput. Appl. Math., 342 (2018), pp. 119–132.
[53] S. Timme, *Adaptive Step Size Control for Polynomial Homotopy Continuation Methods*, preprint, http://arxiv.org/abs/1902.02968v1, 2019.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

[54] L. N. Trefethen, *Approximation Theory and Approximation Practice*, SIAM, Philadelphia, 2019, https://doi.org/10.1137/1.9781611975949.
[55] L. N. Trefethen, *Quantifying the ill-conditioning of analytic continuation*, BIT, to appear (published online January 28, 2020).
[56] A. Trias, *The holomorphic embedding load flow method*, in Proceedings of the 2012 IEEE Power and Energy Society General Meeting, IEEE, Piscataway, NJ, 2012, pp. 1–8.
[57] A. Trias and J. L. Martin, *The holomorphic embedding loadflow method for DC power systems and nonlinear DC circuits*, IEEE Trans. Circuits Systems, 63 (2016), pp. 322–333.
[58] J. van der Hoeven, *Reliable Homotopy Continuation*, Tech. report hal-00589948v4f, LIX, École Polytechnique, 2015.
[59] V. V. Vavilov, V. A. Prokhorov, and S. P. Suetin, *The poles of the m-th row of the Padé table and the singular points of a function*, Math. USSR-Sb., 50 (1985), pp. 457–463.
[60] J. Verschelde, *Algorithm* 795*: PHCpack: A general-purpose solver for polynomial systems by homotopy continuation*, ACM Trans. Math. Software, 25 (1999), pp. 251–276.
[61] J. Verschelde and R. Cools, *Symmetric homotopy construction*, J. Comput. Appl. Math., 50 (1994), pp. 575–592.
[62] J. Verschelde, P. Verlinden, and R. Cools, *Homotopies exploiting Newton polytopes for solving sparse polynomial systems*, SIAM J. Numer. Anal., 31 (1994), pp. 915–930, https://doi.org/10.1137/0731049.
[63] D. Wu and B. Wang, *A Holomorphic Embedding Based Continuation Method for Identifying Multiple Power Flow Solutions*, preprint, https://arxiv.org/abs/1905.01602v1, 2019.
[64] J. Xu, M. Burr, and C. Yap, *An approach for certifying homotopy continuation paths: Univariate case*, in Proceedings of the 2018 ACM International Symposium on Symbolic and Algebraic Computation (ISSAC '18), 2018, ACM, New York, pp. 399–406.

Downloaded 11/10/20 to 128.248.156.45. Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.