SIAM J. NUMER. ANAL.
Vol. 51, No. 6, pp. 3373–3401

CERTIFIED PARALLELOTOPE CONTINUATION FOR ONE-MANIFOLDS*

BENJAMIN MARTIN†, ALEXANDRE GOLDSZTEJN†, LAURENT GRANVILLIERS†, AND CHRISTOPHE JERMANN†

Abstract. Starting from an initial solution, continuation methods efficiently produce a sequence of points on a manifold typically defined as the solution set of an underconstrained system of equations. They have a wide range of applications ranging from curve plotting to polynomial root-finding by homotopy. However, classical methods cannot guarantee that the returned points all belong to the same connected component of the manifold, i.e., they may *jump* from one component to another. Trying to overcome this issue has given birth to several sophisticated heuristics on the one hand and to guaranteed methods based on rigorous computations on the other hand. In this paper we introduce a new rigorous predictor corrector continuation method based on interval computations. Its novelty lies in the fact that it uses parallelotopes as defined in A. Goldsztejn and L. Granvilliers, *A new framework for sharp and efficient resolution of* NCSP *with manifolds of solutions*, Constraints, 15 (2010), pp. 190–212, to enclose consecutive portions of the followed manifold. Though computationally more expensive than regular interval computations, the fact that their orientation can be automatically adapted to the local topology of the followed curve makes parallelotopes a very suitable tool for a certified continuation method, as shown by reported experimental results.

Key words. certified continuation, interval analysis, parallelotope domains

AMS subject classifications. 65G40, 65G20, 65H10, 65H20

DOI. 10.1137/130906544

1. Introduction.

Continuation methods [1] allow exploring step by step a solution manifold, usually defined as the solution set of an underconstrained system of equations $F(x) = 0$ with $F : \mathbb{R}^n \to \mathbb{R}^m$ and $m < n$. This paper focuses on systems where $n - m = 1$, whose solution manifolds are curves, and proposes a robust single-parameter continuation method. It has a wide range of applications, e.g., polynomial root finding via homotopy [3], nonlinear eigenvalue problems [4], biobjective optimization [18, 33], and robot path planning [24]. The most simple and effective continuation method is the predictor corrector algorithm. It includes the simple embedding method and several of its improvements such as the parameter switching [28, 29] and the pseudo-arclength methods that have the definite advantage over the simple embedding method that they naturally track folds [6]. They are all, however, subject to jumping between disconnected components without any prompt. For a given system, there exist some theoretical upper bounds on the step size that enforce the connectivity of the continuation, e.g., Theorem 5.2.1 in [1], but they are impractical. While this may be acceptable in some contexts since solutions are eventually computed, such jumps contradict the essence of continuation. Furthermore, connectivity is mandatory in some applications, e.g., robot command synthesis, where such jumps would result in a nonfeasible command path, or in homotopy methods, where such jumps would prevent computing all the solutions of the original system.

*Received by the editors January 18, 2013; accepted for publication (in revised form) September 16, 2013; published electronically December 17, 2013.
http://www.siam.org/journals/sinum/51-6/90654.html
†Laboratoire Informatique de Nantes Atlantique, Université de Nantes, Nantes 44300, France (benjamin.martin@univ-nantes.fr, alexandre.goldsztejn@univ-nantes.fr, Laurent.Granvilliers@univ-nantes.fr, Christophe.Jermann@univ-nantes.fr).

Several methods have been proposed to certify the connectivity between continuation steps. Smale's α-theory [34] has been used in [3] to derive some maximal step size that certifies the connectivity of a predictor corrector continuation in a homotopy method for computing roots of square polynomial systems in the fields of complex numbers. (It is implemented in the software Macaulay2 [16].) A simplified Kantorovich theorem argument has been proposed and used in [7] in order to derive a maximal step size that certifies the connectivity of a general predictor corrector continuation. It is mainly devoted to quadratic systems of low dimension: While this maximal step size has a simple expression for quadratic systems, it requires heavy formal manipulations for nonquadratic systems. In addition, this maximal step size decreases quadratically with the dimension of the problem, restricting it to low dimensional systems.[1] No overall algorithm or implementation certifying the complete curve connectivity is proposed in [7]. Note that Smale's α-theory and the Kantorovich theorem are known to be closely related [30]. In the more general context of multi-parameter continuation, the reach of a manifold[2] is used in [5] in order to build an isotopic triangulation of the manifold. Although this theoretical framework is not exactly a continuation method, it is close in spirit and it adds connectivity certification to the multiparameter continuation method proposed in [17]. No implementation of [5] is available.

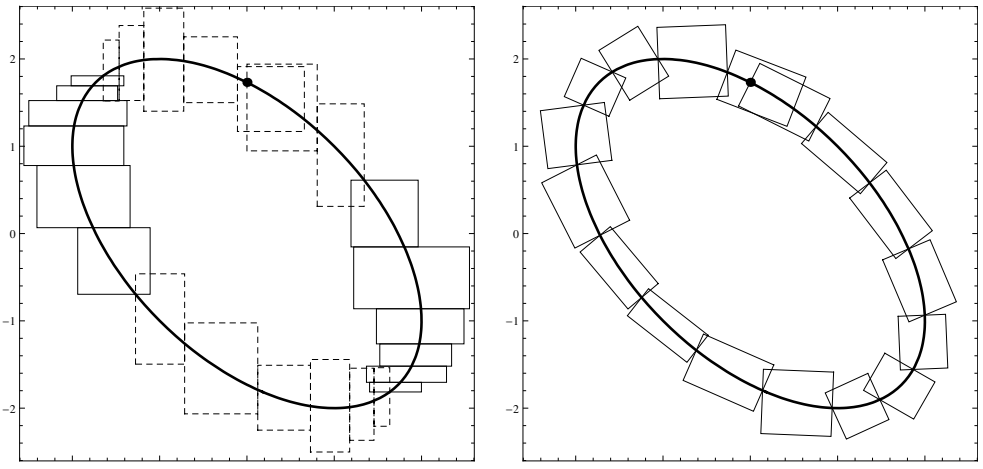
Interval analysis (IA; see [26] and section 2) allows certifying the existence of solutions to systems of equations through the verification of the hypothesis of existence theorems, e.g., the Brouwer fixed point theorem, the Poincaré–Miranda theorem, or the interval Newton existence test. A continuation method with certified connectivity based on IA was investigated in [22], where an embedding method coupled with a strategy of parameter switching was proposed to build a sequence of boxes that encloses the solution path. At each step, a heuristic inflation process allows a parametric version of the interval Newton test to prove the existence of a path along one chosen direction; see Figure 1.1(a), where solid-line boxes certify that the path goes from their bottom side to their top side, while dashed-line boxes certify that the path goes from their left side to their right side. The algorithm proposed in [22] involves several technical hypotheses to be verified, making this algorithm difficult to implement and tune. Indeed, the authors of the present paper could not implement successfully the algorithm proposed in [22], so Figure 1.1(a) actually shows the results of the hybridization of [22] with techniques proposed here.

The main drawback of [22] is intrinsic to the usage of boxes, which do not accurately capture the shape of the manifold. This key issue was tackled in [11], where parallelotopes are used instead of boxes. This allows applying a parametric interval Newton test in an auxiliary basis that better fits the manifold orientation. In [11], parallelotope computations were used in a branch and prune framework that globally explores a manifold, entailing a computational complexity exponential in the number of variables, as is the case of any global search in nonlinear problems. Continuation potentially allows drastically reducing this complexity by instead following locally the manifold, leading to a polynomial time complexity for fixed manifold dimension.

Computations with parallelotopes are used in section 3 to define a continuation algorithm with certified existence and connectivity. The continuation step consists

[1] This maximal step size is proportional to the inverse of the entrywise sum of the absolute value of the system's Hessian, which decreases quadratically with the dimension for nonsparse systems leading to dramatically small step size.

[2] The reach of a manifold has been introduced in [8] and characterizes the size of a neighborhood of the manifold within which any point has a unique projection on the manifold. Roughly speaking, the manifold has a simple behavior within this neighborhood.



(a) Using box domains (b) Using parallelotope domains

FIG. 1.1. *Curve enclosure with certified connectivity. (The initial approximate solution used for the certified continuation is depicted by a point.)*

of an intervalization of the predictor corrector process, where a parametric interval Newton operator is used to build a parallelotope that contains one unique portion of the curve. Continuation steps are interleaved with several checks that ensure the connectivity and the absence of backtrack, while tracking the presence of a loop and certifying it when the curve is actually periodic. The result of this process is illustrated in Figure 1.1(b). Even on this simple example, parallelotopes show a better adaptation to the curve, which will become a key advantage for harder problems. The simplicity of this algorithm is a strong advantage for its implementation and for studying its properties. (Its correctness, its termination, and its asymptotic convergence are investigated in subsection 3.2.) Experimental results presented in section 4 show a drastic efficiency improvement with respect to an interval embedding method with parameter switching close to the one proposed in [22] and demonstrate that the proposed method is quite competitive with respect to the Macaulay2 [16] implementation of [3] when applied to homotopy continuation for complex polynomial root finding.

2. Interval Analysis. IA is a modern branch of numerical analysis born in the 1960s [25]. It replaces computations with real numbers by computations with intervals of real numbers, providing a framework for handling uncertainties and verified computations (see [26, 20] and [21] for a survey).

2.1. Basic definitions and operators. An interval is a closed connected subset of $\mathbb{R}$. Intervals are denoted by boldface symbols, e.g., $\boldsymbol{x} \subseteq \mathbb{R}$. When no confusion is possible, lower and upper bounds of an interval $\boldsymbol{x}$ are denoted by $\underline{x} \in \mathbb{R}$ and $\overline{x} \in \mathbb{R}$ with $\underline{x} \leq \overline{x}$, i.e., $\boldsymbol{x} = [\underline{x}, \overline{x}] = \{x \in \mathbb{R} : \underline{x} \leq x \leq \overline{x}\}$. Furthermore, a real number x will be identified with the degenerated interval $[x, x]$. Interval vectors, also called boxes, are equivalently defined as vectors of intervals $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) \in \mathbb{IR}^n$ with $\boldsymbol{x}_i \in \mathbb{IR}$, or as intervals of vectors $\boldsymbol{x} = [\underline{x}, \overline{x}] \in \mathbb{IR}^n$ with $\underline{x}, \overline{x} \in \mathbb{R}^n$. Similarly, interval matrices are equivalently defined as matrices of intervals or intervals of matrices: $\boldsymbol{A} = (\boldsymbol{a}_{ij})_{1 \leq i \leq n, 1 \leq j \leq m} = [\underline{A}, \overline{A}] \in \mathbb{IR}^{n \times m}$.

The interval hull of a set $E \subseteq \mathbb{R}^n$, denoted by $\square E$, is the smallest box that contains this set, e.g., $\square\{x \in \mathbb{R}^2 : x_1^2 + x_2^2 \leq 1\} = ([-1, 1], [-1, 1])^T$. The interval hull

of two sets E_1 and E_2 is denoted by $E_1 \vee E_2$ and is by definition the smallest box that contains both E_1 and E_2. The center of an interval (or similarly of a box or of an interval matrix) $\boldsymbol{x}$ is $\mathrm{mid}(\boldsymbol{x}) := 0.5(\underline{x} + \overline{x})$, and its magnitude is $|\boldsymbol{x}| = \max\{|\underline{x}|, |\overline{x}|\}$. Unless stated explicitly, the midpoint is computed approximately using floating point operations, although it is assumed that $\mathrm{mid}(\boldsymbol{x}) \in \boldsymbol{x}$ holds (which is necessary for its usual usage together with the interval Newton operator). The interior of an interval (or box) $\boldsymbol{x}$ is denoted by $\mathrm{int}(\boldsymbol{x})$ and its width by $\mathrm{wid}(\boldsymbol{x}) = \overline{x} - \underline{x}$. (The width is a real number in case of intervals, a real vector in case of boxes, or a real matrix in case of interval matrices.) Throughout the paper, the infinite norm is used, so, e.g., $\|\mathrm{wid}\boldsymbol{x}\| = \max_i |\overline{x}_i - \underline{x}_i|$.

Operations $\circ \in \{+, \times, -, \div\}$ are extended to intervals in the following way:

$$[\underline{x}, \overline{x}] \circ [\underline{y}, \overline{y}] := \{x \circ y : x \in [\underline{x}, \overline{x}], y \in [\underline{y}, \overline{y}]\}. \tag{2.1}$$

The division is defined for intervals $[\underline{y}, \overline{y}]$ that do not contain zero. Continuous unary elementary functions like exp, ln, sin, etc., are also extended to intervals similarly for interval arguments contained in their domains: $f(\boldsymbol{x}) := \{f(x) : x \in \boldsymbol{x}\}$, which is an interval since f is continuous. All these elementary interval extensions form the interval arithmetic. As real numbers are identified to degenerated intervals, the Interval Arithmetic actually generalizes the real arithmetic, and mixed operations like $1 + [1, 2] = [2, 3]$ are interpreted using (2.1).

A real function can be extended to intervals replacing its elementary operations by their interval counterparts. The fundamental theorem of IA is that the interval evaluation of a real function yields an enclosure of the range of this function over the interval arguments, e.g., $\boldsymbol{x} + \sin(\boldsymbol{x})\,\boldsymbol{y} \supseteq \{x + \sin(x)y : x \in \boldsymbol{x}, y \in \boldsymbol{y}\}$, or $\boldsymbol{AB} \supseteq \{AB : A \in \boldsymbol{A}, B \in \boldsymbol{B}\}$, where $\left(\boldsymbol{AB}\right)_{ij} = \sum_k \boldsymbol{a}_{ik}\boldsymbol{b}_{kj}$. When the expression evaluated for interval arguments contains only one occurrence of each variable, the computed enclosure of the range is optimal, e.g., $\boldsymbol{AB} = \square\{AB : A \in \boldsymbol{A}, B \in \boldsymbol{B}\}$. However, the interval evaluation of expressions that contain several occurrences of some variable is not optimal anymore in general. Nevertheless, under the mild hypothesis that the real function is Lipschitz continuous inside the interval arguments, the overestimation of interval evaluations decreases proportionally to the width of the interval arguments. When interval arguments are small, the mean-value interval extensions usually give rise to better enclosure whose overestimation decreases quadratically with respect to the width of the interval arguments. For a differentiable real function $F : \mathbb{R}^n \to \mathbb{R}^m$, the mean value extension is defined as $\boldsymbol{F}(\tilde{x}) + \boldsymbol{F'}(\boldsymbol{x})(\boldsymbol{x} - \tilde{x}) \supseteq \{F(x) : x \in \boldsymbol{x}\}$, where $\boldsymbol{F}$ and $\boldsymbol{F'}$ are the (usually natural) interval extensions of the function and its derivatives, and $\tilde{x} \in \boldsymbol{x}$.

Solution existence and uniqueness certification. Given a square system of equations $F(x) = 0$, i.e., $F : \mathbb{R}^n \to \mathbb{R}^n$, the Newton operator consists in solving approximately the linearized equation $F(\tilde{x}_k) + F'(\tilde{x}_k)(x_{k+1} - \tilde{x}_k) = 0$. Given a box $\boldsymbol{x}_k$ and $\tilde{x}_k \in \boldsymbol{x}_k$ (usually $\tilde{x}_k = \mathrm{mid}(\boldsymbol{x}_k)$), the interval Newton operator consists in enclosing the solutions to the interval linearization $\boldsymbol{F}(\tilde{x}_k) + \boldsymbol{F'}(\boldsymbol{x}_k)(x_{k+1} - \tilde{x}_k) \ni 0$, giving rise to a new box $\boldsymbol{x}_{k+1}$. It has the following key properties: Every solution contained inside $\boldsymbol{x}_k$ also belongs to $\boldsymbol{x}_{k+1}$ (so if $\boldsymbol{x}_{k+1}$ is empty then $\boldsymbol{x}_k$ is proved to contain no solution); furthermore, if $\emptyset \neq \boldsymbol{x}_{k+1} \subseteq \mathrm{int}(\boldsymbol{x}_k)$, then $\boldsymbol{x}_k$ is proved to contain a unique solution (that also belongs to $\boldsymbol{x}_{k+1}$). See, e.g., [26] for more details. The enclosure of the solutions to this interval linear system is usually performed using either the Krawczyk operator (somehow an interval version of the Jacobi method) or the interval Gauss–Seidel operator, leading respectively to the so-called Krawczyk and Hansen–Sengupta interval operators:

$$\mathbf{K}(F,\boldsymbol{x}) := \operatorname{mid}(\boldsymbol{x}) - \boldsymbol{F}(\tilde{x}) - (\boldsymbol{F}'(\boldsymbol{x}) - I)(\boldsymbol{x} - \operatorname{mid}(\boldsymbol{x})), \tag{2.2}$$

$$\mathbf{H}(F,\boldsymbol{x}) = \tilde{x} - \boldsymbol{\Gamma}(\boldsymbol{F}'(\boldsymbol{x}), \boldsymbol{x} - \tilde{x}, -\boldsymbol{F}(\tilde{x})) \tag{2.3}$$

$$\text{with } \boldsymbol{\Gamma}_i(\boldsymbol{A},\boldsymbol{x},\boldsymbol{b}) = \frac{1}{\boldsymbol{a}_{ii}}\left(\boldsymbol{b}_i - \sum_{j\neq i} \boldsymbol{a}_{ij}\boldsymbol{x}_j\right), \tag{2.4}$$

where $\tilde{x} \in \boldsymbol{x}$ is usually chosen as its midpoint. We emphasize here that the interval Gauss–Seidel operator usually includes an intersection with the previous domain of $\boldsymbol{x}$, which allows enclosing only those solutions that are inside the initial domain. It is not included here in order to allow the Hansen–Sengupta operator to inflate and shift boxes as well as contract them. The Krawczyk operator has equivalent properties although is proved to be slightly less efficient [26]. It may, however, be efficiently implemented in software handling fast matrix computations (like INTLAB [32]). In the rest of the paper, $\mathbf{N}(F,\boldsymbol{x})$ will denote either of these interval Newton operators.

The interval Newton operators can be extended to a square parametric system of equations $F(x,a) = 0$, i.e., $F : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^n$, simply by evaluating the function and its derivatives over the whole parameters intervals. This gives rise to parametric interval Newton operators, e.g., the parametric Krawczyk operator $\mathbf{K}(F,\boldsymbol{x},\boldsymbol{a}) = \operatorname{mid}(\boldsymbol{x}) - \boldsymbol{F}(\tilde{x},\boldsymbol{a}) - (\boldsymbol{F}'_x(\boldsymbol{x},\boldsymbol{a}) - I)(\boldsymbol{x} - \operatorname{mid}(\boldsymbol{x}))$ or the parametric Hansen–Sengupta operator $\mathbf{H}(F,\boldsymbol{x},\boldsymbol{a}) = \tilde{x} - \boldsymbol{\Gamma}(\boldsymbol{F}'_x(\boldsymbol{x},\boldsymbol{a}), \boldsymbol{x} - \tilde{x}, -\boldsymbol{F}(\tilde{x},\boldsymbol{a}))$. Its properties are similar to its nonparametric version but hold for all values of the parameters inside their domain. In particular, $\mathbf{H}(F,\boldsymbol{x},\boldsymbol{a}) \subseteq \operatorname{int}(\boldsymbol{x})$ implies that $\boldsymbol{x}$ contains a unique solution for each parameter value inside $\boldsymbol{a}$.

Finally, note that every interval Newton operator generally requires a left preconditioning in order to be efficient [26]. This is not detailed here since parallelotope computations presented in the next subsection can advantageously replace such a preconditioning process. (They can actually be interpreted as a right preconditioning process.)

Rounded computations. As real numbers are approximately represented by floating point numbers [9], the IA cannot match the definition (2.1) exactly. In order to preserve the inclusion property, the IA has to be implemented using an outward rounding. For example, $[1,3]/[10,10] = [0.1, 0.3]$, while both 0.1 and 0.3 cannot be exactly represented with standard binary floating point numbers. Therefore, the computed result will be $[0.1^{\triangledown}, 0.3^{\triangle}]$, where $0.1^{\triangledown}$ (respectively, $0.3^{\triangle}$) is a floating point number slightly smaller than 0.1 (respectively, slightly bigger than 0.3). Of course, a good implementation will return the greatest floating point number smaller than 0.1 and the smallest floating point number greater than 0.3. Among other implementations of IA, we can cite the C/C++ libraries PROFIL/BIAS [23] and Gaol [14], the MATLAB toolbox INTLAB [32], and Mathematica [35].

2.2. Manifold certification via parallelotope computation. A parallelotope $\widehat{\boldsymbol{x}}$ is the image of a box $\boldsymbol{w}$ through an affine map $w \to Cw + \tilde{x}$. It is therefore defined by a triple $(C, \boldsymbol{w}, \tilde{x})$, where $C \in \mathbb{R}^{n\times n}$, $\boldsymbol{w} \in \mathbb{IR}^n$, and $c \in \mathbb{R}^n$, whose corresponding parallelotope is $\widehat{\boldsymbol{x}} = \{Cw + \tilde{x} \in \mathbb{R}^n : w \in \boldsymbol{w}\}$. ($C$, $\boldsymbol{w}$, and $\tilde{x}$ are respectively called the characteristic matrix, box, and vector of the parallelotope.) The interval hull $\square\widehat{\boldsymbol{x}}$ of the parallelotope $\widehat{\boldsymbol{x}} = (C, \boldsymbol{w}, \tilde{x})$ is easily computed as $C\boldsymbol{w} + \tilde{x}$. The midpoint of a parallelotope is $\operatorname{mid}(\widehat{\boldsymbol{x}}) := C \operatorname{mid}(\boldsymbol{w}) + \tilde{x}$. Testing whether two parallelotopes $\widehat{\boldsymbol{x}}$ and $\widehat{\boldsymbol{x}}'$ intersect is thus a simple linear program. Alternatively, we will use the following sufficient condition:

$$C^{-1}\big((C'\boldsymbol{w}' + \tilde{x}') - \tilde{x}\big) \cap \boldsymbol{w} = \emptyset \implies \widehat{\boldsymbol{x}} \cap \widehat{\boldsymbol{x}}' = \emptyset. \tag{2.5}$$

A certified enclosure of the inverse of C has to be used in (2.5), e.g.,

$$C^{-1} \in \tilde{C}^{-1} + \frac{\nu}{1-\nu}[-|\tilde{C}^{-1}|, |\tilde{C}^{-1}|] \tag{2.6}$$

with $\nu = \|\tilde{C}^{-1}C - I\|_\infty$, $\tilde{C}^{-1}C - I$ being computed using interval arithmetic in order to provide a rigorous upper bound on the norm, and where $\tilde{C}^{-1}$ designates an approximate inverse of C (see Theorem 4.1.11 in [26]). The sufficient condition (2.5) is quite accurate when $\widehat{\boldsymbol{x}}'$ is very small with respect to $\widehat{\boldsymbol{x}}$, which is the case in our usage of the intersection test.

Parallelotopes are used in [11] in conjunction with IA in order to enclose and certify p-manifolds defined by a system $F(x) = 0$ of m equations and n unknowns with $m < n$. The cornerstone of this approach is to apply usual IA techniques dedicated to boxes within the auxiliary space of the parallelotope, where the original system of equations becomes $G(w) = 0$ with $G(w) = F(Cw + \tilde{x})$ (whose derivative is $G'(w) = F'(Cw+\tilde{x})C$). Inside this auxiliary basis, the last $n-m$ components of w are identified to parameters (w is split into $w = (u, v)$ with $u \in \mathbb{R}^m$ and $v \in \mathbb{R}^{n-m}$), and the aim is to build parallelotopes $\widehat{\boldsymbol{x}} = (C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x})$ that contain a solution for each parameters values in their domain, i.e., satisfying

$$\forall v \in \boldsymbol{v},\ \exists u \in \boldsymbol{u},\ G(u, v) = 0. \tag{2.7}$$

This property expresses that the manifold crosses the whole parallelotope along the parameter subspace. In order to certify property (2.7), [11] applies a parametric interval Newton operator to the auxiliary system $G(u, v) = 0$. The interval Newton for parallelotopes takes as input a system of equations and a parallelotope and outputs a new domain for the nonparametric auxiliary variables u. It can be instantiated using the Krawczyk or the Hansen–Sengupta interval operators:

$$\mathbf{K}(F, \widehat{\boldsymbol{x}}) := \mathrm{mid}(\boldsymbol{u}) - \boldsymbol{b} - (\boldsymbol{A}_u - I)(\boldsymbol{u} - \mathrm{mid}(\boldsymbol{u})), \tag{2.8}$$

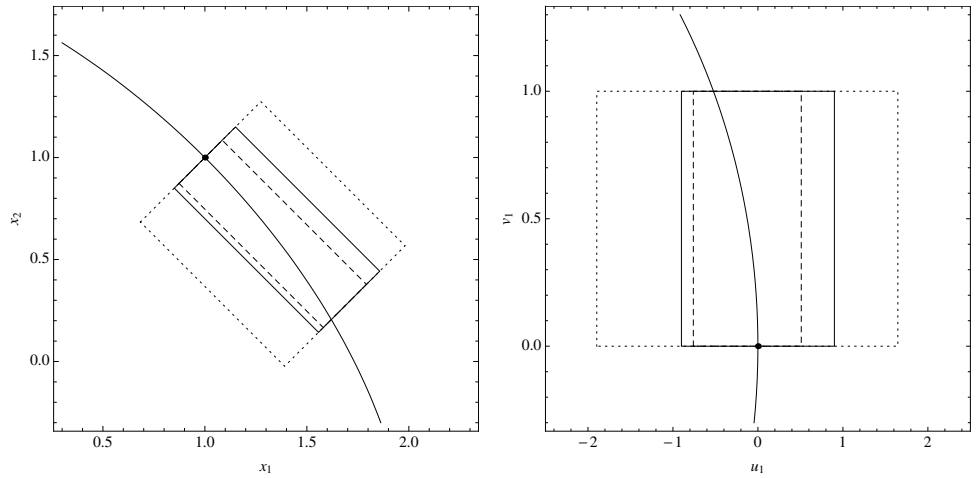
$$\mathbf{H}(F, \widehat{\boldsymbol{x}}) := \mathrm{mid}(\boldsymbol{u}) + \Gamma(\boldsymbol{A}_u, \boldsymbol{u} - \mathrm{mid}(\boldsymbol{u}), -\boldsymbol{b}), \tag{2.9}$$

where $\widehat{\boldsymbol{x}} = (C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x})$, $\boldsymbol{A}_u \supseteq \{G'_u(w) : w \in \boldsymbol{w}\}$, $\boldsymbol{A}_v \supseteq \{G'_v(\mathrm{mid}\boldsymbol{u}, v) : v \in \boldsymbol{v}\}$. (Note that $\boldsymbol{A}_u \in \mathbb{IR}^{m\times m}$ and $\boldsymbol{A}_v \in \mathbb{IR}^{m\times(n-m)}$) and $\boldsymbol{b} = \boldsymbol{G}(\mathrm{mid}(\boldsymbol{w})) + \boldsymbol{A}_v(\boldsymbol{v} - \mathrm{mid}\boldsymbol{v}.)$ In the rest of the paper, $\mathbf{N}(F, \widehat{\boldsymbol{x}})$ will denote either of these interval operators. Since $G(\mathrm{mid}(\boldsymbol{u}), v) \in \boldsymbol{b}$ holds for all $v \in \boldsymbol{v}$, the properties of the parametric interval Newton hold for the auxiliary system $G(w) = 0$. In particular, every solution contained in the initial parallelotope has to belong to the computed parallelotope $(C, (\mathbf{N}(F, \widehat{\boldsymbol{x}}), \boldsymbol{v}), \tilde{x})$, and $\mathbf{N}(F, \widehat{\boldsymbol{x}}) \subseteq \boldsymbol{u}$ implies (2.7), with uniqueness in addition to the existence statement.

The partial derivatives enclosures $\boldsymbol{A}_u$ and $\boldsymbol{A}_v$ can be computed in several ways. The most obvious is to evaluate the derivatives of F over the interval hull of the parallelotopes: $\boldsymbol{A}_u = \boldsymbol{F}'(C\boldsymbol{w} + \tilde{x})C_u$ and $\boldsymbol{A}_v = \boldsymbol{F}'(C(\mathrm{mid}\boldsymbol{u}, \boldsymbol{v}) + \tilde{x})C_v$, where $C = (C_u \,|\, C_v)$. However, this does not take full benefit of parallelotopes. When formal simplifications can be performed, e.g., for polynomial systems, the expression $G(w) = F(Cw + \tilde{x})$ can be formally simplified before performing an automatic differentiation to obtain $\boldsymbol{A}_u$ and $\boldsymbol{A}_v$. When F is twice differentiable, a third option is to use a mean value evaluation of the derivatives of G. This sensibly improves the evaluation over the hull of the parallelotope but requires evaluating second-order derivatives, which can turn out to be too expensive for large systems.

Example 2.1. Consider $F(x) = x_1^2 + x_2^2 + x_1x_2 - 3$, whose solution set is an ellipse, and the parallelotope $(C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x})$ with $\tilde{x} = (1., 1.)$, $\boldsymbol{u} = [-0.9, 0.9]$, $\boldsymbol{v} = [0, 1]$, and

$$C = \begin{pmatrix} 0.16666 & 0.70710 \\ 0.16666 & -0.70710 \end{pmatrix}. \tag{2.10}$$



(a) Parallelotope enclosure: Original basis (b) Parallelotope enclosure: Auxiliary basis

FIG. 2.1. *Application of the interval Krawczyk operator for parallelotopes in Example* 2.1.

The solution set and the parallelotope are depicted by solid lines in Figure 2.1(a). The system expressed in the auxiliary parallelotope basis is $G(w) = 0$ with $G(w) = F(Cw + \tilde{x})$, where $w = (u, v) \in \mathbb{R}^2$. The auxiliary system solution set, which is now approximately aligned with the vertical axis corresponding to the parameter v, and the box $(\boldsymbol{u}, \boldsymbol{v})$ are depicted by solid lines in Figure 2.1(b).

The explicit expression of $G(w)$ is

$$(c_{11}u+c_{12}v+c_1)^2+(c_{21}u+c_{22}v+c_2)^2+(c_{11}u+c_{12}v+c_1)(c_{21}u+c_{22}v+c_2)-3. \tag{2.11}$$

Using automatic differentiation with this expression gives rise to the following enclosure of $G'(w)$: $(\boldsymbol{A}_u, \boldsymbol{A}_v) = ([0.496, 1.504], [-1.637, 2.637])$, which is equivalent to evaluating $F'(x)$ on the interval hull of the parallelotope using automatic differentiation. The Krawczyk operator used with this expression returns $\boldsymbol{u}' = [-1.897, 1.647]$, which is not included inside $\boldsymbol{u}$, hence failing to certify the curve within the parallelotope. The parallelotope $(C, (\boldsymbol{u}', \boldsymbol{v}), \tilde{x})$ is depicted by dotted lines in Figure 2.1.

Expanding (2.11), collecting the occurrences of u and v, and differentiating with respect to w gives rise to the new expression

$$([1^-, 1^+] + [0.166^-, 0.166^+]u + [0^-, 0^+]v, [0^-, 0^+] + [0^-, 0^+]u + [1^-, 1^+]v), \tag{2.12}$$

where $[a^-, a^+]$ denotes the interval obtained using interval arithmetic to enclose rounding errors. Using automatic differentiation with this expression gives rise to the following enclosure of $G'(w)$: $(\boldsymbol{A}_u, \boldsymbol{A}_v) = ([0.849, 1.151], [-0.001, 1.001])$, which is much sharper that the evaluation over the interval hull of the parallelotope. The Krawczyk operator used with this expression returns $\boldsymbol{u}'' = [-0.761, 0.511]$, which is now included inside $\boldsymbol{u}$, hence certifying that a unique curve crosses the parallelotope. The parallelotope $(C, (\boldsymbol{u}'', \boldsymbol{v}), \tilde{x})$ is depicted by dashed lines in Figure 2.1. This system being quadratic, its derivative is linear and the mean value extension of the derivatives results to the same enclosure as the formal simplification.

In order to contract, the interval Newton for parallelotopes requires that $\boldsymbol{A}_u$ is close to the identity and that $\boldsymbol{A}_v$ is close to zero. To this end, the characteristic matrix

Algorithm 1. Contract.

Input: $F : \mathbb{R}^n \to \mathbb{R}^m$; $(C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x}) \in \mathbb{R}^{n\times n} \times \mathbb{IR}^n \times \mathbb{R}^n$;
Parameters: $\overline{k} \in \mathbb{N}$;
1.1 $k \leftarrow 0$
1.2 **repeat**
1.3 $\quad \boldsymbol{u}' = \boldsymbol{u}$
1.4 $\quad \boldsymbol{u} \leftarrow \mathbf{N}(F, (C, (\boldsymbol{u}', \boldsymbol{v}), \tilde{x})) \cap \boldsymbol{u}'$
1.5 $\quad k \leftarrow k + 1$
1.6 **until** $\boldsymbol{u}' = \boldsymbol{u}$ **or** $k \geq \overline{k}$
1.7 **return** $(C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x})$

C proposed in [11] is chosen so that its first m columns are an approximation of a generalized inverse of $F'(\tilde{x})$ and the remaining $n-m$ columns approximate the kernel of $F'(\tilde{x})$, where $\tilde{x}$ is close to the center of the parallelotope. Provided the parallelotope is small enough so that the manifold remains relatively close to its tangent, this characteristic matrix fixes the orientation of the parallelotope similarly to the one of the manifold. While the classical interval Newton operator for square systems of equations is known to require a left preconditioning for successfully contracting a box and hence proving the existence of a solution, applying the interval Newton within the auxiliary space of a parallelotope can be interpreted as a right preconditioning. This choice of the characteristic matrix can then be compared to the classical inverse midpoint (left) preconditioning for interval Newton operators dedicated to square systems of equations. This interval Newton for parallelotopes is used within the two algorithms described below.

Algorithm 1 applies the interval Newton for parallelotopes inductively until some fixed point is reached or a maximum of iterations is reached (by default $\overline{k} = 15$). It aims at reducing a parallelotope without loosing any solution to $F(x) = 0$. Formally, it satisfies

$$x \in \widehat{\boldsymbol{x}} \wedge F(x) = 0 \implies x \in \text{Contract}(F, \widehat{\boldsymbol{x}}), \tag{2.13}$$

which is a direct consequence of the properties of the parametric interval Newton operator.

It turns out also to be useful to inflate a box in such a way that the interval Newton operator strictly contracts it. This is even critical in the context of certified continuation, where the certification has to be performed from an initial nonrigorous guess. In the context of certification of manifolds using parallelotopes, inflating a parallelotope consists in inflating its characteristic box $\boldsymbol{u}$ corresponding to the nonparametric dimensions. To this end, Algorithm 2 implements a two-stage inflation process adapted from a box-inflation process proposed in [12, 19]: The main inflation is performed by the interval Newton operator itself at line 2.4, applying it without intersecting the previous domain so as to allow shifting and inflating the box. When this inflation process succeeds (informally the initial parallelotope has to be small enough and close enough to a regular manifold), it often converges to a limit parallelotope from the inside, which does not allow observing the strict contraction necessary to certify the manifold. Therefore, a static (relative and absolute) inflation is performed at line 2.9, before applying the interval Newton operator, in order to allow the latter to be strictly contracting. (Such a static inflation is similar to the ϵ-inflation described

ALGORITHM 2. N-INFLATE.

Input: $F : \mathbb{R}^n \to \mathbb{R}^m$; $(C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x}) \in \mathbb{R}^{n\times n} \times \mathbb{IR}^n \times \mathbb{R}^n$;
Parameters: $\overline{k} \in \mathbb{N}$; $\overline{\mu} \in \mathbb{R}$ (s.t. $0 < \overline{\mu} < 1$); $\delta \in \mathbb{R}$ (s.t. $1 < \delta$); $\chi \in \mathbb{R}$ (s.t. $0 < \chi$)

2.1 $k, \mu \leftarrow 0$
2.2 **repeat**
2.3 $\quad k \leftarrow k + 1$
2.4 $\quad \boldsymbol{u}' \leftarrow \mathbf{N}(f, (C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x}))$
2.5 $\quad$ *success* $\leftarrow \boldsymbol{u}' \subseteq \mathrm{int}(\boldsymbol{u})$
2.6 $\quad$ **if** $\neg$*success* **then**
2.7 $\quad\quad$ **if** $k \geq 2$ **then** $\mu \leftarrow d_H(\boldsymbol{u}, \boldsymbol{u}')/d$
2.8 $\quad\quad$ $d \leftarrow d_H(\boldsymbol{u}, \boldsymbol{u}')$
2.9 $\quad\quad$ $\boldsymbol{u} \leftarrow \mathrm{mid}(\boldsymbol{u}') + \delta(\boldsymbol{u}' - \mathrm{mid}(\boldsymbol{u}')) + \chi[-1, +1]$
2.10 $\quad$ **end**
2.11 **until** *success* **or** $k > \overline{k}$ **or** $\mu > \overline{\mu}$
2.12 **return** $(success, (C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x}))$

in [31].) A larger static inflation eases the interval Newton contraction but also deteriorates the overall convergence of the inflation iteration. Although the behavior of the process seems to be not very sensitive to reasonable changes of the static inflation parameters δ and χ, experiments have shown that $\delta = 1.1$ (which is compatible with the value recommended in [31]) and $\chi = 10^{-12}$ represent a good setting in general (although χ obviously depends on the machine and the problem). Finally, the iteration is stopped as soon as $\boldsymbol{u} \subseteq \mathrm{int}(\boldsymbol{u})'$, so (2.7) holds, or when divergence is observed, either by monitoring the distance between successive iterates and enforcing a decrease of at least $\overline{\mu}$ (by default $\overline{\mu} = 1.0$, aiming only at detecting divergence), or by enforcing a safeguarding maximum number of steps (by default $\overline{k} = 15$). Algorithm 2 returns a pair of results composed of a boolean indicating whether it was able to strictly contract $\boldsymbol{u}$, i.e., to certify the parallelotope; and the resulting parallelotope. The latter is typically quite large due to the double inflation process, which is in fact advantageous since the existence and uniqueness of the solutions hold inside its whole region. When necessary, tighter parallelotopes could be computed bisecting and contracting them as done in [11]. Note that each iteration of Algorithm 2 involves the interval evaluation of a function, that of a Jacobian matrix and $O(n^3)$ interval operations for the right preconditioning of the interval Jacobian. The properties of Algorithm 2 are summarized in the following theorem (whose proof is derived straightforwardly from [10, 11, 19][3]).

THEOREM 2.1. *Let* $\widehat{\boldsymbol{x}} = (C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x})$ *and* $(\mathbf{true}, (C, (\boldsymbol{u}', \boldsymbol{v}), \tilde{x})) = N\text{-}Inflate(F, \widehat{\boldsymbol{x}})$. *First,* $x \in \widehat{\boldsymbol{x}}$ *and* $F(x) = 0$ *implies* $x \in \widehat{\boldsymbol{x}}'$. *Second, there exists a unique function* $\mu : \boldsymbol{v} \to \boldsymbol{u}'$, *differentiable inside* $\mathrm{int}(\boldsymbol{v})$, *such that*

$$\forall v \in \boldsymbol{v},\ F(C(\mu(v), v) + \tilde{x}) = 0. \tag{2.14}$$

Finally, for all $x \in \widehat{\boldsymbol{x}}$, $F'(x)$ *is full rank.*

3. Parallelotope continuation. The method we propose, called ParCont, interleaves local curve certifications using Algorithm 2 with several checks, which intend

[3]Except for the differentiability of μ, which is a consequence of the implicit function theorem and the fact that $\boldsymbol{A}_u$ is strongly regular, and hence G' and F' are full rank inside $\widehat{\boldsymbol{x}}$.

ALGORITHM 3. PARCONT.

Input: $F : \mathbb{R}^n \to \mathbb{R}^{n-1}$; $\boldsymbol{x}^{\text{init}} \in \mathbb{IR}^n$; $\tilde{x}_0 \in \mathbb{R}^n$; $d \in \{-1, 1\}$
Parameters: $\underline{h}, h \in \mathbb{R}$ (s.t. $0 < \underline{h} < h$); $\alpha, \beta \in \mathbb{R}$ (s.t. $0 < \alpha < 1 < \beta$)
3.1 $k \leftarrow 1$
3.2 *stop* $\leftarrow$ **false**
3.3 **while** $\neg$*stop* **do**
3.4 $\quad \widehat{\boldsymbol{x}}_k \leftarrow (C_k, \boldsymbol{w}_k, \tilde{x}_{k-1})$ using Eq. (3.1)
3.5 $\quad (\textit{success}, \widehat{\boldsymbol{x}}_k) \leftarrow \text{N-Inflate}(F, \widehat{\boldsymbol{x}}_k)$
3.6 $\quad$ **if** *success* **then**
3.7 $\quad\quad$ **if** $k = 1$ **then** $\widehat{\boldsymbol{y}}_0 \leftarrow \text{Contract}\big(F, \text{InputSide}(\widehat{\boldsymbol{x}}_k)\big)$
3.8 $\quad\quad \widehat{\boldsymbol{y}}_k \leftarrow \text{Contract}\big(F, \text{OutputSide}(\widehat{\boldsymbol{x}}_k)\big)$
3.9 $\quad$ **end**
3.10 $\quad$ *valid* $\leftarrow$ *success* **and** (3.9a) **and** (3.9b) **and** (3.9c)
3.11 $\quad h \leftarrow$ Step size updating using (3.10)
3.12 $\quad$ **if** *valid* **then**
3.13 $\quad\quad \tilde{x}_k \leftarrow \text{mid}(\widehat{\boldsymbol{y}}_k)$
3.14 $\quad\quad k \leftarrow k + 1$
3.15 $\quad$ **end**
3.16 $\quad$ *stop* $\leftarrow (\neg(3.9a))$ **or** (3.11a) **or** (3.11b) **or** (3.11c) **or** (3.11d)
3.17 **end**
3.18 **return** $(\widehat{\boldsymbol{y}}_0, \ldots, \widehat{\boldsymbol{y}}_{k-1})$, $(\widehat{\boldsymbol{x}}_1, \ldots, \widehat{\boldsymbol{x}}_{k-1})$

in particular to certify the connections between successive parallelotopes and to ensure no backtrack. It is formally defined in Algorithm 3, whose description resides in subsection 3.1. It takes as inputs the system F whose solution set is the curve to follow, an initial point x_0 on (or very close to) this curve, and an initial box domain $\boldsymbol{x}^{\text{init}}$ within which the curve is to be followed. A direction $d \in \{-1, 1\}$ for the continuation is also provided, in order to be able to perform two continuations processes in different directions. (See the definition of C_k in subsection 3.1.1.) It iteratively builds two sequences of parallelotopes $(\widehat{\boldsymbol{x}}_k)$ and $(\widehat{\boldsymbol{y}}_k)$. Each parallelotope $\widehat{\boldsymbol{x}}_k$ is crossed from one input side to the opposite output side by one unique component of the curve. Each parallelotope $\widehat{\boldsymbol{y}}_k$ encloses the single solution of the curve on the output side of $\widehat{\boldsymbol{x}}_k$ (except $\widehat{\boldsymbol{y}}_0$, which encloses the unique solution on the input side of the first parallelotope). The parallelotopes $\widehat{\boldsymbol{y}}_k$ enclose a unique solution and are therefore tiny (usually of width approximately 10^{-14}). The connection between two consecutive parallelotopes $\widehat{\boldsymbol{x}}_k$ and $\widehat{\boldsymbol{x}}_{k+1}$ is ensured since both enclose the certified solution within parallelotope $\widehat{\boldsymbol{y}}_k$. A special case is $\widehat{\boldsymbol{y}}_0$, which encloses the solution of the curve on the *input* side of $\widehat{\boldsymbol{x}}_1$. It is used to detect loops, i.e., connection between the first and last parallelotopes in the sequence $(\widehat{\boldsymbol{x}}_k)$. The correctness, halting, and asymptotic convergence of ParCont are investigated in subsection 3.2. Finally, the limitations of ParCont are emphasized in subsection 3.3.

3.1. Algorithm description. Algorithm 3 gives a formal definition of ParCont. It consists of one main loop, which iteratively builds a trial parallelotope whose length corresponds to the step size and tries inflating it using Algorithm 2. In case of success it performs several checks to validate the absence of backtrack, the inclusion within the initial domain, and the looping status. When the successfully inflated parallelotope is validated, a tiny parallelotope enclosing the unique solution within its

output side is built using Algorithm 1, which is used to certify the connection with the next parallelotope. The step size is eventually updated, and tests decide whether to perform a new step.

Notation. The parallelotope $\widehat{\boldsymbol{x}}_k$ characteristic matrix and box are respectively denoted by C_k and $\boldsymbol{w}_k = (\boldsymbol{u}_k, \boldsymbol{v}_k)$. The parallelotope $\widehat{\boldsymbol{y}}_k$ is tiny and contains the unique solution within the output side of the parallelotope $\widehat{\boldsymbol{x}}_k$, which is denoted by y_k. For $k \geq 1$, the (approximate) midpoint of the parallelotope $\widehat{\boldsymbol{y}}_k$ is denoted $\tilde{x}_k$, so $\tilde{x}_k$ and y_k are very close to each other. Provided that $\tilde{x}_0$ is an accurate enough approximate solution, it is also very close to y_0. The closeness of $\tilde{x}_k$ and y_k for $k \geq 1$ has no impact on the correctness of the algorithm, but it provides a useful picture of its normal behavior.

3.1.1. Trial parallelotope construction and inflation. Line 3.4 of Algorithm 3 builds a trial parallelotope $\widehat{\boldsymbol{x}}_k = (C_k, \boldsymbol{w}_k, \tilde{x}_{k-1})$, which intends to be a segment tangent to the curve at $\tilde{x}_{k-1}$ of length h. In practice, it is approximately tangent (due to acceptable rounding errors in the derivatives evaluation at $\tilde{x}_{k-1}$ and the approximate null space computation) and is slightly enlarged so as to rigorously contain the tiny parallelotope $\widehat{\boldsymbol{y}}_{k-1}$ and its solution y_{k-1}:

$$C_k := \begin{pmatrix} F'(\tilde{x}_{k-1}) \\ \ker(F'(\tilde{x}_{k-1}))^T \end{pmatrix}^{-1}, \tag{3.1a}$$

$$\boldsymbol{w}_k := \begin{cases} \big(0, \ldots, 0, [0, h]\big) & \text{if } k = 1, \\ \big(C_k^{-1}(\square\widehat{\boldsymbol{y}}_{k-1} - \tilde{x}_{k-1})\big) \vee \big(0, \ldots, 0, [0, h]\big) & \text{if } k \geq 2, \end{cases} \tag{3.1b}$$

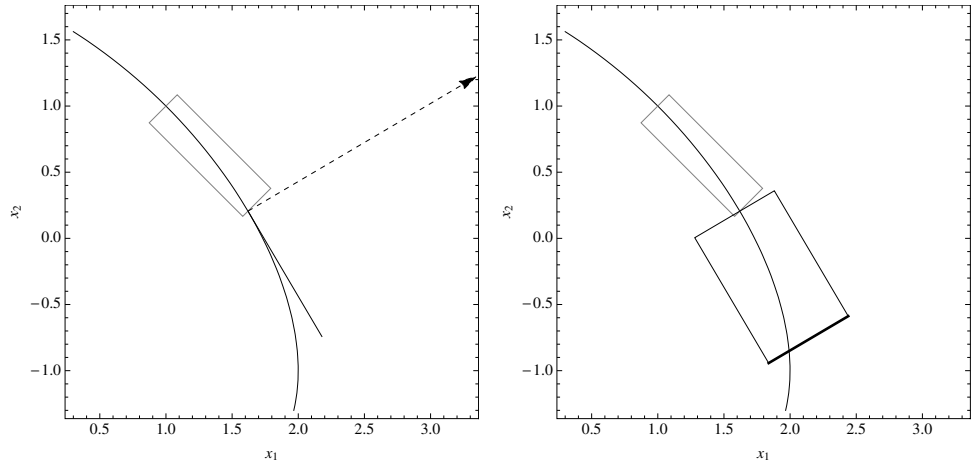
where $\ker(F'(\tilde{x}_{k-1}))$ is one single vector of norm 1 and oriented so that the sign of $\det C_k^{-1}$ is $d \in \{-1, 1\}$ (in order to maintain the same direction for the continuation; see [1]). The characteristic box $\boldsymbol{w}_k$ built using (3.1b) entails that the trial parallelotope $\widehat{\boldsymbol{x}}_k$ actually contains the unique solution within $\widehat{\boldsymbol{y}}_{k-1}$. (Indeed, $\boldsymbol{w}_k \supseteq C_k^{-1}(\square\widehat{\boldsymbol{y}}_{k-1} - \tilde{x}_{k-1})$ implies $\square\widehat{\boldsymbol{y}}_{k-1} \subseteq \widehat{\boldsymbol{x}}_k$.)

Floating point computations can be used in (3.1a), since these components of $\widehat{\boldsymbol{x}}_k$ can be computed approximately. However, interval computations have to be used in (3.1b) so that the last solution of the previous parallelotope provably belongs to the current parallelotope. This entails using a certified enclosure of the inverse of C_k, e.g., using (2.6). Note also that in case the previous iteration failed, it is not necessary to recompute (3.1a). This is not shown explicitly in Algorithm 3 for the sake of simplicity.

Example 3.1. Applying Algorithm 3 with step size $h = 1$ to the system of Example 2.1 results in the parallelotope $\widehat{\boldsymbol{x}}_1 = (C_1, \boldsymbol{w}_1, \tilde{x}_0)$, which has been computed in Example 2.1, together with a tiny parallelotope $\widehat{\boldsymbol{y}}_1 = (C_1, (\boldsymbol{u}_1, 1), \tilde{x}_0)$, where $\boldsymbol{u}_1 = [-0.5227744249483427, -0.5227744249483349]$, which contains the unique solution in its output side. Also, $\tilde{x}_1 = \text{mid}\widehat{\boldsymbol{y}}_1 = (1.6199777103618245, 0.2057641479887291)$, which is a sharp approximation of the solution within $\widehat{\boldsymbol{y}}_1$, and the step size is increased, say, to $h = 1.1$.

A new loop of Algorithm 3 starts by evaluating the derivatives at x_1, using standard double computations, leading to $F'(x_1) \approx (1.72286, 1.01575)$, whose normalized and correctly oriented null space is approximately $(-0.507877, 0.86143)$. This leads to the trial parallelotope $\widehat{\boldsymbol{x}}_2 = (C_2, \boldsymbol{w}_2, \tilde{x}_1)$ with

$$C_2 \approx \begin{pmatrix} 0.215357 & 0.507877 \\ 0.126969 & -0.86143 \end{pmatrix} \tag{3.2}$$



(a) In black, the trial parallelotope $\widehat{\boldsymbol{x}}_2$. ($F'(\tilde{x}_1)$ is represented by the dashed arrow.)

(b) In black, the inflated parallelotope $\widehat{\boldsymbol{x}}_2$. In bold, its output side.

FIG. 3.1. *Construction of the parallelotope in Example* 3.1.

and

$$\boldsymbol{w}_2 = (0, [0, h]) \vee \left(C_2^{-1}(\square\widehat{\boldsymbol{y}}_1 - \tilde{x}_1)\right) \tag{3.3}$$

$$= ([-8.12 \times 10^{-15}, 7.90^{-15}], [-1.87 \times 10^{-15}, 1.1]). \tag{3.4}$$

This trial parallelotope, which is depicted in black in Figure 3.1(a), contains the unique solution y_1 that lies in the output side of $\widehat{\boldsymbol{x}}_1$.

The trial parallelotope contains the previous parallelotope output side solution y_{k-1} for $v \approx 0$. It is inflated aiming to enclose a unique solution for all values of $v \in \boldsymbol{v}_k$. This is done using Algorithm 2 at line 3.5, which furthermore ensures by Theorem 2.1 that the inflated parallelotope also contains y_{k-1} (since no solution can be lost during Algorithm 2).

Example 3.2. Algorithm 2 successfully inflates the trial parallelotope of Example 3.1, giving rise to $\widehat{\boldsymbol{x}}_2 = (C_2, \boldsymbol{w}_2, \tilde{x}_1)$ and

$$\boldsymbol{w}_2 = ([-1.583, 1.211], [-1.87 \times 10^{-15}, 1.1]), \tag{3.5}$$

which is depicted in Figure 3.1(b), showing it indeed contains a unique solution for each $v \in \boldsymbol{v}_2$.

The trial characteristic box built using (3.1b) is generally very thin, which entails too many iterations of Algorithm 2. Making the reasonable assumption that two consecutive parallelotopes will be similar, this can be sensibly improved at low cost using the previous parallelotope characteristic box to enlarge it:

$$\boldsymbol{w}_k \;\leftarrow\; (\boldsymbol{u}_k \vee \boldsymbol{u}_{k-1}, \boldsymbol{v}_k). \tag{3.6}$$

This heuristic is used within line 3.4, just after (3.1b) for $k \geq 2$. It is used only when the previous parallelotope was successfully build, since a failure entails a decrease of the step size, for which the previous characteristic box is most probably too large.

3.1.2. Output side parallelotopes. Once the parallelotope $\widehat{\boldsymbol{x}}_k$ is successfully built, the solution on its output side has to be enclosed sharply inside $\widehat{\boldsymbol{y}}_k$. To this end, $\widehat{\boldsymbol{y}}_k$ is computed at line 3.8 by applying Contract (Algorithm 1) to contract the parallelotope corresponding to the output side of $\widehat{\boldsymbol{x}}_k$. During the first iteration of Algorithm 3, the input solution is similarly enclosed inside $\widehat{\boldsymbol{y}}_0$ at line 3.7. Parallelotopes corresponding to the input and output sides of a given parallelotope are determined as follows:

$$\text{InputSide}(C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x}) = (C, (\boldsymbol{u}, \underline{v}), \tilde{x}), \tag{3.7}$$

$$\text{OutputSide}(C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x}) = (C, (\boldsymbol{u}, \overline{v}), \tilde{x}). \tag{3.8}$$

By Theorem 5.2.2 and Theorem 5.2.5 of [26], addressing respectively the Krawczyk and Hansen–Sengupta operators, these contractions allow computing arbitrarily sharp enclosure of the solutions, according to the computational precision.

Example 3.3. The output side of the parallelotope (3.5) built in Example 3.2 is $(C_2, ([-1.583, 1.211], 1.1), \tilde{x}_1)$, which is depicted in bold in Figure 3.1(b). It is contracted at line 3.8 to $\widehat{\boldsymbol{y}}_2 = (C_2, ([-0.86220253621524, -0.86220253621520], 1.1), \tilde{x}_1)$. The resulting parallelotope is a very sharp enclosure of the unique solution within the output side of $\widehat{\boldsymbol{x}}_2$.

3.1.3. Validation of successfully inflated parallelotopes. The parallelotope successfully inflated has to be validated at line 3.10 for ensuring no backtrack on the curve happens (3.9a) (starting from iteration $k = 2$), for deciding the looping status of the curve (3.9b) (the parallelotope is valid if the presence of a loop is either rigorously disproved by $\widehat{\boldsymbol{y}}_0 \cap \widehat{\boldsymbol{x}}_k = \emptyset$ or proved by $\widehat{\boldsymbol{y}}_0 \subseteq \widehat{\boldsymbol{x}}_k$, starting from iteration $k = 2$), and for checking its inclusion within the initial domain (3.9c) (the parallelotope is valid if rigorously proved to either remain fully inside the domain or to leave the domain):

$$k = 1 \ \textbf{ or } \ \big(\widehat{\boldsymbol{y}}_k \cap \widehat{\boldsymbol{x}}_{k-1} = \emptyset \ \textbf{ and } \ \widehat{\boldsymbol{y}}_{k-2} \cap \widehat{\boldsymbol{x}}_k = \emptyset\big), \tag{3.9a}$$

$$k = 1 \ \textbf{ or } \ \widehat{\boldsymbol{y}}_0 \cap \widehat{\boldsymbol{x}}_k = \emptyset \ \textbf{ or } \ \widehat{\boldsymbol{y}}_0 \subseteq \widehat{\boldsymbol{x}}_k, \tag{3.9b}$$

$$\square\widehat{\boldsymbol{y}}_k \cap \boldsymbol{x}^{\text{init}} = \emptyset \ \textbf{ or } \ \square\widehat{\boldsymbol{x}}_k \subseteq \boldsymbol{x}^{\text{init}}. \tag{3.9c}$$

Parallelotope empty intersections can be checked using (2.5).

Remark 3.1. Since the correct direction is used by checking the sign of the determinant of C_k, (3.9a) is always true provided that C_k is computed with good enough accuracy (evaluation of the derivatives at $\tilde{x}_{k-1}$, kernel computation, and matrix inversion). However, these tests are included in Algorithm 3 in order to enforce its correctness regardless of the accuracy of C_k and of its determinant sign computation.

Example 3.4. The parallelotope built in Example 3.2 is validated as follows. First there is no backtrack since $\widehat{\boldsymbol{y}}_2 \cap \widehat{\boldsymbol{x}}_1 = \emptyset$ and $\widehat{\boldsymbol{y}}_0 \cap \widehat{\boldsymbol{x}}_2 = \emptyset$ (see Figure 3.1(a)). Second, there is no loop since $\widehat{\boldsymbol{y}}_0 \cap \widehat{\boldsymbol{x}}_2 = \emptyset$. Finally, (3.9c) depends on the initial domain $\boldsymbol{x}^{\text{init}}$ but will succeed provided that it is large enough.

3.1.4. Step size update. A step size adjustment is performed by updating h at line 3.11. The only requirement on a step size adjustment strategy is that it is *fair*, i.e., infinitely many consecutive unsuccessful iterations entail that h converges toward zero. ParCont implements the following simple step size control strategy:

$$h \leftarrow \begin{cases} \beta h & \text{if } \textit{valid}, \\ \alpha h & \text{otherwise.} \end{cases} \tag{3.10}$$

This strategy is obviously fair provided that $0 < \alpha < 1$, while $1 < \beta$ allows increasing h in case of validated success. A *pessimistic* strategy with small values for α and β, which intends making smaller steps with easy success of N-Inflate, sounds realistic taking into account that calls to N-Inflate have the same cost whether they succeed or not. Nonreported experiments with various value combinations on several problems have confirmed this hypothesis and they have shown that $\alpha = 0.5$ and $\beta = 1.1$ represent an efficient setting.

3.1.5. Stopping criteria. Four stopping conditions can stop the algorithm. First, the algorithm stops whenever (3.9a) is false, i.e., the absence of backtrack could not be proved. In this case, the parallelotope was not validated, but decreasing the step size will not improve the situation, hence the algorithm has to stop. The last four stopping conditions are given below:

$$\textit{valid} \ \textbf{and} \ k \geq 2 \ \textbf{and} \ \widehat{\boldsymbol{y}}_0 \subseteq \widehat{\boldsymbol{x}}_k, \tag{3.11a}$$

$$\textit{valid} \ \textbf{and} \ \square\widehat{\boldsymbol{y}}_k \cap \boldsymbol{x}^{\text{init}} = \emptyset, \tag{3.11b}$$

$$\textit{valid} \ \textbf{and} \ \inf(\textbf{dist}(\square\widehat{\boldsymbol{y}}_{k-1}, \square\widehat{\boldsymbol{y}}_k)) \leq \underline{h}, \tag{3.11c}$$

$$h \leq \underline{h}. \tag{3.11d}$$

Whenever (3.11a) is satisfied (starting from iteration $k = 2$), the curve is proved to loop (see the proof of Theorem 3.1) and the algorithm stops as a complete connected component of the followed curve has been rigorously covered. The condition (3.11b) allows stopping the algorithm as soon as the curve is proved to leave the initial domain. The conditions (3.11c) and (3.11d) enforce a minimal step size, respectively enforcing a minimal actual step size and avoiding infinitely many successive failures. These last two conditions stand for guaranteeing the termination of the algorithm (see Corollary 3.2 and its proof), allowing it to stop when failing to cover the manifold because, e.g., of a singularity or a too strong curvature.

3.2. Properties of the algorithm. The three statements provided in this subsection involve real interval arithmetic. The first two, related to the correctness and halting of Algorithm 3, remain correct in outwardly rounded floating interval arithmetic. The third statement is about the asymptotic convergence of Algorithm 3, which is valid only in real interval arithmetic, but provides good insight on the normal behavior of the floating point implementation of Algorithm 3. All parameters of the algorithms are assumed to lie inside the domains provided in each algorithm description.

3.2.1. Correctness. Theorem 2.1 shows that each validated parallelotope $\widehat{\boldsymbol{x}}_k$ is crossed by a piece of solution curve $\gamma_k(t)$. Theorem 3.1 is dedicated to showing that they can be glued together to a global solution curve $\gamma(t)$ (using the solutions y_k in parallelotopes $\widehat{\boldsymbol{y}}_k$ that are common to successive parallelotopes $\widehat{\boldsymbol{x}}_k$), hence ensuring the connectivity of the continuation, and that there is no unwanted backtrack during the continuation process (using the validating conditions (3.9a)).

THEOREM 3.1. *Suppose $K \geq 1$. Let $(\widehat{\boldsymbol{x}}_1, \ldots, \widehat{\boldsymbol{x}}_K)$ and $(\widehat{\boldsymbol{y}}_0, \ldots, \widehat{\boldsymbol{y}}_K)$ be some parallelotopes sequences produced by Algorithm 3. There exist $L > 0$ and $\gamma : [0, L] \to \cup_{k=1}^{K} \widehat{\boldsymbol{x}}_k$ such that*

(i) *for all $t \in [0, L]$, $F(\gamma(t)) = 0$ and $F'(\gamma(t))$ is full rank;*
(ii) *γ is differentiable in the interior of $[0, L]$, and $\|\gamma'(t)\| = 1$;*
(iii) *either $\widehat{\boldsymbol{y}}_0 \cap \widehat{\boldsymbol{x}}_K = \emptyset$ or $\widehat{\boldsymbol{y}}_0 \subseteq \widehat{\boldsymbol{x}}_K$. Furthermore,*

(iiia) *If* $K = 1$ *or* $\widehat{\boldsymbol{y}}_0 \cap \widehat{\boldsymbol{x}}_K = \emptyset$*, then* γ *is injective.*

(iiib) *If* $K > 1$ *and* $\widehat{\boldsymbol{y}}_0 \subseteq \widehat{\boldsymbol{x}}_K$*, then* γ *is an injective loop.*[4]

Proof. Let $\mu_k : \boldsymbol{v}_k \to \boldsymbol{u}_k$ be the function given by Theorem 2.1 applied to the parallelotope $\widehat{\boldsymbol{x}}_k$, and define $\tilde{\gamma}_k : \boldsymbol{v}_k \to \widehat{\boldsymbol{x}}_k$ by $\tilde{\gamma}_k(v) = C_k(\mu_k(v), v) + \tilde{x}_{k-1}$, which is also differentiable in the interior of $\boldsymbol{v}_k$. Hence, $x \in \widehat{\boldsymbol{x}}_k \wedge F(x) = 0 \iff \exists v \in [\underline{v}_k, \overline{v}_k], x = \tilde{\gamma}_k(v)$. Furthermore, $\tilde{\gamma}_k'(v) \neq 0$ obviously holds, hence $\tilde{\gamma}_k$ can be reparametrized to $\gamma_k : [0, l_k] \to \widehat{\boldsymbol{x}}_k$ with unit speed (i.e., $\|\gamma_k'(t)\| = 1$), so l_k is the length of the solution path that crosses $\widehat{\boldsymbol{x}}_k$. The functions γ_k obviously inherit the property

$$x \in \widehat{\boldsymbol{x}}_k \wedge F(x) = 0 \iff \exists t \in [0, l_k], x = \gamma_k(t). \tag{3.12}$$

Note that $\tilde{\gamma}_k$ is injective since C_k is nonsingular, and hence so is γ_k.

Now, γ is defined piecewise by connecting the γ_k using the solutions $y_k \in \widehat{\boldsymbol{y}}_k$ shared by consecutive parallelotopes. By construction, $y_{k-1} \in \widehat{\boldsymbol{x}}_k$ and $y_k \in \widehat{\boldsymbol{x}}_k$ for $k \in \{1, \ldots, K\}$. Hence $y_{k-1}, y_k \in \gamma_k([0, l_k])$, and $\underline{t}_k := \gamma_k^{-1}(y_{k-1})$ and $\overline{t}_k := \gamma_k^{-1}(y_k) = l_k$ are well defined. We now prove that $\underline{t}_k < \overline{t}_k$ holds for all $k \in \{1, \ldots, K\}$. For all such k, y_k is the output solution of the parallelotope $\widehat{\boldsymbol{x}}_k$ and hence $\underline{t}_k \leq \overline{t}_k$ holds. The equality $\underline{t}_k = \overline{t}_k$ contradicts (3.11d) and $\underline{h} > 0$ for $k = 1$ or contradicts $\widehat{\boldsymbol{y}}_k \cap \widehat{\boldsymbol{x}}_{k-1} = \emptyset$ in (3.9a) for $k \geq 2$ (since $\widehat{\boldsymbol{y}}_{k-1} \subseteq \widehat{\boldsymbol{x}}_{k-1}$). Define T_k for $k \in \{0, \ldots, K\}$ by $T_0 = 0$ and $T_k = T_{k-1} + \overline{t}_k - \underline{t}_k$, so $0 = T_0 < T_1 < \cdots < T_K$. The function $\gamma : [0, T_K] \to \cup_{k=1}^K \widehat{\boldsymbol{x}}_k$ is finally defined as follows: If $t \in [T_{k-1}, T_k[$, then, $t + \underline{t}_k - T_{k-1} \in [\underline{t}_k, \overline{t}_k[$ and

$$\gamma(t) := \gamma_k(t + \underline{t}_k - T_{k-1}) \tag{3.13}$$

is well defined. As a consequence, γ is differentiable within $]0, T_K[\backslash \{T_1, \ldots, T_{K-1}\}$. For all $t \in [0, T_K]$, $\gamma(t) \in \cup_{k=1}^K \widehat{\boldsymbol{x}}_k$ and $F(\gamma(t)) = 0$ hold by construction, and $F'(\gamma(t))$ is full rank by Theorem 2.1, which prove (i) for any $0 < L \leq T_K$.

To prove (ii), first note also that γ is continuous with $\gamma(T_k) = y_k$. Indeed $\gamma(T_k) = \gamma_{k+1}(\underline{t}_{k+1}) = y_k$, while

$$\lim_{t \to T_k^-} \gamma(t) = \lim_{t \to T_k^-} \gamma_k(t + \underline{t}_k - T_{k-1}) = \gamma_k(\overline{t}_k) = y_k. \tag{3.14}$$

Furthermore, γ is differentiable in $]0, T_K[$, so define $g_k^{\pm} := \gamma'(T_k^{\pm})$ (evaluations at $T_k^{\pm}$ corresponding to limits from above and below, respectively). As a consequence, we just need to prove that $g_k^- = g_k^+$ for $k \in \{1, \ldots, K-1\}$. Since $F'(\gamma(t))\gamma'(t)$ and $\|\gamma'(t)\|$ are both continuous and respectively identically 0 and 1 inside $]0, T_K[\backslash \{T_1, \ldots, T_{K-1}\}$, both $F'(\gamma(T_k))g_k^{\pm} = 0$ and $\|g_k^{\pm}\| = 1$ hold. Therefore, $g_k^- = \pm g_k^+$ (since $\ker F'(\gamma(T_k))$ has dimension 1). Now suppose that $g_k^- = -g_k^+$. Both $\gamma(T_k + t) = \gamma_{k+1}(\underline{t}_k + t)$ and $\gamma(T_k - t) = \gamma_k(\overline{t}_k - t)$ hold inside $[0, \min\{T_{k+1} - T_k, T_k - T_{k-1}\}]$, and hence both are differentiable in the interior of this time domain. Furthermore, both have the same value (namely, $\gamma(T_k)$) and derivative at $t = 0$ (since $g_k^- = -g_k^+$), so Lemma A.1 in the appendix proves that they are equal on the whole time domain. Therefore, $\gamma(T_k + \min\{T_{k+1} - T_k, T_k - T_{k-1}\}) = \gamma(T_k - \min\{T_{k+1} - T_k, T_k - T_{k-1}\})$. Finally, either $T_{k+1} - T_k \leq T_k - T_{k-1}$, then $\gamma(T_{k+1}) = \gamma(T_k - (T_{k+1} - T_k))$ and $T_k - (T_{k+1} - T_k) \in [T_{k-1}, T_k]$, which contradicts $\widehat{\boldsymbol{y}}_{k+1} \cap \widehat{\boldsymbol{x}}_k = \emptyset$ in (3.9a). Or, $T_{k+1} - T_k > T_k - T_{k-1}$, then $\gamma(2T_k - T_{k-1}) = \gamma(T_{k-1})$ and $2T_k - T_{k-1} \in [T_k, T_{k+1}]$, which contradicts $\widehat{\boldsymbol{y}}_{k-1} \cap \widehat{\boldsymbol{x}}_{k+1} = \emptyset$ in (3.9a). Therefore $g_k^- = g_k^+$ and γ is eventually differentiable in $]0, T_K[$, proving (ii) for any $0 < L \leq T_K$.

[4] i.e., it is injective inside $[0, L[$, $\gamma(0) = \gamma(L)$, and $\gamma'(0^+) = \gamma'(L^-)$, evaluations at $t^{\pm}$ corresponding to one-sided limits.

The property (iii) is a direct consequence of the validating condition (3.9b). For (iiia) and (iiib), first note that (iiia) is trivial if $K = 1$. Thus suppose $K > 1$, and consider $0 \leq t_1 < t_2 \leq T_K$ such that $\gamma(t_1) = \gamma(t_2) = x_*$. As previously, note that $F'(x_*)$ being full rank, its kernel is one-dimensional and hence $\gamma'(t_1) = \pm\gamma'(t_2)$. (Left or right derivatives are considered whether $t_1 = 0$ or $t_2 = T_K$.) Suppose first that $\gamma'(t_1) = -\gamma'(t_2)$. Then both $\gamma(t_1 + t)$ and $\gamma(t_2 - t)$ are defined on $[0, t_2 - t_1]$ and have the same value and derivative at $t = 0$. Therefore, Lemma A.1 proves that they are equal on $[0, t_2 - t_1]$. As a consequence, $\gamma'(\frac{1}{2}(t_2 + t_1)) = -\gamma'(\frac{1}{2}(t_2 + t_1))$, which entails $\gamma'(\frac{1}{2}(t_2 + t_1)) = 0$, a contradiction since $\|\gamma'(\frac{1}{2}(t_2 + t_1))\| = 1$. Therefore $\gamma'(t_1) = \gamma'(t_2)$. Then both $\gamma(t_1 - t)$ and $\gamma(t_2 - t)$ are defined on $[0, t_1]$ and have the same value and derivative at $t = 0$. Therefore, Lemma A.1 proves that they are equal on $[0, t_1]$, and in particular $\gamma(0) = \gamma(t_2 - t_1)$ and $\gamma'(0^+) = \gamma'((t_2 - t_1)^-)$ (evaluations at $t^\pm$ corresponding to limits from above and below, respectively). Now, $\widehat{\boldsymbol{y}}_0 \cap \widehat{\boldsymbol{x}}_k = \emptyset$ is enforced by Algorithm 3 for all $k \in \{2, \ldots, K-1\}$, hence $\gamma(t_2 - t_1) \notin \widehat{\boldsymbol{x}}_k$ for any such k. Note furthermore that $t_2 - t_1 \notin [0, T_1]$ since otherwise $\gamma_1(0) = \gamma_1(t_2 - t_1)$, which is impossible since γ_1 is injective. Therefore, $t_2 - t_1 \in [T_{K-1}, T_K]$ so

$$y_0 = \gamma(0) = \gamma(t_2 - t_1) = \gamma_K(t_2 - t_1 + \underline{t}_k - T_{K-1}). \tag{3.15}$$

Two cases arise depending on the halting status of Algorithm 3. First, if $\widehat{\boldsymbol{y}}_0 \cap \widehat{\boldsymbol{x}}_K = \emptyset$, then $t_2 - t_1 \in [T_{K-1}, T_K]$ is contradicted and γ is injective inside $[0, L]$ with $L = T_K$, hence proving (iiia). Second, if $\widehat{\boldsymbol{y}}_0 \subseteq \widehat{\boldsymbol{x}}_K$, then $y_0 \in \widehat{\boldsymbol{x}}_K$ which entails $t_2 - t_1 = \gamma_K^{-1}(y_0) + T_{K-1} - \underline{t}_k$ by (3.15), which has been proved to belong to $[T_{K-1}, T_K]$. So by defining $L = \gamma_K^{-1}(y_0) + T_{K-1} - \underline{t}_k \leq T_K$, we have γ injective inside $[0, L[$ and $\gamma(0) = \gamma(L)$, which eventually proves (iiib). ∎

3.2.2. Halting. Define $R := \{x \in \boldsymbol{x}^{\text{init}} : F(x) = 0 \wedge F'(x) \text{ full rank}\}$ and $R(y_0)$ as the connected component of R that contains y_0. Three cases arise when the algorithm halts at iteration $k = K + 1$ depending on the condition that stopped the algorithm. First, if (3.11a) stopped Algorithm 3, then $R(y_0)$ is proved to be a loop. Second, if (3.11b) stopped Algorithm 3, then the last solution y_k is proved to be outside the domain $\boldsymbol{x}^{\text{init}}$, and therefore Algorithm 3 has fully enclosed $R(y_0)$ from y_0 to one of its sides, depending on the direction chosen for the continuation. The second part of $R(y_0)$ can be computed by starting Algorithm 3 again from y_0 in the opposite direction. Finally, if (3.11d) has stopped Algorithm 3, then either a singularity or a too strong curvature prematurely stopped the algorithm.

Since γ has a unit speed parametrization, L is the length of the 1-manifold between $\gamma(0)$ and $\gamma(L)$, and the following corollary can be proved.

COROLLARY 3.2. *If the step size update is fair and $R(y_0)$ has a finite length, then Algorithm 3 halts.*

Proof. Suppose Algorithm 3 does not halt. Since Algorithms 2 and 1 halt, Algorithm 3 does not halt if it executes infinitely many times the main while loop. Thus the algorithm is either producing infinitely many successive failed iterations or infinitely many successful ones (separated by finitely many failures).

If Algorithm 3 is producing infinitely many successive failures, then the fairness of the step size control implies that h converges to zero. However, this contradicts the stopping criteria (3.11d), since $\underline{h} > 0$. Hence, Algorithm 3 cannot produce infinitely many failures and therefore produces infinitely many successful iterations.

At iteration $K \geq 2$, Theorem 3.1 can be applied: there exists a curve $\gamma : [0, L_K] \to \cup_{k=1}^{k} \widehat{\boldsymbol{x}}_k$ with L_K the length of the curve between y_0 and y_K, where $F'(\gamma(t))$ is full rank, $\gamma(t)$ is connected to y_0, and $\gamma(t)$ is in $\boldsymbol{x}^{\text{init}}$ for all $t \in [0, L_K]$. Therefore, $\gamma(t) \in R$

for all $t \in [0, L_K]$. The stopping criteria (3.11c) ensures that the length of the manifold between two consecutive certified solutions y_{i-1} and y_i is greater than $\|y_i - y_{i-1}\| \geq \underline{h}$. (The length of the curve between two points is longer than the length of the direct line connecting them, which is bounded by the algorithm.) Hence, $L_K \geq K\underline{h}$. Therefore, with infinitely successful iterations, K tends to infinity and so does the length of the curve L_K. This contradicts the assumption that R has a finite length. ∎

Algorithm 3 may not halt whenever the regular manifold connected to y_0 has an infinite length (e.g., consider $F(x) = (x_1 - \cos\left(2\pi/x_3^2\right), x_2 - \sin\left(2\pi/x_3^2\right)) = 0$ whose solution manifold is a spiral that collapses to a circle toward $x_3 = 0$). However, when finite precision floating point arithmetic is used and $\boldsymbol{x}^{\text{init}}$ is bounded, such a bounded curve with infinite length has some accumulation point outside the curve, and parallelotopes with floating point characteristics will eventually contain some singular points entailing stopping Algorithm 3.

3.2.3. Asymptotic convergence. The previous two theorems do not state anything about the ability of Algorithm 3 to actually enclose the curve. Indeed, Algorithm 3 may stop failing to build any parallelotope because $\tilde{x}_0$ is too close to a singularity. The following analysis shows that under appropriate hypotheses, Algorithm 3 actually succeeds in computing the component connected to y_0.

It is conducted for the simplest version of Algorithm 3. The Krawczyk operator is used since it is easier to analyze than the Hansen–Sengupta operator (the latter being known to be more efficient; see [26]). Furthermore, derivatives are all evaluated on the interval hull of the parallelotope. As a consequence, the Newton operator has here the following form:

$$\mathbf{K}(F, \widehat{\boldsymbol{x}}) = (\mathrm{Id}_E - \boldsymbol{G})(\mathrm{mid}\boldsymbol{w}) - (\boldsymbol{A}(\boldsymbol{w}) - E)(\boldsymbol{w} - \mathrm{mid}\boldsymbol{w}), \tag{3.16}$$

where $\widehat{\boldsymbol{x}} = (C, (\boldsymbol{u}, \boldsymbol{v}), \tilde{x})$, $\boldsymbol{A}(\boldsymbol{w}) = (\boldsymbol{A}_u, \boldsymbol{A}_v) = \boldsymbol{F}'(\square\widehat{\boldsymbol{x}})C$, $G(w) = F(Cw + \tilde{x})$, $E = (I \,|\, 0)$, and $\mathrm{Id}_E(w) = Ew$ is the truncated identity map associated to E.

For the asymptotic analysis, we neglect rounding errors due to the floating point arithmetic: We consider that C_k and $\boldsymbol{G}(\mathrm{mid}(\boldsymbol{w}))$ are computed with no rounding error, so $F'(\tilde{x}_{k-1})C_k$ is exactly the $n-1$ first lines of the identity matrix and $\boldsymbol{G}(\mathrm{mid}(\boldsymbol{w})) = G(\mathrm{mid}(\boldsymbol{w}))$. Furthermore, we consider that $\widehat{\boldsymbol{y}}_k = \tilde{x}_k$, and hence $F(\tilde{x}_k) = 0$, which is justified by the fact that Algorithm 1 converges unconditionally and very quickly to the unique solution contained in the parallelotopes' input and output sides. As a consequence, for any trial parallelotope built using (3.1) we have $F(c) = 0$. (The heuristic (3.6) is not considered here.) Note furthermore that since $\widehat{\boldsymbol{y}}_{k-1} = \tilde{x}_{k-1}$, we have $\boldsymbol{w}_k = (0, \ldots, 0, [0, h])$ in (3.1b) at all steps.

Then, the following theorem shows that there exists a threshold step size h_*, which depends only on the Lipschitz constant of $\boldsymbol{F}'$ and the norm of C, under which the certification procedure implemented in Algorithm 2 succeeds.

THEOREM 3.3. *Consider an initial parallelotope $\widehat{\boldsymbol{x}} = (C, (0, \ldots, 0, [0, h]), \tilde{x})$ built as in Algorithm 2, i.e., using (3.1). Note that by hypothesis $1 \leq \delta$ and $0 < \chi$. Suppose that the interval extensions $\boldsymbol{F}'$ of the derivatives of F are λ-Lipschitz continuous[5] with $\lambda > 0$,[6] $\|C\| \leq \kappa$, and $\chi \leq \frac{1}{6\kappa^2\lambda\delta}$, and define h_* as the greatest root of the quadratic polynomial $\kappa^2\lambda\delta(2h)^2 + \chi = (2h)$:*

[5] The natural interval extension of an expression involving elementary functions that are Lipschitz continuous is Lipschitz continuous; see Theorem 2.1.1 of [26].

[6] The limit case $\lambda = 0$, corresponding to a system of linear equations, is uninteresting and can be captured by considering arbitrarily small Lipschitz constants, entailing $h_* = \infty$ for nonsingular linear systems.

$$h_* := \frac{1+\sqrt{1-4\kappa^2\lambda\delta\chi}}{4\kappa^2\lambda\delta}. \tag{3.17}$$

Note that h_* *satisfies* $\frac{1}{4\kappa^2\lambda\delta} \le h_* \le \frac{1}{2\kappa^2\lambda\delta}$. *Finally suppose that* $\left(\frac{3}{4}\right)^{\overline{k}} < \frac{\chi}{4h_*}$ *(such a* $\overline{k}$ *exists since since* $\chi > 0$*) and* $\overline{\mu} \ge \frac{3}{4}$. *Then* $h \le h_*$ *implies N-Inflate*$(F, \widehat{\boldsymbol{x}})$ *succeeds.*

Proof. Denote the kth box computed by Algorithm 2 by $\boldsymbol{u}_k$ and write generally $\boldsymbol{w} = (\boldsymbol{u}, [0,h])$ and $\widehat{\boldsymbol{x}} = (C, \boldsymbol{w}, \tilde{x})$, so $\boldsymbol{w}_k = (\boldsymbol{u}_k, [0,h])$ and $\widehat{\boldsymbol{x}}_k = (C, \boldsymbol{w}_k, \tilde{x})$. Then $\boldsymbol{u}_0 = 0$ and $\boldsymbol{u}_{k+1} = \mathbf{K}_{\delta,\chi}(\boldsymbol{u}_k)$ with

$$\mathbf{K}_{\delta,\chi}(\boldsymbol{u}) = \mathrm{mid}\mathbf{K}(F, \widehat{\boldsymbol{x}}) + \delta(\mathbf{K}(F, \widehat{\boldsymbol{x}}) - \mathrm{mid}\mathbf{K}(F, \widehat{\boldsymbol{x}})) + \chi[-1,1] \tag{3.18}$$

$$= (\mathrm{Id}_E - \boldsymbol{G})(\mathrm{mid}\boldsymbol{w}) - \delta(\boldsymbol{A}(\boldsymbol{w}) - E)(\boldsymbol{w} - \mathrm{mid}\boldsymbol{w}) + \chi[-1,1]. \tag{3.19}$$

Since $\boldsymbol{F}'$ is λ-Lipschitz continuous, $\boldsymbol{A}(\boldsymbol{w})$ is $\kappa^2\lambda$-Lipscitz continuous. As a consequence,

$$\|\mathrm{wid}\boldsymbol{A}(\boldsymbol{w})\| \le 2d\big(\boldsymbol{A}(\boldsymbol{w}), \boldsymbol{A}(\mathrm{mid}\boldsymbol{w})\big) \le 2\kappa^2\lambda d(\boldsymbol{w}, \mathrm{mid}\boldsymbol{w}) = \kappa^2\lambda\|\mathrm{wid}\boldsymbol{w}\|. \tag{3.20}$$

Now define $\mathcal{U} := \{\boldsymbol{u} \in \mathbb{IR}^{n-1} : 0 \in \boldsymbol{u} \wedge \|\mathrm{wid}\boldsymbol{u}\| \le h_*\}$, which is a complete metric space for the usual distance for intervals. We are going to apply the Banach fixed point theorem to $\mathbf{K}_{\delta,\chi}$ inside $\mathcal{U}$.

First, we prove that $\mathbf{K}_{\delta,\chi}(\mathcal{U}) \subseteq \mathcal{U}$ by considering an arbitrary $\boldsymbol{u} \in \mathcal{U}$ and proving that $\mathbf{K}_{\delta,\chi}(\boldsymbol{u}) \in \mathcal{U}$. Note that $0 \in \boldsymbol{u} \iff c \in (C, (\boldsymbol{u}, [0,h]), \tilde{x})$, and by Theorem 2.1 $F(c) = 0$ implies $c \in (C, (\mathbf{K}_{\delta,\chi}(\boldsymbol{u}), [0,h]), \tilde{x})$. So eventually $0 \in \mathbf{K}_{\delta,\chi}(\boldsymbol{u})$. This furthermore entails $E = F'(c)C \in \boldsymbol{A}(\boldsymbol{w})$, and hence using (3.20)

$$\|A(\boldsymbol{w}) - E\| \le \|\mathrm{wid}\boldsymbol{A}(\boldsymbol{w})\| \le \kappa^2\lambda\|\mathrm{wid}\boldsymbol{w}\|. \tag{3.21}$$

Using standard rules for computing the radius of intervals, we obtain

$$\|\mathrm{wid}\mathbf{K}_{\delta,\chi}(\boldsymbol{u})\| = \delta\|A(\boldsymbol{w}) - E\|\,\|\mathrm{wid}\boldsymbol{w}\| + \chi \tag{3.22}$$

$$\le \kappa^2\lambda\delta\|\mathrm{wid}\boldsymbol{w}\|^2 + \chi \tag{3.23}$$

$$\le \kappa^2\lambda\delta\, h_*^2 + \chi. \tag{3.24}$$

Inequality (3.23) is a consequence of (3.21), and inequality (3.24) is a consequence of $\|\mathrm{wid}\boldsymbol{w}\| = \|\mathrm{wid}(\boldsymbol{u}, [0,h])\|$, $\|\mathrm{wid}\boldsymbol{u}\| \le h_*$, and $h \le h_*$. Now, recall that h_* is defined as the greatest root of the quadratic polynomial $\kappa^2\lambda\delta(2h)^2 + \chi = (2h)$. Since $\kappa^2\lambda\delta h_*^2 + \chi = \frac{1}{4}\big(\kappa^2\lambda\delta(2h_*)^2 + \chi\big) + \frac{3}{4}\chi$, we obtain $\|\mathrm{wid}\mathbf{K}_{\delta,\chi}(\boldsymbol{u})\| \le \frac{1}{2}h_* + \frac{3}{4}\chi$. Finally, $\frac{3}{4}\chi \le \frac{1}{8\kappa^2\lambda\delta} \le \frac{1}{2}h_*$, eventually entailing $\|\mathrm{wid}\mathbf{K}_{\delta,\chi}(\boldsymbol{u})\| \le h_*$.

Second, we prove that $\mathbf{K}_{\delta,\chi} : \mathcal{U} \to \mathcal{U}$ is contracting. Consider some arbitrary $\boldsymbol{u}, \boldsymbol{u}' \in \mathcal{U}$, and define $\tilde{\boldsymbol{u}} = \mathbf{K}_{\delta,\chi}(\boldsymbol{u})$ and $\tilde{\boldsymbol{u}}' = \mathbf{K}_{\delta,\chi}(\boldsymbol{u}')$. Then

$$d(\tilde{\boldsymbol{u}}, \tilde{\boldsymbol{u}}') = \|\mathrm{mid}\tilde{\boldsymbol{u}} - \mathrm{mid}\tilde{\boldsymbol{u}}'\| + \|\mathrm{rad}\tilde{\boldsymbol{u}} - \mathrm{rad}\tilde{\boldsymbol{u}}'\|. \tag{3.25}$$

We bound the two summands separately. For the first summand,

$$\|\mathrm{mid}\tilde{\boldsymbol{u}} - \mathrm{mid}\tilde{\boldsymbol{u}}'\| = \|(\mathrm{Id}_E - G)(\mathrm{mid}\boldsymbol{w}) - (\mathrm{Id}_E - G)(\mathrm{mid}\boldsymbol{w}')\| \tag{3.26}$$

$$\le \left(\max_{\substack{\xi = (1-t)\mathrm{mid}\boldsymbol{w} + t\mathrm{mid}\boldsymbol{w}' \\ t \in [0,1]}} \|E - G'(\xi)\|\right) \|\mathrm{mid}\boldsymbol{w} - \mathrm{mid}\boldsymbol{w}'\|. \tag{3.27}$$

Now, $\|E - G'(\xi)\| = \|G'(0) - G'(\xi)\| \le \kappa^2\lambda\|\xi\| \le \kappa^2\lambda h_* \le \frac{1}{2}$, hence eventually

$$\|\mathrm{mid}\tilde{\boldsymbol{u}} - \mathrm{mid}\tilde{\boldsymbol{u}}'\| \le \frac{1}{2}\|\mathrm{mid}\boldsymbol{w} - \mathrm{mid}\boldsymbol{w}'\|. \tag{3.28}$$

For the second summand, $\|\mathrm{rad}\tilde{\boldsymbol{u}} - \mathrm{rad}\tilde{\boldsymbol{u}}'\| = \delta\||\boldsymbol{A}(\boldsymbol{w}) - E|\mathrm{rad}\boldsymbol{w} - |\boldsymbol{A}(\boldsymbol{w}') - E|\mathrm{rad}\boldsymbol{w}'\| + \chi - \chi$, which, by adding the vector $0 = -|\boldsymbol{A}(\boldsymbol{w}) - E|\mathrm{rad}\boldsymbol{w}' + |\boldsymbol{A}(\boldsymbol{w}) - E|\mathrm{rad}\boldsymbol{w}'$ within the norm, and using the triangular inequality, is less than

$$\delta\||\boldsymbol{A}(\boldsymbol{w}) - E|(\mathrm{rad}\boldsymbol{w} - \mathrm{rad}\boldsymbol{w}')\| + \delta\|\big(|\boldsymbol{A}(\boldsymbol{w}) - E| - |\boldsymbol{A}(\boldsymbol{w}') - E|\big)\mathrm{rad}\boldsymbol{w}'\| \tag{3.29}$$

$$\leq \delta\|\boldsymbol{A}(\boldsymbol{w}) - E\|\|\mathrm{rad}\boldsymbol{w} - \mathrm{rad}\boldsymbol{w}'\| + \delta d\big(\boldsymbol{A}(\boldsymbol{w}) - E, \boldsymbol{A}(\boldsymbol{w}') - E\big)\|\mathrm{rad}\boldsymbol{w}'\| \tag{3.30}$$

$$\leq \delta\kappa^2\lambda\|\mathrm{wid}\boldsymbol{w}\|\|\mathrm{rad}\boldsymbol{w} - \mathrm{rad}\boldsymbol{w}'\| + \delta\kappa^2\lambda\|\mathrm{rad}\boldsymbol{w}'\|d(\boldsymbol{w}, \boldsymbol{w}') \tag{3.31}$$

$$\leq \frac{1}{2}\|\mathrm{rad}\boldsymbol{w} - \mathrm{rad}\boldsymbol{w}'\| + \frac{1}{4}d(\boldsymbol{w}, \boldsymbol{w}'), \tag{3.32}$$

where (3.30) follows from $||\boldsymbol{a}| - |\boldsymbol{a}'|| \leq d(\boldsymbol{a}, \boldsymbol{a}')$ (see Proposition 1.7.3 in [26]); (3.31) follows from (3.21), $d\big(\boldsymbol{A}(\boldsymbol{w}) - E, \boldsymbol{A}(\boldsymbol{w}') - E\big) = d\big(\boldsymbol{A}(\boldsymbol{w}), \boldsymbol{A}(\boldsymbol{w}')\big)$, and the Lipschitz constant of $\boldsymbol{A}(\boldsymbol{w})$; and (3.32) follows from the fact that both $\|\mathrm{wid}\boldsymbol{w}\|$ and $\|\mathrm{wid}\boldsymbol{w}'\| = 2\|\mathrm{rad}\boldsymbol{w}'\|$ are less than $h_* \leq \frac{1}{2\kappa^2\lambda\delta}$. Summing the upper bounds of the two summands yields $d(\tilde{\boldsymbol{u}}, \tilde{\boldsymbol{u}}') \leq \frac{3}{4}d(\boldsymbol{w}, \boldsymbol{w}')$, while $d(\boldsymbol{w}, \boldsymbol{w}') = d(\boldsymbol{u}, \boldsymbol{u}')$. We have proved that $\mathbf{K}_{\delta,\chi}$ is contracting and hence that $\boldsymbol{u}_k$ converges toward its unique fixed point.

Finally, we prove that Algorithm 2 stops with success. Since $\mathbf{K}_{\delta,\chi}$ is $\frac{3}{4}$-contracting, $d(\boldsymbol{u}_{k+2}, \boldsymbol{u}_{k+1}) \leq \frac{3}{4}d(\boldsymbol{u}_{k+1}, \boldsymbol{u}_k)$, so the test $\mu > \overline{\mu}$ at line 2.11 of Algorithm 2 never fails and the main loop of Algorithm 2 stops only when *success* is true. We now prove that *success* is actually true for $k = \overline{k}$ satisfying the constraint given in the statement. Define $\boldsymbol{u}'_{k+1} = \mathbf{K}(\boldsymbol{u}_k)$ so that

$$\boldsymbol{u}_{k+1} = \mathrm{mid}\boldsymbol{u}'_{k+1} + \delta(\boldsymbol{u}'_{k+1} - \mathrm{mid}\boldsymbol{u}'_{k+1}) + \chi[-1, 1], \tag{3.33}$$

as defined at lines 2.4 and 2.9 of Algorithm 2. We shall prove that $\boldsymbol{u}'_{k+1} \subseteq \mathrm{int}(\boldsymbol{u}_k)$, which is equivalent to $\mathrm{rad}\boldsymbol{u}'_{k+1} + |\mathrm{mid}\boldsymbol{u}'_{k+1} - \mathrm{mid}\boldsymbol{u}_k| < \mathrm{rad}\boldsymbol{u}_k$. Now, $d(\boldsymbol{u}_k, \boldsymbol{u}_{k+1}) \leq (\frac{3}{4})^k d(\boldsymbol{u}_0, \boldsymbol{u}_1) \leq (\frac{3}{4})^k h_* =: \epsilon$; hence, using (3.33), $\|\mathrm{mid}\boldsymbol{u}_k - \mathrm{mid}\boldsymbol{u}'_{k+1}\| \leq \epsilon$ and $\|\mathrm{rad}\boldsymbol{u}_k - (\delta\mathrm{rad}\boldsymbol{u}'_{k+1} + \frac{\chi}{2})\| \leq \epsilon$, which implies $\mathrm{rad}\boldsymbol{u}'_{k+1} \leq \mathrm{rad}\boldsymbol{u}_k + \epsilon - \frac{\chi}{2}$. We obtain finally $\mathrm{rad}\boldsymbol{u}'_{k+1} + |\mathrm{mid}\boldsymbol{u}'_{k+1} - \mathrm{mid}\boldsymbol{u}_k| \leq \mathrm{rad}\boldsymbol{u}_k + \epsilon - \frac{\chi}{2} + \epsilon$, which is strictly less than $\mathrm{rad}\boldsymbol{u}_k$ provided that $2\epsilon < \frac{\chi}{2}$, holding for $k = \overline{k}$ by assumption. ∎

Note that when the map F is close to singular, $\|C\|$ is very large. Therefore both χ and h_* have to be small so as to satisfy $\chi \leq \frac{1}{6\kappa^2\lambda\delta}$ and $h_* \leq \frac{1}{2\kappa^2\lambda\delta}$. On the other hand, if the system is nonsingular inside a neighborhood of the manifold connected to $\tilde{x}_0$ and included inside $\boldsymbol{x}^{\mathrm{init}}$, then it is compact and $\|C\|$ reaches its lower bound, which therefore has to be strictly positive. As a consequence, there exists a minimal step size which will allow continuing the full curve. Note finally that the bound

$$\overline{k} = \left\lceil \frac{\log 4h_* - \log\chi}{\log 4 - \log 3} \right\rceil \tag{3.34}$$

provided by Theorem 3.3 is quite pessimistic. On the one hand, it shows that the static absolute inflation $\chi > 0$ is enough to enforce the certification. On the other hand, the static relative inflation $\delta \geq 1$, although not necessary, strongly speeds up the certification.

3.3. Limitations of ParCont. Algorithm 3 is restricted to certifying nonsingular curves. Indeed, the certification of existence comes with a certification of regularity of the system's derivatives. If the curve contains a singular point, then Algorithm 2 will fail each time a parallelotope contains this singular point. Therefore, the step

size will keep on decreasing, leading to a sequence of smaller and smaller parallelotopes, which will converge to this singularity, eventually halting when the prescribed minimal step size is reached.

When the curve is regular, Theorem 3.3 shows that there exists a step size h_* that enforces the success of all certifications. However, this analysis does not take into account the finite precision floating point computations. In practice, the required step size could be too small with respect to the computational precision, leading to a failure similar to the singular case. When using standard floating point computations, the computational precision depends on the magnitude of the involved numbers, hence translating a problem can impact the resolution using floating point numbers. This is, however, inherent to all floating point implementations of numerical algorithms.

Noncertified continuation methods can benefit from quasi-Newton-like algorithms, which allow handling very large systems. Each step of Algorithm 3 involves operations of cubic complexity with respect to the number of variables which are involved in the interval Newton steps. This cubic complexity restricts the scope of Algorithm 3 to smaller problems than noncertified methods.

4. Experiments. ParCont has been implemented in C++ using the RealPaver [15] API, implementing routines such as interval Newton methods and many other constraint solving techniques, which uses Gaol [14] for interval arithmetic and Lapack [2] for linear algebra (matrix inversions, kernel computations, etc.). All the experiments have been run on a machine under Linux Ubuntu version 11.10, with an Intel i5-2400 3.10-GHz processor and 4 GB of RAM. In the following experiments, the parameters of N-Inflate and Contract are set as described in section 2 (i.e., $\chi = 10^{-12}$, $\delta = 1.1$, $\overline{k} = 15$, $\overline{\lambda} = 1$)—except when explicitly mentioned—and as described in section 3.1 for parameters of the main algorithm (i.e., $\alpha = 0.5$, $\beta = 1.1$, $\underline{h} = 10^{-8}$). The derivatives in N-Inflate are by default computed using the box hull of the input parallelotope. In addition, some results are shown when considering two other ways of computing the derivatives, namely, mean value form of the derivatives (hence using the second derivatives of F evaluated at $\square\widehat{\boldsymbol{x}}$) and formal form (a formal condensed expression of $G(w)$ is computed beforehand).

Measuring the performances of ParCont by counting the number of produced parallelelotopes is not relevant. Indeed, an iteration of the algorithm makes use of the iterative procedures N-Inflate and Contract whose induced computational cost is not fixed. Moreover, unsuccessful steps have to be counted since they cost as much as successful ones. Hence, it is more suitable to count the number of iterations performed by N-Inflate and Contract. The time complexity of both N-Inflate and Contract is $O(n^3)$ per iteration, which corresponds to the interval matrix multiplication performed when evaluating the derivatives in the auxiliary basis. The complexity of Contract can be reduced to $O(n^2)$ by reusing the last interval evaluation of the derivatives in N-Inflate. Thus, the number of N-Inflate iterations is used as a performance measure. In the following, both computational times and number of N-Inflate iterations are presented.

Our goal is to illustrate the strengths (and limitations) of parallelotopes with respect to boxes. To this end, we propose a comparison of ParCont with a continuation based on boxes as in [22]. The algorithm in [22] can be seen as an intervalization of the parameter-embedding continuation: boxes are constructed along the manifold by first selecting one of its components as a parameter. The certification process is based on a parametric interval Krawczyk operator and a generate-and-try inflation. However, from the information provided in [22] we have not been able to tune the inflation process correctly in order to reach any satisfying results. Therefore, we

have implemented a version of this algorithm, called BoxCont, replacing the inflation process by the one defined in section 2 and using a heuristic for guessing good initial boxes as in (3.6).

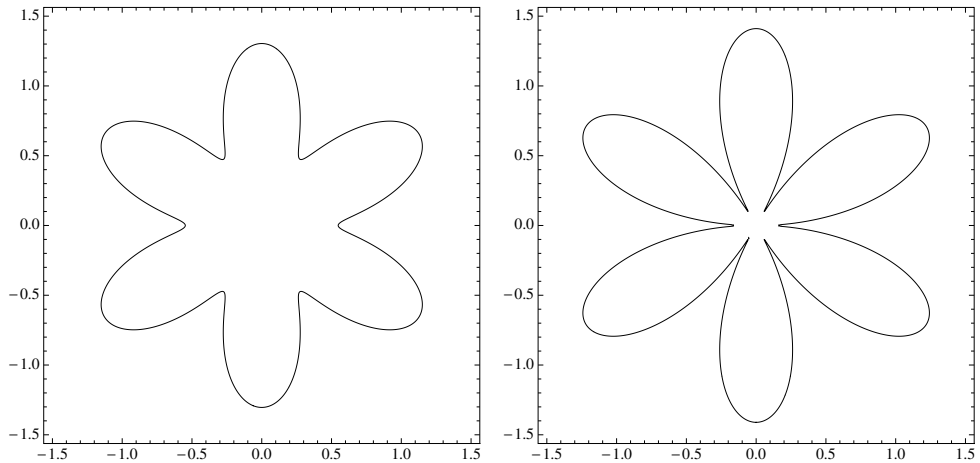
Different experiments are presented in the following. First, the two methods are applied to a problem with increasingly difficult topology. Second, another problem with increasingly difficult conditioning is proposed. Third, the algorithms are applied to track a particular manifold embedded in higher and higher dimensional space. Last, two applications of certified continuation are presented: homotopy continuation for tracking the roots of a complex polynomial—results are compared with another certified method from the literature—and robot command synthesis.

4.1. Influence of the manifold topology. Consider the following system depending on the parameter $\epsilon \in [0, 1]$:

$$x^8 - (1-\epsilon)x^6 + 4x^6y^2 - (3+15\epsilon)x^4y^2 + 6x^4y^4 - (3-15\epsilon)x^2y^4 + 4x^2y^6 - (1+\epsilon)y^6 + y^8. \tag{4.1}$$

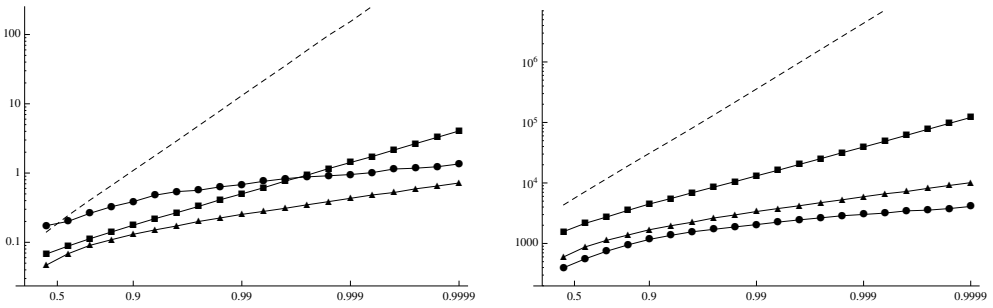
The solution manifold of this system has a six-petals-flower shape. The parameter ϵ controls the length of the petals, hence the acuity of the curvature between two consecutive petals (see Figure 4.1). In particular $\epsilon = 0$ corresponds to a circle and $\epsilon = 1$ leads to six petals connected by a singularity at (0,0). We solve several instances of (4.1) using varying ϵ values closer and closer to 1 and starting from the initial solution $(\sqrt{1-\epsilon}, 0)^T$. The nonlinearity of this problem is strong enough to allow using $\chi = 0$ in N-Inflate and $\underline{h} \simeq 0$, hence enabling tackling of very strong curvatures.

Figure 4.2 confirms that as ϵ increases, more computations are required. This is seen as the number of N-Inflate iterations and hence computational times are increasing with ϵ. The continuation is indeed adapting to the acuity of the curvature by reducing the average step length, hence increasing the number of iterations. BoxCont appears to adapt badly to the difficult topology, whereas ParCont handles them well. Indeed, the growth factor in number of iterations as ϵ increases is much higher when considering simple box continuation. On the other hand, manipulating



(a) $\epsilon = 0.7$ (b) $\epsilon = 0.99$

FIG. 4.1. *Solution manifold of* (4.1).



(a) Time versus ϵ (b) N-Inflate iterations versus ϵ

FIG. 4.2. *Comparison of ParCont (plain lines) with BoxCont (dashed line) on the manifold induced by* (4.1) *(logarithmic scales). Square, triangle, and circle markers represent different evaluation of the derivatives, respectively: hull of parallelotopes (default mode), mean value form, and formal form.*

parallelotopes is computationally more costly, hence the computation times for the problems with low ϵ are quite comparable. Nevertheless, Figure 4.2 clearly shows that parallelotopes asymptotically have better timings: $t_{\text{par}} \approx 0.062(1-\epsilon)^{-0.45}$ while $t_{\text{box}} \approx 0.099(1-\epsilon)^{-1.06}$, which are quite accurate approximations in view of the almost linear curves in the log scale graphic.

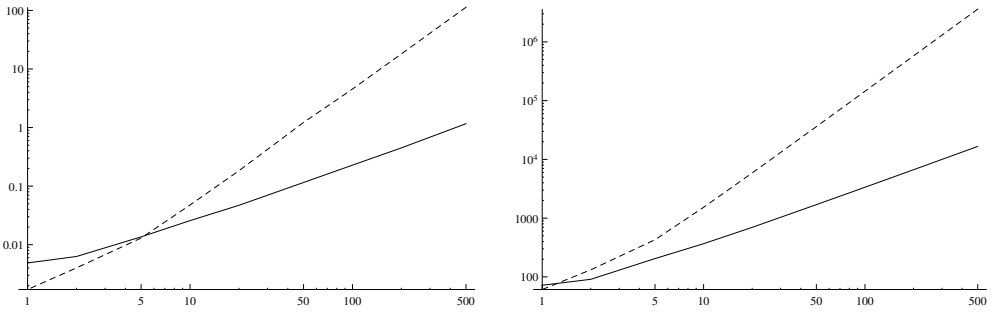
Exploiting better parallelotopes using more accurate derivative computations improves the asymptotic behavior of ParCont, as shown on Figure 4.2. Computing derivatives using a formal auxiliary expression of (4.1) strengthens the adaptation to strong curvatures. On the other hand, this expression is complex, hence its evaluation and differentiation are more computationally expensive. This way of computing derivatives is worth using only when the curvatures are very sharp. Computing the derivatives using a mean form requires second derivatives of (4.1). Nevertheless, the small number of variables makes this technique computationally reasonable and overall very efficient.

4.2. Influence of the conditioning. Consider the following problem:

$$\begin{pmatrix} (x_1+\epsilon)^2 + x_2^2 + x_3^2 - (1+\epsilon^2) \\ (x_1-\epsilon)^2 + x_2^2 + x_3^2 - (1+\epsilon^2) \end{pmatrix} = 0. \tag{4.2}$$

The problem in (4.2) consists of two spheres, at distance 2ϵ from each other, whose intersection is the unit circle in the plane (x_2, x_3). By shifting ϵ toward 0, the system becomes more and more close to be singular. The norm of matrices C are increasing, following a factor $\simeq 1/\epsilon$, whereas the manifold of solutions remains the same. We solve this problem for different ϵ values, with $(0, 1, 0)$ as starting points, and compare the performances of BoxCont and ParCont.

Figure 4.3 shows, as expected, that performances deteriorate as the problem becomes more singular. However, BoxCont suffers more from this difficulty than ParCont. Whereas the usage of BoxCont seems reasonable when the problem is well defined, showing better timings than ParCont, its performances significantly reduce with the conditioning of the system. The growths in computational time and in N-Inflate iterations in ParCont are indeed much lower than in BoxCont. Results for sharper derivatives evaluations are not reported here. There is in fact no gain in using these techniques here since the manifold to track is simple.



(a) Time versus $1/\epsilon$ (b) N-Inflate iterations versus $1/\epsilon$

FIG. 4.3. *Comparison of ParCont (solid line) with BoxCont (dashed line) on the manifold induced by* (4.2) *(logarithmic scales).*

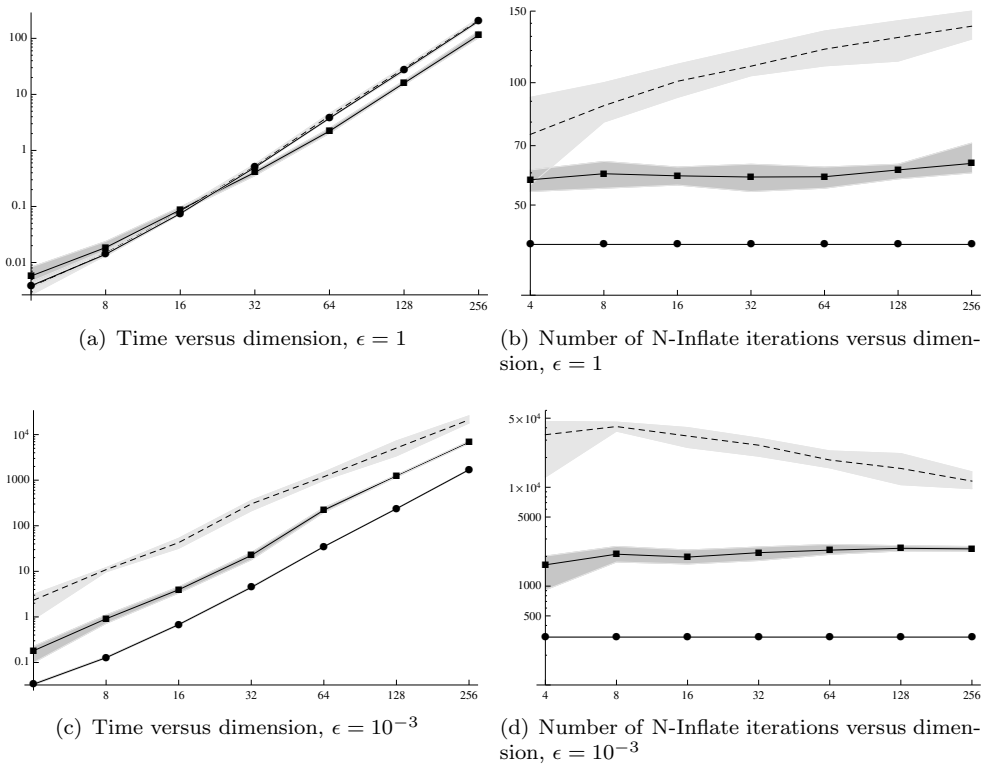
4.3. Influence of the embedding space dimension. Consider the following system depending on the parameters $n \in [2, +\infty)$ and $\epsilon \in (0, 1]$:

$$\begin{pmatrix} x^T Q' x - 1 \\ A' x \end{pmatrix} = 0 \tag{4.3}$$

with $Q' = M^T Q M$, $A' = AM$, $Q = \begin{bmatrix} I_{n-1\times n-1} & 0 \\ 0 & \epsilon \end{bmatrix}$, $A = (I_{n-2\times n-2}|0|0)$ and M an orthonormal matrix. With $M = I$, the solution manifold of (4.3) consists in a two-dimensional ellipse embedded in the subspace spanned by (e_{n-1}, e_n) of $\mathbb{R}^n$ and whose shape is controlled by ϵ. ($\epsilon = 1$ yields a circle, while $\epsilon \to 0$ yields an ellipse more and more stretched toward infinity.) Applying an isometry with $M \neq I$ enforces the different variables to be nontrivially involved in the solutions and preserves the length and shape of the manifold. The following results are given upon a set of 10 random transformation matrices M per tested dimension, obtained by applying a Gramm–Schmidt orthogonalization to random matrices. Initial solutions are $M^T(0, 0, \ldots, 0, 1, 0)$. For each dimension, Figure 4.4 shows the average timings and number of N-Inflate iterations.

Consider first the results for $\epsilon = 1$ (i.e., a circle) reported in Figures 4.4(a) and 4.4(b). They show that the two methods have similar computational time as the dimension increases. However, ParCont performs fewer steps than BoxCont. The latter also shows a growth of the number of N-Inflate iterations as the dimension increases that impacts its computational time. (ParCont runs faster for dimensions above 32.) The number of N-Inflate iterations of ParCont remains overall stable.

Consider now the results for $\epsilon = 10^{-3}$ (i.e., a stretched ellipse) in Figures 4.4(c) and 4.4(d). It appears ParCont requires less computation time than BoxCont and its number of N-Inflate iterations remains stable. However, BoxCont shows a growth of computation time slightly lower than ParCont. This is explained by a surprising decrease of the number of steps as the dimension increases. This effect is due to the parameter embedding approach used in BoxCont: moving along one parameter with step size h entails that the certified manifold inside a box produced by N-Inflate is proportional to approximately $h\sqrt{n}$ (the diagonal of an n-dimensional box of width h). Hence, since the total length of the manifold remains constant by construction and the average step length remains quite stable as the dimension increases, the corresponding reduction of the number of N-Inflate iterations, due to a reduction of the number of steps, can be observed.



(a) Time versus dimension, $\epsilon = 1$

(b) Number of N-Inflate iterations versus dimension, $\epsilon = 1$

(c) Time versus dimension, $\epsilon = 10^{-3}$

(d) Number of N-Inflate iterations versus dimension, $\epsilon = 10^{-3}$

FIG. 4.4. *Results of ParCont (solid lines) and BoxCont (dashed lines) on the manifold induced by* (4.3) *(axis y in logarithmic scale). Gray areas represent the range (maximum and minimum) of the different measures among the* 10 *random transformation matrices M. Square and circle markers represent respectively the differentiation using the hull of parallelotopes (default mode) and a formal expression of* $G'(u)$.

Computing derivatives using a mean form, hence computing second-order derivatives, is not reasonable here due to the large dimension of the problems. (It is in fact much slower than the default ParCont after dimension 8.) On the other hand, on this particular problem, we can construct a very simple formal expression of the derivatives in the auxiliary space of parallelotopes, using

$$G'(w) = \begin{pmatrix} 2w^T(C^TQ'C) + (\tilde{x}^TQ')C \\ A'C \end{pmatrix}. \tag{4.4}$$

This latter expression involves more matrix-matrix multiplications than $F'(\square\widehat{\boldsymbol{x}})C$, hence increasing the cost of the derivative evaluation. However the results become totally stable: For any rotation and any dimension, the number of N-Inflate iterations (and number of steps) remains the same. Furthermore, the overall computational time is sensibly decreased with respect to the default derivative evaluation.

To conclude, using parallelotopes instead of boxes guarantees stability in following a manifold pushed into several dimensions, or, otherwise said, the behavior of ParCont depends essentially on the manifold shape and not on its embedding dimension. Coupled with the results from the previous experiments, the use of parallelotopes ensures the stability of the continuation process, adapts well to difficult topologies, and is less sensitive to the conditioning of the system.

4.4. Homotopy continuation. ParCont is used here to solve a polynomial system through homotopy. Its results are compared against the method NAG4M2 [3], a certified homotopy continuation implemented in the software Macaulay2 [16].

The problem Katsura_n is one of the standard scalable benchmarks for homotopy solving methods. It consists of a system of $n+1$ variables $(z_0, z_1, \ldots, z_n) \in \mathbb{C}^{n+1}$. For instance, $\text{Katsura}_2(z)$ consists of

$$\text{Katsura}_2(z) = \begin{pmatrix} 1 - z_0 - 2z_1 - 2z_2 \\ z_1 - 2z_0 z_1 - 2z_1 z_2 \\ z_0 - z_0^2 - 2z_1^2 - 2z_2^2 \end{pmatrix} = 0. \tag{4.5}$$

The aim is to find all the complex roots of Katsura_n. Therefore, the system to solve is the following linear homotopy:

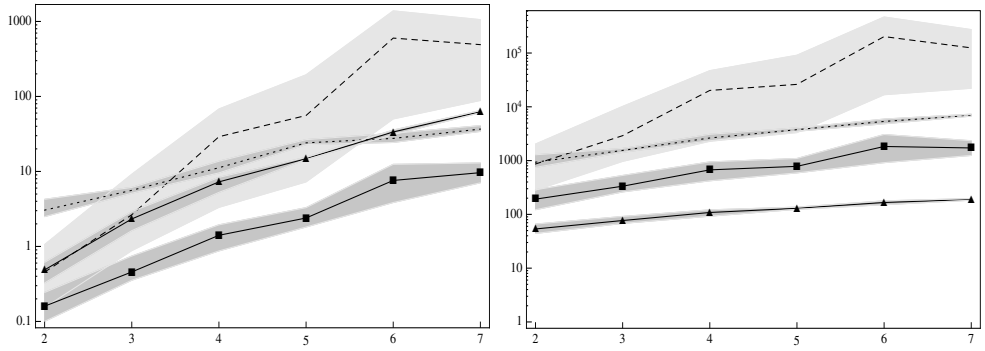
$$(1-t)h_n(z) + t\mu\text{Katsura}_n(z) = 0, \tag{4.6}$$

where $\mu \in \mathbb{C}$ is randomly selected on the complex circle, $t \in [0,1]$ the induced homotopy variable, and h_n an initial system matching the degrees of Katsura_n, that is, $h_n(z) = \left(z_0 - 1, z_1^2 - 1, \ldots, z_n^2 - 1\right)^T$. In order to solve the system with ParCont, the real and imaginary parts have to be separated, doubling the number of variables and equations. Hence, with z in $\mathbb{C}^{n+1}$, there are $2(n+1)+1$ variables (real and imaginary part of z plus t) and $2(n+1)$ equations. Input solutions are roots of h_n (there are 2^n such roots) and $t = 0$. The aim is to follow the paths of solutions of (4.6) from $t = 0$ in order to reach solutions to Katsura_n at $t = 1$. Hence, the domain of the variables is not limited, except for t which must remain in $[0,1]$.

The results obtained by ParCont are compared against those of the certified homotopy tracking method NAG4M2 from [3]. There is no performance measure in NAG4M2 that can be compared with the number of N-Inflate iterations. Therefore, we instead compare the computational time and the number of steps of the main algorithm (counting also unsuccessful ones for ParCont). Indeed, a step of NAG4M2 builds a new solution, whereas a step of ParCont is an attempt to build a new parallelotope. We randomly selected a sample of five values of μ such that no one leads to an incomplete homotopy or diverging paths. For each value of μ and each root of the polynomial system, we measured the number of steps (including unsuccessful steps for ParCont) and computation time. Figure 4.5 reports mean, min, and max values of these measures for different problem sizes.

Figure 4.5 shows that the performances of BoxCont are worse, on average, than ParCont and NAG4M2 with n greater than 4. Noticeably, BoxCont shows high instability for the different values of μ both in iterations and computation time. Undeniably, the difficulty to adapt well to the topology of solution paths makes the use of boxes unadvised compared to parallelotopes on this benchmark. Considering ParCont and NAG4M2, Figure 4.5 shows that the former requires fewer steps and less computational time than the latter. However, whereas the growth rate of steps is approximately proportional between the two methods, computational time of ParCont appears to increase more quickly than NAG4M2 when the dimension increases. In addition, NAG4M2 appears to be more stable than ParCont between the different values of μ.

Formal expressions in auxiliary space of (4.6) are too large to be handled by our implementation, making it impossible to exploit better derivatives with this technique. Mean value form can still be used. The improved derivative computation depicted in Figure 4.5 appears to be more stable than default ParCont. However, computing



(a) Time versus dimension (b) Number of steps versus dimension

FIG. 4.5. *Comparison of the different methods on the manifold induced by* (4.6) *(axis y in logarithmic scale). Solid, dashed, and dotted lines corresponds respectively to ParCont, BoxCont, and NAG4M2. Gray areas represent the range (maximum and minimum) of the different measures among the μ values. Square and triangle markers represent respectively differentiation using the hull of parallelotopes (default mode) and mean value form.*

second-order derivatives become too computationally expensive when the dimension increases: computation times are always higher than default ParCont and higher than NAG4M2 after $n = 6$.

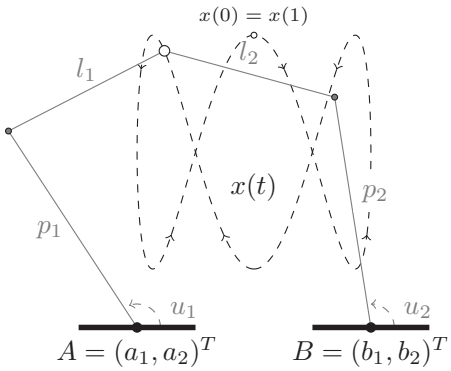
All in all, one can see that ParCont is competitive with NAG4M2. The latter makes use of techniques well suited for homotopy on polynomial systems (complex arithmetic computations and use of homogeneous systems and projective spaces), ensuring a more stable behavior with respect to the value of μ. Hence, it seems promising to incorporate these techniques in ParCont, for instance, by using complex interval arithmetic as in [27]. Finally, better derivative computation increase the stability of ParCont but would require some dedicated and computationally cheaper evaluation techniques to handle high dimensional problems.

4.5. Command synthesis. A robot is defined by a system $F(x, u) = 0$ called the kinematic model, where $F : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$, $x \in \mathbb{R}^n$, is the pose of the end effectors of the robot, and $u \in \mathbb{R}^n$ are the command variables. Given a trajectory for the effector defined by $x : [0, 1] \to \mathbb{R}^n$ (from $x(0)$ to $x(1)$), the aim of command synthesis is to find the command function $u(t)$ required to follow this trajectory, that is, all the solutions to $F(x(t), u) = 0$, starting from one given input control $u(0) = u^0$ satisfying $F(x(0), u^0) = 0$. Obtaining a certified continuous command function is critical in order to avoid false trajectory tracking or inconsistent sequence of commands. Therefore, ParCont is suitable to solve this problem

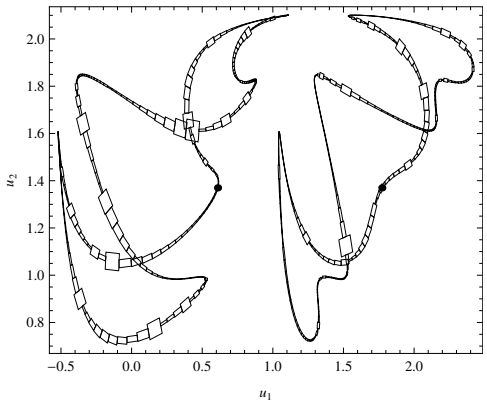
Consider the robot as presented in Figure 4.6(a), called $\underline{\text{R}}\text{RRRR}\underline{\text{R}}$. The end effector is controlled by two articulated arms, each made of two parts linked by a revolute joint. The two arms are controlled by their angle u_1 and u_2. One can easily see that the pose of the end effector may be attained by two different commands per articulated arm. The kinematic model of this robot is

$$F(x,u) = \begin{pmatrix} (x_1 - (\cos(u_1)p_1 + a_1))^2 + (x_2 - (\sin(u_1)p_1 + a_2))^2 - l_1^2 \\ (x_1 - (\cos(u_2)p_2 + b_1))^2 + (x_2 - (\sin(u_2)p_2 + b_2))^2 - l_2^2 \end{pmatrix} = 0, \tag{4.7}$$

where $x = (x_1, x_2)^T$ is the position of the end effector of the robot, $A = (a_1, a_2)^T$ (respectively, $B = (b_1, b_2)^T$) is the position of the fixed joint of the first



(a) Scheme of the robot. The trajectory $x(t)$ is represented in dashed lines.



(b) Computed command sequence projected in the command space. Initial commands u^0 are represented by points.

FIG. 4.6. *Example of a two-armed robot* $\underline{R}RRR\underline{R}$.

(respectively, second) articulated arm. The first and second parts of the first arm are respectively of length p_1 and l_1 (p_2 and l_2 for the second arm). For this benchmark, $x(t) = (\sin(2\pi t), \cos(3(2\pi t)) + 1.5)$, with t in $[0, 1]$ (shown in Figure 4.6(a)).

The geometric parameters of the robot are $A = (-1, 0)^T$, $B = (1, 0)^T$, p_1 and p_2 set to 2, and l_1 and l_2 set to 1.5. Given the system $F(x, u)$ and the initial pose $x(0) = (0, 2.5)^T$, ParCont is applied with two initial commands $u^0 = (0.6085, 1.3699)^T$ and $u^0 = (1.7716, 1.3699)^T$. The sequence of commands following the trajectory obtained with ParCont is represented in Figure 4.6(b) and fully certified to be continuous. Each command sequence has been computed in approximately 0.14 seconds.

5. Conclusion. The proposed method, ParCont, uses parallelotope continuation in order to fully certify the continuation of a curve defined by a system of $n - 1$ equations and n unknowns. It is simple to implement, given the use of any interval arithmetic library, and requires only a very few parameters to be tuned, some robust values of these parameters being proposed. Experiments have demonstrated its superiority over a variant of the algorithm that uses boxes instead of parallelotopes. In addition, experiments have shown that the proposed algorithm compares favorably with respect to the Macaulay2 [16] implementation of NAG4M2 [3] for certified polynomial homotopy continuation.

Due to its applicability to a broad range of one-manifold continuation problems, ParCont can be improved by coupling techniques dedicated to specific categories of problems. For instance, its performances for polynomial homotopy continuation can be improved using complex interval arithmetic. Noteworthily, using the framework defined in [13], ParCont is readily applicable to one parameter continuation of solutions to ordinary differential equations, like periodic solutions or solutions to boundary value problems. Nevertheless, k-manifolds are out of range of ParCont. Although there are no theoretical limitations in building a certified enclosure of a portion of such manifolds with a parellelotope, guaranteeing the overall connectivity of the continuation is challenging since it requires rethinking the merging of successive parallelotopes.

Appendix A. Technical lemma.

LEMMA A.1. *Let $\alpha : [0, L_\alpha] \to \mathbb{R}^{n+1}$ and $\beta : [0, L_\beta] \to \mathbb{R}^{n+1}$ be continuously differentiable curves (whenever t is some interval endpoint, right or left derivatives are considered), and let $F : \mathbb{R}^{n+1} \to \mathbb{R}^n$ be continuously differentiable. Suppose the following:*

1. *For all t in $[0, m]$ with $m = \min\{L_\alpha, L_\beta\}$,*
   (i) *$F(\alpha(t)) = 0$ and $F(\beta(t)) = 0$;*
   (ii) *$F'(\alpha(t))$ and $F'(\beta(t))$ are full (row) rank;*
   (iii) *$\|\alpha'(t)\| = 1$ and $\|\beta'(t)\| = 1$.*
2. *$\alpha(0) = \beta(0)$ and $\alpha'(0) = \beta'(0)$.*

Then, $\alpha(t) = \beta(t)$, and thus $\alpha'(t) = \beta'(t)$, holds for all $t \in [0, m]$.

Proof. First, we prove that if $\alpha(t^*) = \beta(t^*) =: x^*$ and $\alpha'(t^*) = \beta'(t^*)$ both hold for some $t^* \in (0, m)$, then $\alpha(t) = \beta(t)$ (and therefore $\alpha'(t) = \beta'(t)$) holds in a neighborhood of t^*. Since $F'(x^*)$ is full rank, it contains a nonsingular $n \times n$ submatrix. Without loss of generality, suppose that the last n columns of $F'(x^*)$ form a nonsingular square matrix, i.e., $F'(x^*) = (a|A)$ with $A \in \mathbb{R}^{n\times n}$ nonsingular and $a \in \mathbb{R}^n$. Therefore, by the implicit function theorem, there exists a neighborhood U of x^* and a differentiable function $\phi : \mathbb{R} \longrightarrow \mathbb{R}^n$ such that $x \in U$ and $F(x) = 0$ is equivalent to $x = (x_1, \phi(x_1))$.

Since $F(\alpha(t)) = 0$ we have $(a|A)\,\alpha'(t^*) = 0$, and thus $\alpha'_{>1}(t^*) = -\alpha'_1(t^*)A^{-1}a$ (where the vector α' is split into $(\alpha_1, \alpha'_{>1})$). Therefore $\alpha'_1(t^*) = 0$ entails $\alpha'(t^*) = 0$, which contradicts $\|\alpha'(t^*)\| = 1$. As a conclusion, $\alpha'_1(t^*) \neq 0$ and the inverse function theorem proves there exists a neighborhood X_1 of x_1^* and a differentiable function α_1^{-1} defined inside X_1 such that $\alpha_1(\alpha_1^{-1}(x_1)) = x_1$. Since α, α_1^{-1}, and β are continuous, there exists a neighborhood T^* of t^* such that $\beta_1(T^*) \subseteq X_1$, $\alpha(\alpha_1^{-1}(\beta_1(T^*))) \subseteq U$, and $\beta(T^*) \subseteq U$. Fix an arbitrary $t \in T^*$ and write $u := \alpha(\tau(t))$ with $\tau(t) := \alpha_1^{-1}(\beta_1(t))$ and $v := \beta(t)$. Then $u_1 = v_1$ while $F(u) = 0$ and $F(v) = 0$ entail $u = (u_1, \phi(u_1))$ and $v = (v_1, \phi(v_1))$. Therefore $u = v$, i.e., $\alpha(\tau(t)) = \beta(t)$.

Differentiating $\alpha(\tau(t)) = \beta(t)$, we obtain $\alpha'(\tau(t))\tau'(t) = \beta'(t)$ for all $t \in T^*$. Since both $\|\alpha'(\tau(t))\| = 1$ and $\|\beta'(t)\| = 1$, we have $|\tau'(t)| = 1$. Note that $\alpha_1(t^*) = \beta_1(t^*) = x_1^*$ entails $\tau(t^*) = t^*$, and then $\alpha'(t^*) = \beta'(t^*)$ entails $\tau'(t^*) = 1$. Since τ' is continuous we have $\tau'(t) = 1$ for all $t \in T^*$, and eventually $\tau(t) = t$ and $\alpha(t) = \beta(t)$ (and therefore $\alpha'(t) = \beta'(t)$).

This argument based on the implicit function theorem also holds for $t = 0$ and $t = m$ and furthermore proves that α and β can be extended to a neighborhood of $[0, m]$ with the same properties.

Second, define $T = \{t \in [0, m] : \alpha(t) = \beta(t) \wedge \alpha'(t) = \beta'(t)\}$. By the first part of the proof, T is open in the topology of $[0, m]$. By the continuity of $\alpha(t) - \beta(t)$ and its derivative, T is also closed. Since $[0, m]$ is connected, it has only two clopen subsets, namely, $\emptyset$ and $[0, m]$. Since $0 \in T$, we finally have $T = [0, m]$. ∎

REFERENCES

[1] E. L. ALLGOWER AND K. GEORG, *Introduction to Numerical Continuation Methods*, SIAM, Philadelphia, 2003.

[2] E. ANDERSON, Z. BAI, C. BISCHOF, S. BLACKFORD, J. DEMMEL, J. DONGARRA, J. DU CROZ, A. GREENBAUM, S. HAMMARLING, A. MCKENNEY, AND D. SORENSEN, *LAPACK Users' Guide*, 3rd ed., Philadelphia, 1999.

[3] C. BELTRÁN AND A. LEYKIN, *Certified numerical homotopy tracking*, Experiment. Math., 21 (2012), pp. 69–83.

[4] W.-J. Beyn, C. Effenberger, and D. Kressner, *Continuation of eigenvalues and invariant pairs for parameterized nonlinear eigenvalue problems*, Numer. Math., 119 (2011), pp. 489–516.
[5] J.-D. Boissonnat and A. Ghosh, *Triangulating smooth submanifolds with light scaffolding*, Math. Comput. Sci., 4 (2010), pp. 431–461.
[6] K. I. Dickson, C. T. Kelley, I. C. F. Ipsen, and I. G. Kevrekidis, *Condition estimates for pseudo-arclength continuation*, SIAM J. Numer. Anal., 45 (2007), pp. 263–276.
[7] D. Faudot and D. Michelucci, *A new robust algorithm to trace curves*, Reliab. Comput., 13 (2007), pp. 309–324.
[8] H. Federer, *Curvature Measures*, Trans. Amer. Math. Soc., 93 (1959), pp. 418–491.
[9] D. Goldberg, *What every computer scientist should know about floating-point arithmetic*, Comput. Surveys, 23 (1991), pp. 5–48.
[10] A. Goldsztejn, *A branch and prune algorithm for the approximation of non-linear AE-solution sets*, in Proceedings of ACM SAC 2006, pp. 1650–1654.
[11] A. Goldsztejn and L. Granvilliers, *A new framework for sharp and efficient resolution of* NCSP *with manifolds of solutions*, Constraints, 15 (2010), pp. 190–212.
[12] A. Goldsztejn and L. Jaulin, *Inner approximation of the range of vector-valued functions*, Reliab. Comput., 14 (2010), pp. 1–23.
[13] A. Goldsztejn, O. Mullier, D. Eveillard, and H. Hosobe, *Including ordinary differential equations based constraints in the standard CP framework*, in Proceedings of CP 2010, Lecture Notes in Comput. Sci. 6308, Springer, New York, 2010, pp. 221–235.
[14] F. Goualard, *Gaol* 3.1.1*: Not Just Another Interval Arithmetic Library*, Laboratoire d'Informatique de Nantes-Atlantique, version 4.0, 2006.
[15] L. Granvilliers and F. Benhamou, *Algorithm* 852*: Realpaver: An interval solver using constraint satisfaction techniques*, ACM Trans. Math. Softw., 32 (2006), pp. 138–156.
[16] D. R. Grayson and M. E. Stillman, *Macaulay*2, http://www.math.uiuc.edu/Macaulay2/.
[17] M. E. Henderson, *Multiple parameter continuation: Computing implicitly defined k-manifolds*, Internat. J. Bifur. Chaos, 12 (2002), pp. 451–476.
[18] C. Hillermeier, *Generalized homotopy approach to multiobjective optimization*, J. Optim. Theory Appl., 110 (2001), pp. 557–583.
[19] D. Ishii, A. Goldsztejn, and C. Jermann, *Interval-Based Projection Method for Under-Constrained Numerical Systems*, Constraints, 17 (2012), pp. 432–460.
[20] L. Jaulin, M. Kieffer, O. Didrit, and E. Walter, *Applied Interval Analysis with Examples in Parameter and State Estimation, Robust Control and Robotics*, Springer, New York, 2001.
[21] R. B. Kearfott, *Interval Computations: Introduction, Uses, and Resources*, Euromath Bulletin, 2 (1996), pp. 95–112.
[22] R. B. Kearfott and Z. Xing, *An interval step control for continuation methods*, SIAM J. Numer. Anal., 31 (1994), pp. pp. 892–914.
[23] O. Knueppel, *PROFIL/BIAS—A fast interval library*, Computing, 53 (1994), pp. 277–287.
[24] J. P. Merlet, *Parallel Robots*, Kluwer, Dordrecht, the Netherlands, 2000.
[25] R. Moore, *Interval Analysis*, Prentice-Hall, Englewood Cliffs, NJ, 1966.
[26] A. Neumaier, *Interval Methods for Systems of Equations*, Encyclopedia Math. Appl., Cambridge University Press, Cambridge, UK, 1991.
[27] M. C. Recchioni, *Modified newton method in circular interval arithmetic*, J. Optim. Theory Appl., 86 (1995), pp. 223–244.
[28] W. C. Rheinboldt, *Solution fields of nonlinear equations and continuation methods*, SIAM J. Numer. Anal., 17 (1980), pp. 221–237.
[29] W. C. Rheinboldt, *Numerical analysis of continuation methods for nonlinear structural problems*, Comput. & Structures, 13 (1981), pp. 103–113.
[30] W. C. Rheinboldt, *On a theorem of S. Smale about Newton's method for analytic mappings*, Appl. Math. Lett., 1 (1988), pp. 69–72.
[31] S. M. Rump, *A note on epsilon-inflation*, Reliab. Comput., 4 (1998), pp. 371–375.
[32] S. M. Rump, *INTLAB—INTerval LABoratory*, in Developments in Reliable Computing, T. Csendes, ed., Kluwer Academic Publishers, Dordrecht, the Netherlands, 1999, pp. 77–104.
[33] O. Schütze, C. A. C. Coello, S. Mostaghim, E.-G. Talbi, and M. Dellnitz, *Hybridizing evolutionary strategies with continuation methods for solving multi-objective problems*, Engrg. Optim., 40 (2008), pp. 383–402.
[34] S. Smale, *Newton's method estimates from data at one point*, in The Merging of Disciplines in Pure, Applied and Computational Mathematics, Springer, New York, 1986, pp. 185–196.
[35] *Mathematica*, version 7.0, Wolfram Research, Champaign, IL, 2008.