SIAM J. NUMER. ANAL.
Vol. 41, No. 6, pp. 2350–2373

EXISTENCE VERIFICATION FOR HIGHER DEGREE SINGULAR ZEROS OF NONLINEAR SYSTEMS*

R. BAKER KEARFOTT† AND JIANWEI DIAN‡

Abstract. Finding approximate solutions to systems of n nonlinear equations in n real variables is a much studied problem in numerical analysis. Somewhat more recently, researchers have developed numerical methods to provide mathematically rigorous error bounds on such solutions. (We say that we "verify" existence of the solution within those bounds on the variables.) However, when the Jacobi matrix is singular at the solution, no computational techniques to verify existence can handle the general case. Nonetheless, computational verification that one or more solutions exists within a region in complex space containing the real bounds is possible by computing the topological degree. In a previous paper, we presented theory and algorithms for the simplest case, when the rank-defect of the Jacobian matrix at the solution is 1 and the topological index is 2. Here, we generalize that result to arbitrary topological index $d \geq 2$: We present theory, algorithms, and experimental results. We also present a heuristic for determining the degree, obtaining a value that we can subsequently verify with our algorithms. Although execution times are slow compared to corresponding bound verification processes for nonsingular systems, the order with respect to system size is still cubic.

Key words. complex nonlinear systems, interval computations, verified computations, singularities, topological degree

AMS subject classifications. 65G10, 65H10

DOI. 10.1137/S0036142901386057

1. Introduction.

Solution of linear and nonlinear systems of equations is a fundamental problem in numerical analysis, underlying much, if not most, of modern scientific computation. A system of n equations in n unknowns, where the expressions defining the system are defined in some closed, bounded subset $\mathbf{D}$ of n-dimensional space, may be expressed mathematically by

$$F(x) = 0, \quad F : \mathbf{D} \subset \mathbb{R}^n \to \mathbb{R}^n. \tag{1.1}$$

Throughout scientific computing, floating point arithmetic is used to solve equations (1.1) approximately. If F is linear, for example, then various direct (Gaussian elimination–based) methods, or iterative methods such as the preconditioned conjugate gradient method, are used. If F is nonlinear, then numerical solution of (1.1) involves various iterative methods, and the corresponding computer code can be sophisticated or involve numerous heuristics. In both the linear and nonlinear cases, the result of the computation is an approximate solution vector $\check{x} \in \mathbb{R}^n$, $F(\check{x}) \approx 0$. Hopefully, $\check{x}$ is near an exact or "true" solution x^*, $F(x^*) = 0$, such that $\|\check{x} - x^*\|$ is small.[1] However, with a few exceptions, the computation that produces the approximate solution $\check{x}$ does not give a bound on $\|\check{x} - x^*\|$. Indeed, it is not hard to find

*Received by the editors March 2, 2001; accepted for publication (in revised form) June 16, 2003; published electronically December 17, 2003. This work was partially supported by National Science Foundation grant DMS-9701540.

http://www.siam.org/journals/sinum/41-6/38605.html

†Department of Mathematics, University of Louisiana at Lafayette, Lafayette, LA 70504 (rbk@louisiana.edu).

‡Hewlett–Packard Company, 3000 Waterview Parkway, Richardson, TX 75080 (jianwei_dian@hp.com).

[1] Except when explicitly noted, we assume only that the norm is some fixed norm (independent of x), since the norm is discussed in terms of the order $\mathcal{O}(\|\cdot\|)$ and since we are working in finite-dimensional spaces.

instances of practical problems for which the output vector $\breve{x}$ of an algorithm to solve a nonlinear system of the form (1.1) is not near a true solution at all, and for which the modeler does not recognize this fact; see, for example, [5].

On the other hand, efficient methods have been available for some time to construct bounds about such approximate solutions $\breve{x}$ at which a true solution is known to exist. Specifically, an interval vector

$$\boldsymbol{x} = ([\underline{x}_1, \overline{x}_1], [\underline{x}_2, \overline{x}_2], \ldots, [\underline{x}_n, \overline{x}_n])^T \tag{1.2}$$

is found such that each width $\mathrm{w}([\underline{x}_i, \overline{x}_i]) = \overline{x}_i - \underline{x}_i$ is small (a small multiple of the machine precision, depending on the problem), and such that the computational process has proven mathematically (with no uncertainty due to roundoff error) that there is an exact solution $x^* \in \boldsymbol{x}$. Although it is not universally recognized within the general numerical analysis community, such methods can be developed to be practical more often than not and can give rigorous bounds that both are tighter than heuristic error estimates and are obtained with less effort; see, for example, [15]. An explanation of these methods appears in [6, 11, 17] and in numerous other works. The mathematical assumptions under which such verification methods can be expected to be successful are basically that the Jacobi matrix for the system is continuous and nonsingular at the solution; see the aforementioned references for a precise statement of the assumptions. For a practical implementation of such methods (with interval arithmetic), the function residuals and Jacobi matrix need to be representable as a computer program.

Although these verification methods involve interval arithmetic, notorious for impracticality due to overestimation when naively used, the intervals (the coordinates of $\boldsymbol{x}$) in a posteriori verification computations are small. It is known, from both theory and practice, that the overestimation in such small intervals is asymptotically insignificant, making such methods more generally applicable.

In this work, we consider not finding an approximate solution $\breve{x}$ but constructing and verifying bounds $\boldsymbol{x}$ about such a point $\breve{x}$ (however found) such that an exact solution x^* lies within $\boldsymbol{x}$. Specifically, we address the following problem.

(1.3) Given $F : \mathbf{D} \to \mathbb{R}^n$, where $\mathbf{D}$ is some closed, bounded subset of $\mathbb{R}^n$ with nonempty interior, and given an approximate solution $\breve{x} \in \mathbf{D}$, construct bounds $\boldsymbol{x} \in \mathbb{IR}^n$, $\breve{x} \in \boldsymbol{x}$, with $\boldsymbol{x}$ as in (1.2), for which we *rigorously* verify

- there exists an $x^* \in \boldsymbol{x}$ such that $F(x^*) = 0$.

Throughout this paper, by "rigorous" we mean "with the same standard as for a traditional mathematical proof." Our algorithms for such verification will employ techniques derived from traditional floating point computations but will use directed roundings to take the finite nature of floating point arithmetic into account.

As is seen in [6, 11, 17] and elsewhere, when the Jacobian matrix $F'(x^*)$ is well-conditioned and not too quickly varying, interval computations have no trouble proving that there is a *unique* solution within small boxes with x^* reasonably near the center. (Various techniques, such as those in [16], can be used to initially construct the bounds over which the verification algorithm proceeds.) However, when $F'(x^*)$ is ill-conditioned or singular, in general, no computational techniques can verify the existence of a solution within a given region $\boldsymbol{x}$ of $\mathbb{R}^n$. Indeed, common thinking among researchers in such verification methods has been that verification is not possible in the singular case. Nonetheless, in [14] we introduced an algorithm for computational but rigorous verification, in the singular case, that a given number of true solutions exists within a region in complex space containing $\boldsymbol{x}$. There we studied the simplest case, when the rank-defect of the Jacobian matrix at the solution is one, and we developed and experimentally validated algorithms for the case when the topological index is 2. There, we also proved the special case of Theorem 3.1 (see section 3 below) when $d = 2$. Under the same assumptions as those in section 2 below, we developed specialized versions of the algorithms in section 4 below, and we presented varying-dimensional experimental results in [14].

We were surprised and pleased that the results in [14] could be generalized so easily. In particular, we developed an alternate simple, general proof for Theorem 3.1 below. Furthermore, the algorithms in section 4 below, although not taking advantage of special efficiencies in the degree-2 case as in [14], are similar in structure and have the same computational complexity as the algorithms in [14].

The developments below proceed by thinking of the function F in terms of a model of the form

$$F(x) = M(x) + R(x), \tag{1.4}$$

where $M(x)$ is a Taylor approximation to F about x^* and $R(x)$ is the error term. The number of solutions to $F(x) = 0$ is determined according to the topological degree (reviewed in section 1.2 below) of F. In Theorem 3.1 (see section 3 below) we show that, if $F(x) = M(x)$, where $M(x)$ has some verifiable properties, then the topological degree of F must equal d. (This proves existence, since the topological degree over a region in complex space is equal to the number of solutions in the region, counting multiplicities.) Basing the computations on the structure of M, we use a heuristic test to guess the integer d. Speaking roughly, we then take account of both roundoff error and the error term $R(x)$ with interval computations. In particular, we use the structure of M to efficiently arrange an exhaustive search that rigorously verifies that the topological degree actually is d. Even though the search is exhaustive, completion of the search requires only the same order of magnitude of computational work as a step of Newton's method on the system; this is due to the postulated structure of M and the way we have arranged the search.

As explained in [14, section 1.4], if d is even, it is meaningless to discuss the existence of a solution in $\mathbb{R}^n$ within the framework of errors in the data, model, and floating point system, since the topological degree in real space in such cases may be equal to 0. The even d case is a generalization of the situation with $f(x) = x^2$ at $x = 0$: The function f itself has a unique solution at $x = 0$, yet perturbations of f result in either no solutions or two solutions near $x = 0$. In contrast, $f(z) = (1+\epsilon_1)z^2+\epsilon_2 z+\epsilon_3$, $|\epsilon_i|$ small for $i = 1, 2, 3$, has two solutions, counting multiplicities, in all sufficiently large (but with diameters that can be chosen to be $\mathcal{O}(\epsilon)$ as $\epsilon \to 0$) open sets in $\mathbb{C}$ containing $z = 0$. This illustrates a general phenomenon: Whereas small perturbations of the data change the existence of (one or more) solutions near a particular point in $\mathbb{R}^n$, the solutions vary continuously with perturbations of complex extensions. We have presented one precise statement of this in Theorem 3.1 of [4]: Under the assumptions in that theorem (essentially, that the Jacobi matrix have rank defect 1 at the solution, and that certain derivative tensors up to order d vanish), $\mathrm{d}(F, \boldsymbol{x}, 0) = 0$ for a box $\boldsymbol{x} \in \mathbb{R}^n$ whenever d is even (and $\mathrm{d}(F, \boldsymbol{x}, 0) = \pm 1$ over such a box when d is odd). Thus, in that case when d is even (and, we believe, in many fairly general cases) verifying the value of the topological degree within the real context cannot verify existence of the solution. In contrast, $\mathrm{d}(F, \boldsymbol{z}, 0)$ must always be nonzero if there is a $z \in \boldsymbol{z}$ with $F(\boldsymbol{z}) = 0$ and $0 \notin \boldsymbol{\partial z}$, for $\boldsymbol{z} \subset \mathbb{C}^n$. Also, the *number* of solutions counting multiplicities can change under perturbations in $\mathbb{R}^n$, even for odd-order functions such as x^3, for which we can use techniques such as those in [4] to prove existence; in contrast, $\mathrm{d}(F, \boldsymbol{z}, 0)$ gives the exact number of solutions within $\boldsymbol{z}$, counting multiplicities, for complex-valued functions F of complex variables $\boldsymbol{z}$.

In this paper, we consider the case of general d, to verify the existence of solutions in small neighborhoods of $\mathbb{C}^n$, as illustrated in problem (1.5) below. Our hope is that such verification will be useful in analysis of systems having even-order roots, even though the validation is in a different space; in any case, rigorous validation of such systems in the original space may not be possible and may not be meaningful if the system was derived from measurements with errors. (We present special theory, analysis, and algorithms for the real case and odd-order roots in [4].)

(1.5)

> Given F, $\mathbf{D}$, and $\check{x}$ as in problem (1.3), consider an analytic extension $\tilde{F}$ of F to a domain $\tilde{\mathbf{D}} \subseteq \mathbb{C}^n$, $\mathbf{D} \subset \tilde{\mathbf{D}}$. Construct bounds $\boldsymbol{x}$ as in problem (1.3) and $\boldsymbol{y} = ([\underline{y}_1, \overline{y}_1], \ldots, [\underline{y}_n, \overline{y}_n])$, $0 \in \boldsymbol{y}$, for which we *rigorously* verify the following:
>
> - there exists a $z^* \in \boldsymbol{z}$ such that $\tilde{F}(z^*) = 0$, where
> - $\boldsymbol{z} = \big\{(x_1 + iy_1, x_2 + iy_2, \ldots, x_n + iy_n)^T \in \mathbb{C}^n \;\big|\; x_j \in \boldsymbol{x}_j,\, y_j \in \boldsymbol{y}_j,\, 1 \le j \le n\big\}$.

Hiding detail and revealing overall ideas, we have simplified the notation in this work, compared to that in [14].

After introducing our notation in section 1.1, we briefly review the relevant portions of topological degree theory in sections 1.2 and 1.3. We introduce our use of the structure of the model $M(x)$ in section 2. We present our scheme for setting the coordinate bounds $\boldsymbol{x}$ within which we prove the existence of solutions in section 2.2; though related, this scheme is improved and works more generally than that in [14]. In section 3, we show that the degree must be equal to d if $F(x) = M(x)$ (i.e., $R(x) = 0$), within the context introduced in section 2. In section 4, we present the algorithm that verifies that the degree is d for nonzero $R(x)$, within the framework introduced in section 2. In section 5, we present an easily implemented heuristic computation for guessing the value of d, necessary for the verification algorithm in section 4. Finally, in section 6.3 we present results of trying the computations on several examples; these results illustrate that the algorithm can be practical for a variety of problems, and that the computation does not necessarily increase rapidly with the dimension of the problem.

1.1. Notation.

We assume familiarity with the fundamentals of interval arithmetic; see [1, 6, 11, 17, 19] for introductory material.

Throughout, scalars and vectors will be denoted by lower case, while matrices will be denoted by upper case. Intervals, interval vectors (also called "boxes"), and interval matrices will be denoted by boldface. For instance, $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ denotes an interval vector, $A = (a_{i,j})$ denotes a point matrix, and $\boldsymbol{A} = (\boldsymbol{a}_{i,j})$ denotes an interval matrix. The midpoint of an interval or interval vector $\boldsymbol{x}$ will be denoted by $\mathrm{m}(\boldsymbol{x})$. As in section 1, $\mathrm{w}(\boldsymbol{x})$ denotes the width of an interval $\boldsymbol{x} = [\underline{x}, \overline{x}]$, that is, $\mathrm{w}(\boldsymbol{x}) = \overline{x} - \underline{x}$; if $\boldsymbol{x}$ represents an interval vector, then the midpoint $\mathrm{m}(\boldsymbol{x})$ and width $\mathrm{w}(\boldsymbol{x})$ will be real vectors, understood componentwise. Real n-space will be denoted by $\mathbb{R}^n$, while complex n-space will be denoted by $\mathbb{C}^n$. The set of real interval vectors will be denoted by $\mathbb{IR}^n$, while the set of complex interval vectors will be denoted by $\mathbb{IC}^n$.

Suppose $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ is an n-dimensional real box, where $\boldsymbol{x}_k = [\underline{x}_k, \overline{x}_k]$. The nonoriented boundary of $\boldsymbol{x}$, denoted by $\boldsymbol{\partial x}$, consists of $2n$ $(n-1)$-dimensional real boxes

$$\boldsymbol{x}_{\underline{k}} \equiv (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{k-1}, \underline{x}_k, \boldsymbol{x}_{k+1}, \ldots, \boldsymbol{x}_n) \quad \text{and} \quad \boldsymbol{x}_{\overline{k}} \equiv (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{k-1}, \overline{x}_k, \boldsymbol{x}_{k+1}, \ldots, \boldsymbol{x}_n),$$

where $k = 1, \ldots, n$.

The *orientation* of a region $\mathbf{D} \subset \mathbb{R}^n$ and of its boundary $\boldsymbol{\partial}\mathbf{D}$ is a generalization of the concept of orientation of a region and its boundary (counterclockwise being positive orientation) in complex analysis, or of the concepts of orientation of a region and its boundary when applying Green's theorem or Stokes' theorem; see [3, pp. 4–10] or [2], for example, for a detailed formal definition. In particular, a simplex $\langle a^{(0)}, a^{(1)}, \ldots, a^{(n)} \rangle$, $a^{(k)} \in \mathbb{R}^n$, $0 \le k \le n$, is positively oriented, provided that a certain determinant formed from the coordinates of the points $a^{(k)}$ is positive, and is negatively oriented if that determinant is negative; polygonal regions formed by juxtaposing such oriented simplexes have a positive orientation, provided that each component simplex is positively oriented.

To explain the algorithms in this paper, we need concern ourselves only with the derived orientation of the boundary of an interval vector (i.e., of a box) $\boldsymbol{x}$. The following "definition" can be derived as a theorem from the general definition of a positively oriented polygonal region. (For a more detailed presentation, see our technical report [13, pp. 7–8].)

Definition 1.1. *Suppose that a box $\boldsymbol{x}$ as in* (1.2) *is positively oriented. Then the positively oriented boundary $b(\boldsymbol{x})$ is given by the formal sum*

$$\sum_{k=1}^{n} \left\{ (-1)^k \boldsymbol{x}_{\underline{k}} + (-1)^{k+1} \boldsymbol{x}_{\overline{k}} \right\}$$

of the $2n$ $(n-1)$-dimensional boxes $\boldsymbol{x}_{\underline{k}}$ and $\boldsymbol{x}_{\overline{k}}$.

Our model $M(x)$ of $F(x)$ as in (1.4) is a multivariate Taylor polynomial. In particular we will write a component f_i of F as

$$f_i(x) = f_i(\check{x}) + \sum_{j=1}^{d} \frac{1}{j!} D^j f_i(\check{x})[x - \check{x}, \ldots, x - \check{x}] + \mathcal{O}\left(\|x - \check{x}\|\right)^{d+1}, \tag{1.6}$$

where

$$\begin{aligned} &D^j f_i(\check{x})[x - \check{x}, \ldots, x - \check{x}] \\ &= \sum_{k_1=1}^{n} \cdots \sum_{k_j=1}^{n} \frac{\partial^j f_i}{\partial x_{k_1} \cdots \partial x_{k_j}}(\check{x})(x_{k_1} - \check{x}_{k_1}) \cdots (x_{k_j} - \check{x}_{k_j}) \end{aligned} \tag{1.7}$$

is the jth derivative tensor.

In our verification algorithms, the domains will be interval vectors, i.e., rectangular boxes $\boldsymbol{x}$. However, we state some of the known topological degree theory results more generally, in terms of the closed, bounded set $\mathbf{D}$ with nonempty interior that we introduced above.

1.2. Formulas from degree theory.

In [14], we reviewed the topological degree in the context of this paper. Also see [2, 3, 8, 9, 18, 20]. Here, we repeat several properties used in the proofs in subsequent sections.

Although a formal definition of the topological degree is somewhat cumbersome, one obtains an intuitive understanding of the topological degree from its properties. In particular, for $n = 1$, the topological degree of F at 0 over an interval $\boldsymbol{x}$, denoted $\mathrm{d}(F, \boldsymbol{x}, 0)$, is the number of times the graph of F crosses the x-axis in the positive direction, minus the number of times the graph of F crosses the x-axis in the negative direction. If $F : \mathbb{C} \to \mathbb{C}$ and $\mathbf{D}$ is a simply connected region (a region without holes, such as a disk) containing the origin in $\mathbb{C}$, then the topological degree $\mathrm{d}(F, \mathbf{D}, 0)$ is equal to the winding number of F with respect to the curve bounding $\mathbf{D}$. Because of this fact, $\mathrm{d}(p_d, \mathbf{D}, 0) = d$, where p_d is any polynomial of degree d and $\mathbf{D}$ is any sufficiently large simply connected domain in $\mathbb{C}$ with $0 \in \mathbf{D}$ (where the size depends on the particular p_d). Thus, in $\mathbb{C}$, the topological degree roughly corresponds to the notion of algebraic degree, which is the same as the number of solutions, counting multiplicity. If we think of $\mathbf{D}$ as being the closure of a very small region containing a solution z^*, $F(z^*) = 0$, then $\mathrm{d}(F, \mathbf{D}, 0)$ is termed the *topological index* of z^*; the topological index corresponds to the multiplicity of z^*. For example, the topological index of z^d at $z^* = 0$ is equal to d. In this paper, we prove the existence of solutions within small domains $\mathbf{D}$ by verifying, essentially, that the topological index is nonzero.

Formal definitions of the topological degree can be given analytically (in terms of an integral) as in [18, Chapter 6], or in terms of fundamental concepts of algebraic topology, as in [2]. In either case, either definition can be obtained as a theorem, starting with the other one as the definition. We can actually think of the degree in terms of the following.

THEOREM 1.2 (see [18, p. 150]). *Suppose that F is continuous, and suppose that the Jacobian matrix $F'(x)$ is defined and nonsingular at each zero of F within a domain $\mathbf{D}$, which is the closure of an open region in $\mathbb{R}^n$, and suppose that $F(x) \neq 0$ when $x \in \boldsymbol{\partial}\mathbf{D}$. Then, the degree* $\mathrm{d}(F, \mathbf{D}, 0)$ *is equal to the number of zeros of F at which the determinant of the Jacobian matrix $F'(x)$ is positive, minus the number of zeros of F at which the determinant of the Jacobian matrix $F'(x)$ is negative.*

Basically, Theorem 1.2 states that the degree is an algebraic number of zeros of F in $\mathbf{D}$ when the Jacobian matrix is nonsingular at each zero. However, the degree does not change as F is perturbed, and we can imagine the degree remaining defined as two or more zeros of F coalesce into a single zero at which the Jacobian matrix is singular (important in our context here). Similarly, F need only be continuous (not necessarily differentiable) for $\mathrm{d}(F, \mathbf{D}, 0)$ to be defined. To define the degree for arbitrary continuous functions that do not vanish on the boundary $\boldsymbol{\partial}\mathbf{D}$, the analytic definition as in [18, Chapter 6] uses an integral and mollifying functions, whereas the topological definition approximates the image of the boundary $\boldsymbol{\partial}\mathbf{D}$ with a piecewise-linear simplicial complex (similar to how engineers approximate an object with triangles for the finite element method, except that the topologist's simplicial complex is oriented).

Starting either from the analytical definition of [18] or from the algebraic-topological definition of [2], we obtain the following properties of the degree. These properties are what will concern us in our verification procedures.

THEOREM 1.3 (see [18, p. 150]). *Let F, $G : \mathbf{D} \subset \mathbb{R}^n \to \mathbb{R}^n$ be two continuous functions that do not vanish on $\boldsymbol{\partial}D$. If $F(x) = G(x)$ for $x \in \boldsymbol{\partial}\mathbf{D}$, then* $\mathrm{d}(F, \mathbf{D}, 0) = \mathrm{d}(G, \mathbf{D}, 0)$.

Theorem 1.3 states one of the most important properties of degree: The degree depends only on the function values on the boundary.

THEOREM 1.4 (see [18, p. 157]). *Let F, $G : \mathbf{D} \subset \mathbb{R}^n \to \mathbb{R}^n$ be two continuous functions. If*

$$0 \notin \{tF(x) + (1-t)G(x) | x \in \boldsymbol{\partial}\mathbf{D} \text{ and } t \in [0, 1]\},$$

then

$$\mathrm{d}(F, \mathbf{D}, 0) = \mathrm{d}(G, \mathbf{D}, 0).$$

Theorem 1.4 is the famous Poincaré–Bohl theorem. It is a particular case of the homotopy invariant property of the topological degree. Since $\mathbf{D}$ is compact, this homotopy invariance implies, without too much argument, that the degree is a continuous function of F.

COROLLARY 1.5. *Suppose $F : \mathbf{D} \subset \mathbb{R}^n \to \mathbb{R}^n$ is continuous and $\mathrm{d}(F, \mathbf{D}, 0) = d$. Then there is an $\epsilon > 0$ such that, for all continuous $G : \mathbf{D} \to \mathbb{R}^n$ with $|F(x) - G(x)| < \epsilon$ for $x \in \mathbf{D}$, $\mathrm{d}(F, \mathbf{D}, 0) = \mathrm{d}(G, \mathbf{D}, 0)$.*

Suppose $F : \mathbf{D} \subset \mathbb{C}^n \to \mathbb{C}^n$ is analytic, and view the real and imaginary components of F and its argument $z \in \mathbb{C}^n$ as real components in $\mathbb{R}^{2n}$. Let $z = x + iy$ and $F(z) = u(x, y) + iv(x, y)$, where $x = (x_1, \ldots, x_n)$, $y = (y_1, \ldots, y_n)$, $u(x, y) = (u_1(x, y), \ldots, u_n(x, y))$, and $v(x, y) = (v_1(x, y), \ldots, v_n(x, y))$. We define $\tilde{\mathbf{D}}$ by

$$\tilde{\mathbf{D}} \equiv \{(x_1, y_1, \ldots, x_n, y_n) | (x_1 + iy_1, \ldots, x_n + iy_n) \in \mathbf{D}\}$$

and $\tilde{F} : \tilde{\mathbf{D}} \subset \mathbb{R}^{2n} \to \mathbb{R}^{2n}$ by $\tilde{F} = (u_1, v_1, \ldots, u_n, v_n)$. We then have the following properties.

THEOREM 1.6 (see [14]). *Suppose that $F : \mathbf{D} \subset \mathbb{C}^n \to \mathbb{C}^n$ is analytic, with $F(z) \neq 0$ for any $z \in \boldsymbol{\partial}\mathbf{D}$, and suppose that $\tilde{\mathbf{D}}$ and $\tilde{F} : \tilde{\mathbf{D}} \to \mathbb{R}^{2n}$ are defined as above. Then $\mathrm{d}(\tilde{F}, \tilde{\mathbf{D}}, 0)$ is nonnegative and is equal to the number of solutions $z^* \in \mathbf{D}$, $F(z^*) = 0$, counting multiplicities.*

1.3. A basic degree computation formula. Theorem 1.7 below relates the basic theory of the topological degree to the computational verification procedures in section 4 below. Theorem 1.7 is similar to Theorem 2.5 of [14]. We can obtain Theorem 1.7 from formulas (4.12) and (4.14) in [20], by taking into account the orientations of the faces of $\boldsymbol{x}$.

Theorem 1.7 characterizes $\mathrm{d}(F, \boldsymbol{x}, 0)$ in terms of certain components of F on $\boldsymbol{\partial x}$. In particular, set

$$F_{\neg k}(\boldsymbol{x}) \equiv \big(f_1(\boldsymbol{x}), \ldots, f_{k-1}(\boldsymbol{x}), f_{k+1}(\boldsymbol{x}), \ldots, f_n(\boldsymbol{x})\big).$$

Then we have the following result.

THEOREM 1.7. *Let $s \in \{-1, 1\}$ be fixed arbitrarily, suppose $F \neq 0$ on $\boldsymbol{\partial x}$, and suppose that there is a p, $1 \leq p \leq n$, such that*

1. *$F_{\neg p} \equiv (f_1, \ldots, f_{p-1}, f_{p+1}, \ldots, f_n) \neq 0$ on $\boldsymbol{\partial x}_{\underline{k}}$ or $\boldsymbol{\partial x}_{\overline{k}}$, $k = 1, \ldots, n$; and*
2. *the Jacobi matrices of $F_{\neg p}$ are nonsingular at all solutions of $F_{\neg p} = 0$ on $\boldsymbol{\partial x}$ and are continuous in a neighborhood of such solutions.*

Then

$$\begin{aligned}\mathrm{d}(F, \boldsymbol{x}, 0) \; &= (-1)^{p-1} s \left\{ \sum_{k=1}^{n} (-1)^k \sum_{\substack{x \in \boldsymbol{x}_{\underline{k}} \\ F_{\neg p}(x) = 0 \\ \mathrm{sgn}(f_p(x)) = s}} \mathrm{sgn} \left| \frac{\partial F_{\neg p}}{\partial x_1 x_2 \cdots x_{k-1} x_{k+1} \cdots x_n}(x) \right| \right. \\ & \qquad \left. + \sum_{k=1}^{n} (-1)^{k+1} \sum_{\substack{x \in \boldsymbol{x}_{\overline{k}} \\ F_{\neg p}(x) = 0 \\ \mathrm{sgn}(f_p(x)) = s}} \mathrm{sgn} \left| \frac{\partial F_{\neg p}}{\partial x_1 x_2 \cdots x_{k-1} x_{k+1} \cdots x_n}(x) \right| \right\}.\end{aligned}$$

2. Assumptions and choice of box. In this section, we present the basic assumptions. We also introduce how we choose the coordinate bounds $\boldsymbol{x}_i = [\underline{x}_i, \overline{x}_i]$ to satisfy the assumptions and enable more efficient algorithms. When the rank of $F'(x^*)$ is $n-p$ for some $p > 0$, an appropriate preconditioner can be used to reduce $\boldsymbol{F'}(\boldsymbol{x})$ to approximately the pattern shown in Figure 2.1. (See [11] and [14] for details on preconditioning.)

$$Y\boldsymbol{F'}(\boldsymbol{x}) \approx \begin{pmatrix} 1 & 0 & \cdots & 0 & \overbrace{*\cdots*}^{p} \\ 0 & 1 & 0\cdots & 0 & *\cdots* \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & 1 & *\cdots* \\ 0 & \cdots & 0 & 0 & 0\cdots 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 & 0\cdots 0 \end{pmatrix}.$$

FIG. 2.1. *An approximate form for a preconditioned singular interval system of approximate rank $n-p$, where "$*$" represents a nonzero element.*

In the analysis to follow, we assume that the system has already been preconditioned, so that it is, to within second-order terms with respect to $\mathrm{w}(\boldsymbol{x})$, of the form in Figure 2.1. That is, we assume that the preconditioned system is of the form seen in Figure 2.1 if we interpret "$*$" to represent any interval, "1" to represent intervals of the form $[1-\mathcal{O}(\|x-x^*\|), 1+\mathcal{O}(\|x-x^*\|)]$, and "0" to represent intervals of the form $[-\mathcal{O}(\|x-x^*\|), \mathcal{O}(\|x-x^*\|)]$. Here as in [14], we concentrate on the case $p=1$.

2.1. The basic assumptions. As in the special case $d=2$ of [14], we assume

1. $F : \mathbf{D} \subset \mathbb{R}^n \to \mathbb{R}^n$ can be extended to an analytic function in $\mathbb{C}^n$.
2. $\boldsymbol{x} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_n) = ([\underline{x}_1, \overline{x}_1], \dots, [\underline{x}_n, \overline{x}_n])$ is a small box constructed to be centered at an approximate solution $\check{x}$, i.e., $\mathrm{m}(\boldsymbol{x}) = (\check{x}_1, \dots, \check{x}_n)$.
3. $\check{x}$ is near a point x^* with $F(x^*) = 0$ such that $\|\check{x} - x^*\|$ is much smaller than the norm of the width of the box $\boldsymbol{x}$, and the width of the box $\boldsymbol{x}$ is small enough that mean value interval extensions lead, after preconditioning, to a system like Figure 2.1, with small intervals replacing the zeros.
4. F has been preconditioned as in Figure 2.1, and $F'(x^*)$ has null space of dimension 1.

Define

$$\begin{aligned}
\alpha_k &\equiv \frac{\partial f_k}{\partial x_n}(\check{x}), \qquad 1 \le k \le n-1, \\
\alpha_n &\equiv -1, \\
\Delta_1 &\equiv \left| \frac{\partial F}{\partial x_1 \cdots \partial x_n}(\check{x}) \right| \\
\Delta_l &\equiv \sum_{k_1=1}^{n} \cdots \sum_{k_l=1}^{n} \frac{\partial^l f_n}{\partial x_{k_1} \cdots \partial x_{k_l}}(\check{x}) \alpha_{k_1} \cdots \alpha_{k_l}, \qquad 2 \le l.
\end{aligned}$$

The following representation of $F(x)$ near $\check{x}$ is appropriate under these assumptions:

$$f_k(x) = (x_k - \check{x}_k) + \alpha_k(x_n - \check{x}_n) + \mathcal{O}\left(\|x - \check{x}\|\right)^2 \quad \text{for } 1 \le k \le n-1, \tag{2.1}$$

$$f_n(x) = \sum_{\ell=2}^{d} \frac{1}{\ell!} D^\ell f_n(\check{x})[x - \check{x}, \ldots, x - \check{x}] + \mathcal{O}\left(\|x - \check{x}\|\right)^{d+1}. \tag{2.2}$$

Here and below, "d" is a fixed constant that represents the postulated topological index (obtained, say, with the heuristic in section 5 below); the index d will be verified with our proposed algorithms (in section 4 below).

We now introduce additional notation to describe the complex extensions. For $F : \mathbb{R}^n \to \mathbb{R}^n$, extend F to complex space: $x+iy$, with y in a small box $\boldsymbol{y} = \left(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n\right) = \left([\underline{y}_1, \overline{y}_1], \ldots, [\underline{y}_n, \overline{y}_n]\right)$, where $\boldsymbol{y}$ is centered at $(0, \ldots, 0)$. As in Theorem 1.6 above, define $\tilde{\boldsymbol{x}} \equiv (\boldsymbol{x}, \boldsymbol{y}) \equiv (\boldsymbol{x}_1, \boldsymbol{y}_1, \ldots, \boldsymbol{x}_n, \boldsymbol{y}_n) = ([\underline{x}_1, \overline{x}_1], [\underline{y}_1, \overline{y}_1], \ldots, [\underline{x}_n, \overline{x}_n], [\underline{y}_n, \overline{y}_n])$, $u_k(x,y) \equiv \Re(f_k(x+iy))$ and $v_k(x,y) \equiv \Im(f_k(x+iy))$. With this, define

$$\tilde{F}(x,y) \equiv (u_1(x,y), v_1(x,y), \ldots, u_n(x,y), v_n(x,y)) : \mathbb{R}^{2n} \to \mathbb{R}^{2n}.$$

Also define

$$\tilde{F}_{\neg u_n}(x,y) \equiv \left(u_1(x,y), v_1(x,y), \ldots, u_{n-1}(x,y), v_{n-1}(x,y), v_n(x,y)\right).$$

Then, based on (2.1) and (2.2), for $1 \le k \le (n-1)$,

$$\left.\begin{array}{rcl} u_k(x,y) & = & (x_k - \check{x}_k) + \alpha_k(x_n - \check{x}_n) \\ & & + \, \mathcal{O}\left(\|(x - \check{x}, y)\|\right)^2, \\ v_k(x,y) & = & y_k + \alpha_k y_n + \mathcal{O}\left(\|(x - \check{x}, y)\|\right)^2, \end{array}\right\} \tag{2.3}$$

or

$$\left.\begin{array}{rcl} u_k(x,y) & \approx & (x_k - \check{x}_k) + \alpha_k(x_n - \check{x}_n), \\ v_k(x,y) & \approx & y_k + \alpha_k y_n. \end{array}\right\} \tag{2.4}$$

2.2. Choosing the coordinate bounds.

In our verification algorithms below, we drastically reduce the amount of computation required by astutely choosing the ratios of coordinate widths of the boxes $\boldsymbol{x}$ and $\boldsymbol{y}$. We will use a scheme similar to that of section 5 of [14]. In particular, having defined $\boldsymbol{x}_{\underline{k}}$ and $\boldsymbol{x}_{\overline{k}}$ in section 1.1, we define $\boldsymbol{y}_{\underline{k}}$ and $\boldsymbol{y}_{\overline{k}}$ similarly:

$$\begin{aligned} \boldsymbol{y}_{\underline{k}} &\equiv (\boldsymbol{x}_1, \boldsymbol{y}_1, \ldots, \boldsymbol{x}_{k-1}, \boldsymbol{y}_{k-1}, \boldsymbol{x}_k, \underline{y}_k, \boldsymbol{x}_{k+1}, \boldsymbol{y}_{k+1}, \ldots, \boldsymbol{x}_n, \boldsymbol{y}_n) \quad \text{and} \\ \boldsymbol{y}_{\overline{k}} &\equiv (\boldsymbol{x}_1, \boldsymbol{y}_1, \ldots, \boldsymbol{x}_{k-1}, \boldsymbol{y}_{k-1}, \boldsymbol{x}_k, \overline{y}_k, \boldsymbol{x}_{k+1}, \boldsymbol{y}_{k+1}, \ldots, \boldsymbol{x}_n, \boldsymbol{y}_n). \end{aligned}$$

To compute the degree $\mathrm{d}(\tilde{F}, \tilde{\boldsymbol{x}}, 0)$, we will consider $\tilde{F}_{\neg u_n}$ on the boundary of $\tilde{\boldsymbol{x}}$. This boundary consists of the $4n$ faces $\boldsymbol{x}_{\underline{1}}$, $\boldsymbol{x}_{\overline{1}}$, $\boldsymbol{y}_{\underline{1}}$, $\boldsymbol{y}_{\overline{1}}$, $\ldots$, $\boldsymbol{x}_{\underline{n}}$, $\boldsymbol{x}_{\overline{n}}$, $\boldsymbol{y}_{\underline{n}}$, $\boldsymbol{y}_{\overline{n}}$. We will set $\boldsymbol{x}_n$ and $\boldsymbol{y}_n$ so that the coordinate widths $\mathrm{w}(\boldsymbol{x}_k)$ obey

$$\mathrm{w}(\boldsymbol{x}_n) \le \frac{1}{2} \min_{1 \le k \le n-1} \left\{ \frac{\mathrm{w}(\boldsymbol{x}_k)}{|\alpha_k|} \right\} \quad \text{and} \quad \mathrm{w}(\boldsymbol{y}_n) \le \frac{1}{2} \min_{1 \le k \le n-1} \left\{ \frac{\mathrm{w}(\boldsymbol{y}_k)}{|\alpha_k|} \right\}. \tag{2.5}$$

In the above two relationships, when $\alpha_k = 0$ for some k, that particular k can be ignored in obtaining the minima, and $\mathrm{w}(\boldsymbol{x}_k)$ and $\mathrm{w}(\boldsymbol{y}_k)$ can be set to any small positive values as long as the assumptions in section 2.1 are met. If $\alpha_k = 0$ for $k = 1, \ldots, n-1$, then $\mathrm{w}(\boldsymbol{x}_k)$ and $\mathrm{w}(\boldsymbol{y}_k)$, $k = 1, \ldots, n$, can independently be set to any small positive values, as long as the assumptions in section 2.1 are met.

Constructing the box widths this way will make it unlikely that $u_k(x, y) = 0$ on either $\boldsymbol{x}_{\underline{k}}$ or $\boldsymbol{x}_{\overline{k}}$ and unlikely that $v_k(x, y) = 0$ on either $\boldsymbol{y}_{\underline{k}}$ or $\boldsymbol{y}_{\overline{k}}$, for $k = 1, \ldots, n-1$. This, in turn, will allow us to replace searches on $4n - 4$ of the $4n$ faces of $\boldsymbol{\partial\tilde{x}}$ by simple interval evaluations, reducing the total computational cost dramatically. See [14] for details.

A difference between the scheme used here and that of [14] is the way the ratio $\mathrm{w}(\boldsymbol{y}_n)/\mathrm{w}(\boldsymbol{x}_n)$ is chosen. In [14], $\mathrm{w}(\boldsymbol{y}_n)$ was chosen large relative to $\boldsymbol{x}_n$, to arrange no solutions of $u_n = 0$ on $\boldsymbol{y}_{\underline{n}}$ and $\boldsymbol{y}_{\overline{n}}$. When the degree is odd, that is not possible, and we have found the strategy represented by formula (4.1) below, implying $\mathrm{w}(\boldsymbol{y}_n)$ small relative to $\mathrm{w}(\boldsymbol{x}_n)$, as in Figure 4.1 below, to be more convenient.

3. When the polynomial model is exact.

In [14] we proved that, under the assumptions in section 2, if the $\mathcal{O}\left(\|x - \check{x}\|\right)^2$ term is absent in (2.1) and the $\mathcal{O}\left(\|x - \check{x}\|\right)^{d+1}$ term is absent in (2.2) with $d = 2$, and if $\Delta_1 = 0$ but $\Delta_2 \neq 0$, then $\mathrm{d}(\tilde{F}, \tilde{\boldsymbol{x}}, 0) = 2$. Here, we generalize that result to $\Delta_1 = \cdots = \Delta_{d-1} = 0$, $\Delta_d \neq 0$. Since the degree doesn't change under small perturbations of the function $\tilde{F}$ (see Theorem 1.5 above), the conclusion in Theorem 3.1 below also holds for more general continuous functions for which the $\mathcal{O}\left(\|x - \check{x}\|\right)^2$ and $\mathcal{O}\left(\|x - \check{x}\|\right)^{d+1}$ terms are not absent but are sufficiently small. In our computational existence verification algorithm in the next section, we use interval arithmetic to rigorously encompass the $\mathcal{O}\left(\|x - \check{x}\|\right)^2$ and $\mathcal{O}\left(\|x - \check{x}\|\right)^{d+1}$ terms. In this way, Theorem 3.1 below provides guidance for construction of our general algorithm.

THEOREM 3.1. *Suppose that*

1. $\tilde{\boldsymbol{x}}$ *is a nondegenerate box in* $\mathbb{R}^{2n}$ *as defined in section* 2;
2. $(\check{x}, \check{y}) = (\check{x}_1, \check{y}_1, \ldots, \check{x}_n, \check{y}_n)$ *is the midpoint of* $\tilde{\boldsymbol{x}}$*;*
3. F *and* $\tilde{F}$ *are as in section* 2;
4. F *is such that the* $\mathcal{O}\left(\|x - \check{x}\|\right)^2$ *and* $\mathcal{O}\left(\|x - \check{x}\|\right)^{d+1}$ *terms in* (2.1) *and* (2.2) *are absent; and*
5. $\Delta_1 = \cdots = \Delta_{d-1} = 0$, $\Delta_d \neq 0$, *where* $2 \leq d$.

Then $\mathrm{d}(\tilde{F}, \tilde{\boldsymbol{x}}, 0) = d$.

In contrast to the proof in [14], we use a homotopy argument to prove Theorem 3.1.

Proof. Let $z = (z_1, \ldots, z_n) = (x_1 + iy_1, \ldots, x_n + iy_n)$. Then

$$F(z) = (f_1(z), \ldots, f_{n-1}(z), f_n(z)),$$

where

$$\begin{aligned} f_k(z) &= (z_k - \check{z}_k) + \frac{\partial f_k}{\partial x_n}(\check{x})(z_n - \check{z}_n) \\ &= (z_k - \check{z}_k) + \alpha_k(z_n - \check{z}_n) \\ &\qquad \text{for } 1 \leq k \leq n-1, \end{aligned}$$

$$f_n(z) = \sum_{\ell=2}^{d} \frac{1}{\ell!} D^\ell f_n(\check{x})[z - \check{z}, \ldots, z - \check{z}]. \tag{3.1}$$

We construct $A : \mathbb{C}^n \to \mathbb{C}^n$ by

$$A(z) = (a_1(z), \ldots, a_{n-1}(z), a_n(z)),$$

where

$$\begin{aligned} a_k(z) &= (z_k - \check{z}_k) + \alpha_k(z_n - \check{z}_n) \quad \text{for } 1 \le k \le n-1, \\ a_n(z) &= \frac{(-1)^d \Delta_d}{d!}(z_n - \check{z}_n)^d. \end{aligned} \tag{3.2}$$

Let $r_k(x,y) \equiv \Re(a_k(x+iy))$ and $s_k(x,y) \equiv \Im(a_k(x+iy))$. With this, define $\tilde{A} : \mathbb{R}^{2n} \to \mathbb{R}^{2n}$ by

$$\tilde{A}(x,y) \equiv (r_1(x,y), s_1(x,y), \ldots, r_n(x,y), s_n(x,y)).$$

We construct $G : \mathbb{C}^n \to \mathbb{C}^n$ by

$$G(z) = (g_1(z), \ldots, g_{n-1}(z), g_n(z)),$$

where

$$\begin{aligned} g_k(z) &= (z_k - \check{z}_k) \quad \text{for } 1 \le k \le n-1, \\ g_n(z) &= \frac{(-1)^d \Delta_d}{d!}(z_n - \check{z}_n)^d. \end{aligned} \tag{3.3}$$

Let $p_k(x,y) \equiv \Re(g_k(x+iy))$ and $q_k(x,y) \equiv \Im(g_k(x+iy))$. With this, define $\tilde{G} : \mathbb{R}^{2n} \to \mathbb{R}^{2n}$ by

$$\tilde{G}(x,y) \equiv (p_1(x,y), q_1(x,y), \ldots, p_n(x,y), q_n(x,y)).$$

We will prove $\mathrm{d}(\tilde{F}, \tilde{\boldsymbol{x}}, 0) = \mathrm{d}(\tilde{A}, \tilde{\boldsymbol{x}}, 0) = \mathrm{d}(\tilde{G}, \tilde{\boldsymbol{x}}, 0)$. First, we prove $\mathrm{d}(\tilde{F}, \tilde{\boldsymbol{x}}, 0) = \mathrm{d}(\tilde{A}, \tilde{\boldsymbol{x}}, 0)$.

Define

$$\begin{aligned} &\tilde{H}_1((x,y),t) \equiv t\tilde{F}(x,y) + (1-t)\tilde{A}(x,y) \\ \text{and} \qquad &H_1(z,t) \equiv tF(z) + (1-t)A(z). \end{aligned}$$

We will prove that $\tilde{H}_1((x,y),t) \ne 0$ when $(x,y) \in \partial\tilde{\boldsymbol{x}}$ and $t \in [0,1]$. It is clear that $\tilde{H}_1((x,y),t) = 0$ is equivalent to $H_1(z,t) = 0$, so we consider $H_1(z,t)$. The definition of H_1 and some rearrangement of terms give

$$\begin{aligned} H_1(z,t) = \Big(&(z_1 - \check{z}_1) + \alpha_1(z_n - \check{z}_n), \ldots, (z_{n-1} - \check{z}_{n-1}) + \alpha_{n-1}(z_n - \check{z}_n), \\ &tf_n(z) + (1-t)\frac{(-1)^d \Delta_d}{d!}(z_n - \check{z}_n)^d\Big). \end{aligned}$$

Thus, $H_1(z,t) = 0$ implies $z_k = \check{z}_k - \alpha_k(z_n - \check{z}_n)$ for $k = 1, \ldots, n-1$. By definition, $\alpha_n = -1$, and thus $z_n = \check{z}_n - \alpha_n(z_n - \check{z}_n)$. Substituting $z_k - \check{z}_k = -\alpha_k(z_n - \check{z}_n)$ for each such k ($k = 1, 2, \ldots, n$) in the derivative tensor evaluation $D^\ell f_n(\check{x})[z - \check{z}, \ldots, z - \check{z}]$ in (3.1), we obtain

$$D^\ell f_n(\check{x})[z - \check{z}, \ldots, z - \check{z}] = (-1)^\ell \Delta_\ell (z_n - \check{z}_n)^\ell, \quad 2 \le \ell \le d. \tag{3.4}$$

Since we are assuming that Δ_ℓ, $\ell < d$, vanish, (3.4) and (3.1) give

$$f_n(z) = \frac{(-1)^d \Delta_d}{d!}(z_n - \check{z}_n)^d. \tag{3.5}$$

Thus, the last component of $H_1(z,t)$ is

$$\begin{aligned} tf_n(z) &+ (1-t)\frac{(-1)^d\Delta_d}{d!}(z_n - \check{z}_n)^d \\ &= t\frac{(-1)^d\Delta_d}{d!}(z_n - \check{z}_n)^d + (1-t)\frac{(-1)^d\Delta_d}{d!}(z_n - \check{z}_n)^d \\ &= \frac{(-1)^d\Delta_d}{d!}(z_n - \check{z}_n)^d. \end{aligned}$$

Then, $H_1(z,t) = 0$ implies $(z_n - \check{z}_n)^d = 0$, and consequently, $z_n - \check{z}_n = 0$ or $z_n = \check{z}_n$. This implies $z_k = \check{z}_k - \alpha_k(z_n - \check{z}_n) = \check{z}_k$ for $k = 1, \ldots, n-1$.

Now we know that $H_1(z,t)$ has a unique zero at $(\check{z}_1, \ldots, \check{z}_{n-1}, \check{z}_n)$. This is saying that $\tilde{H}_1((x,y),t)$ has a unique zero at $(\check{x}, \check{y})$, which is the midpoint of nondegenerate box $\tilde{\boldsymbol{x}}$. Thus, $\tilde{H}_1((x,y),t) \neq 0$ for $(x,y) \in \boldsymbol{\partial\tilde{x}}$ and $t \in [0,1]$. Then, by Theorem 1.4,

$$\mathrm{d}(\tilde{F}, \tilde{\boldsymbol{x}}, 0) = \mathrm{d}(\tilde{A}, \tilde{\boldsymbol{x}}, 0).$$

Next, we prove $\mathrm{d}(\tilde{A}, \tilde{\boldsymbol{x}}, 0) = \mathrm{d}(\tilde{G}, \tilde{\boldsymbol{x}}, 0)$. Define

$$\begin{aligned} \tilde{H}_2((x,y),t) &\equiv t\tilde{A}(x,y) + (1-t)\tilde{G}(x,y) \\ \text{and} \qquad H_2(z,t) &\equiv tA(z) + (1-t)G(z). \end{aligned}$$

We will prove that $\tilde{H}_2((x,y),t) \neq 0$ when $(x,y) \in \partial\tilde{\boldsymbol{x}}$ and $t \in [0,1]$. It is clear that $\tilde{H}_2((x,y),t) = 0$ is equivalent to $H_2(z,t) = 0$, so we consider $H_2(z,t)$. The definition of H_2 and some rearrangement of terms give

$$\begin{aligned} H_2(z,t) = \Big(&(z_1 - \check{z}_1) + t\alpha_1(z_n - \check{z}_n), \ldots, (z_{n-1} - \check{z}_{n-1}) + t\alpha_{n-1}(z_n - \check{z}_n), \\ &\frac{(-1)^d\Delta_d}{d!}(z_n - \check{z}_n)^d\Big). \end{aligned}$$

Because of the last component of $H_2(z,t)$, $H_2(z,t) = 0$ implies $z_n = \check{z}_n$. Then, from the first $n-1$ components of $H_2(z,t)$, $H_2(z,t) = 0$ implies $z_k = \check{z}_k - t\alpha_k(z_n - \check{z}_n) = \check{z}_k$ for $k = 1, \ldots, n-1$. Thus, $H_2(z,t)$ has a unique zero at $(\check{z}_1, \ldots, \check{z}_{n-1}, \check{z}_n)$. This is saying that $\tilde{H}_2((x,y),t)$ has a unique zero at $(\check{x}, \check{y})$, which is the midpoint of the nondegenerate box $\tilde{\boldsymbol{x}}$. Thus, $\tilde{H}_2((x,y),t) \neq 0$ for $(x,y) \in \boldsymbol{\partial\tilde{x}}$ and $t \in [0,1]$. Then, by Theorem 1.4,

$$\mathrm{d}(\tilde{A}, \tilde{\boldsymbol{x}}, 0) = \mathrm{d}(\tilde{G}, \tilde{\boldsymbol{x}}, 0).$$

Next, we prove $\mathrm{d}(\tilde{G}, \tilde{\boldsymbol{x}}, 0) = d$. Perturb $G(z)$ by an arbitrary small ϵ to define

$$G_\epsilon(z) = (g_{1\epsilon}(z), \ldots, g_{(n-1)\epsilon}(z), g_{n\epsilon}(z)),$$

where

$$\begin{aligned} g_{k\epsilon}(z) &= g_k(z) = (z_k - \check{z}_k) \quad \text{for } 1 \leq k \leq n-1, \\ g_{n\epsilon}(z) &= g_n(z) + \epsilon = \frac{(-1)^d\Delta_d}{d!}(z_n - \check{z}_n)^d + \epsilon. \end{aligned} \tag{3.6}$$

Let $p_{k\epsilon}(x,y) \equiv \Re(g_{k\epsilon}(x+iy))$ and $q_{k\epsilon}(x,y) \equiv \Im(g_{k\epsilon}(x+iy))$. With this, define

$$\tilde{G}_\epsilon(x,y) \equiv (p_{1\epsilon}(x,y), q_{1\epsilon}(x,y), \ldots, p_{n\epsilon}(x,y), q_{n\epsilon}(x,y)).$$

It is obvious that $p_{k\epsilon}(x,y) = x_k - \check{x}_k$ and $q_{k\epsilon}(x,y) = y_k - \check{y}_k$ for $k = 1, \ldots, n-1$. Assume that ϵ is small enough. Then $G_\epsilon(z)$, and thus $\tilde{G}_\epsilon(x,y)$, have d zeros $\tilde{z} = (\tilde{z}_1, \ldots, \tilde{z}_{n-1}, \tilde{z}_n)$, or $\tilde{x} = (\tilde{x}_1, \tilde{y}_1, \ldots, \tilde{x}_{n-1}, \tilde{y}_{n-1}, \tilde{x}_n, \tilde{x}_n)$ in $\tilde{\boldsymbol{x}}$, with $\tilde{z}_k - \check{z}_k = 0$, or $\tilde{x}_k - \check{x}_k = 0$ and $\tilde{y}_k - \check{y}_k = 0$ for $k = 1, \ldots, n-1$, and $(\tilde{z}_n - \check{z}_n)^d = \frac{d!\epsilon}{(-1)^{d+1}\Delta_d} \neq 0$. $\frac{\partial g_{n\epsilon}}{\partial z_n}(\tilde{z}) = \frac{(-1)^d \Delta_d}{(d-1)!}(\tilde{z}_n - \check{z}_n)^{d-1} \neq 0$.

$$\left|\frac{\partial \tilde{G}_\epsilon}{\partial x_1 \partial y_1 \ldots \partial x_n \partial y_n}(\tilde{x})\right| = \begin{vmatrix} 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \\ 0 & 0 & \cdots & 1 & 0 & 0 \\ 0 & 0 & \cdots & 0 & \frac{\partial p_{n\epsilon}}{\partial x_n} & \frac{\partial p_{n\epsilon}}{\partial y_n} \\ 0 & 0 & \cdots & 0 & \frac{\partial q_{n\epsilon}}{\partial x_n} & \frac{\partial q_{n\epsilon}}{\partial y_n} \end{vmatrix}$$

$$(3.7) \qquad = \begin{vmatrix} \frac{\partial p_{n\epsilon}}{\partial x_n} & \frac{\partial p_{n\epsilon}}{\partial y_n} \\ \frac{\partial q_{n\epsilon}}{\partial x_n} & \frac{\partial q_{n\epsilon}}{\partial y_n} \end{vmatrix} = \begin{vmatrix} \frac{\partial p_{n\epsilon}}{\partial x_n} & \frac{\partial p_{n\epsilon}}{\partial y_n} \\ -\frac{\partial p_{n\epsilon}}{\partial y_n} & \frac{\partial p_{n\epsilon}}{\partial x_n} \end{vmatrix}$$

$$= \left(\frac{\partial p_{n\epsilon}}{\partial x_n}\right)^2 + \left(\frac{\partial p_{n\epsilon}}{\partial y_n}\right)^2 = \left|\frac{\partial g_{n\epsilon}}{\partial z_n}(\tilde{z})\right|^2 > 0.$$

Thus, by Theorem 1.2, $\mathrm{d}(\tilde{G}_\epsilon, \tilde{\boldsymbol{x}}, 0) = d$, and then $\mathrm{d}(\tilde{G}, \tilde{\boldsymbol{x}}, 0) = d$ by Theorem 1.5. Finally,

$$\mathrm{d}(\tilde{F}, \tilde{\boldsymbol{x}}, 0) = \mathrm{d}(\tilde{G}, \tilde{\boldsymbol{x}}, 0) = d. \qquad \square$$

Unless the components of F are exactly linear and degree-d polynomials, the $\mathcal{O}\left(\|x - \check{x}\|\right)^2$ and $\mathcal{O}\left(\|x - \check{x}\|\right)^{d+1}$ terms in (2.1) and (2.2) are not absent. However, since $\mathrm{d}(F, \boldsymbol{z}, 0)$ is a continuous function of F, $\mathrm{d}(F, \boldsymbol{z}, 0)$ will still be equal to d if the widths $\mathrm{w}(\boldsymbol{x}_k - \check{x}_k)$ (and hence $\|x_k - \check{x}_k\|$) are small, for $1 \le k \le n$. Nonetheless, the proof of Theorem 3.1 does not lead to a practical computational verification technique that the degree is d for such more general F: If we try to verify $H(z,t) \neq 0$ or $\tilde{H}((x,y),t) \neq 0$ when $(x,y) \in \partial\tilde{\boldsymbol{x}}$ and $t \in [0,1]$, then it would require an inordinate amount of work for a verification process that would normally require only a single step of an interval Newton method in the nonsingular case. First, we would need to compute Δ_d, which involves all partial derivatives of order 1 and order d. This is expensive when both n and d are large. Second, we would need to know where the solutions of $u_n(x) = 0$ and $v_n(x) = 0$ are on $\boldsymbol{x}_{\underline{n}}$, $\boldsymbol{x}_{\overline{n}}$, $\boldsymbol{y}_{\underline{n}}$, and $\boldsymbol{y}_{\overline{n}}$ when $z_k = \check{z}_k - t\alpha_k(z_n - \check{z}_n)$, and the search process for such solutions is expensive.

We could try to verify $H(z,t) \neq 0$ when $(x,y) \in \partial\tilde{\boldsymbol{x}}$ and $t \in [0,1]$ in another way: verify that $H(z,t) = 0$ has a unique solution in the interior of $\tilde{\boldsymbol{x}}$ when $t \in [0,1]$. However, we will run into the singular situation again if we do that.

In fact, there is an alternative algorithm to verify that the degree is d. That will be the subject of the next section.

4. Algorithm to verify a nonzero topological degree.

The algorithm we present here is similar to the algorithm in [14]. Based on Theorem 1.7 in section 1.2, the following theorem underlies our algorithm.

THEOREM 4.1. *Suppose that*

1. *$u_k \neq 0$ on $\boldsymbol{x}_{\underline{k}}$ and $\boldsymbol{x}_{\overline{k}}$, and $v_k \neq 0$ on $\boldsymbol{y}_{\underline{k}}$ and $\boldsymbol{y}_{\overline{k}}$, $k = 1, \ldots, n-1$;*
2. *$\tilde{F}_{\neg u_n} = 0$ has solutions, if there are any, on $\boldsymbol{x}_{\underline{n}}$ and $\boldsymbol{x}_{\overline{n}}$ with y_n in the interior of $\boldsymbol{y}_n$, and $\tilde{F}_{\neg u_n} = 0$ has solutions, if there are any, on $\boldsymbol{y}_{\underline{n}}$ and $\boldsymbol{y}_{\overline{n}}$ with x_n in the interior of $\boldsymbol{x}_n$;*
3. *$u_n \neq 0$ at the solutions of $\tilde{F}_{\neg u_n} = 0$ in condition* 2; *and*
4. *the Jacobi matrices of $\tilde{F}_{\neg u_n}$ are nonsingular at the solutions of $\tilde{F}_{\neg u_n} = 0$ in condition* 2.

Then, for a fixed $s \in \{-1, 1\}$,

$$
\begin{aligned}
\mathrm{d}(\tilde{F}, \tilde{\boldsymbol{x}}, 0) = -s & \sum_{\substack{x_n = \underline{x}_n \\ \tilde{F}_{\neg u_n}(x,y)=0 \\ \mathrm{sgn}(u_n(x,y))=s}} \mathrm{sgn} \left| \frac{\partial \tilde{F}_{\neg u_n}}{\partial x_1 y_1 \cdots x_{n-1} y_{n-1} y_n}(x, y) \right| \\
+ s & \sum_{\substack{x_n = \overline{x}_n \\ \tilde{F}_{\neg u_n}(x,y)=0 \\ \mathrm{sgn}(u_n(x,y))=s}} \mathrm{sgn} \left| \frac{\partial \tilde{F}_{\neg u_n}}{\partial x_1 y_1 \cdots x_{n-1} y_{n-1} y_n}(x, y) \right| \\
+ s & \sum_{\substack{y_n = \underline{y}_n \\ \tilde{F}_{\neg u_n}(x,y)=0 \\ \mathrm{sgn}(u_n(x,y))=s}} \mathrm{sgn} \left| \frac{\partial \tilde{F}_{\neg u_n}}{\partial x_1 y_1 \cdots x_{n-1} y_{n-1} x_n}(x, y) \right| \\
- s & \sum_{\substack{y_n = \overline{y}_n \\ \tilde{F}_{\neg u_n}(x,y)=0 \\ \mathrm{sgn}(u_n(x,y))=s}} \mathrm{sgn} \left| \frac{\partial \tilde{F}_{\neg u_n}}{\partial x_1 y_1 \cdots x_{n-1} y_{n-1} x_n}(x, y) \right| .
\end{aligned}
$$

Proof. Condition 1 implies $\tilde{F} \neq 0$ on $\boldsymbol{x}_{\underline{k}}$, $\boldsymbol{x}_{\overline{k}}$, $\boldsymbol{y}_{\underline{k}}$, and $\boldsymbol{y}_{\overline{k}}$, $k = 1, \ldots, n-1$, and conditions 2 and 3 imply $\tilde{F} \neq 0$ on $\boldsymbol{x}_{\underline{n}}$, $\boldsymbol{x}_{\overline{n}}$, $\boldsymbol{y}_{\underline{n}}$, and $\boldsymbol{y}_{\overline{n}}$. Thus, $\tilde{F} \neq 0$ on $\boldsymbol{\partial} \tilde{\boldsymbol{x}}$. Now, condition 1 implies $\tilde{F}_{\neg u_n} \neq 0$ on $\boldsymbol{\partial x}_{\underline{k}}$, $\boldsymbol{\partial x}_{\overline{k}}$, $\boldsymbol{\partial y}_{\underline{k}}$, and $\boldsymbol{\partial y}_{\overline{k}}$, $k = 1, \ldots, n-1$. $\boldsymbol{\partial x}_{\underline{n}}$ consists of $2(n-1)$ $(2n-2)$-dimensional boxes, each of which is either embedded in some $\boldsymbol{x}_{\underline{k}}$, $\boldsymbol{x}_{\overline{k}}$, $\boldsymbol{y}_{\underline{k}}$, or $\boldsymbol{y}_{\overline{k}}$, $1 \leq k \leq n-1$, or is embedded in $\boldsymbol{\partial y}_{\underline{n}}$ or $\boldsymbol{\partial y}_{\overline{n}}$. Thus, by 1 and 2, $\tilde{F}_{\neg u_n} \neq 0$ on $\boldsymbol{\partial x}_{\underline{n}}$. Similarly, $\tilde{F}_{\neg u_n} \neq 0$ on $\boldsymbol{\partial x}_{\overline{n}}$, $\boldsymbol{\partial y}_{\underline{n}}$, and $\boldsymbol{\partial y}_{\overline{n}}$. Thus, condition 1 in Theorem 1.7 is satisfied. Finally, with condition 4, all the conditions of Theorem 1.7 are satisfied. The formula is thus obtained. □

By constructing the box $\tilde{\boldsymbol{x}}$ according to (2.5), we can verify $u_k \neq 0$ on $\boldsymbol{x}_{\underline{k}}$ and $\boldsymbol{x}_{\overline{k}}$, and $v_k \neq 0$ on $\boldsymbol{y}_{\underline{k}}$ and $\boldsymbol{y}_{\overline{k}}$, $k = 1, \ldots, n-1$, since $u_k(x, y) \approx (x_k - \check{x}_k) + \alpha_k (x_n - \check{x}_n) \neq 0$ on $\boldsymbol{x}_{\underline{k}}$ and $\boldsymbol{x}_{\overline{k}}$, and $v_k(x, y) \approx y_k + \alpha_k y_n \neq 0$ on $\boldsymbol{y}_{\underline{k}}$ and $\boldsymbol{y}_{\overline{k}}$. This needs only $4n-4$ interval evaluations. Then, we need to search only the four faces $\boldsymbol{x}_{\underline{n}}, \boldsymbol{x}_{\overline{n}}, \boldsymbol{y}_{\underline{n}}$, and $\boldsymbol{y}_{\overline{n}}$ for solutions of $\tilde{F}_{\neg u_n}(x, y) = 0$, regardless of how large n is. The four faces $\boldsymbol{x}_{\underline{n}}$, $\boldsymbol{x}_{\overline{n}}$, $\boldsymbol{y}_{\underline{n}}$, and $\boldsymbol{y}_{\overline{n}}$ remaining to be searched are $(2n-1)$-dimensional boxes. However, exploitation of (2.3) will reduce the search for solutions of $\tilde{F}_{\neg u_n}(x, y) = 0$ on the $(2n-1)$-dimensional boxes to a one-dimensional search. We use $\boldsymbol{x}_{\underline{n}}$ as an example to explain this.

On $\boldsymbol{x}_{\underline{n}}$, $x_n = \underline{x}_n$. We know from (2.3) that if x_n is known precisely, formally solving $\boldsymbol{u}_k(\boldsymbol{x}, \boldsymbol{y}) = 0$ for x_k gives sharper bounds $\tilde{\boldsymbol{x}}_k$ with $\mathrm{w}(\tilde{\boldsymbol{x}}_k) = \mathcal{O}\left(\|(\boldsymbol{x} - \check{x}, \boldsymbol{y})\|\right)^2$,

$1 \le k \le n-1$. Then, we can divide $\boldsymbol{y}_n$ into smaller subintervals. For a small subinterval $\boldsymbol{y}_n^0$ of $\boldsymbol{y}_n$, we can formally solve $\boldsymbol{v}_k(\boldsymbol{x},\boldsymbol{y})=0$ for y_k to get sharper bounds $\tilde{\boldsymbol{y}}_k$ with $\mathrm{w}(\tilde{\boldsymbol{y}}_k) = \mathcal{O}(\max(\|(\boldsymbol{x}-\check{x},\boldsymbol{y})\|^2, \|\boldsymbol{y}_n^0\|))$, $1 \le k \le n-1$. Thus, we have reduced the search to searching the one-dimensional interval $\boldsymbol{y}_n$, much less costly than searching a $(2n-1)$-dimensional box when n is large. Furthermore, if we know approximately where the solutions of $\tilde{F}_{\neg u_n}(x,y)=0$ are, we can reduce even the cost of the one-dimensional search. To this end, we will next analyze the solutions of $\tilde{F}_{\neg u_n}(x,y)=0$ on the four faces $\boldsymbol{x}_{\underline{n}}$, $\boldsymbol{x}_{\overline{n}}$, $\boldsymbol{y}_{\underline{n}}$, and $\boldsymbol{y}_{\overline{n}}$.

To expedite the search, we obtain approximate locations of the places on $\boldsymbol{x}_{\underline{n}}$, $\boldsymbol{x}_{\overline{n}}$, $\boldsymbol{y}_{\underline{n}}$, and $\boldsymbol{y}_{\overline{n}}$ where $\tilde{F}_{\neg u_n}(x,y)=0$. To obtain these locations, we assume that the $\mathcal{O}\left(\|x-\check{x}\|\right)^2$ terms in (2.1) and the $\mathcal{O}\left(\|x-\check{x}\|\right)^{d+1}$ terms in (2.2) are absent. Proceeding as in the proof of Theorem 3.1, we plug $z_k - \check{z}_k = -\alpha_k(z_n-\check{z}_n)$, $k = 1,\ldots,n-1$, into $f_n(z)$ to obtain

$$f_n(z) = \frac{(-1)^d\Delta_d}{d!}(z_n-\check{z}_n)^d$$

as before. Thus, $u_n(x,y) = \Re(f_n(z)) = \left\{(-1)^d\Delta_d/d!\right\}\Re((z_n-\check{z}_n)^d)$ and $v_n(x,y) = \Im(f_n(z)) = \left\{(-1)^d\Delta_d/d!\right\}\Im((z_n-\check{z}_n)^d)$. Setting $z_n - \check{z}_n = r(\cos(\theta)+i\sin(\theta))$, we obtain $u_n(x,y) = \left\{(-1)^d\Delta_d/d!\right\}r\cos(d\theta)$ and $v_n(x,y) = \left\{(-1)^d\Delta_d/d!\right\}r\sin(d\theta)$, so $u_n(x,y)=0$ is equivalent to $\cos(d\theta)=0$ and $v_n(x,y)=0$ is equivalent to $\sin(d\theta)=0$. If we choose $\boldsymbol{x}_n$ and $\boldsymbol{y}_n$ such that

$$\text{(4.1)} \qquad \frac{\mathrm{w}(\boldsymbol{y}_n)}{\mathrm{w}(\boldsymbol{x}_n)} = \tan\left(\frac{\pi}{4d}\right), \qquad \text{that is,} \qquad \mathrm{w}(\boldsymbol{y}_n) = \tan\left(\frac{\pi}{4d}\right)\mathrm{w}(\boldsymbol{x}_n),$$

then all solutions of $v_n(x,y)=0$, and consequently all solutions of $\tilde{F}_{\neg u_n}(x,y)=0$, are arranged in a known pattern on $\boldsymbol{x}_{\underline{n}}$, $\boldsymbol{x}_{\overline{n}}$, $\boldsymbol{y}_{\underline{n}}$, and $\boldsymbol{y}_{\overline{n}}$. In particular, on $\boldsymbol{x}_{\underline{n}}$, $\tilde{x}_n = \underline{x}_n$. $v_n(x,y)=0$ has a unique solution $\tilde{y}_n=0$. Substituting these into the conditions

$$\text{(4.2)} \qquad \left.\begin{array}{rcl} x_k &=& \check{x}_k - \alpha_k(x_n-\check{x}_n), \\ y_k &=& -\alpha_k y_n, \end{array}\right\} \qquad 1 \le k \le n-1,$$

we get the unique solution of $\tilde{F}_{\neg u_n}(x,y)=0$ with

$$(\tilde{x},\tilde{y}) = (\check{x}_1 - \alpha_1(\underline{x}_n-\check{x}_n), 0, \ldots, \check{x}_{n-1}-\alpha_{n-1}(\underline{x}_n-\check{x}_n), 0, \underline{x}_n, 0)\,.$$

Similarly, $\tilde{F}_{\neg u_n}(x,y)=0$ has a unique solution on $\boldsymbol{x}_{\overline{n}}$ with

$$(\tilde{x},\tilde{y}) = (\check{x}_1 - \alpha_1(\overline{x}_n-\check{x}_n), 0, \ldots, \check{x}_{n-1}-\alpha_{n-1}(\overline{x}_n-\check{x}_n), 0, \overline{x}_n, 0).$$

On $\boldsymbol{y}_{\underline{n}}$, $\tilde{y}_n = \underline{y}_n$. $v_n(x,y)=0$ has $d-1$ solutions with

$$\text{(4.3)} \qquad \tilde{x}_n = \check{x}_n + \frac{\mathrm{w}(\boldsymbol{y}_n)}{2\tan\left(\frac{m\pi}{d}\right)}, \quad m = d-1, d-2, \ldots, 1.$$

Substituting these into (4.2) gives the $d-1$ solutions $(\tilde{x},\tilde{y})$ of $\tilde{F}_{\neg u_n}(x,y)=0$ with

$$\begin{aligned}(\tilde{x},\tilde{y}) = \big(&\check{x}_1 - \alpha_1\left(\tilde{x}_n-\check{x}_n\right), \alpha_1\underline{y}_n, \ldots, \check{x}_{n-1}-\alpha_{n-1}\left(\tilde{x}_n-\check{x}_n\right), \\ &-\alpha_{n-1}\underline{y}_n, \tilde{x}_n, \underline{y}_n\big).\end{aligned}$$

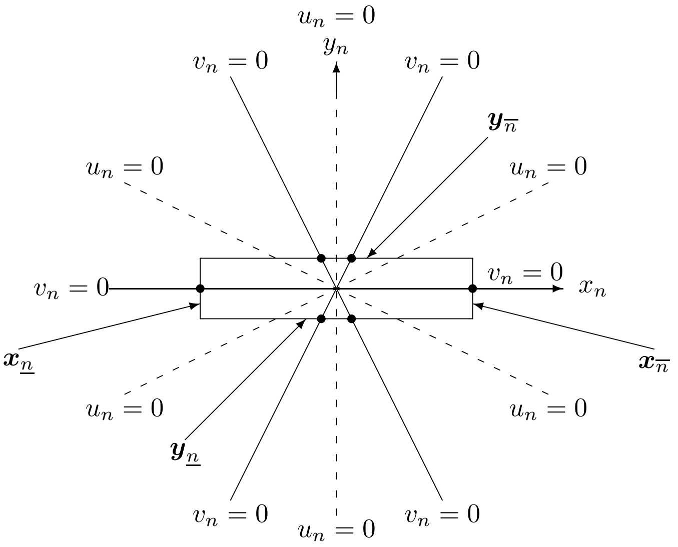
$u_n = 0$
y_n
$v_n = 0$
$v_n = 0$
$\boldsymbol{y}_{\overline{n}}$
$u_n = 0$
$u_n = 0$
$v_n = 0$
$v_n = 0$
x_n
$\boldsymbol{x}_{\underline{n}}$
$\boldsymbol{x}_{\overline{n}}$
$u_n = 0$
$u_n = 0$
$\boldsymbol{y}_{\underline{n}}$
$v_n = 0$
$v_n = 0$
$u_n = 0$

FIG. 4.1. *The zero structure when d is odd. Here, $d = 3$. $v_n = 0$ on solid lines, and $u_n = 0$ on dashed lines. The thick dots are the solutions of $\tilde{F}_{\neg u_n}(x, y) = 0$ on $\boldsymbol{\partial\tilde{x}}$.*

Similarly, $\tilde{F}_{\neg u_n}(x, y) = 0$ has $d - 1$ solutions on $\boldsymbol{y}_{\overline{n}}$ with

$$\begin{aligned}(\tilde{x}, \tilde{y}) = (&\check{x}_1 - \alpha_1 (\tilde{x}_n - \check{x}_n), \ldots, \check{x}_{n-1} - \alpha_{n-1} (\tilde{x}_n - \check{x}_n), \\ &- \alpha_{n-1}\overline{y}_n, \tilde{x}_n, \overline{y}_n).\end{aligned}$$

For example, Figure 4.1 gives the solutions of $v_n(x, y) = 0$ on the four faces $\boldsymbol{x}_{\underline{n}}$, $\boldsymbol{x}_{\overline{n}}$, $\boldsymbol{y}_{\underline{n}}$, and $\boldsymbol{y}_{\overline{n}}$ when $d = 3$.

To use the above analysis to find approximations to the solutions of $\tilde{F}_{\neg u_n} = 0$ on the faces we search, we need to know d; we present a heuristic for d in section 5 below.

Now, we present our algorithm. The algorithm consists of three phases:

1. the box-construction phase, where we set $\tilde{\boldsymbol{x}}$,
2. the elimination phase, where we use interval evaluations to verify that $u_k \neq 0$ on $\boldsymbol{x}_{\underline{k}}$ and $\boldsymbol{x}_{\overline{k}}$, and $v_k \neq 0$ on $\boldsymbol{y}_{\underline{k}}$ and $\boldsymbol{y}_{\overline{k}}$, where $1 \leq k \leq n - 1$, and thus eliminate those $4n - 4$ faces, and
3. the search phase, where we
   (a) search $\boldsymbol{x}_{\underline{n}}$, $\boldsymbol{x}_{\overline{n}}$, $\boldsymbol{y}_{\underline{n}}$, and $\boldsymbol{y}_{\overline{n}}$ to locate the solutions of $\tilde{F}_{\neg u_n}(x, y) = 0$,
   (b) compute the signs of u_n and determinants of the Jacobi matrices of $\tilde{F}_{\neg u_n}$ at those solutions,
   (c) compute the degree contributions of each of the four faces $\boldsymbol{x}_{\underline{n}}$, $\boldsymbol{x}_{\overline{n}}$, $\boldsymbol{y}_{\underline{n}}$, and $\boldsymbol{y}_{\overline{n}}$ according to Theorem 4.1, and
   (d) finally sum up to get the degree.

ALGORITHM 1.

INPUT: An approximate solution $\check{x} \in \mathbf{D} \subseteq \mathbb{R}^n$ and a heuristically derived guess d for the topological index of the solution to $\tilde{F}(z) = 0$ near $\check{x}$. (See section 5 below.)

OUTPUT: Either "`A solution is verified`" or "`Verification failed`." If a solution is verified, then also output real bounds $\boldsymbol{x} \subset \mathbb{R}^n$, $\check{x} \in \boldsymbol{x}$, and imaginary bounds $\boldsymbol{y} \in \mathbb{R}^n$, $0 \in \boldsymbol{y}$, such that a solution of $\tilde{F}(z) = 0$ must lie in $(\boldsymbol{x}_1 + i\boldsymbol{y}_1, \ldots, \boldsymbol{x}_n + i\boldsymbol{y}_n) \in \mathbb{IC}^n$.

Box-setting phase.

```
1. Compute the preconditioner of the original system, using Gaussian elimination with full pivoting.
2. Set the widths of $\boldsymbol{x}_k$ and $\boldsymbol{y}_k$ (see explanation below), for $1 \le k \le n-1$.
3. Set the width of $\boldsymbol{x}_n$ as in (2.5).
4. Set the width of $\boldsymbol{y}_n$ to be the minimum of that obtained from conditions (2.5) and (4.1).
```

Elimination phase.

```
Do for $1 \le k \le n-1$
1. DO for $\boldsymbol{x}_{\underline{k}}$ and $\boldsymbol{x}_{\overline{k}}$
   (a) Compute the mean-value extension of $\boldsymbol{u}_k$ over that face.
   (b) IF $0 \in \boldsymbol{u}_k$, THEN STOP and signal failure.
   END DO
2. DO for $\boldsymbol{y}_{\underline{k}}$ and $\boldsymbol{y}_{\overline{k}}$
   (a) Compute the mean-value extension of $\boldsymbol{v}_k$ over that face.
   (b) IF $0 \in \boldsymbol{v}_k$, THEN STOP and signal failure.
   END DO
```

Search phase.

```
1. Set the value of $s \in \{+1, -1\}$.
      (a) Initialize $s$ to be $+1$. Initialize search_lower and search_upper to be false.
          (See the second note below.)
      (b) DO for $\boldsymbol{x}_{\underline{n}}$ and $\boldsymbol{x}_{\overline{n}}$
            i. Use mean-value extensions for $\boldsymbol{u}_k(\boldsymbol{x}, \boldsymbol{y}) = 0$ to solve for $x_k$ to get sharper bounds $\tilde{\boldsymbol{x}}_k$ with width $\mathcal{O}\left(\|(\boldsymbol{x} - \check{x}, \boldsymbol{y})\|\right)^2$, $1 \le k \le n-1$, and thus to get a subface $\boldsymbol{x}^0_{\underline{n}}$ (or $\boldsymbol{x}^0_{\overline{n}}$) of $\boldsymbol{x}_{\underline{n}}$ (or $\boldsymbol{x}_{\overline{n}}$.).
           ii. IF $\tilde{\boldsymbol{x}}_k \cap \boldsymbol{x}_k = \emptyset$, THEN CYCLE.
          iii. Compute the mean-value extension $\boldsymbol{u}_n$ over $\boldsymbol{x}^0_{\underline{n}}$ (or $\boldsymbol{x}^0_{\overline{n}}$).
           iv. IF $\boldsymbol{u}_n$ contains 0, THEN
                 A. set search_lower (or search_upper) to be true
                 B. CYCLE.
               END IF
            v. IF $\boldsymbol{u}_n$ does not contain 0, THEN set $s = -\mathrm{sgn}(\boldsymbol{u}_n)$.
          END DO
      (c) IF $\boldsymbol{u}_n$ does not contain 0 on both $\boldsymbol{x}_{\underline{n}}$ and $\boldsymbol{x}_{\overline{n}}$,
          THEN set $s$ to be the opposite sign to the sign of $\boldsymbol{u}_n$ on $\boldsymbol{x}_{\overline{n}}$, and
          IF $\boldsymbol{u}_n$ has different signs on $\boldsymbol{x}_{\underline{n}}$ and $\boldsymbol{x}_{\overline{n}}$,
          THEN set search_lower to be true.
2. For $\boldsymbol{x}_{\underline{n}}$ (or $\boldsymbol{x}_{\overline{n}}$), IF search_lower (or search_upper) is true,
   THEN apply Algorithm 2 with $\boldsymbol{x}_{\underline{n}}$ (or $\boldsymbol{x}_{\overline{n}}$) and 0 as input, to compute the degree contribution of $\boldsymbol{x}_{\underline{n}}$ (or $\boldsymbol{x}_{\overline{n}}$).
3. For $\boldsymbol{y}_{\underline{n}}$ (or $\boldsymbol{y}_{\overline{n}}$)
   (a) Use (4.3) to compute the $\tilde{x}^m_n$, $m = d-1, d-2, \ldots, 1$, $\tilde{x}^{d-1}_n < \tilde{x}^{d-2}_n < \cdots < \tilde{x}^1_n$, corresponding to the $d-1$ approximate solutions of $\tilde{F}_{\neg u_n} = 0$ on $\boldsymbol{y}_{\underline{n}}$.
   (b) Divide $\boldsymbol{x}_n$ into $d-1$ parts $\boldsymbol{x}^m_n$, $m = 1, \ldots, d-1$, as follows:
       $\boldsymbol{x}^1_n = [\underline{x}_n, (\tilde{x}^{d-1}_n + \tilde{x}^{d-2}_n)/2]$,
       $\boldsymbol{x}^m_n = [(\tilde{x}^{d-(m-1)}_n + \tilde{x}^{d-m}_n)/2, (\tilde{x}^{d-m}_n + \tilde{x}^{d-(m+1)}_n)/2]$
       for $m = 2, \ldots, d-2$, and $\boldsymbol{x}^{d-1}_n = [(\tilde{x}^2_n + \tilde{x}^1_n)/2, \overline{x}_n]$.
   (c) DO for $m = 1, \ldots, d-1$
         i. Set a subface $\boldsymbol{y}^m_{\underline{n}}$ of $\boldsymbol{y}_{\underline{n}}$ (or $\boldsymbol{y}^m_{\overline{n}}$ of $\boldsymbol{y}_{\overline{n}}$) by replacing $\boldsymbol{x}_n$ by $\boldsymbol{x}^m_n$.
```

```
            ii. Apply Algorithm 3 with $\boldsymbol{y}_{\underline{n}}^m$ and $\tilde{x}_n^m$ as inputs, to compute the degree
                contribution of $\boldsymbol{y}_{\underline{n}}^m$ (or $\boldsymbol{y}_{\overline{n}}^m$).
            END DO
        (d) Add the degree contributions in the last step to get the degree contribution of $\boldsymbol{y}_{\underline{n}}$ (or $\boldsymbol{y}_{\overline{n}}$).
    4. Add the degree contributions of $\boldsymbol{x}_{\underline{n}}$, $\boldsymbol{x}_{\overline{n}}$, $\boldsymbol{y}_{\underline{n}}$, and $\boldsymbol{y}_{\overline{n}}$ to get the overall degree.
```

Notes for Algorithm 1.

1. In step 3 of the box-setting phase, the width $\mathrm{w}(\boldsymbol{x}_n)$ of $\boldsymbol{x}_n$ depends on the accuracy of the approximate solution $\check{x}$ of the system $F(x) = 0$: $\mathrm{w}(\boldsymbol{x}_n)$ should be much larger than $|\check{x}_k - x^*{}_k|$, but also should be small enough to make a quadratic model accurate over the box.
2. We may set s to minimize the amount of work required to evaluate the sum in Theorem 4.1. In particular, if we know $\mathrm{sgn}(u_n) = \sigma$ on a large number of faces, then setting $s = -\sigma$ will eliminate the need to search those faces.

Algorithm 2.

```
INPUT: $\boldsymbol{x}_{\underline{n}}$ (or $\boldsymbol{x}_{\overline{n}}$) and $\boldsymbol{y}$ from Algorithm 1.
OUTPUT: The contribution of $\boldsymbol{x}_{\underline{n}}$ (or $\boldsymbol{x}_{\overline{n}}$) to the degree in Algorithm 1.
    1. (a) Use mean-value extensions for $\boldsymbol{u}_k(\boldsymbol{x}, \boldsymbol{y}) = 0$ to solve for $x_k$ to get sharper
           bounds $\tilde{\boldsymbol{x}}_k$ with width $\mathcal{O}\left(\|(\boldsymbol{x} - \check{x}, \boldsymbol{y})\|\right)^2$, $1 \le k \le n-1$.
       (b) IF $\tilde{\boldsymbol{x}}_k \cap \boldsymbol{x}_k = \emptyset$,
           THEN RETURN the degree contribution of that face as 0.
       (c) Update $\boldsymbol{x}_k$.
    2. (a) Compute the mean-value extension $\boldsymbol{u}_n$ over that face.
       (b) IF $s \times \mathrm{sgn}(\boldsymbol{u}_n) < 0$,
           THEN RETURN the degree contribution of that face as 0.
    3. Construct a small subinterval $\boldsymbol{y}_n^0$ of $\boldsymbol{y}_n$ centered at $\check{y}_n$.
    4. (Steps 4 to 9 are identical to steps 1(d) to 1(i), respectively, of the search phase
       in the algorithm in [14]. These steps are repeated here for completeness.)
       (a) Use mean-value extensions for $\boldsymbol{v}_k(\boldsymbol{x}, \boldsymbol{y}) = 0$ to solve for $y_k$ to get sharper
           bounds $\tilde{\boldsymbol{y}}_k$ with width $\mathcal{O}(\max(\|(\boldsymbol{x} - \check{x}, \boldsymbol{y})\|^2, \|\boldsymbol{y}_n^0\|))$, $1 \le k \le n-1$, thus
           getting a subface $\boldsymbol{x}_{\underline{n}}^0$ (or $\boldsymbol{x}_{\overline{n}}^0$) of $\boldsymbol{x}_{\underline{n}}$ (or $\boldsymbol{x}_{\overline{n}}$.)
       (b) IF $\tilde{\boldsymbol{y}}_k \cap \boldsymbol{y}_k = \emptyset$,
           THEN STOP and signal failure.
    5. (a) Set up an interval Newton method for $\tilde{F}_{\neg u_n}$ to verify existence and
           uniqueness of a zero in the subface $\boldsymbol{x}_{\underline{n}}^0$ (or $\boldsymbol{x}_{\overline{n}}^0$).
       (b) IF the zero cannot be verified,
           THEN STOP and signal failure.
    6. Inflate $\boldsymbol{y}_n^0$ as much as possible subject to verification of existence and uniqueness
       of the zero of $\tilde{F}_{\neg u_n}$ over the corresponding subface, and thus get a subinterval
       $\boldsymbol{y}_n^1$ of $\boldsymbol{y}_n$.
    7. In this step, we verify that $\tilde{F}_{\neg u_n} = 0$ has no solutions when $y_n \in \boldsymbol{y}_n \setminus \boldsymbol{y}_n^1$.
       $\boldsymbol{y}_n \setminus \boldsymbol{y}_n^1$ has two separate parts; we denote the lower part by $\boldsymbol{y}_n^l$ and the upper
       part by $\boldsymbol{y}_n^u$. We present the processing of only the lower part. The upper
       part can be processed similarly.
       (a) DO
            i. Use mean-value extensions for $\boldsymbol{v}_k(\boldsymbol{x}, \boldsymbol{y}) = 0$ to solve for $y_k$ to get
               sharper bounds for $y_k$, $1 \le k \le n-1$, and thus to get a subface of
               $\boldsymbol{x}_{\underline{n}}$ (or $\boldsymbol{x}_{\overline{n}}$).
           ii. Compute the mean-value extensions $\tilde{\boldsymbol{F}}_{\neg u_n}$ over the subface obtained
               in the last step.
            iii. IF $0 \in \tilde{\boldsymbol{F}}_{\neg u_n}$, THEN
                 A. bisect $\boldsymbol{y}^l_n$, update the lower part as a new $\boldsymbol{y}^l_n$;
                 B. CYCLE.
                 END IF
                 IF $0 \notin \tilde{\boldsymbol{F}}_{\neg u_n}$, THEN EXIT the loop.
            END DO
       (b) DO
            i. IF $\underline{y}^1_n \leq \overline{y}^l_n$, THEN EXIT the loop.
            ii. $\boldsymbol{y}^l_n \longleftarrow [\overline{y}^l_n, \overline{y}^l_n + \mathrm{w}(\boldsymbol{y}^l_n)]$.
            iii. Use mean-value extensions for $\boldsymbol{v}_k(\boldsymbol{x},\boldsymbol{y}) = 0$ to solve for $y_k$ to get
                 sharper bounds for $y_k$, $1 \leq k \leq n-1$, and thus to get a subface of
                 $\boldsymbol{x}_{\underline{n}}$ (or $\boldsymbol{x}_{\overline{n}}$).
            iv. Compute the mean-value extensions $\tilde{\boldsymbol{F}}_{\neg u_n}$ over the subface obtained
                 in the last step.
            v. IF $0 \notin \tilde{\boldsymbol{F}}_{\neg u_n}$, THEN CYCLE.
                 IF $0 \in \tilde{\boldsymbol{F}}_{\neg u_n}$, THEN
                 A. $\boldsymbol{y}^l_n \longleftarrow [\underline{y}^l_n, \mathrm{mid}(\boldsymbol{y}^l_n)]$;
                 B. CYCLE.
                 END IF
            END DO
  8. (a) Compute the mean-value extension of $\boldsymbol{u}_n$ over $\boldsymbol{x}^0_{\underline{n}}$ (or $\boldsymbol{x}^0_{\overline{n}}$).
     (b) IF $\boldsymbol{u}_n < 0$,
         THEN RETURN the degree contribution of that face as 0.
  9. (a) Compute $|\frac{\partial \tilde{F}_{\neg u_n}}{\partial x_1 y_1 \ldots x_{n-1} y_{n-1} y_n}(\boldsymbol{x}^0_{\underline{n}})|$ (or $|\frac{\partial \tilde{F}_{\neg u_n}}{\partial x_1 y_1 \ldots x_{n-1} y_{n-1} y_n}(\boldsymbol{x}^0_{\overline{n}})|$).
     (b) IF $0 \in |\frac{\partial \tilde{F}_{\neg u_n}}{\partial x_1 y_1 \ldots x_{n-1} y_{n-1} y_n}(\boldsymbol{x}^0_{\underline{n}})|$ (or $0 \in |\frac{\partial \tilde{F}_{\neg u_n}}{\partial x_1 y_1 \ldots x_{n-1} y_{n-1} y_n}(\boldsymbol{x}^0_{\overline{n}})|$),
         THEN STOP and signal failure.
 10. Apply Theorem 4.1 to compute the degree contribution of $\boldsymbol{x}_{\underline{n}}$ or $\boldsymbol{x}_{\overline{n}}$.
```

Algorithm 3.

```
INPUT: $\boldsymbol{y}_{\underline{n}}$ (or $\boldsymbol{y}_{\overline{n}}$) and $\boldsymbol{x}$.
OUTPUT: The contribution of $\boldsymbol{y}_{\underline{n}}$ (or $\boldsymbol{y}_{\overline{n}}$) to the degree in Algorithm 1.
  1. (a) Use mean-value extensions for $\boldsymbol{v}_k(\boldsymbol{x},\boldsymbol{y}) = 0$ to solve for $y_k$ to get sharper
         bounds $\tilde{\boldsymbol{y}}_k$ with width $\mathcal{O}\left(\|(\boldsymbol{x}-\check{x},\boldsymbol{y})\|\right)^2$, $1 \leq k \leq n-1$.
     (b) IF $\tilde{\boldsymbol{y}}_k \cap \boldsymbol{y}_k = \emptyset$,
         THEN RETURN the degree contribution of that face as 0.
     (c) Update $\boldsymbol{y}_k$.
  2. (a) Compute the mean-value extension $\boldsymbol{u}_n$ over that face.
     (b) IF $s \times \mathrm{sgn}(\boldsymbol{u}_n) < 0$,
         THEN RETURN the degree contribution of that face as 0.
  3. Construct a small subinterval $\boldsymbol{x}^0_n$ of $\boldsymbol{x}_n$ which is centered at $\check{x}_n$.
  4. (Steps 4 to 9 are identical to steps 2(d) to 2(i), respectively, of the search
     phase in the algorithm in [14], but are included here for completeness.) Same
     as step 4 of Algorithm 2, except change $y_k$ to $x_k$, $\tilde{\boldsymbol{y}}_k$ to $\tilde{\boldsymbol{x}}_k$, $\boldsymbol{y}_k$ to $\boldsymbol{x}_k$, $\boldsymbol{x}^0_{\underline{n}}$ to
     $\boldsymbol{y}^0_{\underline{n}}$, $\boldsymbol{x}^0_{\overline{n}}$ to $\boldsymbol{y}^0_{\overline{n}}$, $\boldsymbol{x}_{\underline{n}}$ to $\boldsymbol{y}_{\underline{n}}$, and $\boldsymbol{x}_{\overline{n}}$ to $\boldsymbol{y}_{\overline{n}}$.
  5. Same as step 5 of Algorithm 2, except change $\boldsymbol{x}^0_{\underline{n}}$ to $\boldsymbol{y}^0_{\underline{n}}$ and $\boldsymbol{x}^0_{\overline{n}}$ to $\boldsymbol{y}^0_{\overline{n}}$.
  6. Same as step 6 of Algorithm 2, except change $\boldsymbol{y}^0_n$ to $\boldsymbol{x}^0_n$, $\boldsymbol{y}^1_n$ to $\boldsymbol{x}^1_n$, and $\boldsymbol{y}_n$
     to $\boldsymbol{x}_n$.
  7. Same as step 7 of Algorithm 2, except change $\boldsymbol{y}_n \setminus \boldsymbol{y}^1_n$ to $\boldsymbol{x}_n \setminus \boldsymbol{x}^1_n$.
```

8. Same as step 8 of Algorithm 2, except change $\boldsymbol{x}^0_{\underline{n}}$ to $\boldsymbol{y}^0_{\underline{n}}$ and $\boldsymbol{x}^0_{\overline{n}}$ to $\boldsymbol{y}^0_{\overline{n}}$.
9. Same as step 9 of Algorithm 2, except change $|\frac{\partial \tilde{F}_{\neg u_n}}{\partial x_1 y_1 \ldots x_{n-1} y_{n-1} y_n}(\boldsymbol{x}^0_{\underline{n}})|$ to $|\frac{\partial \tilde{F}_{\neg u_n}}{\partial x_1 y_1 \ldots x_{n-1} y_{n-1} x_n}(\boldsymbol{y}^0_{\underline{n}})|$ and $|\frac{\partial \tilde{F}_{\neg u_n}}{\partial x_1 y_1 \ldots x_{n-1} y_{n-1} y_n}(\boldsymbol{x}^0_{\overline{n}})|$ to $|\frac{\partial \tilde{F}_{\neg u_n}}{\partial x_1 y_1 \ldots x_{n-1} y_{n-1} x_n}(\boldsymbol{y}^0_{\overline{n}})|$.
10. Same as step 10 of Algorithm 2.

Notes for Algorithms 2 *and* 3.

1. Algorithms 2 and 3 are identical to steps 1 and 2 of the search phase of the algorithm in [14], except, in Algorithm 2, $\check{y}_n$ can be any interior point of $\boldsymbol{y}_n$, while $\check{y}_n$ is assumed to equal zero in step 1 of the search phase in the algorithm in [14]. Similarly, in Algorithm 3, $\check{x}_n$ can be any interior point of $\boldsymbol{x}_n$, whereas $\check{x}_n$ is assumed to equal the center of $\boldsymbol{x}_n$ in step 2 of the search phase in the algorithm in [14].
2. In the overall algorithm, Algorithm 1, the actual inputs are $\boldsymbol{y}^m_{\underline{n}}$ and $\check{x}^m_n$ when Algorithm 3 is applied. However, for notational simplicity, we use $\boldsymbol{y}_{\underline{n}}$ and $\check{x}_n$ as inputs in the presentation of Algorithm 3.

In a certain sense, the computational complexity of Algorithms 1, 2, and 3 is $\mathcal{O}(n^3)$. (See [14] for detailed analysis.) Thus, the computational complexity of the overall algorithm, Algorithm 1, is $\mathcal{O}(n^3)$. This is the best possible order, since computing preconditioners of the original system and the system $\tilde{F}_{\neg u_n}$ is necessary and computing each preconditioner is of order $\mathcal{O}(n^3)$.

5. A heuristic for the degree.

The algorithms in section 4 require a value for d to locate the approximate positions of solutions of $\tilde{F}_{\neg u_n} = 0$ on the faces we search. Here, we present a practical heuristic for the value of d.

Proceeding as in the proof of Theorem 3.1, we assume that the $\mathcal{O}\left(\|x - \check{x}\|\right)^2$ terms in (2.1) and the $\mathcal{O}\left(\|x - \check{x}\|\right)^{d+1}$ terms in (2.2) are absent, and we substitute $x_k - \check{x}_k = -\alpha_k(x_n - \check{x}_n)$, $k = 1, \ldots, n-1$, into f_n to enable us to define the univariate function

$$g(x_n - \check{x}_n) = \frac{(-1)^d \Delta_d}{d!}(x_n - \check{x}_n)^d = \frac{\Delta_d}{d!}(\check{x}_n - x_n)^d. \tag{5.1}$$

Setting

$$K(r, x_n - \check{x}_n) \equiv \frac{g(x_n - \check{x}_n)}{(x_n - \check{x}_n)^r} = \frac{\Delta_d}{d!}(\check{x}_n - x_n)^{d-r},$$

it is clear that $K(d, x_n - \check{x}_n) = \Delta_d/d!$ is independent of x_n, while $K(r, x_n - \check{x}_n)$ depends on x_n for any other r value. Letting δ be a heuristically chosen constant, we have the following ratios:

$$\frac{K(d, \delta(x_n - \check{x}_n))}{K(d, x_n - \check{x}_n)} = \frac{\frac{\Delta_d}{d!}}{\frac{\Delta_d}{d!}} = 1, \quad \text{while}$$

$$R(r) = \frac{K(r, \delta(x_n - \check{x}_n))}{K(r, x_n - \check{x}_n)} = \frac{\frac{\Delta_d}{d!}(\delta(\check{x}_n - x_n))^{d-r}}{\frac{\Delta_d}{d!}(\check{x}_n - x_n)^{d-r}} = \delta^{d-r}$$

for any other r value. The first ratio $R(d)$ always equals 1, but $R(r)$, $r \neq d$, depends on the δ value. We can choose δ to distinguish d from other r values. For example, if we choose $\delta = 100$, then $R(r)$ is not smaller than 100 when r is smaller than d, and is not larger than 0.01 when r is larger than d. Both values are sufficiently different from 1. We can also vary the δ value to check our detection of d. Thus, $R(r)$ is a good heuristic to determine the value of d.

The above discussion is based on the assumptions in section 2. However, unless the first $n-1$ components of F are exactly linear and the last component is a homogeneous degree-d polynomial of n variables, those assumptions are only approximately true. In practice, if $g(x_n - \check{x}_n) \approx \frac{\Delta_d}{d!}(\check{x}_n - x_n)^d$ is an accurate approximation, then $(\check{x}_n - x_n)^d$ should dominate the value of $g(x_n - \check{x}_n)$. Actually,

$$g(x_n - \check{x}_n) = \sum_{k=1}^{d-1} c_k \Delta_k (x_n - \check{x}_n)^k + c_d \Delta_d (x_n - \check{x}_n)^d + \sum_{k=d+1}^{\infty} c_k \Delta_k (x_n - \check{x}_n)^k,$$

where, approximately, $\Delta_1 = \cdots = \Delta_{d-1} = 0$, $\Delta_d \neq 0$. Thus, $x_n - \check{x}_n$ and $\delta(x_n - \check{x}_n)$ should not be too small, since $\sum_{k=1}^{d-1} c_k \Delta_k (x_n - \check{x}_n)^k$ could dominate otherwise. They should not be too big either, since $\sum_{k=d+1}^{\infty} c_k \Delta_k (x_n - \check{x}_n)^k$ could dominate otherwise. If $\Delta_k \approx 0, k = 1, \ldots, d-1$, are quite accurate, then we can choose $x_n - \check{x}_n$ very small, so both $\sum_{k=1}^{d-1} c_k \Delta_k (x_n - \check{x}_n)^k$ and $\sum_{k=d+1}^{\infty} c_k \Delta_k (x_n - \check{x}_n)^k$ can be ignored in the detection of d.

The choice of $x_n - \check{x}_n$ is independent of the settings of $\boldsymbol{x}_k$, $k = 1, \ldots, n$, since we only want to know what d is at that stage.

An alternative choice for detecting d is to compute the values of Δ_k, $k = 1, 2, \ldots$, by interval evaluations until we get some Δ_{k_0} that is sufficiently different from 0. Then, we can decide $d = k_0$. The obvious disadvantage of this method is that it is too expensive for just detecting the value of d, since computation of Δ_k involves computations of all kth-order derivatives. Furthermore, even if we actually evaluate Δ_k, $k = 1, 2, \ldots$, spending much time in the process, we still can not detect the value of d if the magnitudes of Δ_k, $k = 1, \ldots, d-1, d$, are not sufficiently different either due to the problem itself or due to the range overestimation in interval computations.

6. Numerical results.

In this section, we present numerical results for the algorithm in section 4.

The testing described in this section is not meant to be exhaustive, but is meant to illustrate that the algorithms are programmable and do succeed for a variety of problems, as well as to illustrate that the technique can be practical for higher-dimensional problems. We emphasize that, unless there are programming blunders, the implementation can never give an incorrect result. (That is, the degree can never be incorrectly verified to be d.) The only ways that the algorithms can fail are by either asserting that they cannot verify that the degree is d or by running out of computer resources (typically, CPU time limits).

6.1. Test problems.

Our test problems are represented in Examples 1 through 5 below. This set includes both simple problems, such as Example 1, and slightly more realistic problems, such as Example 4. There are both lower degree problems, like Example 2, and slightly higher degree problems, like Example 5.

Consistent with the analysis and algorithms in this paper, the null-space of the Jacobi matrix at the solution has dimension 1 in all of these examples. (We discuss the higher-order rank defect case in [12].)

Examples 2, 3, and 4 are variable-dimension examples coming from finite difference discretization of a bifurcation problem. In choosing these three problems, we looked for a simple way to vary both the actual topological index at the solution and the dimension of the problem. Actual verification procedures for differential equation models should differ somewhat from what is seen here, since the discretization error should also be taken into account, to be able to assert properties about the solutions to the differential equation itself, rather than just properties about solutions of the discretization.

Although Examples 2, 3, and 4 have a special structure (tridiagonal systems), this actual structure was not used in the present algorithms; that is, dense linear algebra was used throughout. In this sense, the observed dependence of computational time on dimension is representative, although the precise form of the nonlinearity conceivably could make a difference.

Example 1.

$$\begin{aligned} f_1(x_1, x_2) &= x_1^2 - x_2, \\ f_2(x_1, x_2) &= x_1^2 + x_2. \end{aligned}$$

Example 2 (the same as Example 3 from [14], motivated from considerations in [7]). *Set $F(x) = H(x,t) = (1-t)(Ax - x^2) - tx$, where $A \in \mathbb{R}^{n \times n}$ is the matrix corresponding to central difference discretization of the boundary value problem $-u'' = 0$, $u(0) = u(1) = 0$, and $x^2 = (x_1^2, \dots, x_n^2)^T$. t was chosen to be equal to $t_1 = \lambda_1/(1+\lambda_1)$, where λ_1 is the largest eigenvalue of A.*

In Example 2, if we change the exponent of x from 2 to 3 and 4, then we get Examples 3 and 4.

Example 3. This example is identical to Example 2, except that we set $F(x) = H(x,t) = (1-t)(Ax - x^3) - tx$.

Example 4. This example is identical to Example 2, except that we set $F(x) = H(x,t) = (1-t)(Ax - x^4) - tx$.

We tested with $n = 5$, 10, 20, 40, 80, and 160 for Examples 2, 3, and 4.

Example 5.

$$\begin{aligned} f_1(x_1, x_2, x_3) &= x_1^5 + x_2 + x_2^6 + 3x_3, \\ f_2(x_1, x_2, x_3) &= 4x_1^5 + 5x_2 - 4x_2^6 + 5x_3 - x_3^6, \\ f_3(x_1, x_2, x_3) &= 7x_1^5 + 8x_2 - 100x_2^7 + 10x_3 + 50x_3^6. \end{aligned}$$

For each test problem, we used $(0, 0, \dots, 0)$, the exact solution to $F(x) = 0$, as the approximate solution to the problem $F(x) = 0$. For each problem except Example 4, we set the widths $\mathrm{w}(\boldsymbol{x}_k)$ and $\mathrm{w}(\boldsymbol{y}_k)$ to 10^{-2} for $1 \le k \le n-1$; then the algorithm automatically computed $\mathrm{w}(\boldsymbol{x}_n)$ and $\mathrm{w}(\boldsymbol{y}_n)$. For Example 4, we set the widths $\mathrm{w}(\boldsymbol{x}_k)$ and $\mathrm{w}(\boldsymbol{y}_k)$ to 10^{-1}, instead of 10^{-2}, for $1 \le k \le n-1$. The reason for this setting for Example 4 is that the system $F(x)$ is flatter near the singular solution, since the degree is higher. Because of the flatness, the condition number of the Jacobian matrix of the system $\tilde{F}_{\neg u_n}$ is larger. Then, because of this ill-conditioning, the interval Newton method to verify the unique solutions of $\tilde{F}_{\neg u_n}$ in step 5 of Algorithm 2 and step 5 of Algorithm 3 is less efficient: More iterations can be expected. We tried 10^{-2} first, but the interval Newton method was not able to verify the solutions when the maximum allowed number of iterations was set to be the same as for Examples 2 and 3.

6.2. Test environment. We programmed the algorithms in section 4 in the Fortran 90 environment developed and described in [10, 11]. Similarly, the test functions were programmed using the same Fortran 90 system, which generated internal symbolic representations of the functions. In the actual tests, generic routines then interpreted the internal representations to obtain both floating point and interval values.

The Sun Fortran 95 compiler, version 6.0, was used on a Sparc Ultra-1 model 140 (with a 140 megaHertz clock) with optimization level 0 (that is, with no optimization).

Table 6.1
Numerical results.

		Heuristic		Verified		
Problem	n	degree	Success	degree	CPU time	Time ratio
Example 1	2	2	Yes	2	0.13	-
Example 2	5	2	Yes	2	1.13	-
Example 2	10	2	Yes	2	5.99	5.30
Example 2	20	2	Yes	2	38.40	6.41
Example 2	40	2	Yes	2	273.61	7.13
Example 2	80	2	Yes	2	2198.14	8.03
Example 2	160	2	Yes	2	13033.22	5.93
Example 3	5	3	Yes	3	39.27	-
Example 3	10	3	Yes	3	10.31	0.26
Example 3	20	3	Yes	3	74.32	7.21
Example 3	40	3	Yes	3	481.23	6.48
Example 3	80	3	Yes	3	3805.06	7.91
Example 3	160	3	Yes	3	33944.20	8.92
Example 4	5	4	Yes	4	23.02	-
Example 4	10	4	Yes	4	154.00	6.69
Example 4	20	4	Yes	4	115.55	0.75
Example 4	40	4	Yes	4	3867.51	33.47
Example 4	80	4	Yes	4	6671.20	1.72
Example 4	160	4	-	-	-	-
Example 5	3	5	Yes	5	16.43	-

Execution times were measured with the Port library routine `ETIME`. All times are given in CPU seconds.

6.3. Test results. We present the numerical results in Table 6.1. The column labels of the table are as follows:

Problem: names of the problems identified in section 6.1,

n: number of independent variables,

Heuristic degree: the heuristic value of the degree computed by the heuristic described in section 5,

Success: whether the algorithm was successful,

Verified degree: topological degree verified by the algorithm,

CPU time: CPU time in seconds of the algorithm,

Time ratio: the ratio of two successive CPU times. This column is only meaningful for Examples 2, 3, and 4.

The algorithm, that is, existence verification, succeeded for all problems except Example 4 when $n = 160$. For that problem, we aborted the program after it ran for 36 hours.

We can see from the CPU time ratios that the algorithm is approximately of order $\mathcal{O}(n^3)$ for Examples 2 and 3. However, as we pointed out at the end of section 6.1, when the degree is higher, the system $F(x)$ is flatter near the singular solution. Because of this ill-conditioning, the interval Newton method to verify the unique solutions of $\tilde{F}_{\neg u_n}$ in step 5 of Algorithm 2 and step 5 of Algorithm 3 will be less efficient: More iterations should be expected, and more irregularity in timing could occur. We can see this from the timing results of Example 4. The experimental results are consistent with our expectations.

In certain preliminary experiments, the heuristic failed to compute the correct value of d. The subsequent verification then returned fairly rapidly with "failure to verify" (generally due to failure to verify that there were no solutions to $u_k = 0$ on $\boldsymbol{x}_{\underline{k}}$ or $\boldsymbol{x}_{\overline{k}}$ or to $v_k = 0$ on $\boldsymbol{y}_{\underline{k}}$ or $\boldsymbol{y}_{\overline{k}}$). The heuristic is the weakest part of the verification process.

Although we arranged $\check{x}$ to be exactly the solution x^*, this should not be crucial to the functioning of the algorithm, as long as the box center $\check{x}$ is a sufficiently accurate approximation to an actual root x^* to allow us to choose a box that is large in relationship to this accuracy but small enough to satisfy our other criteria.

Finally, we expect that additional tuning (selection of initial box size, maximum number of inner iterations in the interval Gauss–Seidel method, etc.) could significantly change timing and success for particular problems. The actual times could improve significantly with a more efficient interval arithmetic environment than that of [10, 11], such as direct use of Sun's interval data type in Fortran.

Acknowledgments. We wish to thank both referees for their careful reading and their valuable suggestions to make the exposition more lucid.

REFERENCES

[1] G. Alefeld and J. Herzberger, *Introduction to Interval Computations*, Academic Press, New York, 1983.

[2] P. S. Alexandrov and H. Hopf, *Topologie*, Springer, Berlin, 1935.

[3] J. Cronin, *Fixed Points and Topological Degree in Nonlinear Analysis*, American Mathematical Society, Providence, RI, 1964.

[4] J. Dian and R. B. Kearfott, *Existence verification for singular and non-smooth zeros of real nonlinear systems*, Math. Comp., 72 (2003), pp. 757–766.

[5] C.-Y. Gau, J. F. Brennecke, and M. A. Stadtherr, *Reliable parameter estimation in VLE modeling,* Fluid Phase Equilib., 168 (2000), pp. 1–18.

[6] E. R. Hansen, *Global Optimization Using Interval Analysis*, Marcel Dekker, New York, 1992.

[7] H. Jürgens, H.-O. Peitgen, and D. Saupe, *Topological perturbations in the numerical nonlinear eigenvalue and bifurcation problems*, in Analysis and Computation of Fixed Points, S. M. Robinson, ed., Academic Press, New York, 1980, pp. 139–181.

[8] R. B. Kearfott, *Computing the Degree of Maps and a Generalized Method of Bisection*, Ph.D. thesis, University of Utah, Salt Lake City, UT, 1977.

[9] R. B. Kearfott, *An efficient degree-computation method for a generalized method of bisection*, Numer. Math., 32 (1979), pp. 109–127.

[10] R. B. Kearfott, *A Fortran* 90 *environment for research and prototyping of enclosure algorithms for nonlinear equations and global optimization*, ACM Trans. Math. Software, 21 (1995), pp. 63–78.

[11] R. B. Kearfott, *Rigorous Global Search: Continuous Problems*, Kluwer, Dordrecht, The Netherlands, 1996.

[12] R. B. Kearfott and J. Dian, *Verifying topological indices for higher-order rank deficiencies*, J. Complexity, 18 (2002), pp. 589–611.

[13] R. B. Kearfott, J. Dian, and A. Neumaier, *Existence verification for singular zeros of nonlinear systems*, Technical report, University of Louisiana at Lafayette, Lafayette, LA, 1999; available online at http://interval.louisiana.edu/preprints/singular_existence.ps.

[14] R. B. Kearfott, J. Dian, and A. Neumaier, *Existence verification for singular zeros of complex nonlinear systems*, SIAM. J. Numer. Anal., 38 (2000), pp. 360–379.

[15] C. F. Korn and Ch. Ullrich, *Extending LINPACK by verification routines for linear systems*, Math. Comput. Simulation, 39 (1995), pp. 21–37.

[16] G. Mayer, *Epsilon-inflation in verification algorithms*, J. Comput. Appl. Math., 60 (1994), pp. 147–169.

[17] A. Neumaier, *Interval Methods for Systems of Equations*, Cambridge University Press, Cambridge, UK, 1990.

[18] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.

[19] H. Ratschek and J. Rokne, *New Computer Methods for Global Optimization*, Wiley, New York, 1988.

[20] F. Stenger, *Computing the topological degree of a mapping in* $\mathbb{R}^n$, Numer. Math., 25 (1975), pp. 23–38.