A

Algorithm XXX: An algorithm and software for computing multiplicity structures at zeros of nonlinear systems

Wenrui Hao, University of Notre Dame
Andrew J. Sommese, University of Notre Dame
Zhonggang Zeng, Northeastern Illinois University

A Matlab implementation, MULTIPLICITY, of a numerical algorithm for computing the multiplicity structure of a nonlinear system at an isolated zero is presented. The software incorporates a newly developed equation-by-equation strategy that significantly improves the efficiency of the closedness subspace algorithm and substantially reduces the storage requirement. The equation-by-equation strategy is actually based on a variable-by-variable closedness subspace approach. As a result, the algorithm and software can handle much larger nonlinear systems and higher multiplicities than their predecessors, as shown in computational experiments on the included test suite of benchmark problems.

Categories and Subject Descriptors: G.1.5 [**Mathematics of Computing**]: Roots of Nonlinear Equations - Polynomials, methods for Systems of equations; G.4 [**Mathematical Software**]: Algorithm design and analysis, Certification and testing

Additional Key Words and Phrases: polynomial, multiple roots, multiplicity.

ACM Reference Format:
W. Hao, A. Sommese, Z. Zeng, An algorithm and software for computing multiplicity structures at zeros of nonlinear systems ACM Trans. Embedd. Comput. Syst. V, N, Article A (January YYYY), 16 pages.
DOI = 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

1. INTRODUCTION

Solving nonlinear systems of equations in the form of

$$\begin{cases} f_1(x_1,\ldots,x_s) &= 0 \\ \quad\vdots \qquad\qquad & \vdots\ \vdots \\ f_t(x_1,\ldots,x_s) &= 0 \end{cases} \quad \text{with} \quad t \geq s \tag{1}$$

is one of the most basic problems in applied mathematics. It is well known that a solution of the system (1) is multiple or singular when the Jacobian is rank-deficient at the solution. In other words, this particular solution is a converging point of a few, a few dozens or even hundreds of solutions. Finding the characteristics of the solution such as multiplicity is a natural component of nonlinear system solving. However, the basic concept of multiplicity of the solution and its identification are largely unknown to numerical analysts and scientific computing practitioners. None of the numerical

W. Hao and A. Sommese are supported by the Duncan Chair of the University of Notre Dame. Z. Zeng is supported in part by NSF under Grant DMS-0715127.
Author's addresses: W. Hao and A. Sommese, Department of Applied and Computational Mathematics and Statistics, University of Notre Dame, Notre Dame, IN 46556; Z. Zeng Department of Mathematics, Northeastern Illinois University, Chicago, IL 60625.

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

analysis textbooks in all levels we have seen elaborates the multiplicity beyond the single univariate equations.

On the other hand, the study of multiplicity for systems of polynomial equations is one of the fundamental subjects of algebraic geometry with a long history dating back to Newton's era, see [Dayton and Zeng 2005; Fulton 1984; Macaulay 1916; Morgan 1987 & 2009; Stetter 2004; Zeng 2009] and the references provided in them. There have been rigorous theoretical formulations of multiplicity over the years. Using the Gröbner basis, the standard method for identifying the intersection multiplicity relies on symbolic computation and requires exact solutions that are often unattainable due to Abel's Impossibility Theorem.

In this paper, we introduce a numerical algorithm and its Matlab implementation MULTIPLICITY that calculates the multiplicity structure of a given nonlinear system at an isolated solution with approximate values. For numerical analysts, the software provides a convenient tool for analyzing singularity and bifurcation whenever a nonlinear system needs to be solved and multiple solutions may exist. For the field computational algebraic geometry, this software produces structural invariants such as dual spaces and Hilbert functions that can not be computed by symbolic algorithms for polynomial ideals due to inexact numerical values of computed zeros.

Nonlinear systems of equations especially polynomial systems arise in many applications: robotics, computer vision, kinematics, chemical kinetics, truss design, geometric modeling, and many others (see [Morgan 1987 & 2009; Verschelde 1996]). Multiple zeros possess a rich spectrum of structural invariants besides their locations and multiplicities, such as the Hilbert function, dual space, depth, breadth [Dayton and Zeng 2005], and regularity [Bates et al. 2006].

Computational algorithms have been developed for identifying the multiplicity. The most efficient one in these algorithms is identifying the dual spaces through computing the kernels of the Macaulay matrices [Dayton and Zeng 2005; Macaulay 1916]. The resulting algorithm has been developed in [Dayton and Zeng 2005] and implemented in the software package *ApaTools* [Zeng 2008] and *Bertini* [Bates et al. 2006]. However, memory issues may rise to prohibitive levels when the multiplicity is high or the number of variables is large. Even if the problem is within the memory capacity, the matrix size is still the main bottleneck that slows down the overall computation. The closedness subspace method [Zeng 2009] improves the efficiency substantially in both storage space and execution time in computing the multiplicity structure. However, the matrix sizes used for computing closedness subspaces still exceed the limit of memory storage when the number of variables increases. In this paper, we introduce a newly developed equation-by-equation strategy for computing closedness subspaces with reduced matrix sizes throughout computation. As a result, the numerical results of the revised algorithm show a remarkable improvement in handling larger systems and higher multiplicities.

In the following sections we shall give an elementary introduction to multiplicity and multiplicity structures, an overview of multiplicity algorithms, details of the algorithm including the incorporated new strategy, an introduction to the software, a presentation of computational experiments and benchmark problems.

We would like to thank the referees of this article for their helpful comments.

2. THE NOTION OF MULTIPLICITY

For a general nonlinear system of equations as in (1) a zero $\hat{\mathbf{x}} = (\hat{x}_1, \ldots, \hat{x}_s)$ of $F = \{f_1, \ldots, f_t\}$ is known to be a *multiple solution* of (1) if the Jacobian is rank-deficient. The question is: What is the multiplicity?

Absent in all the numerical analysis textbooks we have seen, the basic concept of multiplicity for a solution to a multivariate nonlinear system is largely unknown to

scientific computing practitioners. On the other hand, the multiplicity of a polynomial system has been rigorously formulated and recognized since Newton's era as one of the fundamental concepts of algebraic geometry. This gap between numerical analysis and algebraic geometry is bridged recently in [Dayton et al. 2011] where the Macaulay formulation of multiplicity is elaborated and extended along with an implementable numerical algorithm. While standard intersection multiplicity is quite abstract and algebraically demanding, the equivalent Macaulay formulation can be easily explained as a generalization of univariate cases.

A well-known notion of the multiplicity for a solution x_* to a single univariate equation $f(x) = 0$ is the count of consecutive vanishing derivatives at x_*: The multiplicity is m if

$$f(x_*) = f'(x_*) = \cdots = f^{(m-1)}(x_*) = 0 \quad \text{and} \quad f^{(m)}(x_*) \neq 0.$$

Needless to mention in this definition, these vanishing derivatives satisfy linear independence and a crucial closedness condition: A vanishing derivative $f^{(k)}(x_*) = 0$ is counted toward multiplicity if and only if all the reduced derivatives $f^{(k-1)}$, $f^{(k-2)}$, ... also vanish at x_*.

To understand the multiplicity for multivariate nonlinear systems, consider the nonlinear system $F = \{f_1, f_2\}$ as an example where

$$\begin{cases} f_1(x_1, x_2) &= \sin x_1 - x_1 \\ f_2(x_1, x_2) &= \sin^2 x_1 - x_2^3 \end{cases}. \tag{2}$$

It is obvious that f_1, f_2 and some of the derivatives vanish at $(0, 0)$, for example

$$\left.\frac{\partial^2 f_i}{\partial x_1 \partial x_2}\right|_{(x_1,x_2)=(0,0)} = 0 \quad \text{for} \quad i = 1, 2.$$

Actually, the origin $(0, 0)$ is a zero of the system with multiplicity 9 since there are 9 linearly independent vanishing derivatives of both $f_1(x_1, x_2)$ and $f_2(x_1, x_2)$ at $(x_1, x_2) = (0, 0)$:

$$\partial_{00},\ \partial_{10},\ \partial_{01},\ \partial_{11},\ \partial_{02},\ \partial_{20} - \partial_{03},\ \partial_{12},\ \partial_{04} - \partial_{21},\ \partial_{05} - \partial_{22}. \tag{3}$$

Here $\partial_{ij} = \partial_{x_1^i x_2^j} = \frac{1}{i!j!}\frac{\partial^{i+j}}{\partial x_1^i \partial x_2^j}$. For instance, $\partial_{04} - \partial_{21}$ is a vanishing derivative for $F = \{f_1, f_2\}$ at $(0, 0)$ since

$$\big(\partial_{04} - \partial_{21}\big) f_j(0, 0) \ \equiv \ \left[\frac{1}{4!}\frac{\partial^4}{\partial x_2^4} f_j(x_1, x_2) - \frac{1}{2!}\frac{\partial^3}{\partial x_1^2 \partial x_2} f_j(x_1, x_2)\right]_{(x_1,x_2)=(0,0)} = 0$$

for $j = 1, 2$. Furthermore, these derivatives satisfy the closedness condition: any reduced derivative from the list (3) also vanishes for $F = \{f_1, f_2\}$ at $(0, 0)$. For example, reducing the x_2 partial derivative of $\partial_{04} - \partial_{21}$ yields $\partial_{03} - \partial_{20}$, which also vanishes for $F = \{f_1, f_2\}$ since $-(\partial_{03} - \partial_{20})$ is in (3) and so does reducing its x_1 partial derivative resulting in $0 - \partial_{11}$ in the list.

More precisely, the closedness condition for a vanishing derivative is defined using the *linear anti-differentiation operator*

$$\Phi_1(\partial_{ij}) = \begin{cases} 0 & \text{if } i = 0, \\ \partial_{i-1,j} & \text{otherwise} \end{cases} \quad \text{and } \Phi_2(\partial_{ij}) = \begin{cases} 0 & \text{if } j = 0, \\ \partial_{i,j-1} & \text{otherwise}. \end{cases}$$

We say $\partial_{04} - \partial_{21}$ satisfies the closedness condition because $F = \{f_1, f_2\}$ has the reduced derivatives

$$\Phi_1(\partial_{04} - \partial_{21}) = -\partial_{11}, \ \ \Phi_2(\partial_{04} - \partial_{21}) = \partial_{03} - \partial_{20},$$

Table I: Clusters of 9 zeros of the perturbed system (F_ϵ) in (4) near the 9-fold zero $\hat{\mathbf{x}} = (0, 0)$ corresponding to different ϵ values.

$\epsilon = 10^{-8}$	
(0.001957432±0.003390374i,	0.023345740∓0.008492212i)
(0.001957432±0.003390374i,	-0.019027341∓0.015971898i)
(0.001957432±0.003390374i,	-0.004318400±0.024464111i)
(-0.003914871,	0.012417083±0.021507018i)
(-0.003914869,	-0.024834165)
$\epsilon = 10^{-16}$	
(0.000004217163±0.000007304341i,	0.00038936143∓0.00014171579i)
(0.000004217163±0.000007304341i,	-0.00031741019∓0.00026633899i)
(0.000004217163±0.000007304341i,	-0.00007195124±0.00040805478i)
(-0.000008434327,	0.00020717471±0.00035883713i)
(-0.000008434327,	-0.00041434943)

and all the subsequently reduced derivatives, vanishing at $(0, 0)$.

The concept of multiplicity is far beyond a single positive integer. For the nonlinear system (2) and its zero $(0, 0)$, each differential operator ∂_{ij} induces a monomial differential functional at the zero $(0, 0)$

$$\partial_{ij}[(0,0)] \ : \ g \longrightarrow \frac{1}{i!j!}\frac{\partial^{i+j} g(x_1, x_2)}{\partial x_1^i \partial x_2^j}\bigg|_{(x_1,x_2)=(0,0)}$$

on differentiable bivariate function $g(x_1, x_2)$. Linear combinations

$$\sum c_{ij}\, \partial_{ij}[(0, 0)]$$

vanishing on both f_1 and f_2 while satisfying the closedness condition form a vector space denoted by $\mathcal{D}_{(0,0)}(F)$. In this example

$$\begin{aligned}\mathcal{D}_{(0,0)}(F) \ = \ \mathrm{span}\{&\partial_{00}[(0,0)],\ \partial_{10}[(0,0)],\ \partial_{01}[(0,0)],\ \ldots,\\ &\partial_{04}[(0,0)] - \partial_{21}[(0,0)],\ \partial_{05}[(0,0)] - \partial_{22}[(0,0)]\},\end{aligned}$$

and this spanning set of differential functionals forms a basis for the vector space $\mathcal{D}_{(0,0)}(F)$, whose dimension is thus the multiplicity 9 of the zero $(0, 0)$ to $F = \{f_1, f_2\}$. Furthermore, the multiplicity 9 is partitioned in a so-called Hilbert function $H = \{1, 2, 2, 2, 1, 1\}$, with breadth 2 and depth 5.

The term "multiplicity" intuitively means the number of solutions that coalesce into the single (multiple) solution. This is only correct when we have a system of n equations in n variables occurring in a sufficiently general family of polynomial systems. In the case of a system of n equations in n variables occurring as a member of a sufficiently general family of systems, the multiple solution becomes a cluster of simple solutions when the system is perturbed, and the number of solutions in the cluster equals to the multiplicity if the perturbation is tiny.

As shown in Table I, exactly 9 solutions emanate from the multiplicity 9 solution $(0, 0)$ of the system

$$F_\epsilon \ = \ \{f_1 - \epsilon, f_2 - \epsilon\} \tag{4}$$

perturbed from (2).

Detailed and rigorous definition of multiplicity and multiplicity structure will be given in Section 3.

3. FORMAL DEFINITIONS

The general nonlinear system (1) can be represented as a set $F = \{f_1, \cdots, f_t\}$ of smooth functions in variables $x_1, \cdots, x_s$. For an integer array $\mathbf{j} = (j_1, \ldots, j_s) \in \mathbb{N}^s$, where $\mathbb{N}$ is the set of nonnegative integers, denote the differential monomial functional $\partial_{\mathbf{j}}[\hat{\mathbf{x}}]$ on differentiable functions at a given point $\hat{\mathbf{x}}$ with order $|\mathbf{j}| = j_1 + \cdots + j_s$ as

$$\partial_{\mathbf{j}}[\hat{\mathbf{x}}] : f \longrightarrow \frac{1}{j_1! \cdots j_s!} \frac{\partial^{j_1+\cdots+j_s}}{\partial_{x_1^{j_1}} \cdots \partial_{x_s^{j_s}}} f(\hat{\mathbf{x}}).$$

A *differential functional* is a linear combination of the monomial functionals in the form of

$$c \;=\; c_1 \partial_{\mathbf{j_1}}[\hat{\mathbf{x}}] + \cdots + c_k \partial_{\mathbf{j_k}}[\hat{\mathbf{x}}].$$

When a functional c is applied to the system F, it generates a set of numbers $c(F) = \{c(f_1), \cdots, c(f_t)\}$. For each variable x_i, the *linear anti-differentiation transformation* [definition 8.21, pp. 330][Stetter 2004], Φ_i is defined by

$$\Phi_i\Big(\sum_{\mathbf{j}} c_{\mathbf{j}} \partial_{\mathbf{j}}[\hat{\mathbf{x}}]\Big) \;=\; \sum_{\mathbf{j}} c_{\mathbf{j}} \Phi_i(\partial_{\mathbf{j}}[\hat{\mathbf{x}}]),$$

with

$$\Phi_i(\partial_{j_1 \cdots j_s}[\hat{\mathbf{x}}]) = \begin{cases} 0, & \text{if } j_i = 0 \\ \partial_{j_1,\ldots,j_{i-1},j_i-1,j_{i+1},\ldots,j_s}[\hat{\mathbf{x}}], & \text{if } j_i > 0 \end{cases}$$

for $i = 1, \cdots, s$. The α-th order *dual subspace* of the system F at $\hat{\mathbf{x}}$ consists of differential functionals that vanish on F and satisfy the closedness condition:

$$\mathcal{D}^{\alpha}_{\hat{\mathbf{x}}}(F) = \left\{ c = \sum_{\mathbf{j} \in \mathbb{N}^s, |\mathbf{j}| \le \alpha} c_{\mathbf{j}} \partial_{\mathbf{j}}[\hat{\mathbf{x}}] \,\middle|\, c(F) = \{0\}, \Phi_i(c) \in \mathcal{D}^{\alpha-1}_{\hat{\mathbf{x}}}(F), \forall i = 1, \cdots, s \right\}, \tag{5}$$

where $\alpha \ge 1$ and $\mathcal{D}^0_{\hat{\mathbf{x}}}(F) = \text{span}\{\partial_{0\cdots0}\}$. Moreover, $\mathcal{D}^{\alpha}_{\hat{\mathbf{x}}}(F) \subset \mathcal{D}^{\alpha+1}_{\hat{\mathbf{x}}}(F)$. With these notations, the multiplicity at a zero $\hat{\mathbf{x}}$ can be defined as follows.

Definition 3.1. *[Dayton et al. 2011]* Let $\hat{\mathbf{x}}$ be a zero of the nonlinear system $F = \{f_1, \cdots, f_t\}$ of functions with derivatives of order $\gamma \ge 1$. If $\mathcal{D}^{\gamma}_{\hat{\mathbf{x}}}(F) = \mathcal{D}^{\gamma-1}_{\hat{\mathbf{x}}}(F)$, then the vector space

$$\mathcal{D}_{\hat{\mathbf{x}}}(F) \;=\; \mathcal{D}^0_{\hat{\mathbf{x}}}(F) \cup \mathcal{D}^1_{\hat{\mathbf{x}}}(F) \cup \cdots \cup \mathcal{D}^{\gamma-1}_{\hat{\mathbf{x}}}(F) = \mathcal{D}^{\gamma}_{\hat{\mathbf{x}}}(F)$$

is called the **dual space** of the system F at $\hat{\mathbf{x}}$. The dimension of $\mathcal{D}_{\hat{\mathbf{x}}}(F)$ is called the **multiplicity** of F at $\hat{\mathbf{x}}$.

Notice that the dual subspaces $\mathcal{D}^{\alpha}_{\hat{\mathbf{x}}}(F)$ strictly enlarge as α increases until reaching a certain number δ for which $\mathcal{D}^{\delta}_{\hat{\mathbf{x}}}(F) = \mathcal{D}^{\delta+1}_{\hat{\mathbf{x}}}(F)$. Thus all functionals in $\mathcal{D}^{\delta+1}_{\hat{\mathbf{x}}}(F)$ are of differential orders up to δ. Moreover, the crucial *closedness condition*, namely,

$$\Phi_i\big(\mathcal{D}^{\alpha+1}_{\hat{\mathbf{x}}}(F)\big) \subset \mathcal{D}^{\alpha}_{\hat{\mathbf{x}}}(F) \;\text{ for } i = 1, \cdots, s, \tag{6}$$

guarantees that there are no functionals in the dual subspace with a differential order $\alpha > \delta + 1$. Thus

$$\mathcal{D}^0_{\hat{\mathbf{x}}}(F) \subset \mathcal{D}^1_{\hat{\mathbf{x}}}(F) \subset \cdots \subset \mathcal{D}^{\delta}_{\hat{\mathbf{x}}}(F) = \mathcal{D}^{\delta+1}_{\hat{\mathbf{x}}}(F) = \cdots = \mathcal{D}^{\gamma}_{\hat{\mathbf{x}}}(F).$$

Here, the integer δ, called the depth, is the highest order of differential functionals in the dual space.

The type of singularity of a multiple zero consists of more than a single integer multiplicity. There is a rich spectrum of algebraic invariants that are essential in the

characterization of the nonlinear system. This *multiplicity structure* includes the dual space, the Hilbert function, the depth and breath. See [Dayton et al. 2011] for more detailed and in-depth discussions. The Hilbert function of the dual space is not the Hilbert function computed by symbolic programs using the graded monomial ordering. See [Griffin et al. 2011, §1].

Remark: In algebraic geometry, the multiplicity structure is not only defined for square systems (n equations with n unknowns), but is also defined for general systems. Intuitively, we would expect that taking a multiplicity μ isolated solution $\hat{\mathbf{x}}$ of a polynomial system on $\mathbb{C}^n$ with the induced scheme structure, we could put it in a flat family of schemes so that nearby schemes consisted of μ nonsingular points near to $\hat{\mathbf{x}}$ on $\mathbb{C}^n$. Though this is true trivially if $n = 1$, the proof is not trivial when $n = 2$, and the statement is in fact false in general for $n \geq 3$: See [Göttsche 1994] for a detailed discussion of zero dimensional scheme. These considerations are closely connected with Kahan's pejorative manifolds [Kahan 1972] (see [Zeng 2003] for a discussion). Indeed, in one variable, the family of monic degree d polynomials having exactly k roots with prescribed multiplicities $d_1, \ldots, d_k$ and $d = d_1 + \cdots + d_k$ is a flat family with parameter space, a classical pejorative manifold of Kahan. In algebraic geometry, the stratification of a family into flat subfamilies generalizes pejorative manifolds, e.g., see [Morgan and Sommese 1989]. The equisingular decomposition of [Hauenstein and Wampler 2010] is a step in the direction of numerically computing the analogue of pejorative manifolds related to the operation of deflation.

4. PROPERTIES AND APPLICATIONS OF MULTIPLICITY AND MULTIPLICITY STRUCTURE

There are some basic properties of the multiplicity and multiplicity structure of zeros of nonlinear systems, such as: [Dayton et al. 2011]

1. *Local Finiteness*: The zero is isolated if and only if the multiplicity is finite. This not only ensures that it takes finitely many steps for computing the multiplicity at an isolated zero, but also provides a mechanism for identifying nonisolated zeros of nonlinear systems [Bates et al. 2009].
2. *Consistence with intersection multiplicity*: The multiplicity formulated in Definition 1 is identical to the intersection multiplicity of a solution of a polynomial system in algebraic geometry.
3. *Perturbation invariance*: In the special case of an isolated solution of a system of n polynomials in n variables, the number of zeros, counting multiplicities, is locally invariant under a small perturbation to the nonlinear system. More precisely, giving a small perturbation within the family to the nonlinear system, there is a cluster of exactly m zeros (counting multiplicities) in a neighborhood of an m-fold zero.
4. *Depth deflation*: Multiple zeros can be accurately computed by a deflation method [Dayton et al. 2011; Leykin et al. 2006] using floating-point arithmetic even if the system is perturbed, with the number of deflation steps bounded by the depth of the zero.

The multiplicity structure identification is indispensable in numerical algebraic geometry. The previous version of the Macaulay algorithm (c.f. [Dayton et al. 2011; Dayton and Zeng 2005] and §5) has been effectively applied in a numerical local dimensional test [Bates et al. 2009] for positive dimensional solutions of polynomial systems. The solutions of a polynomial system may consist of nontrivial components, e.g., curves, surfaces, etc., each with its own dimension. One of the fundamental problems in numerical algebraic geometry is to characterize the solution sets known as algebraic sets or varieties. A novel algorithm in [Bates et al. 2009] takes advantage of the local finiteness property of multiplicity and becomes a robust tool for identifying dimensions of solution sets of polynomial systems, based on the simple fact: A k-dimensional va-

riety in $\mathbb{C}^N$ intersects a generic hyperplane of dimension $N-k$ at isolated points that are zeros of finite multiplicities to the polynomial system augmented with k linear equations.

Moreover, it is shown in [Hauenstein 2010] that many basic algebraic operations performed on homogeneous polynomial ideals can be carried out as operations performed on the dual spaces, such as the ideal membership problem. A polynomial f is in a polynomial ideal if and only if all the differential functionals forming bases for the dual spaces vanish at f. This property provides a numerical method for testing ideal membership. The paper [Hauenstein 2010] also introduces the elimination operator on the dual basis in order to show the relationship between the vector spaces consisting of the homogeneous polynomials with different degrees. The algorithms of these operations depend upon the efficient computation of the dual basis.

One of the areas of scientific computing where multiple zeros inevitably occur is the theory and computation of bifurcation points. Consider the steady state system of a dynamical problem

$$F(\lambda, u) \;=\; 0,$$

where λ is a parameter and u is a steady state solution dependent on λ. When the parameter reaches a bifurcation point λ_0, the system $F(\lambda_0, u)$ has a singular solution u_0 whose multiplicity represents the number of solutions in u converging to u_0 as λ reaches λ_0. Conversely, a small change from λ_0 to $\lambda_0 + \epsilon$ can be considered a small perturbation

$$F(\lambda_0 + \epsilon, u) \;=\; F(\lambda_0, u) + \epsilon G(\lambda_0, u),$$

from the system $F(\lambda_0, u)$, and the perturbation invariance property of multiplicity ensures that the number of solution branches emanating from the bifurcation point (λ_0, u_0) is identical to the multiplicity.

There are also studies of relations between multiplicity and bifurcation in ordinary differential equations (see, e.g. [Ward 2005; Ward 2007]) and partial differential equations [Duo et al. 2008]. We shall present a computational experiment on the bifurcation of a system of ordinary differential equations in Section 7.

It is the objective of this paper and the accompanying software to provide a computational tool for the study of multiplicity and multiplicity structure in scientific computing and numerical algebraic geometry.

5. AN OVERVIEW OF MULTIPLICITY ALGORITHMS

A differential functional $c = \sum_{|\mathbf{j}|\le\alpha} c_{\mathbf{j}} \partial_{\mathbf{j}}[\hat{\mathbf{x}}]$ belongs to the dual subspace $\mathcal{D}^{\alpha}_{\hat{\mathbf{x}}}(F)$ if and only if the coefficients $c_{\mathbf{j}}$ satisfy the system of equations ([Dayton et al. 2011; Dayton and Zeng 2005; Macaulay 1916])

$$\sum_{|\mathbf{j}|\le\alpha} c_{\mathbf{j}} \cdot \partial_{\mathbf{j}}[\hat{\mathbf{x}}] \left((\mathbf{x}-\hat{\mathbf{x}})^{\mathbf{k}} f_i(\mathbf{x}) \right) \;=\; 0, \qquad (7)$$
$$\text{for } \mathbf{k} = (k_1, \cdots, k_s) \in \mathbb{N}^s, |\mathbf{k}| < \alpha,$$

where $(\mathbf{x}-\hat{\mathbf{x}})^{\mathbf{k}} = (x_1 - \hat{x}_1)^{k_1} \cdots (x_s - \hat{x}_s)^{k_s}, 1 \le i \le s$. Then (7) can be written as a homogeneous linear system $S_\alpha \mathbf{c} = \mathbf{0}$, where the vector $\mathbf{c} \in \mathbb{C}^{n_\alpha}$ consists of the coefficients $c_{\mathbf{j}}, |\mathbf{j}| \le \alpha$. This simple observation leads to the Macaulay Algorithm for computing the multiplicity structure:

The Macaulay Algorithm has been implemented in the package *ApaTools* [Zeng 2008] (*MultiplicityStructure* module) and *Bertini* [Bates et al. 2006]. However, the Macaulay matrix S_α, which is of size $m_\alpha \times n_\alpha$ with $m_\alpha = \frac{(\alpha-1+s)!}{s!(\alpha-1)!}$ and $n_\alpha = \frac{(\alpha+s)!}{s!\alpha!}$, can be huge. For example, the KSS system, in Section 7.1, generates a Macaulay ma-

ALGORITHM 1: Macaulay Algorithm

Input: The nonlinear system F and zero $\hat{\mathbf{x}}$
Output: Multiplicity m = nullity(S_α), basis for the dual space $\mathcal{D}_{\hat{\mathbf{x}}}(F)$
Initialize $S_0 = \mathbf{0}_{t\times 1}$
for $\alpha = 1, 2, \cdots$ **do**
  Construct the Macaulay matrix S_α
  Obtain $\mathcal{D}^\alpha_{\hat{\mathbf{x}}}(F)$ by computing the kernel of S_α
  if *nullity*(S_α) = *nullity*($S_{\alpha-1}$) **then**
    Set $\mathcal{D}_{\hat{\mathbf{x}}}(F) = \mathcal{D}^\alpha_{\hat{\mathbf{x}}}(F)$, exit
  end
end

ALGORITHM 2: Closedness Subspace Algorithm

Input: The nonlinear system F and zero
Output: Multiplicity m = nullity(W_α), basis for the dual space $\mathcal{D}_{\hat{\mathbf{x}}}(F)$
Initialize $\mathcal{D}^0_{\hat{\mathbf{x}}}(F) = \mathcal{C}^0_{\hat{\mathbf{x}}}(F) = \text{span}\{\partial_{0\cdots 0}\}$, and $W_0 = \mathbf{0}_{t\times 1}$
for $\alpha = 1, 2, \cdots$ **do**
  Compute a basis for the closedness subspace $\mathcal{C}^\alpha_{\hat{\mathbf{x}}}(F)$
  Construct the matrix W_α of the linear system (8)
  Obtain $\mathcal{D}^\alpha_{\hat{\mathbf{x}}}(F)$ by computing the kernel of W_α
  if *nullity*(W_α) = *nullity*($W_{\alpha-1}$) **then**
    Set $\mathcal{D}_{\hat{\mathbf{x}}}(F) = \mathcal{D}^\alpha_{\hat{\mathbf{x}}}(F)$, exit
  end
end

trix of the size $218,790 \times 48,620$ for $n = 9$. The large sizes of Macaulay matrices slow down the computation even if the storage is within the memory capacity.

The closedness subspace method developed in [Zeng 2009] improves the efficiency substantially in both storage space and execution time. The strategy in [Zeng 2009] comes from an observation that the closedness condition (6) defines a *closedness subspace* $\mathcal{C}^\alpha_{\hat{\mathbf{x}}}(F)$ of low dimension. Let $\{\phi_1, ..., \phi_m\}$ be a basis for this subspace $\mathcal{C}^\alpha_{\hat{\mathbf{x}}}(F)$, every functional c in the dual subspace $\mathcal{D}^\alpha_{\hat{\mathbf{x}}}(F)$ is a linear combination

$$c = u_1\phi_1 + ... + u_m\phi_m$$

satisfying

$$\sum_{j=1}^{m} u_j\phi_j(f_i) = 0, \quad 1 \le i \le t \tag{8}$$

which is equivalent to a homogeneous linear system $W_\alpha u = 0$. The matrix W_α is of small size $t \times m$ due to the low dimension of the closedness subspace. The closedness subspace algorithm can be described in the following pseudo-code.

In this algorithm, matrix sizes are substantially reduced in kernel computation. For example, table II shows that the largest matrix W_α in the KSS system for $n = 9$ is of only 9×265, which is much smaller compared to the $11,440 \times 24,310$ Macaulay matrix. Instead of constructing the Macaulay matrix, the closedness subspace method comes down to construction of the closedness subspace. The equation-by-equation method that will be presented in Section 6 is efficient in computing the closedness subspace by generating smaller matrices at each step and computing their kernels.

Table II: Matrix size comparison

α	size of S_α	size of W_α
1	1×10	9×10
2	10×55	9×46
3	55×220	9×102
4	220×715	9×172
5	$715 \times 2,002$	9×228
6	$2,002 \times 5,005$	9×256
7	$5,005 \times 11,440$	9×264
8	$11,440 \times 24,310$	9×265

Example

Consider the multiplicity of polynomial system

$$f_1(x,y) \;=\; y - x^2,\; f_2(x,y) \;=\; y^2$$

at the zero $(x,y) \;=\; (0,0)$. The dual subspaces are

$$\mathcal{D}^0_{\mathbf{0}} \;=\; \mathbf{span}\{\partial_{00}\},\; \mathcal{D}^1_{\mathbf{0}} \;=\; \mathcal{D}^0_{\mathbf{0}} \;\oplus\; \mathbf{span}\{\partial_{10}\},\; \mathcal{D}^2_{\mathbf{0}} \;=\; \mathcal{D}^1_{\mathbf{0}} \;\oplus\; \mathbf{span}\{\partial_{01} + \partial_{20}\},$$
$$\mathcal{D}_{\mathbf{0}} \;=\; \mathcal{D}^3_{\mathbf{0}} \;=\; \mathcal{D}^3_{\mathbf{0}} \;=\; \mathcal{D}^2_{\mathbf{0}} \;\oplus\; \mathbf{span}\{\partial_{11} + \partial_{30}\}.$$

Figure 2 shows the expansion of Macaulay matrices and the corresponding kernels while Figure 1 shows that of matrices W_α.

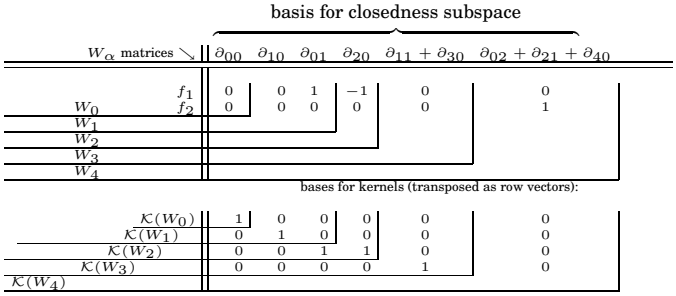

Fig. 1: The W_α matrices for Example.

6. THE CLOSEDNESS SUBSPACE ALGORITHM: DETAILS AND FURTHER DEVELOPMENT

For simplicity of exposition, we omit $[\hat{\mathbf{x}}]$ in the notation $\partial_{\mathbf{j}}[\hat{\mathbf{x}}]$ in this section. For the purpose of computing the closedness subspace $\mathcal{D}^\alpha_{\hat{\mathbf{x}}}(F)$, let

$$S^\alpha_{\hat{\mathbf{x}}}(F) \;=\; \left\{ \partial_{\mathbf{j}} \middle| \exists c = \sum_{|\mathbf{i}| \le \alpha} c_{\mathbf{i}} \partial_{\mathbf{i}} \in \mathcal{D}^\alpha_{\hat{\mathbf{x}}}(F) \text{ such that } c_{\mathbf{j}} \neq 0 \right\},$$

denote the α-th order *dual support* and

$$M^\alpha_{\hat{\mathbf{x}}}(F) \;=\; \left\{ \partial_{\mathbf{j}} \middle| \Phi_\delta(\partial_{\mathbf{j}}) \in S^{\alpha-1}_{\hat{\mathbf{x}}}(F) \text{ for } \delta = 1, \cdots, s \right\} \;\cup\; \left\{ \partial_{0\cdots 0} \right\}$$

be the α-th order *closedness support*. Here we have the following relationship

$$\mathcal{D}^\alpha_{\hat{\mathbf{x}}}(F) \;\subset\; \mathcal{C}^\alpha_{\hat{\mathbf{x}}}(F) \;\subset\; \mathrm{span}\{M^\alpha_{\hat{\mathbf{x}}}(F)\}.$$

Macaulay matrices ↘		column index: ∂_{00}	∂_{10}	∂_{01}	∂_{20}	∂_{11}	∂_{02}	∂_{30}	∂_{21}	∂_{12}	∂_{03}	∂_{04}	∂_{13}	∂_{22}	∂_{31}	∂_{40}
	f_1	0	0	1	−1	0	0	0	0	0	0	0	0	0	0	0
S_0	f_2	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
S_1																
	xf_1	0	0	0	0	1	0	−1	0	0	0	0	0	0	0	0
	xf_2	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
	yf_1	0	0	0	0	0	1	0	−1	0	0	0	0	0	0	0
S_2	yf_2	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
	x^2f_1	0	0	0	0	0	0	0	1	0	0	−1	0	0	0	0
	x^2f_2	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
	xyf_1	0	0	0	0	0	0	0	0	1	0	0	−1	0	0	0
	xyf_2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
	y^2f_1	0	0	0	0	0	0	0	0	0	1	0	0	−1	0	0
S_3	y^2f_2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
	x^3f_1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
	x^3f_2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	x^2yf_1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
	x^2yf_2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	xy^2f_1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
	xy^2f_2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	y^3f_1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
S_4	y^3f_2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
bases for kernels (transposed as row vectors):																
	$\mathcal{K}(S_0)$	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	$\mathcal{K}(S_1)$	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	$\mathcal{K}(S_2)$	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
	$\mathcal{K}(S_3)$	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0
	$\mathcal{K}(S_4)$															

Fig. 2: The Macaulay matrices for Example.

Using these notations, we give a sketch of the closedness subspace method for computing the multiplicity structure of the nonlinear system F at $\hat{\mathbf{x}}$:

1 **Initialize** $\alpha = 0$, W_0 as a $t \times 1$ zero matrix, the dual subspace $\mathcal{D}^0_{\hat{\mathbf{x}}}(F) = \text{span}\{\partial_{0\cdots0}\}$, the dual support $S^0_{\hat{\mathbf{x}}}(F) = \{\partial_{0\cdots0}\}$ and the closedness support $M^1_{\hat{\mathbf{x}}}(F) = \{\partial_{0\cdots0}\} \cup \{\partial_{1\cdots0}, \cdots, \partial_{0\cdots1}\}$.

2 **Compute the closedness subspace**: Denote $M^{\alpha+1}_{\hat{\mathbf{x}}}(F) = \{\partial_{0\cdots0}\} \cup \{\partial_{\mathbf{j}_1}, \cdots, \partial_{\mathbf{j}_m}\}$ and $\mathcal{D}^{\alpha}_{\hat{\mathbf{x}}}(F) = \text{span}\{c_1, \cdots, c_n\}$. Since the closedness subspace $\mathcal{C}^{\alpha+1}_{\hat{\mathbf{x}}}(F)$ is a subspace of $\text{span}\{M^{\alpha+1}_{\hat{\mathbf{x}}}(F)\}$, the closedness condition yields

$$\Phi_\sigma\left(\sum_{i=1}^{m} z_i \partial_{\mathbf{j}_i}\right) = \sum_{j=1}^{n} y_{\sigma_j} c_j, \text{ for } \sigma = 1, \cdots, s, \text{ where } z_i, y_{\sigma j} \in \mathbb{C}. \tag{9}$$

Matching the coefficients of $\partial_{\mathbf{j}_i}$ in (9) leads to linear equations in matrix-vector form

$$\mathbf{z} = A\mathbf{y} \text{ and } B\mathbf{y} = 0, \tag{10}$$

where $\mathbf{z} = [z_1, \cdots, z_m]^T \in \mathbb{C}^m$, $\mathbf{y} = [y_{11}, \cdots, y_{1n}, \cdots, y_{s1}, \cdots, y_{sn}]^T \in \mathbb{C}^{sn}$. The set of vectors $\mathbf{z}$ satisfying (10) forms a vector space representation of the closedness subspace $\mathcal{C}^{\alpha+1}_{\hat{\mathbf{x}}}(F)$. If the dimension of $\mathcal{C}^{\alpha+1}_{\hat{\mathbf{x}}}(F)$ is the same as $\mathcal{C}^{\alpha}_{\hat{\mathbf{x}}}(F)$, then terminate and output the multiplicity.

3 **Compute the dual space**: Let $\{h_1, \cdots, h_d\}$ be the basis of $\mathcal{C}^{\alpha+1}_{\hat{\mathbf{x}}}(F)$ computed in step 2. Compute the matrix $W_{\alpha+1} = (w_{ij})_{t\times d}$, where $w_{ij} = h_j(f_i)$, and set $\mathcal{D}^{\alpha+1}_{\hat{\mathbf{x}}}(F)$ by isomorphism to the kernel of the matrix $W_{\alpha+1}$. If the dimension of $\mathcal{D}^{\alpha+1}_{\hat{\mathbf{x}}}(F)$ is the same as $\mathcal{D}^{\alpha}_{\hat{\mathbf{x}}}(F)$, then terminate and output the multiplicity. Otherwise, update the dual support $S^{\alpha+1}_{\hat{\mathbf{x}}}(F)$ and set $\alpha = \alpha + 1$.

4 **Compute the closedness support**: Find all $\partial_{\mathbf{j}}$ satisfying $\Phi_\sigma(\partial_{\mathbf{j}}) \in S^{\alpha}_{\hat{\mathbf{x}}}(F)$ and set up the closedness support $M^{\alpha+1}_{\hat{\mathbf{x}}}(F)$. Go to step 2.

Although the method is substantially more efficient than the Macaulay algorithm, the sizes of matrices A and B in (10) can still be a bottleneck in computation. We can improve the efficiency even further with a new equation-by-equation strategy.

At the stage where $\mathcal{D}^{\alpha-1}_{\hat{\mathbf{x}}}(F), \mathcal{C}^{\alpha-1}_{\hat{\mathbf{x}}}(F), M^{\alpha-1}_{\hat{\mathbf{x}}}(F)$ and $S^{\alpha-1}_{\hat{\mathbf{x}}}(F)$ are computed for $\alpha > 0$, we can write

$$\begin{aligned} M^{\alpha}_{\hat{\mathbf{x}}}(F) &= \{\partial_{\mathbf{j}_1}, \partial_{\mathbf{j}_2}, \cdots, \partial_{\mathbf{j}_m}\} \cup \{\partial_{0\cdots 0}\} \quad \text{and} \\ \mathcal{D}^{\alpha-1}_{\hat{\mathbf{x}}}(F) &= \operatorname{span}\{c_1, c_2, \cdots, c_n\}, \end{aligned}$$

where $c_i \in \operatorname{span}\{S^{\alpha-1}_{\hat{\mathbf{x}}}(F)\}$. Because of $\mathcal{C}^{\alpha}_{\hat{\mathbf{x}}}(F) \in \operatorname{span}\{M^{\alpha}_{\hat{\mathbf{x}}}(F)\}$ and the closedness condition, the identification of $\mathcal{C}^{\alpha}_{\hat{\mathbf{x}}}(F)$ is equivalent to finding $z_{\mathbf{j}_i}$'s in $\mathbb{C}$ such that there exist $y_{\sigma k}$'s in $\mathbb{C}$ satisfying

$$\begin{aligned} \Phi_\sigma(z_{\mathbf{j}_1}\partial_{\mathbf{j}_1} + z_{\mathbf{j}_2}\partial_{\mathbf{j}_2} + \cdots + z_{\mathbf{j}_m}\partial_{\mathbf{j}_m}) &= y_{\sigma 1}c_1 + y_{\sigma 2}c_2 + \cdots + y_{\sigma n}c_n, \\ &\quad \sigma = 1, \cdots, s. \end{aligned} \tag{11}$$

In particular, equation (11) for $\sigma = 1$ leads to two sets of equations

$$\begin{aligned} z_{\mathbf{j}_i} &= d_{i11}y_{11} + d_{i12}y_{12} + \cdots + d_{i1n}y_{1n} \\ &\quad \text{for } i = 1, \cdots, m, \text{ where } \Phi_\sigma(\partial_{\mathbf{j}_i}) \neq 0 \end{aligned} \tag{12}$$

and

$$\begin{aligned} 0 &= d_{i11}y_{11} + d_{i12}y_{12} + \cdots + d_{i1n}y_{1n} \\ &\quad \text{for } i = 1, \cdots, m, \text{ where } \Phi_\sigma(\partial_{\mathbf{j}_i}) = 0 \end{aligned} \tag{13}$$

Equation (12) leads to

$$\mathbf{z}^{(1)} = A_1\mathbf{y}^{(1)},$$

where the components of the vector $\mathbf{z}^{(1)}$ have subscripts belonging to the index set $\mathbf{I}^{(1)} = \{\mathbf{j}_i \mid \Phi_1(\partial_{\mathbf{j}_i}) \neq 0\}$. Equation (13) yields

$$B_1\mathbf{y}^{(1)} = 0,$$

where $\mathbf{y}^{(1)} = [y_{11}, \cdots, y_{1n}]$. Solve $B_1\mathbf{y}^{(1)} = 0$ for the kernel of B_1. We have $\mathbf{y}^{(1)} = N_1\mathbf{u}^{(1)}$. The columns of N_1 form a basis for the kernel of B_1. Therefore

$$\mathbf{z}^{(1)} = A_1N_1\mathbf{u}^{(1)} \doteq M_1\mathbf{u}^{(1)}.$$

Similarly, applying (11) for $\sigma = 2$, we obtain two equations

$$\mathbf{z}^{(2)} = A_2\mathbf{y}^{(2)} \text{ and } B_2\mathbf{y}^{(2)} = 0,$$

where the components of the vector $\mathbf{z}^{(2)}$ have subscripts belong to the index set $\mathbf{I}^{(2)} = \{\mathbf{j}_i \mid \Phi_2(\partial_{\mathbf{j}_i}) \neq 0\}$ and $\mathbf{y}^{(2)} = [y_{21}, \cdots, y_{2n}]$. Similarly, we can get

$$\mathbf{y}^{(2)} = N_2\mathbf{u}^{(2)}, \mathbf{z}^{(2)} = A_2N_2\mathbf{u}^{(2)}.$$

If $\mathbf{I}^{(1)} \cap \mathbf{I}^{(2)} \neq \emptyset$, then the common components in $\mathbf{z}^{(1)}$ and $\mathbf{z}^{(2)}$ are equal. Extracting those equal components in $\mathbf{z}^{(1)} = A_2N_2\mathbf{u}^{(1)}$ and $\mathbf{z}^{(2)} = A_2N_2\mathbf{u}^{(2)}$ yields

$$D_1\mathbf{u}^{(1)} = D_2\mathbf{u}^{(2)},$$

which leads to the homogeneous equation

$$[D_1, -D_2]\begin{bmatrix} \mathbf{u}^{(1)} \\ \mathbf{u}^{(2)} \end{bmatrix} = 0.$$

1. Initialize $A_1 = O$, $B_1 = O$.
2. Construct rows of A_1, B_1 from (12) and (13) respectively. Compute the kernel of B_1 and generate M_1.
3. for $\delta = 2, 3, \cdots, s$ do
   (a) Initialize $A_\delta = O$, $B_\delta = O$.
   (b) Construct rows of A_δ, B_δ and compute the kernel of B_δ.
   (c) Find the intersection of $\mathbf{I}^{(\delta)}$ and $\mathbf{I}^{(\delta-1)}$; extract rows of D_δ and $D_{\delta-1}$.
   (d) Compute the kernel of $[D_\delta, -D_{\delta-1}]$ and generate the matrix M_δ.
   (e) Combine components of $\mathbf{z}^{(\delta-1)}$ and $\mathbf{z}^{(\delta)}$ to form a redefined $\mathbf{z}^{(\delta)}$, and update M_δ in (14).
4. Compute the kernel of $G^H M_s$, and output $\mathcal{C}^{\alpha}_{\hat{\mathbf{x}}}(F)$.

Equivalently, (by finding the kernel of $[D_1, -D_2]$)

$$\begin{bmatrix} \mathbf{u}^{(1)} \\ \mathbf{u}^{(2)} \end{bmatrix} = M_2 \mathbf{v}^{(2)}.$$

Therefore,

$$\begin{aligned} \mathbf{z}^{(1)} &= M_1 \mathbf{u}^{(1)} = [M_1, 0] \begin{bmatrix} \mathbf{u}^{(1)} \\ \mathbf{u}^{(2)} \end{bmatrix} = [M_1, 0] M_2 \mathbf{v}^{(2)}, \\ \mathbf{z}^{(2)} &= [0, A_2 N_2] M_2 \mathbf{v}^{(2)}. \end{aligned} \tag{14}$$

Combine the components of $\mathbf{z}^{(1)}$ and $\mathbf{z}^{(2)}$ in the above two equations into the redefined vector $\mathbf{z}^{(2)}$ whose components have subscripts belong to the index set $\mathbf{I}^{(2)} := \mathbf{I}^{(2)} \cup \mathbf{I}^{(1)}$. Then rewrite (14) as $\mathbf{z}^{(2)} = M_2 \mathbf{u}^{(2)}$ and continue this procedure to the next $\sigma = 3$ till $\sigma = s$.

The process of computing the closedness subspace with an equation-by-equation method can now be summarized in the following pseudo-code.

7. COMPUTATIONAL EXPERIMENTS

In this section, we present test results of MULTIPLICITY and compare it with the previous version of the closedness algorithm in [Zeng 2009] with the same output (multiplicity, basis of dual space and the Hilbert function). All experiments are carried out using a 2.33 GHz quad-core Xeon 5410 processor running 64-bit Linux. All timing measures are elapsed time in seconds. The objective of developing the new method in this paper is to advance the computational efficiency for large systems. For this purpose, we compute multiplicity structures for the three benchmark problems in [Zeng 2009] and the benchmark problems in [Dayton and Zeng 2005] in Table III. The benchmark problems in [Dayton and Zeng 2005] with selected multiple zeros are listed below. Most of them are widely used for testing algorithms in polynomial system solving.

1. *cbms1* [Sturmfels 2002]: $x^3 - yz = 0$, $y^3 - xz = 0$, $z^3 - xy = 0$, $(0, 0, 0)$
2. *cbms2* [Sturmfels 2002]: $x^3 - 3x^2y + 3xy^2 - y^3 - z^2 = 0$, $z^3 - 3z^2x + 3zx^2 - x^3 - y^2 = 0$, $y^3 - 3y^2z + 3yz^2 - z^3 - x^2 = 0$, $(0, 0, 0)$
3. *mth191* [Leykin et al. 2006]: $x^3 + y^2 + z^2 - 1 = 0$, $x^2 + y^3 + z^2 - 1 = 0$, $x^2 + y^2 + z^3 - 1 = 0$, $(0, 1, 0)$
4. *decker2* [Decker et al. 1983]: $x + y^3 = 0$, $x^2y - y^4 = 0$, $(0, 0)$
5. *Ojika2* [Ojika 1987]: $x^2 + y + z - 1 = 0$, $x + y^2 + z - 1 = 0$, $x + y + z^2 - 1 = 0$, $(0, 0, 1)$, $(1, 0, 0)$

Table III: Test results on benchmark problems of [Dayton and Zeng 2005]

system	zero	Hilbert function	multiplicity	previous time	new time
cmbs1	(0,0,0)	1, 3, 3, 3, 1	11	.367	.04
cmbs2	(0,0,0)	1, 3, 3, 1	8	.289	.03
mth191	(0,1,0)	1, 2, 1	4	.104	.02
Decker2	(0,0)	1,1,1,1	4	.107	.02
Ojika2	(0, 0, 1)	1, 1	2	.051	.02
	(1, 0, 0)	1, 1	2	.10	.02
Ojika3	(0, 0, 1)	1, 1, 1, 1	4	.198	.03
	(-5/2, 5/2, 1)	1, 1	2	.065	.01
Caprasse	(2, -i$\sqrt{3}$, 2, i$\sqrt{3}$)	1, 2, 1	4	.026	.04
DZ1	(0, 0, 0, 0)	1, 4, 10, 16, 22, 25, 22, 16, 10, 4, 1	131	20.46	12.06
DZ2	(0, 0, -1)	1, 2, 3, 3, 2, 2, 2, 1	16	.89	.06

6. *Ojika3* [Ojika 1987]: $x+y+z-1=0$, $2x^3+5y^2-10z+5z^3+5=0$, $2x+2y+z^2-1=0$, $(0,0,1)$, $(-\frac{5}{2},\frac{5}{2},1)$
7. *Caprasse* [Moritsugu and Kuriyama 2009]: $-x_1^3x_3+4x_1x_2^2x_3+4x_1^2x_2x_4+2x_2^3x_4+4x_1^2-10x_2^2+4x_1x_3-10x_2x_4+2=0$, $-x_1x_3^3+4x_2x_3^2x_4+4x_1x_3x_4^2+2x_2x_4^3+4x_1x_3+4x_3^2-10x_2x_4-10x_4^2+2=0$, $x_2^2x_3+2x_1x_2x_4-2x_1-x_3, 2x_2x_3x_4+x_1x_4^2-x_1-2x_3=0$, $(2,-i\sqrt{3},2,i\sqrt{3})$
8. *DZ1* [Dayton and Zeng 2005]: $x_1^4-x_2x_3x_4=0$, $x_2^4-x_1x_3x_4=0$, $x_3^4-x_1x_2x_4=0$, $x_4^4-x_1x_2x_3=0$, $(0,0,0,0)$
9. *DZ2* [Dayton and Zeng 2005]: $x^4=0, x^2y+y^4=0$, $z+z^2-7x^3-8x^2=0$, $(0,0,1)$

7.1. The KSS systems

The *KSS systems* of $n\times n$ at zero $(1,1,\cdots,1)$ are defined as [Zeng 2009]

$$x_i^2+\sum_{j=1}^{n}x_j-2x_i-(n-1) \;=\; 0, \text{ for } i=1,2,\cdots,n$$

at the zero $(1,\cdots,1)$. The multiplicity increases from 4 to 256 as n goes from 3 to 9 and the memory demand intensifies even further.

Table IV: Timing comparison on KSS systems

system size n	5	6	7	8	9	13
multiplicity(depth)	16(4)	42(6)	64(6)	163(8)	256(8)	2407(7)
previous time	2.08	13.12	38.03	1149.48	6517.54	–
new time	1.08	8.80	18.14	153.77	432.26	2679.48

7.2. The cyclic cubic systems

The *cyclic cubic systems* of $n\times n$ are defined as [Zeng 2009]

$$x_i^3-x_{i+1}x_{i+2} \;=\; 0, \text{ for } i=1,2,\cdots,n, \text{ with } x_{n+1}=x_1 \text{ and } x_{n+2}=x_2$$

at zero $(0,\cdots,0)$. The multiplicity of the cyclic cubic increases rapidly, and the memory is only enough for the previous code up to $n=7$.

7.3. The ten-fold triangle systems

The *ten-fold triangle systems* of $n\times n$ are defined as [Zeng 2009]

$$\begin{aligned} x_1+\cdots+x_i \;&=\; 0, \text{ for } i=1,2,\cdots,n-2,\\ (x_1+\cdots+x_{n-1})^2 \;&=\; 0, \quad (x_1+\cdots+x_n)^5 \;=\; 0 \end{aligned}$$

Table V: Timing comparison on cyclic cubic systems

system size n	3	4	5	6	7	8	9
multiplicity(depth)	11(4)	30(6)	62(7)	153(9)	338(10)	798(12)	1811(14)
previous time	0.34	2.13	9.32	72.32	733.29	–	–
new time	0.16	1.97	7.24	49.41	226.48	2035.92	33785.46

at zero $(0, \cdots, 0)$. This system has a fixed multiplicity 10 and depth 5 for all $n \geq 2$. But the problem size increases in terms of n.

Table VI: Timing comparison on ten-fold triangle systems

system size n	10	20	30	40	60	80	100	300
multiplicity(depth)	10(5)	10(5)	10(5)	10(5)	10(5)	10(5)	10(5)	10(5)
previous time	3.42	15.07	49.19	178.98	1317.54	5032.08	15258.53	–
new time	0.61	1.75	8.76	31.61	237.98	1050.12	2948.26	7849.34

7.4. Bifurcation problem

In scientific computing, many systems depend on parameters. As parameters vary, the qualitative behavior of solution may change dramatically at a "bifurcation point" where the solution of the system is multiple. This is an intersection point of several parameterized solution branches. From the perturbation invariance property of the multiplicity, the number of branches converging to the bifurcation point equal to the multiplicity. Here we look at a simple bifurcation problem

$$\begin{cases} \dfrac{dx}{dt} &= (x-p)^2(y-p)^2 - px, \\ \dfrac{dy}{dt} &= x^2 - y^2 - py, \end{cases}$$

where $x(t)$ and $y(t)$ are the functions of time t, p is a parameter. For this dynamic system, we consider the steady state system, i.e., time t goes to infinity. It becomes

$$\begin{cases} (x-p)^2(y-p)^2 - px, \\ x^2 - y^2 - py. \end{cases}$$

When $p = 0$, the steady state system has one solution $(0, 0)$. Using MULTIPLICITY, we know that the multiplicity of the system at $(0, 0)$ is 8 as follows.

```
    Depth          Hilbert function
       0                  1
       1                  2
       2                  2
       3                  2
       4                  1
-------------------------------------------
Depth:4    Multiplicity:8
```

When $p = 10^{-3}$, we solved the steady state system using *Bertini* and found that it has 8 solutions: four of them are real and the others are complex. We used the parameter tracking of *Bertini* to track the parameter p from 10^{-3} to -10^{-3}. It can be seen that the 8 solutions merge to $(0, 0)$ as p goes to 0. Here 4 paths of real solutions are plotted in Figure 3, which shows that $(0, 0)$ is a bifurcation point and there are 4 different branches on $\mathbb{R}^2$.

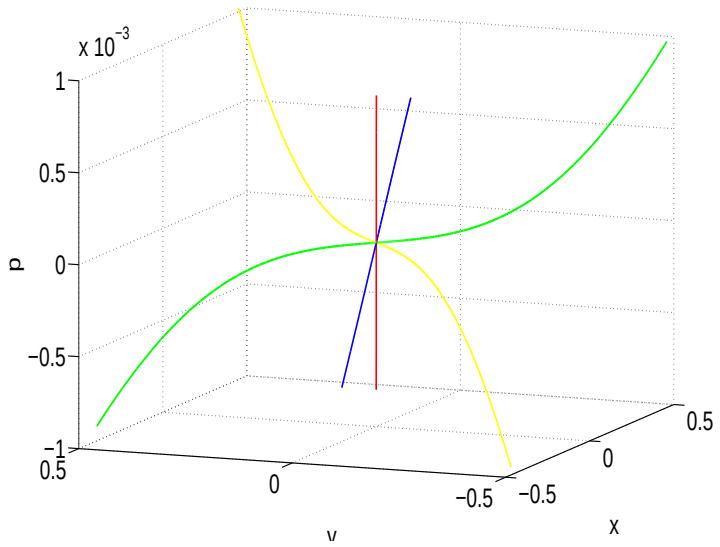

Fig. 3: Real solutions v.s. p

Conclusion

The equation-by-equation method provides a way to compute the closedness subspace that significantly reduces what is often a huge amount of computation. MULTIPLICITY is a program which implements this method to compute the multiplicity structure of a given system of nonlinear functions at a given zero. The experimental results on the benchmark systems clearly show the effectiveness of this new strategy.

REFERENCES

BATES D. J., HAUENSTEIN J. D., SOMMESE A. J., AND WAMPLER C. W. 2006. Bertini: Software for numerical algebraic geometry, *available at www.nd.edu/∼sommese/bertini*.

BATES D. J., HAUENSTEIN J. D., PETERSON C., AND SOMMESE A. J. 2009. A numerical local dimension test for points on the solution set of a system of polynomial equations, *SIAM J. Numer. Anal.*. 47, 3608-3623.

BATES D. J., PETERSON C., AND SOMMESE A. J. 2006. A numerical-symbolic algorithm for computing the multiplicity of a component of an algebraic set, *J. of Complexity*, 22, 475-489.

DAYTON B. H., LI T.-Y., AND ZENG Z. 2011. Multiple zeros of nonlinear systems, *Math. of Comp.*, 80, 2143-2168.

DAYTON B. H. AND ZENG Z. 2005. Computing the multiplicity structure in solving polynomial systems. In *Proceedings of ISSAC 05, ACM Press*, 116-123.

DECKER D. W., KELLER H. B., AND KELLY C. T. 1983. Convergence rate for Newton's method at singular points, *SIAM J. Numer. Anal.*, 20, 296-314.

DUO J., SHI J., AND WANG Y. 2008. Structure of the solution set for a class of semilinear elliptic equations with asymptotic linear nonlinearity, *Nonlinear Analysis*, 69, 2369-2378.

FULTON W. 1984. Intersection Theory, *Springer Verlag, Berlin*.

GÖTTSCHE L. 1994. Hilbert schemes of zero-dimensional subschemes of smooth varieties, *Lecture Notes in Mathematics*, 1572, *Springer Verlag, Berlin*.

2011. GRIFFIN Z.A., HAUENSTEIN J.D., PETERSON C., AND SOMMESE A.J. Numerical computation of the Hilbert function of a zero-scheme. *Available at* http://www.math.nd.edu/∼sommeset/preprints.

HAUENSTEIN J. D. 2010. Algebraic computations using Macaulay dual spaces, *submitted, available at* http://www.math.tamu.edu/∼jhauenst/preprints/index.html.

HAUENSTEIN J. D., AND WAMPLER C. W. 2010. Isosingular sets and deflation, *submitted, available at* http://www.math.tamu.edu/∼jhauenst/preprints/index.html.

KAHAN W. 1972. Conserving confluence curbs ill-condition, Computer Science, Technical Report 6, University of California, Berkeley, 1972.

LEYKIN A., VERSCHELDE J., AND ZHAO A. 2006. Newton's method with de ation for isolated singularities of polynomial systems, *Theoretical Computer Science*, 359, 111-122.

MACAULAY F. S. 1916. The Algebraic Theory of Modular Systems, *Cambridge Univ. Press, London*.

MORA T. 2004. Solving Polyonmial Equation Systems II, *Cambridge Univ. Press, London*.

MORGAN A. P. 1987 & 2009. Solving Polynomial Systems Using Continuation for Engineering and Scientific Problems, *Prentice-Hall, Inc., Upper Saddle River, NJ*. Reprinted as Vol. 57, *Classics in Applied Mathematics*, *SIAM*.

MORGAN A., AND SOMMESE A.J. 1989. Coefficient-parameter polynomial continuation, *Appl. Math. Comput.* 29 (1989), 123-160; Erratum, 51 (1992), p. 207.

MORITSUGU S., AND KURIYAMA K. 2009. On multiple zeros of systems of algebraic equations. In *Proceedings of ISSAC '99, ACM Press*, 23-30.

OJIKA T. 1987. Modified deflation algorithm for the solution of singular problems, *J. Math. Anal. Appl.*, 123, 199-221.

SOMMESE A. J., AND WAMPLER C. W. 1996. Numerical algebraic geometry, *in The Mathematics of Numerical Analysis*, Renegar J., Shub M., and Smale S., Eds. Vol. 32 of *Lectures in Applied Mathematics*, 749-763.

SOMMESE A. J., AND VERSCHELDE J. 2000. Numerical homotopies to compute generic points on positive dimensional algebraic sets, *J. of Complexity*, 16, 572-602.

STETTER H. J. 2004. Numerical Polynomial Algebra, *SIAM*.

STURMFELS B. 2002. Solving Systems of polynomial Equations, *Number 97 in CBMS Reginal Conference Series in Mathematics, AMS*.

VERSCHELDE J. 1996. Homotopy continuation methods for solving polynomial systems, *Ph.D. Dissertation, Department of Computer Science, Katholieke Universiteit Leuven, Leuven, Belgium*.

WARD J. R. 2005. Rotation numbers and global bifurcation in systems of ordinary differential equations, *Advanced Nonlinear Studies*, 5, 375-392.

WARD J. R. 2007. Existence, multiplicity, and bifurcation in systems of ordinary differential equations. In *Proceedings of Sixth Mississippi State Conference on Differential Equations and Computational Simulations*, 399-415.

ZENG Z. 2003. A method computing multiple roots of inexact polynomials, *in Proceedings of the 2003 international symposium on Symbolic and algebraic computation*, ISSAC 2003, 266-272.

ZENG Z. 2008. ApaTools: A Maple and Matlab toolbox for approximate polynomial algebra, *in Software for Algebraic Geometry, IMA Volume 148, Stillman M., Takayama N., and Verschelde J., Eds., Springer*, 149-167.

ZENG Z. 2009. The closedness subspace method for computing the multiplicity structure of a polynomial system, *in Interactions of Classical and Numerical Algebraic Geometry, D. Bates et al., eds., Contemp. Math., 496, Amer. Math. Soc.*, 347-362.