Inverse multivariate polynomial root-finding: numerical implementations of the affine and projective Buchberger-Möller algorithm.

Kim Batselier, Ngai Wong[1]

Department of Electrical and Electronic Engineering, The University of Hong Kong, Hong Kong

Abstract

We address the inverse problem of multivariate polynomial root-finding: given a finite set $\mathcal{Z}$ of points in $\mathbb{C}^n$, find the minimal set of multivariate polynomials that vanish on $\mathcal{Z}$. Two SVD-based algorithms are presented: one algorithm works only for affine roots and as a result almost always returns an overdetermined set of polynomials. This issue is resolved in the second algorithm by introducing projective points and hence adding roots at infinity. In addition, we show how the use of multiplicity structures is required to describe roots with multiplicities. We also derive a suitable tolerance that needs to be used when the roots are not known with infinite precision. A measure of how well the resulting polynomials vanish approximately on $\mathcal{Z}$ is shown to be the smallest singular value of a particular matrix. Both affine and projective implementations of our algorithm are applied to the problem of computing continuous-time polynomial dynamical systems from a given set of fixed points, demonstrating the effectiveness and robustness of our proposed methods.

Keywords: numerical Buchberger-Möller algorithm, singular value decomposition, reduced Gröbner basis, polynomial dynamical systems, fixed points
2010 MSC: 13P99, 65F99, 93A30

[1]This work was supported in part by the Hong Kong Research Grants Council under General Research Fund (GRF) Project 17208514, and the University Research Committee of The University of Hong Kong.

1. Introduction

By now it has been well established that for the case of zero-dimensional varieties the problem of finding all roots of a set of multivariate polynomials is a problem in linear algebra. Indeed, for the univariate case we have that a root $z \in \mathbb{C}$ of a degree d polynomial $p(x) = \sum_{i=0}^{d} a_i\, x^i\, (a_d \neq 0)$ can be retrieved from the eigenvalue problem

$$\begin{pmatrix} 1 \\ z \\ z^2 \\ \vdots \\ z^{d-1} \end{pmatrix} z = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ -\frac{a_0}{a_d} & -\frac{a_1}{a_d} & -\frac{a_2}{a_d} & \cdots & -\frac{a_{d-1}}{a_d} \end{pmatrix} \begin{pmatrix} 1 \\ z \\ z^2 \\ \vdots \\ z^{d-1} \end{pmatrix},$$

where the matrix on the right-hand side is called the companion matrix of the polynomial $p(x)$. If none of the roots have multiplicities, then the companion matrix is diagonalized by the Vandermonde matrix corresponding with the roots $z_1, \ldots, z_d$. In their seminal paper [1], Auzinger and Stetter generalized this idea to the multivariate case. For this we need to consider a set of multivariate polynomials $f_1(\mathbf{x}), \ldots, f_s(\mathbf{x})$, with $\mathbf{x} \equiv (x_1, \ldots, x_n) \in \mathbb{C}^n$, for which a root $\mathbf{z} \equiv (z_1, \ldots, z_n) \in \mathbb{C}^n$ is a point such that $f_1(\mathbf{z}) = \cdots = f_s(\mathbf{z}) = 0$. Then, given a monomial basis $B = \{b_1, \ldots, b_l\}$ for the quotient ring $\mathcal{C}^n/\langle f_1, \ldots, f_n\rangle$, where $\mathcal{C}^n$ denotes the ring of n-variate polynomials and $\langle f_1, \ldots, f_s\rangle$ is the polynomial ideal generated by $f_1, \ldots, f_s$, we retrieve the ith component of $\mathbf{z}$ from the eigenvalue problem

$$\begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_l \end{pmatrix} z_i = A_i \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_l \end{pmatrix}.$$

The $A_i\, (i = 1, \ldots, n)$ matrices express how the products of the monomials B with x_i modulo $f_1, \ldots, f_s$ are themselves representable in terms of the monomial basis B [2, 3]. The B monomials are typically found by explicitly computing a Gröbner basis for the polynomial ideal $\langle f_1, \ldots, f_n\rangle$. An alternative way is by considering the linearly dependent leading monomials in the row space of the Macaulay matrix [4].

It is much less known that the inverse root-finding problem of finding a minimal set of polynomials that vanish on a given finite set of roots is also a problem in linear algebra. The univariate case is trivially solved. Given a set of m roots $\mathcal{Z} = \{z_1, \ldots, z_m\}$, then the polynomial $p(x) = \prod_{i=1}^{m}(x - z_i)$ is the unique polynomial of minimal degree that vanishes on $\mathcal{Z}$. In terms of linear algebra, one needs to find a single vector, unique up to a scaling factor, that lies in the left null space of the $(m + 1) \times m$ confluent Vandermonde matrix V on the roots $z_1, \ldots, z_m$. The extension of this problem to the multivariate case is a bit more involved. Essentially, a reduced Gröbner basis for the polynomial ideal that vanishes on the given set of roots needs to be computed. This reduced Gröbner basis is unique and contains the minimal number of polynomials of minimal degrees for a given monomial ordering. It turns out that the polynomials of this reduced Gröbner basis also lie in the left null space of a confluent Vandermonde matrix, generalized to the multivariate case.

Möller and Buchberger were the first to solve this problem with their symbolic Buchberger-Möller algorithm [5]. Their algorithm however only works for the affine case and none of the roots can have multiplicities. In addition, since it is a symbolic algorithm, the roots can only be integers or rational numbers. An implementation can be found in the COCOA computer algebra system [6]. An extension of the symbolic Buchberger-Möller algorithm to the projective case together with the derivation of a suitable stopping criterion appeared in [7]. Again, no multiplicities of the roots are supported for this symbolic projective algorithm. This is also newly addressed in this article by the introduction of multiplicity structures. A numerical-symbolic hybrid method that generates exact polynomials using inexact intermediate numerical results obtained from homotopy methods is described in [8]. A numerical implementation of the affine Buchberger-Möller algorithm is described in [9]. Just as in this article, their implementation uses the singular value decomposition (SVD), which can be computed in a numerically backward stable way [10]. After the computation of the SVD, an additional Gaussian Elimination (GE) step is required. Their method consequently needs two tolerances ϵ, τ for the SVD and GE steps respectively. A set of reduced Gröbner basis polynomials $g_1, \ldots, g_t$ are computed such that $|g_i(\mathcal{Z})| \leq \delta = \epsilon\sqrt{t} + \tau t(t + \mu)\sqrt{m}\,(i = 1, \ldots, t)$. This bound only applies when the roots lie in the interval $[-1, 1]^n$. Also, the numbers t and μ are not known beforehand and depend on the tolerances ϵ, τ in an unpredictable manner, making it very impractical to choose the tolerances in function of

the required δ. Our algorithm does not require the GE step. Furthermore, we derive a suitable tolerance to be used with the SVD when the roots are not known with infinite precision. This uniquely determines the number of computed Gröbner basis polynomials and provides a measure on how well they vanish approximately on the given finite set of roots. For the application that we describe in Section 5, it will be necessary to obtain a "square" system of polynomials. This means that there are an equal amount of polynomials as there are variables. The affine Buchberger-Möller will however almost never compute a square system. We will explain how by adding multiplcities or roots at infinities a square system of polynomials can be obtained. To summarize, the contributions of this article are

- two new numerical affine and projective Buchberger-Möller algorithms are derived,
- we describe how by adding multiplicities and/or roots at infinity a square polynomial system can be obtained,
- multiple roots are supported,
- no extra GE step is required,
- no scaling of the roots is required,
- we derive a tolerance for the SVD that depends on the absolute uncertainty on the given roots.

The case of positive-dimensional solution sets, i.e. infinite solutions, is not supported. The outline of this article is as follows. In Section 2 we will define some notations and introduce some necessary concepts from algebraic geometry. In addition, we will define the confluent multivariate Vandermonde matrix and demonstrate that it is possible to retrieve the required reduced Gröbner basis from it. In Section 3, we derive our SVD-based affine Buchberger-Möller algorithm by means of a running example. We also show a suitable tolerance can be chosen for the case when the roots are not known with infinite precision. In addition, we discuss the computational complexity of the method and explain how it also provides a measure on how well the resulting polynomials vanish on the given set of fixed points. In Section 4 the affine algorithm is extended to the projective case. This allows us to add roots at infinity, which is in most cases a necessary requirement

in order to obtain a system of n polynomials. In Section 5, the application of both algorithms is illustrated on a particular modeling problem: given a set of fixed points, find a corresponding minimal polynomial dynamical system. We also present the case where the roots are known to be perturbed and compare the results with the unperturbed case. All algorithms are implemented in MATLAB [11]/Octave[12] and are freely available at `https://github.com/kbatseli/PNLA_MATLAB_OCTAVE`.

Another application of the algorithms described in this article are data-mining applications of finding polynomial relations in measured data from both the oil and steel industry [9]. There the argument is made that measurements in industrial applications invariably contain errors. Hence, numerical algorithms that compute approximate results from approximate data are preferred over symbolic methods that compute exact results from exact data. Finally, another application is found in the realization of multi-dimensional linear state space systems as described in [13].

2. Notation and facts

Before deriving our algorithm, we first introduce the notation and provide some basic facts on polynomials and Gröbner bases. The main goal of this section is to show where this confluent multivariate Vandermonde matrix, which is central to our algorithm, comes from. Good reference works on this subject matter are [14, 15, 16, 17, 3].

2.1. Vector space of polynomials and projective coordinates

We introduce the shorter notation $\mathcal{C}^n$ for the ring of multivariate polynomials in n variables with complex coefficients. The subset $\mathcal{C}_d^n$ of $\mathcal{C}^n$, which contains all n-variate polynomials of degree at most d is a vector space. Also, a monomial basis will be used to describe $\mathcal{C}_d^n$ and since there are $\binom{d+n}{n}$ n-variate monomials of degree at most d, we set $q \equiv \dim \mathcal{C}_d^n = \binom{d+n}{n}$.

Roots at infinity are described in the projective setting with homogeneous polynomials. A polynomial of degree d is homogeneous when every term is of degree d. The set of all homogeneous polynomials in $n+1$ variables is denoted $\mathcal{P}^n$. Unlike the non-homogeneous case, limiting $\mathcal{P}^n$ to all homogeneous polynomials of degrees 0 up to d does not result in a vector space. The set $\mathcal{P}_d^n$ of all homogeneous polynomials of degree d however forms a vector space. A non-homogeneous polynomial can easily be made homogeneous by introducing an extra variable x_0 [15, p. 373].

Definition 2.1. *Let $f \in \mathcal{C}_d^n$, then its homogenization $f^h \in \mathcal{P}_d^n$ is the polynomial obtained by multiplying each term of f with a power of x_0 such that its degree becomes d.*

In order to describe solution sets of systems of homogeneous polynomials, projective space needs to be introduced. First, an equivalence relation $\sim$ on the nonzero points of $\mathbb{C}^{n+1}$ is defined as $(x'_0, \ldots, x'_n) \sim (x_0, \ldots, x_n)$ if there is a nonzero $\lambda \in \mathbb{C}$ such that $(x'_0, \ldots, x'_n) = \lambda\,(x_0, \ldots, x_n)$. Projective space is then defined as the set of resulting equivalence classes in the following way.

Definition 2.2. *([15, p. 368]) n-dimensional projective space $\mathbb{P}^n$ is the set of equivalence classes of $\sim$ on $\mathbb{C}^{n+1} - \{0\}$. Each nonzero $(n+1)$-tuple $(x_0, \ldots, x_n)$ defines a point p in $\mathbb{P}^n$, and we say that $(x_0, \ldots, x_n)$ are homogeneous coordinates of p.*

Note that the origin $(0, \ldots, 0) \in \mathbb{C}^{n+1}$ is not a point in projective space. Because of the equivalence relation $\sim$, an infinite number of projective points $(x_0, \ldots, x_n)$ can be associated with 1 affine point $(x_1, \ldots, x_n)$. Affine space $\mathbb{C}^n$ can be retrieved as a 'slice' of projective space $\mathbb{C}^n = \{(x_0, x_1, \ldots, x_n) \in \mathbb{P}^n \ : \ x_0 = 1\}$. This means that given a projective point $p = (x_0, \ldots, x_n)$ with $x_0 \neq 0$, its affine counterpart is $(1, \frac{x_1}{x_0}, \ldots, \frac{x_n}{x_0})$. The projective points for which $x_0 = 0$ are called points at infinity.

2.2. Polynomial ideals and Gröbner bases

The left null space of the Vandermonde matrix turns out to be related to polynomial ideals and their generators. Here we give a quick introduction to polynomial ideals and Gröbner bases.

Definition 2.3. *([15, p. 30]) Let $f_1, \ldots, f_s \in \mathcal{C}^n$. Then we set*

$$\langle f_1, \ldots, f_s \rangle \ = \ \left\{ \sum_{i=1}^{s} h_i f_i : h_1, \ldots, h_s \in \mathcal{C}^n \right\} \tag{1}$$

and call it the ideal generated by $f_1, \ldots, f_s$.

The ideal hence contains all polynomial combinations $\sum_{i=1}^{s} h_i f_i$ without any constraints on the degrees of $h_1, \ldots, h_s$. For this reason, the polynomials $f_1, \ldots, f_s$ are also called the generators of the polynomial ideal. We will denote all polynomials of the ideal $\langle f_1, \ldots, f_s \rangle$ with a degree of at most d by $\langle f_1, \ldots, f_s \rangle_d$. Observe that this implies that $\langle f_1, \ldots, f_s \rangle_d \subset \mathcal{C}_d^n$ and

$\langle f_1, \ldots, f_s \rangle_d$ is therefore also a vector space. Likewise, for the homogeneous ideal $\langle f_1^h, \ldots, f_s^h \rangle$ we can consider the vector space $\langle f_1^h, \ldots, f_s^h \rangle_d$ of all homogeneous polynomials of degree d. The set of generators is not unique for a given polynomial ideal. An important set of generators for a given polynomial ideal $\langle f_1, \ldots, f_s \rangle$ is a Gröbner basis. Their most useful property is also their defining property.

Definition 2.4. *([15, p. 77]) Given a set of multivariate polynomials $f_1, \ldots, f_s$ and a monomial ordering, then a finite set of polynomials $G = \{g_1, \ldots, g_t\} \in \langle f_1, \ldots, f_s \rangle$ is a Gröbner basis of $\langle f_1, \ldots, f_s \rangle$ if*

$$\forall\, p \in \langle f_1, \ldots, f_s \rangle, \exists\, g \in G \text{ such that } LM(g) \text{ divides } LM(p),$$

where $LM(g)$, $LM(p)$ denote the leading monomials of g and p respectively, according to the given monomial ordering. In addition, a Gröbner basis is called reduced if no monomial in any element of the basis is divisible by the leading monomials of the other elements of the basis.

Note from the definition that a Gröbner basis depends on the monomial ordering. For a formal definition of monomial orderings together with a detailed description of some relevant orderings in computational algebraic geometry see [15, p. 54]. The monomial ordering used in this article is the graded xel ordering [18, p.3], which is sometimes also called the degree negative lexicographic monomial ordering. This ordering is graded because it first compares the degrees of the two monomials and applies the xel ordering when there is a tie. The defining property of a Gröbner basis is hence that the leading monomial of every polynomial in $\langle f_1, \ldots, f_s \rangle$ is divisible by at least one of the leading monomials of the Gröbner basis.

Another important concept is the quotient ring of a polynomial ideal. A monomial basis for the quotient ring will turn out to be the building blocks of a reduced Gröbner basis.

Definition 2.5. *([3, p. 29] for a given polynomial $I = \langle f_1, \ldots, f_s \rangle$, the set $[p]_I = \{r \in \mathcal{C}^n : p - r \in I\}$ is called the residue class of p mod I. The set of all such residue classes is called the quotient ring of I and denoted $\mathcal{C}^n/I$.*

Observe that the elements r of $[p]_I$ are remainders of polynomials p modulo the ideal I. The quotient ring is therefore the ring of all remainders for any polynomial p. It is a classical result that for a polynomial system $f_1, \ldots, f_s$

with a finite number of m affine roots that its quotient ring $\mathcal{C}^n/I$ is a finite-dimensional vector space [3, p. 31]. Furthermore, the dimension of $\mathcal{C}^n/I$ is the number of affine roots, including multiplicities. Another way of expressing this result is that there are m linearly independent monomials that form a basis for $\mathcal{C}^n/I$. These basis monomials are also called the standard monomials. Since standard monomials form remainders, it is important to observe that their set B is described by $B = \{b_i\,(i = 1, \ldots, m) : b_i \notin \langle \mathrm{LM}(I)\rangle\}$. If we denote the set of leading monomials of a Gröbner basis by $A = \{a_1, \ldots, a_t\}$, then a reduced Gröbner basis are polynomials of the form

$$g_i = a_i + \sum_{j=1}^{m} \beta_j\, b_j \quad (\beta_j \in \mathbb{C}, b_j \in B). \tag{2}$$

Polynomials of the form (2) will be computed numerically by the algorithms in this article. Hence, it is necessary to determine the monomials $a \in A$ and $b \in B$. This will be done by inspection of the rows of the confluent Vandermonde matrix, described in the next subsection.

2.3. Dual vector space

We are now ready to describe the confluent multivariate Vandermonde matrix, which plays a central role in our algorithm, as a basis for a particular dual vector space of $\langle f_1, \ldots, f_s\rangle_d$. The dual of a vector space $\mathcal{V}$ over a field k is a vector space $\mathcal{V}'$ of linear functionals $l : \mathcal{V} \to k$ that map each vector of $\mathcal{V}$ to an element of the field k. In our case, the field k will always be $\mathbb{C}$. Consequently, the dual vector space of $\mathcal{C}_d^n$ is $\mathcal{C}_d^{n'}$. The most interesting and useful dual vector space in this article is the annihilator.

Definition 2.6. *([19, p. 26]) The annihilator $\mathcal{V}^o$ of a vector space $\mathcal{V}$ is the set of linear functionals $l \in \mathcal{V}'$ such that $l(v) = 0$ for all $v \in \mathcal{V}$.*

We are interested in the annihilator $\langle f_1, \ldots, f_s\rangle_d^o$ of $\langle f_1, \ldots, f_s\rangle_d$, since a basis for it is given by the confluent multivariate Vandermonde matrix. We now need to show that from a certain degree d the annihilator $\langle f_1, \ldots, f_s\rangle_d^o$ is isomorphic to $\mathcal{C}^n/I$. This isomorphism then implies that we can determine the standard monomials B and consequently a reduced Gröbner basis from the Vandermonde matrix. The dimension of an annihilator is given by the following theorem.

Theorem 2.1. *([19, p. 26]) If $\mathcal{M}$ is an r-dimensional subspace of an n-dimensional vector space $\mathcal{V}$, then $\mathcal{M}^o$ is an $(n-r)$-dimensional subspace of $\mathcal{V}'$.*

Theorem 2.1 implies that if $\dim \langle f_1, \ldots, f_s \rangle_d = r$, then $\dim \langle f_1, \ldots, f_s \rangle_d^o = q - r$. In addition, we also have that

$$\begin{aligned} \dim \langle f_1, \ldots, f_s \rangle_d^o &= q - r, \\ &= \dim \mathcal{C}_d^n - \dim \langle f_1, \ldots, f_s \rangle_d, \\ &= \dim \mathcal{C}_d^n / \langle f_1, \ldots, f_s \rangle_d = \mathrm{HF}(d). \end{aligned} \tag{3}$$

The function $\mathrm{HF}(d)$ that expresses $\dim \langle f_1, \ldots, f_s \rangle_d^o$ as a function of the degree d is known in the literature as the Hilbert Function [15, p. 457]. For the homogeneous case the Hilbert Function $\mathrm{HF}(d)$ is defined as the dimension of the vector space $\mathcal{P}_d^n / \langle f_1^h, \ldots, f_s^h \rangle_d$. There is a degree $d^\star$ such that $\mathrm{HF}(d)$ is a polynomial for all $d \geq d^\star$. This polynomial is consequently called the Hilbert Polynomial $\mathrm{HP}(d)$ and $d^\star$ is called the degree of regularity [15, p. 459]. More importantly is the fact that the degree of this polynomial is the dimension of the solution set.

Theorem 2.2. *([15, p. 461]) Let $\langle f_1, \ldots, f_s \rangle$ be a polynomial ideal and V be its solution set, then $\dim V = \deg HP(d)$.*

A similar theorem applies to the projective case [15, p. 464]. Since we start from a given finite set of roots, we always have that $\dim V = 0$. Theorem 2.2 then implies that for all $d \geq d^\star$ $\dim \langle f_1, \ldots, f_s \rangle_d^o = \dim \mathcal{C}^n / I$ or in other words $\mathcal{C}^n / I \cong \langle f_1, \ldots, f_s \rangle_d^o$, $\forall d \geq d^\star$. This has the important implication that we can use $\langle f_1, \ldots, f_s \rangle_d^o$ to determine the standard monomials, which we will need to determine a reduced Gröbner basis. Constructing a basis for the annihilator is very straightforward. Since all functionals of $\langle f_1, \ldots, f_s \rangle_d^o$ map all polynomials in $\langle f_1, \ldots, f_s \rangle_d$ to zero, a basis for the dual vector space is then described by linear functionals that evaluate polynomials in their roots. This leads us to the following definition.

Definition 2.7. *([3, p 8]) Let $j \in \mathbb{N}_0^n$ and $z \in \mathbb{C}^n$, then the differential functional $\partial_j|_z \in \mathcal{C}_d^{n'}$ is defined by*

$$\partial_j|_z \triangleq \frac{1}{j_1! \ldots j_n!} \frac{\partial^{j_1 + \ldots + j_n}}{\partial x_1^{j_1} \ldots \partial x_n^{j_n}}|_z$$

where $|_z$ stands for evaluation in $\boldsymbol{z} = (x_1, \ldots, x_n)$.

When $j_1 = j_2 = \cdots = j_n = 0$, then the functional evaluates all polynomials in the point $\mathbf{z}$. If any of the j_i is nonzero, then the partial derivative with respect to x_i of the polynomials is evaluated in $\mathbf{z}$. We will need these higher order derivatives to define multiple roots. Being elements of the dual vector space, these differential functionals $\partial_j|_z$ can be represented as vectors. Stacking all these vectors together in a matrix results in the confluent multivariate Vandermonde matrix, which we illustrate in the following example.

Example 1. *In $\mathcal{C}_3^{2'}$ the functionals $\partial_{00}|_z, \partial_{10}|_z, \partial_{01}|_z, \partial_{20}|_z, \partial_{11}|_z$ and $\partial_{02}|_z$ have the following coefficient vectors*

$$K = \begin{array}{c} \begin{array}{cccccc} \partial_{00}|_z & \partial_{10}|_z & \partial_{01}|_z & \partial_{20}|_z & \partial_{11}|_z & \partial_{02}|_z \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ x_1 & 1 & 0 & 0 & 0 & 0 \\ x_2 & 0 & 1 & 0 & 0 & 0 \\ x_1^2 & 2x_1 & 0 & 1 & 0 & 0 \\ x_1x_2 & x_2 & x_1 & 0 & 1 & 0 \\ x_2^2 & 0 & 2x_2 & 0 & 0 & 1 \\ x_1^3 & 3x_1^2 & 0 & 3x_1 & 0 & 0 \\ x_1^2x_2 & 2x_1x_2 & x_1^2 & x_2 & 2x_1 & 0 \\ x_1x_2^2 & x_2^2 & 2x_1x_2 & 0 & 0 & x_1 \\ x_2^3 & 0 & 3x_2^2 & 0 & 0 & 3x_2 \end{pmatrix} \end{array}, \tag{4}$$

with $\boldsymbol{z} = (x_1, x_2) \in \mathbb{C}^2$.

Observe that we had to specify the degree d of $\mathcal{C}_d^{n'}$ in order to be able to write down K. We will make no further distinction between the differential functionals $\partial_j|_z$ and their coefficient vectors. As mentioned earlier, the higher order derivatives are necessary to define multiplicities of roots.

Definition 2.8. *([3, p. 328]) For a polynomial ideal $\langle f_1, \ldots, f_s \rangle$ with a finite number of roots, consider a root $\boldsymbol{z} \in \mathcal{C}^n$. The multiplicity m of $\boldsymbol{z}$ is then defined as the number of linearly independent functionals $c_\mu \triangleq \sum_j \beta_{\mu j} \partial_j|_z$ that form a closed vector space $\mathcal{D}_z \triangleq span(c_\mu, \mu = 1, \ldots, m)$. The functionals c_μ are then said to define the multiplicity structure of z.*

The problem of finding the multiplicity structure numerically for the multiple roots of a given polynomial system $f_1, \ldots, f_s$ is solved in [20, 21, 22]. A numerical-symbolic method that approaches the notion of multiplicity in

a different but related way can be found in [23]. Observe also that the representation of the multiplicity structure of a root is not unique. Indeed, if we denote the matrix of coefficient vectors of c_μ by K, then for any nonsingular $m \times m$ matrix T we have that K and $K\,T$ share the same the column space. As Definition 2.8 already points out, it is not sufficient to take m functionals for a given root $\mathbf{z}$ and assume it specifies a polynomial ideal with a root $\mathbf{z}$ of multiplicity m. The additional requirement of closedness of the corresponding vector space is also necessary.

Definition 2.9. *([3, p. 44]) A vector space $\mathcal{D}$ of functionals on $\mathcal{C}^n$ is closed if and only if $\forall q \in \mathcal{C}^n$*

$$l(p) = 0,\ \forall l \in \mathcal{D} \Rightarrow l(qp) = 0,\ \forall l \in \mathcal{D}.$$

An easy way that allows us to check whether a given basis of functionals $\partial_j|_z$ spans a closed vector space $\mathcal{D}$ is by applying anti-differentiation operators s_σ to these functionals.

Definition 2.10. *([3, p. 330]) The anti-differentiation operators $s_\sigma, \sigma = 1, \ldots, n$ are defined by*

$$s_\sigma\, \partial_j|_z \triangleq \begin{cases} \partial_{j-e_\sigma}|_z & \text{if} \quad j_\sigma > 0, \\ 0\text{-}\textit{functional} & \text{if} \quad j_\sigma = 0, \end{cases}$$

where e_σ is the σ-th unit vector and

$$s_\sigma\,(\sum_j \beta_j\, \partial_j|_z) \triangleq \sum_j \beta_j\, s_\sigma\, \partial_j|_z.$$

Once the anti-differential operators are defined, then checking whether a set of m differential functionals c_μ define a closed subspace can be done by using the following theorem.

Theorem 2.3. *([3, p. 330]) In $\mathcal{C}^n$, consider the linear space $\mathcal{D}_z$ spanned by m differential functionals $c_\mu = \sum_j \beta_{\mu j}\, \partial_j|_z, (\mu = 1, \ldots, m)$. Then $\mathcal{D}_z$ is closed if and only if $c \in \mathcal{D}_z \ \Rightarrow\ s_\sigma\, c \in \mathcal{D}_z$, $\sigma = 1, \ldots, n$.*

Example 2. *Consider the following basis K for $\mathcal{D}_{(1,2,3)}$*

$$K \ = \ \left(\ \partial_{000}|_{(1,2,3)} \quad \partial_{100}|_{(1,2,3)} \quad \partial_{010}|_{(1,2,3)} \quad \partial_{110}|_{(1,2,3)} - 2\partial_{020}|_{(1,2,3)} \ \right).$$

Table 1 lists the application of all anti-differentiation operators s_1, s_2, s_3 to the basis functionals in K. It is clear that all entries of the table are elements of $\mathcal{D}_{(1,2,3)}$, which proves its closedness.

Table 1: Application of anti-differentiation operators to the functionals.

	$\partial_{000}\vert_{(1,2,3)}$	$\partial_{100}\vert_{(1,2,3)}$	$\partial_{010}\vert_{(1,2,3)}$	$\partial_{110}\vert_{(1,2,3)} - 2\partial_{020}\vert_{(1,2,3)}$
$s_1\partial_j\vert_z$	0	$\partial_{000}\vert_{(1,2,3)}$	0	$\partial_{010}\vert_{(1,2,3)}$
$s_2\,\partial_j\vert_z$	0	0	$\partial_{000}\vert_{(1,2,3)}$	$\partial_{100}\vert_{(1,2,3)} - 2\partial_{010}\vert_{(1,2,3)}$
$s_3\,\partial_j\vert_z$	0	0	0	0

We end this section with the formal definition of the affine confluent multivariate Vandermonde matrix.

Definition 2.11. *For a given set of affine roots $\mathcal{Z} = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m\} \in \mathbb{C}^n$ each with a certain multiplicity structure and given degree d, the confluent multivariate Vandermonde matrix K is the matrix of differential functionals $c_\mu \in \mathcal{C}_d^{n'}$ for each of the roots in $\mathcal{Z}$, such that their multiplicity structures form closed subspaces.*

In Section 4 we will extend this notion of the affine confluent Vandermonde matrix to the projective case.

3. Numerical affine Buchberger-Möller algorithm

3.1. Derivation of the algorithm

In the previous section we discussed all necessary ingredients to numerically determine a reduced Gröbner basis $f_1, \ldots, f_s$ using a basis of the annihilator $\langle f_1, \ldots, f_s\rangle_d^o$. We will illustrate the derivation of the algorithm by means of a small running example.

Example 3. *Suppose that $\mathcal{Z} = \{(0,0),(3,5)\}$ with multiplicity structures $\partial_{00}, \partial_{10}, \partial_{20}$ and $\partial_{00}, \partial_{01}$, respectively. The closedness of the dual vector space is easily verified using Theorem 2.3. Since there are in total 5 differential functionals, this means that $dim\mathcal{C}^n/I = 5$. We set $d = 3$ and construct the Vandermonde matrix K with respect to the graded xel ordering*

$$K = \begin{array}{c} \\ 1 \\ x_1 \\ x_2 \\ x_1^2 \\ x_1x_2 \\ x_2^2 \\ x_1^3 \\ x_1^2x_2 \\ x_1x_2^2 \\ x_2^3 \end{array} \begin{array}{c} \begin{array}{ccccc} \partial_{00}|_{(0,0)} & \partial_{10}|_{(0,0)} & \partial_{20}|_{(0,0)} & \partial_{00}|_{(3,5)} & \partial_{01}|_{(3,5)} \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 3 & 0 \\ 0 & 0 & 0 & 5 & 1 \\ 0 & 0 & 1 & 9 & 0 \\ 0 & 0 & 0 & 15 & 3 \\ 0 & 0 & 0 & 25 & 10 \\ 0 & 0 & 0 & 27 & 0 \\ 0 & 0 & 0 & 45 & 9 \\ 0 & 0 & 0 & 75 & 30 \\ 0 & 0 & 0 & 125 & 75 \end{pmatrix} \end{array}$$

where both the rows and columns are labeled accordingly.

Observe that we can associate a monomial with each row of K. These are simply the entries of ∂_{00} according to the graded xel ordering. The rank of K in Example 3 is 5. This implies that 5 linearly independent monomials, corresponding with particular rows of K, can be found. If these are found by checking the rows of K for linear independence starting from top to bottom, then it can be proved that the obtained linearly independent monomials are of lowest total degree [18]. These monomials will in fact be a basis for $\mathcal{C}^n/I$. Algorithm 1 presents an algorithm in pseudo-code that finds a maximal set of linearly independent row indices B and linearly dependent row indices A for a given Vandermonde matrix, starting from the top row r_1 where r_i denotes the index of the ith row. Consequently, if the algorithm returns m linearly independent monomials, where m is the total number of fixed points including multiplicities, then a basis for $\mathcal{C}^n/I$ is found. Because of the bijection between the row indices of K and monomials, $a \in A$ and $b \in B$ will denote both the row indices as their corresponding monomials. It should be clear from the context which interpretation is used.

Algorithm 1. *Find a maximal set of linearly independent rows*
Input*: confluent multivariate Vandermonde matrix K*
Output*: linearly independent monomials B and linearly dependent monomials A*

$A, B \leftarrow \emptyset$
if $r_1 \neq 0$ ***then***
  $B \leftarrow [B\,,\, r_1]$
end if

for *i from* 2 *up to* $\binom{d+n}{n}$ ***do***
  if $K(r_i,:)$ *linearly independent with respect to* $\{r_1,\ldots,r_{i-1}\}$ ***then***
    $B \leftarrow [B\,,\, r_i]$
  else
    $A \leftarrow [A\,,\, r_i]$
  end if
end for

We will now revisit Example 3 to demonstrate how the pseudo-code of Algorithm 1 can be made more explicit and how it can be adjusted such that it also returns the desired polynomials $g_1,\ldots,g_t$.

Example 4. *Running Algorithm 1 on K from Example 3 results in*

$$\begin{aligned} B &= \{1, x_1, x_2, x_1^2, x_2^2\}, \\ A &= \{x_1x_2, x_1^3, x_1^2x_2, x_1x_2^2, x_2^3\}. \end{aligned}$$

The rank of K being 5 implies that the set B contains the five standard monomials. Interpreting B as a set of row indices of K, this can also be written using MATLAB notation as $\mathrm{rank}(K(B,:)) = 5$. The first element of A is $a_1 = x_1x_2$ and from (2) it then follows that the first polynomial g_1 of the desired reduced Gröbner basis has the following form

$$g_1 = c_{11}\,x_1x_2 + c_{20}\,x_1^2 + c_{01}\,x_2 + c_{10}\,x_1 + c_{00},$$

with unknown coefficients $c_{00},\ldots,c_{11}$. These coefficients can be uniquely determined by solving the following linear system

$$\begin{pmatrix} K(B,:)^T & K(a_1,:)^T \end{pmatrix} \begin{pmatrix} c_{00} \\ c_{10} \\ c_{01} \\ c_{20} \\ c_{11} \end{pmatrix} = 0, \tag{5}$$

which is guaranteed to have a solution since (5) expresses the linear dependence of the row $K(a_1,:)$ with respect to $K(B,:)$. The matrix on the left-hand side of (5) can be written compactly using MATLAB notation as $K([B,a_1],:)^T$. We will use this notation when discussing the numerical implementation of our algorithm.

The linear system (5) can be solved for each element of A. This results in a polynomial system of $q - m = 10 - 5 = 5$ polynomials. This number of polynomials however can be reduced by using the defining property of a Gröbner basis. Indeed, since the leading monomials of Gröbner basis polynomials divide all leading monomials in the ideal $\langle g_1, \ldots, g_t \rangle$, it is sufficient to retain only those monomials of A that are not divisible by any other element of A. Or equivalently in Algorithm 1, as soon as an element of A is found, none of its monomial multiples need to be checked anymore for linear dependence. We now apply this reduction step to the monomials of our running example.

Example 5. *The set of linearly dependent monomials A was*

$$A \;=\; \{x_1x_2, x_1^3, x_1^2x_2, x_1x_2^2, x_2^3\}.$$

By removing the monomials that are divisible by other elements of A we obtain the following reduced set $A^\star \;=\; \{x_1x_2, x_1^3, x_2^3\}$. Solving the linear systems (5) *results in the following polynomials*

$$G \;=\; \begin{cases} g_1 &= 0.9487\, x_2 - 0.3162\, x_1\, x_2, \\ g_2 &= 0.9908\, x_2 - 0.0991\, x_2^2 - 0.0917\, x_1^3, \\ g_3 &= 0.9278\, x_2 - 0.3711\, x_2^2 + 0.0371\, x_2^3. \end{cases} \tag{6}$$

The only thing that remains is a stop condition that tells us when d is large enough. This is easily derived, again using the defining property of Gröbner bases.

Lemma 3.1. *Let A, B be the monomial sets obtained from running Algorithm 1 on the Vandermonde matrix K for a degree d. Furthermore, let $A^\star$ be the subset of reduced linearly dependent monomials and T_{d+1}^n be the set of all $\binom{d+n}{n-1}$ n-variate monomials of total degree $d+1$. If the cardinality of B equals $\dim \mathcal{C}^n/I$ and all monomials in T_{d+1}^n are divisible by elements of $A^\star$, then d is sufficiently large.*

Proof. Since the cardinality of B equals $\dim \mathcal{C}^n/I$, all standard monomials are found. If all elements of T_{d+1}^n are divisible by elements of $A^\star$, then all elements of T_{d+k}^n for $k \geq 2$ are also divisible by elements of $A^\star$. This implies that no new linearly dependent monomials can be found anymore for any degree larger than d. □

We conclude our running example with the application of Lemma 3.1 to determine whether $d = 3$ is large enough.

Example 6. *The cardinality of B is $dim \mathcal{C}^n/I = 5$ and*

$$A^\star \;=\; \{x_1x_2, x_1^3, x_2^3\},\; T_2^4 = \{x_1^4, x_1^3x_2, x_1^2x_2^2, x_1x_2^3, x_2^4\}.$$

Every monomial of T_2^4 is divisible by an element of $A^\star$ since

$$x_1^4 = x_1\,(x_1^3), x_1^3x_2 = x_2\,(x_1^3), x_1^2x_2^2 = x_1x_2\,(x_1x_2), x_1x_2^3 = x_1\,(x_2^3), x_2^4 = x_2\,(x_2^3).$$

Lemma 3.1 is satisfied and $G = \{g_1, g_2, g_3\}$ is a reduced Gröbner basis.

Observe that the Gröbner basis G in (6) consists of 3 polynomials, while $n = 2$. The application described in Section 5 requires that the number of polynomials equals the number of variables n. This issue is addressed in Section 4 by extending the affine algorithm to the projective case.

3.2. Numerical algorithm and implementation

Our SVD-based numerical implementation of the affine Buchberger-Möller algorithm is presented in Algorithm 2. The MATLAB/Octave implementation in the freely downloadable PNLA package is abma.m. Since the first row of the Vandermonde matrix K is always linearly independent, the monomial 1 will always be an element of B. The algorithm then recursively checks monomials of increasing total degree for linear independence until the stop condition of Lemma 3.1 is satisfied. A graded monomial ordering is therefore assumed. The defining property of a Gröbner basis is taken into account each recursion by removing monomial multiples of the elements of $A^\star$. Checking whether a monomial is linearly independent and solving the linear system (5) for the unknown coefficients is achieved by one SVD. When not all m standard monomials have been found, a criterion is needed to decide whether the current monomial under investigation a is linearly independent with respect to all other elements of B. This is done by inspection of the singular values of $K([B, a], :)^T$, obtained from computing its SVD

$$K([B, a], :)^T \;= U\,\Sigma\,V^T \;=\; U \begin{pmatrix} \overline{\sigma} & & \\ & \ddots & \\ & & \underline{\sigma} \end{pmatrix} V^T,$$

where U, V are orthogonal matrices and Σ is a diagonal matrix containing the singular values $\overline{\sigma} \geq \ldots \geq \underline{\sigma}$. When the smallest singular value $\underline{\sigma}$ is smaller than a chosen tolerance τ, then a is determined to be linearly dependent. If the numerical values of the roots are known with infinite precision then the tolerance τ is set to $m\,\overline{\sigma}\,e$, where m is the number of columns of $K([B,a],:)$, $\overline{\sigma}$ is the largest singular value and e is the machine precision. For double precision we have that $e \approx 10^{-16}$. If the roots are only known up to a certain accuracy, then the tolerance τ needs to take this into account. We discuss how to choose the tolerance in this case in the next subsection. For each iteration in the for-loop, the tolerance τ is recomputed. When the monomial a is found to be linearly dependent, then the right singular vector $\underline{v}$ corresponding with the smallest singular value is the vector of coefficients that solves (5). The smallest singular value $\underline{\sigma}$ then also serves as a measure of how well the retrieved polynomial vanishes on the given fixed points. Indeed, writing $K([B,a],:)$ as the product of a row selection matrix S and K we have that

$$||\underline{v}^T\ K([B,a],:)||_2\ =\ ||\underline{v}^T\ S\ K||_2\ =\ ||\underline{\tilde{v}}^T\ K||_2\ =\ \underline{\sigma}.$$

The $\underline{\tilde{v}}^T \equiv \underline{v}^T\,S$ in the last equation is the desired coefficient vector of the reduced Gröbner basis polynomial.

Removing the monomial multiples of $A^\star$ limits the total number of SVDs to $m + n_{A^\star}$, where $n_{A^\star}$ is the cardinality of $A^\star$. Since the left singular vectors U do not need to be computed, the computational complexity of each SVD is upper bounded by $4m(m+1)^2 + 8(m+1)^3$ flops [10, p. 254]. In fact, once the m standard monomials are known during the algorithm, no decision on the linear dependence of the remaining monomials needs to be made anymore. The only remaining task is therefore the computation of the single null vector of $K([B,a],:)^T$. We have not taken this optimization into account in Algorithm 2.

Algorithm 2. *Numerical affine Buchberger-Möller algorithm (abma.m)*
Input*: set of affine roots $\mathcal{Z}$ with corresponding multiplicity structures, monomial ordering $<$*
Output*: $A^\star, B$ and reduced Gröbner basis G*

$A^\star, G \leftarrow \varnothing$
$B \leftarrow \{1\}$
$K \leftarrow$ *first row of Vandermonde matrix corresponding with monomial 1*
$d \leftarrow 1$

$X \leftarrow T_n^d$
while $X \neq \varnothing$ ***do***
  append K *with rows corresponding with monomials of* X
  for *all monomials in* X ***do***
    $a \leftarrow$ *smallest monomial in* X *according to monomial ordering* $<$
    $[U,\ S,\ V] \leftarrow SVD(K([B, a], :)^T)$
    $\tau \leftarrow m\,\overline{\sigma}\,e$ *or* (9)
    if $\underline{\sigma} < \tau$ ***then***
      append a *to* $A^\star$ *and remove it from* X
      append $\tilde{\underline{v}}^T$ *to* G
    else
      append a *to* B *and remove it from* X
    end if
  end for
  $d \leftarrow d + 1$
  $X \leftarrow T_d^n$
  remove all monomial multiples of $A^\star$ *from* X
end while

The matrix K is a generalization of a confluent Vandermonde matrix to the multivariate case. Vandermonde matrices appear in polynomial interpolation problems and can be very ill-conditioned. This ill-conditioning is due to the fact that a monomial basis was chosen to represent polynomials. We then have in the univariate case that the degree of the monomials in the Vandermonde matrix grows linearly with the number of points. This is however not the case when $n > 1$ and is due to the curse of dimensionality, which states that there are $\binom{d+n}{n}$ number of n-variate monomials of degrees 0 up to d. Indeed, supposing for the sake of argument that the first m monomials are the standard monomials, then the degree $\hat{d}$ that satisfies

$$\binom{\hat{d} - 1 + n}{n} \leq m < \binom{\hat{d} + n}{n}$$

will be much smaller than m. If we take for example $m = 100$ and $n = 8$, then $\hat{d} = 3$, since $\binom{2+8}{8} = 45 \leq 100 < \binom{3+8}{8} = 165$. The $K([B, a], :)$ matrix will therefore contain only monomials up to degree 3 while for the univariate case this would have been 100.

Example 7. *In this example we demonstrate the run times in seconds of Algorithm 2 for an increasing number of roots and number of variables. All*

computations were done in Matlab on a 64-bit 4-core 3.3 GHz desktop computer with 16 GB RAM. Figure 1 shows how the total run time increases as the number of roots m and number of variables n increases. The total run time is determined by the total number of Gröbner basis polynomials that need to be computed. Figure 1 shows that up until 20 roots a doubling of the number of variables results in a 10-fold increase in run time. Finding the reduced Gröbner basis for 41 roots and 20 variables takes about 15 minutes and computes 210 polynomials.

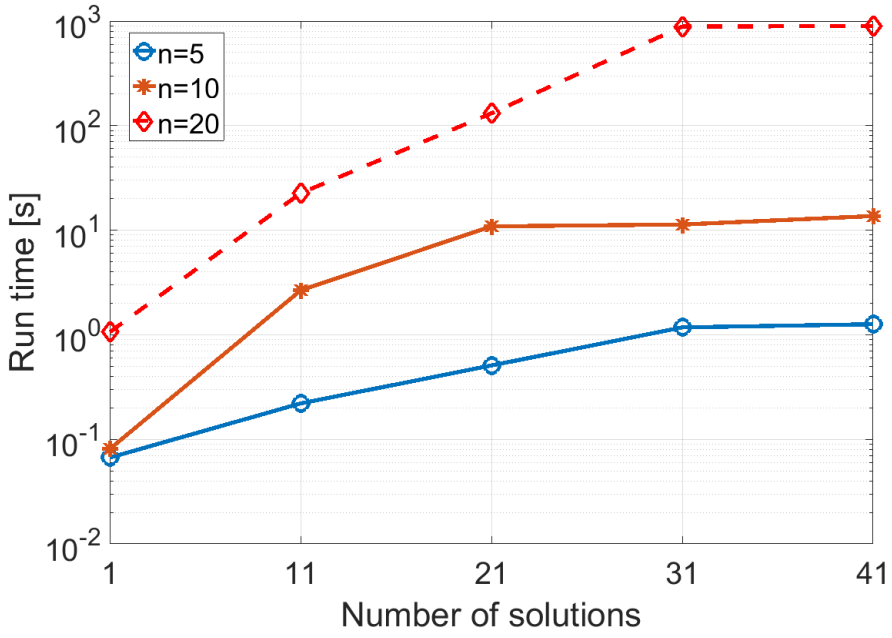

Figure 1: Run times of the abma.m algorithm as a function of the number of roots m.

3.3. Choosing a suitable tolerance τ

When the numerical values of the roots are not known with infinite precision, then the tolerance τ cannot be set to $m \,\overline{\sigma}\, e$. Knowledge on the error should be take into account instead. We will assume that each of the components z_i are perturbed by e_i and that an upper bound ϵ on the absolute error e_i is known such that

$$\hat{z}_i = z_i + e_i, \qquad \text{and} \qquad |\hat{z}_i - z_i| \leq \epsilon \quad \forall i = 1, \ldots, n.$$

Consequently, the matrix $K([B,a],:)$ is then also perturbed as

$$\hat{K}([B,a],:) = K([B,a],:) + E.$$

If σ and $\hat{\sigma}_i$ denote any singular value of $([B,a],:)$ and $\hat{K}([B,a],:)$ respectively, then Weyl's Theorem [24] states that

$$|\hat{\sigma}_i - \sigma_i| \leq ||E||_2.$$

Finding an approximation for $||E||_2$ is therefore key to choosing a good tolerance. If E_i denotes the i-th column of E, then

$$||E||_2 \leq \sqrt{m}\, \max_i ||E_i||_2. \tag{7}$$

Each row of K contains the evaluation of a monomial x^α in one of the roots. Interpreting a monomial as a multivariate function $f(z_1, \ldots, z_n)$, the linear approximation of $f(z_1 + \epsilon, \ldots, z_n + \epsilon)$ is then given by

$$f(z_1 + \epsilon, \ldots, z_n + \epsilon) \;\approx\; f(z_1, \ldots, z_n) + \nabla f(z_1, \ldots, z_n)\, \epsilon,$$

where ∇ denotes the vector differential operator. Each row r_i of the matrix E contains the entries $\nabla f(z_1, \ldots, z_n)\, \epsilon$, where $f(z_1, \ldots, z_n)$ is the monomial that corresponds with r_i. The column of E with the maximal 2-norm will correspond with the fixed point that has maximal 2-norm. Hence, if we denote $\mathbf{z} = \underset{\mathbf{z}_i \in \mathcal{Z}}{\operatorname{argmax}} ||\mathbf{z}_i||_2$, then

$$\max_i ||E_i||_2 \;=\; \epsilon \sqrt{\sum_{B,a} (\nabla f(\mathbf{z}))^2}, \tag{8}$$

where the summation goes over all standard monomials in B and the monomial a that is being checked for linear independence. Combining (7) and (8) we set the tolerance τ to

$$\tau \;=\; \epsilon \sqrt{m \sum_{B,a} (\nabla f(\mathbf{z}))^2}. \tag{9}$$

Example 8. *Suppose* $m = 9, \epsilon = 10^{-5}, B = \{1, x_1, x_1^2\}, a = x_1\, x_2^2$ *and* $\boldsymbol{z} = max_i\, \boldsymbol{z}_i = (-5, 3)$. *Application of* (8) *results in*

$$max_i\, ||E_i||_2 \;=\; 10^{-5}\sqrt{1^2 + (2 \cdot -5)^2 + (9 + 2 \cdot -5 \cdot 3)^2} = 2.3281 \times 10^{-4}.$$

The tolerance is then set to $\tau = \sqrt{9} \cdot 2.3281 \times 10^{-4} = 6.9843 \times 10^{-4}$.

Further application of this tolerance is demonstrated in Section 5.

4. Numerical projective Buchberger-Möller algorithm

In this section we will address the issue that for a given set of affine roots, Algorithm 2 will almost always determine a reduced Gröbner basis $g_1, \ldots, g_t$ with $t > n$. This can be problematic, as indicated by the application in Section 5 where only square polynomial systems are desired for which $t = n$. The solution in obtaining square polynomial systems lies in using homogeneous polynomials and projective coordinates and is stated in Bézout's theorem.

Theorem 4.1. *([14, p. 91]) Let $f_1^h, \ldots, f_n^h$ be a polynomial system of n homogeneous polynomials of degrees $d_1, \ldots, d_n$ with a finite number of roots over $\mathbb{P}^n$. Then the total number of roots of $f_1^h, \ldots, f_n^h$, counted with multiplicities, is $\prod_{i=1}^n d_i$.*

Hence, the solution to obtain n polynomials is rather straightforward. One needs to make sure the total number of projective roots is $\prod_{i=1}^n d_i$ and by Bézout's Theorem it is then guaranteed that a homogeneous polynomial system of n polynomials exists. There is however, as we will demonstrate, a sensitivity to the chosen multiplicity structure in the sense that only certain multiplicity structures will result in n polynomials. Observe that these n homogeneous polynomials will be in $n+1$ variables, but this is easily resolved by considering the corresponding affine polynomials by setting $x_0 = 1$.

There are two ways to obtain $d_1 \cdots d_n$ projective roots: either extend the given roots with additional multiplicities, or add multiple roots at infinity. Note that roots with multiplicities are special because they are the result of a careful configuration of parameters. A small perturbation of the coefficients of the polynomials $f_1, \ldots, f_n$ are guaranteed to destroy the multiplicity structure.

Only minimal adjustments of Algorithm 2 are required to make it work in the projective case. Again, we will revisit the running example of Section 3 and discuss it now in the projective setting.

Example 9. *We take the roots $\mathcal{Z} = \{(0,0),(3,5)\}$ of Example 3 and transform them into projective coordinates $\mathcal{Z}^h = \{(1,0,0),(1,3,5)\}$. Suppose we are looking for 2 polynomials g_1, g_2 with degrees $d_1 = d_2 = 2$, which sets the total number of projective roots to $d_1\, d_2 = 4$. Suppose we do not wish to increase the multiplicity of the affine roots. Adding 2 additional projective roots at infinity is then required. Hence $\mathcal{Z}^h = \{(1,0,0),(1,3,5),(0,1,0),(0,0,1)\}$ and since each of the projective roots has no multiplicities, the dual vector*

space is closed. Again we set $d = 3$ and construct the now homogeneous multivariate Vandermonde matrix K of differential functionals that evaluate homogeneous polynomials of degree 3

$$K = \begin{array}{c} \\ x_0^3 \\ x_0^2x_1 \\ x_0^2x_2 \\ x_0x_1^2 \\ x_0x_1x_2 \\ x_0x_2^2 \\ x_1^3 \\ x_1^2x_2 \\ x_1x_2^2 \\ x_2^3 \end{array} \begin{array}{c} \begin{array}{cccc} \partial_{000}|_{(1,0,0)} & \partial_{000}|_{(1,3,5)} & \partial_{000}|_{(0,1,0)} & \partial_{000}|_{(0,0,1)} \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 5 & 0 & 0 \\ 0 & 9 & 0 & 0 \\ 0 & 15 & 0 & 0 \\ 0 & 25 & 0 & 0 \\ 0 & 27 & 1 & 0 \\ 0 & 45 & 0 & 0 \\ 0 & 75 & 0 & 0 \\ 0 & 125 & 0 & 1 \end{pmatrix} \end{array}.$$

A first important observation is that the projective Vandermonde matrix K in Example 9 has the same number of rows as in the affine case. This is because the number of $n+1$-variate homogeneous monomials of degree d is also given by $\binom{d+n}{n}$. Indeed, going from the projective case to the affine case or vice-versa is simply a relabeling of the monomials. Another important observation is that roots at infinity give rise to linearly independent rows that are located at the bottom of K. In Example 9 these linearly independent rows correspond with the monomials x_1^3 and x_2^3. Note that these standard monomials corresponding with roots at infinity will be the monomials that will have a degree d after setting $x_0 = 1$.

Our projective Buchberger-Möller algorithm will also recursively check monomials for linear independence for increasing total degrees. There is however a subtle difference due to the difference between the affine and projective definitions of the Hilbert Function. This difference implies that in the projective case the linearly independent monomials in B need to be recomputed for each degree d. The stopping condition of Lemma 3.1 therefore does not apply to the projective case. In order to formulate the new stopping condition we need to introduce the concept of connected monomials.

Definition 4.1. *([7, p. 348]) Let $B \subset T_d^{n+1}$. Two monomials b, b' are connected in B if there is a sequence of monomials $b_0, b_1, \ldots, b_r \in B$ with $b_0 = b$ and $b_r = b'$ such that for each $i = 1, \ldots, r$ there exists $\alpha, \beta \in \{0, \ldots, n\}$ satisfying $b_i = b_{i-1} \cdot x_\alpha / x_\beta$.*

Or in other words, two monomials in B are connected if one can pass from one b_i to the next by replacing one variable by another. We then call the connected component of a monomial $b \in B$ the set of all monomials in B that are connected with b. The projective stopping condition can now be formulated completely in terms of the connectivity of monomials in the set of linearly independent monomials.

Lemma 4.1. *([7, p. 349]) Let B be the monomial set of linearly independent monomials obtained from running Algorithm 1 on the homogeneous K for degree d. Then d is large enough if*

1. *the cardinality of B is m, the total number of projective roots including multiplicities and*
2. *for all i, every monomial in the connected components of x_i^d in B is divisible by x_i.*

Another stopping condition is formulated in terms of computing Hilbert Series and can be found in [17, p. 401]. The whole SVD-based projective Buchberger-Möller algorithm is presented in Algorithm 3 and implemented in the PNLA package as pbma.m. The same remarks as made for Algorithm 2 on the tolerance, accuracy and computational complexity apply here as well. The total number of SVD's is larger compared to the affine algorithm due to the fact that the standard monomials B are recomputed for each degree.

Algorithm 3. *Numerical projective Buchberger-Möller algorithm (pbma.m)*
***Input**: set of projective fixed points $\mathcal{Z}$ with corresponding multiplicity structures, monomial ordering $<$*
***Output**: $A^\star, B$ and reduced Gröbner basis G*

$A^\star, G \leftarrow \varnothing$
$d \leftarrow 0$
while** Lemma 4.1 not satisfied **do
  $d \leftarrow d + 1$
  $B = \varnothing$
  $X \leftarrow T_d^{n+1}$
  remove all monomial multiples of $A^\star$ from X
  construct K with rows corresponding with monomials of X
  for** all monomials in X **do

$a \leftarrow$ *smallest monomial in* X *according to monomial ordering* $>$
$[U,\ S,\ V] \leftarrow SVD(K([B, a], :)^T)$
$\tau \leftarrow m\,\overline{\sigma}\,e$ *or* (9)
if $\underline{\sigma} < \tau$ ***then***
append a *to* $A^\star$ *and remove it from* X
append $\tilde{\underline{v}}^T$ *to* G
else
append a *to* B *and remove it from* X
end if
end for
end while

We revisit our running example and apply Algorithm 3.

Example 10. *At* $d = 2$*, 2 homogeneous polynomials are found and Algorithm 3 stops at* $d = 3$*, where a third polynomial is computed. The resulting reduced Gröbner basis is*

$$G \;=\; \begin{cases} g_1 &=& -0.8575\,x_0\,x_1 + 0.5145\,x_0\,x_2, \\ g_2 &=& -0.9806\,x_0\,x_1 + 0.1961\,x_1\,x_2, \\ g_3 &=& -0.9487\,x_0^2\,x_1 + 0.3162\,x_0\,x_1^2. \end{cases}$$

The obtained reduced Gröbner basis consists of 3 polynomials while only 2 were expected. This set of generators however is not minimal. In order for a set of generators $g_1, \ldots, g_t$ of degrees $d_1 \leq d_2 \leq \ldots \leq d_t$ to be minimal, the condition that $g_i \notin \langle g_1, \ldots, g_{i-1}\rangle$ for all $i = 1, \ldots, t$ needs to be satisfied. Checking whether $g_i \notin \langle g_1, \ldots, g_{i-1}\rangle$ is in essence also a rank test. Indeed, let M be the matrix with row space $\langle g_1, \ldots, g_{i-1}\rangle_{d_i}$, then if

$$\text{rank} \begin{pmatrix} M \\ g_i \end{pmatrix} \;=\; \text{rank}\,(M) \tag{10}$$

we also have that $g_i \in \langle g_1, \ldots, g_{i-1}\rangle$. One way to check whether (10) applies would be to bring the matrix $\begin{pmatrix} M^T & g_i^T \end{pmatrix}$ into reduced row echelon form using Gaussian elimination. If the column g_i is reduced to zero then the rank of M does not increase and the equality applies. The SVD however is a more robust method to determine the numerical rank and therefore it is our method of choice. The tolerance for this rank test needs to be chosen the same as the tolerance in Algorithm 2 or 3. More details on how to construct

the (Macaulay) matrix M and check whether $g_i \notin \langle g_1, \ldots, g_{i-1} \rangle$ applies can be found in [4]. We now apply this SVD-based test to g_3 of our running example to check whether indeed the obtained Gröbner basis is not minimal.

Example 11. *Since $d_1 = d_2 = 2$, the rows of the 6×20 matrix M are the coefficient vectors of $x_i\, g_1, x_i\, g_2$ $(i = 0, \ldots, 2)$, with a smallest singular value $\underline{\sigma} = 0.2254$. Appending the coefficient vector of g_3 and recomputing the singular values results in $\underline{\sigma} = 2.11 \times 10^{-17}$, which indicates that g_3 lies in $\langle g_1, g_2 \rangle$. We can therefore delete g_3 from G. Setting $x_0 = 1$ results in the final polynomial system*

$$G = \begin{cases} g_1 &= -0.8575\, x_1 + 0.5145\, x_2, \\ g_2 &= -0.9806\, x_1 + 0.1961\, x_1\, x_2. \end{cases} \tag{11}$$

Observe that $d_1 = 1$ after setting $x_0 = 1$, while originally we choose the roots at infinity so that $d_1 = d_2 = 2$ would apply. This does not however pose any problem since we know that any pair of polynomials g_1', g_2' that are linear combinations of g_1, g_2 lie in the ideal $\langle g_1, g_2 \rangle$. We will further discuss this in the next section.

5. Application

In this section we will discuss the application of our proposed algorithms to the problem of computing a square polynomial dynamical system for a given set of fixed points. Polynomial dynamical systems are ubiquitous throughout all fields of science and engineering. Examples of such systems can be found in gene-regulatory networks [25, 26], ecology [27], economics [28], neural physiology [29, 30], chemistry [31], atmospheric science [32] and many others. These systems are described by n state variables $x_i \in \mathbb{C}$ and each of their dynamics are described by a multivariate polynomial $f_i(\mathbf{x}) \in \mathcal{C}^n$. The state dynamics can then be written as

$$\begin{cases} \dot{x}_1 &= f_1(\mathbf{x}), \\ \vdots & \quad \vdots \\ \dot{x}_n &= f_n(\mathbf{x}). \end{cases} \tag{12}$$

Observe that at the roots we have that $\dot{x}_1 = \dot{x}_2 = \cdots = \dot{x}_n = 0$. For this reason the roots of the polynomial system are called the fixed points of the dynamical system. In electrical engineering, these points are also commonly

known as DC operating points. We will now use Algorithms 2 and 3 to numerically compute for a given set of fixed points $\mathcal{Z}$, the smallest set of n polynomials that vanish on $\mathcal{Z}$. Or in other words, to check whether a square polynomial dynamical system in the form of (12) exists for a given set of fixed points $\mathcal{Z}$. All experiments were run in Matlab on a dual-core 1.66 GHz laptop with 2 GB RAM and took less than 0.1 seconds to run.

5.1. Lotka-Volterra systems

Suppose we want to determine a dynamical system with 2 affine fixed points $\mathcal{Z} = \{(0,0),(3,5)\}$ and described by 2 polynomials of degree 2. What is the most general form of such a dynamical system that satisfies these conditions? As demonstrated in Section 4, 2 fixed points at infinity need to be added to $\mathcal{Z}$ to make this possible. The whole set of second degree polynomials that have these fixed points is described by

(13)
$$\begin{cases} g_1' &= a_{11}\,(-0.8575\,x_1 + 0.5145\,x_2) + a_{12}\,(-0.9806\,x_1 + 0.1961\,x_1\,x_2), \\ g_2' &= a_{21}\,(-0.8575\,x_1 + 0.5145\,x_2) + a_{22}\,(-0.9806\,x_1 + 0.1961\,x_1\,x_2). \end{cases}$$

An additional constraint on the coefficients a_{ij} is required such that the matrix $A = (a_{ij})$ is regular and that $a_{12} \neq 0, a_{22} \neq 0$. Indeed, the regularity of A is required in order for g_1', g_2' to still be a set of generators for the polynomial ideal and the condition $a_{12} \neq 0, a_{22} \neq 0$ ensures that both g_1' and g_2' are of degree 2.

Also observe that the choice of the a_{ij} coefficients completely determines the stability of the affine fixed points. For example, the stability of $(0,0)$ is determined by the eigenvalues of the following Jacobian J

$$J \;=\; \begin{pmatrix} -a_{11}\,0.8575 - a_{12}\,0.9806 & a_{11}\,0.5145 \\ -a_{21}\,0.8575 - a_{22}\,0.9806 & a_{21}\,0.5145 \end{pmatrix}.$$

If Δ and γ denote the determinant and trace respectively of J, then all dynamical systems where $(0,0)$ is a saddle point are described by the set of coefficients a_{ij} such that $\Delta < 0$ is satisfied. Likewise, all dynamical systems with a stable fixed point $(0,0)$ are described by the set of coefficients a_{ij} such that both $\Delta > 0$ and $\gamma < 0$ are satisfied. One particular choice of $a_{11} = 0, a_{12} = -10.12, a_{21} = -5.831, a_{22} = 5.099$ leads to the following dynamical system

$$\begin{cases} \dot{x}_1 &= 10\,x_1 - 2\,x_1\,x_2, \\ \dot{x}_2 &= -3\,x_2 + x_1\,x_2, \end{cases}$$

which is the well-known Lotka-Volterra system. Observe from (13) that only the monomials $x_1, x_2, x_1\,x_2$ can appear in any of the dynamical systems. The given set of fixed points with their multiplicity structures hence strongly dictate what monomials are allowed to be present in the state equations. This is easily demonstrated by applying Algorithm 3 to $\mathcal{Z} = \{(1,0,0),(1,3,5),(0,1,0)\}$ with multiplicity structures

$$\partial_{000}|_{(1,0,0)},\ \partial_{000}|_{(1,3,5)},\ \partial_{000}|_{(0,1,0)},\ \partial_{100}|_{(0,1,0)}.$$

The minimal set of polynomials is then

$$\begin{cases} g_1 &= -0.9487\,x_0\,x_1 + 0.3162\,x_1\,x_2, \\ g_2 &= -0.9806\,x_0\,x_2 + 0.1961\,x_2^2, \\ g_3 &= -0.8575\,x_0^2\,x_1 + 0.5145\,x_0^2\,x_2. \end{cases}$$

Since the number of generators is 3, this multiplicity structure does not allow a polynomial dynamical system realization. Also note that the $x_1\,x_2$ monomial is replaced by x_2^2. Changing the multiplicity structure of $(0,1,0)$ to $\partial_{000}, \partial_{001}$ does lead to 2 polynomials in the monomials x_1, x_2, x_2^2 after setting $x_0 = 1$. Similarly, adding $(0,0,1)$ with multiplicity structure $\partial_{000}, \partial_{100}$ results in minimal 3 generators where the $x_1\,x_2$ monomial is now replaced by x_1^2 and changing the multiplicity structure of the fixed point at infinity to $\partial_{000}, \partial_{010}$ results in 2 polynomials in the monomials x_1, x_2, x_1^2 after setting $x_0 = 1$.

5.2. Van der Pol oscillator

Next we demonstrate the use of multiplicities in order to obtain a polynomial system of specific degrees. For this particular example we set $n = 2$ and $d_1 = 1, d_2 = 3$ and aim at modeling an oscillator. A simple oscillatory system has one affine fixed point at the origin that undergoes a Hopf bifurcation when a particular parameter is changed, resulting in periodic solutions. Modeling such a system therefore means that we are interested in a polynomial dynamical system with one affine fixed point $\mathcal{Z} = \{(0,0)\}$. In order to obtain a polynomial of degree three we add the fixed point at infinity $(0,1,0)$ to $\mathcal{Z}$ with a multiplicity of 2 and multiplicity structure $\partial_{000}, \partial_{100}$. Running Algorithm 3 results in the very simple reduced Gröbner basis $g_1 = x_2, g_2 = x_0^2 x_1$. The only allowable first-degree polynomial is $a\,g_1$ while the remaining polynomial of degree 3 is described by $(c_0\,x_0^2 + c_1\,x_0 x_1 + c_2\,x_0 x_2 + c_3\,x_1^2 + c_4\,x_1 x_2 + c_5\,x_2^2) g_1 + b\,g_2$.

Choosing $a = 1, b = -1, c_0 = \mu, c_3 = -\mu$ and setting the remaining coefficients to zero together with $x_0 = 1$ results in the system

$$\begin{cases} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= \mu\,(1 - x_1^2)\,x_2 - x_1, \end{cases}$$

which is the well-known Van der Pol oscillator.

5.3. FitzHugh-Nagumo systems

The projective algorithm is not always necessary. For the set of fixed points $\mathcal{Z} = \{(0, -1), (\sqrt{2}, -\sqrt{2} - 1), (-\sqrt{2}, \sqrt{2} - 1)\}$ it is possible to retrieve a set of 2 polynomials g_1, g_2 of degrees $1, 3$, respectively. Algorithm 2 returns

$$\begin{cases} g_1 &= 0.5774 - 0.5774\,x_1 - 0.5774\,x_2, \\ g_2 &= -0.8944\,x_1 + 0.4474\,x_1^3. \end{cases}$$

The whole set of polynomials that vanish on $\mathcal{Z}$ is given by

$$\begin{cases} g_1' &= a_1\,g_1, \\ g_2' &= (a_{11} + a_{12}\,x_1 + a_{13}\,x_2)\,g_1 + a_{14}\,g_3, \end{cases}$$

where $a_1 \neq 0$ and $a_{14} \neq 0$. Choosing $a_1 = -1/0.5774, a_{11} = 1.7321, a_{14} = -2.2361$ results in the FitzHugh-Nagumo system [29, 30]

$$\begin{cases} \dot{x}_1 &= x_1 - x_1^3 - x_2 + 1, \\ \dot{x}_2 &= x_1 - 1 + x_2, \end{cases}$$

with unity external current. Note that for a set of 3 randomly chosen fixed points it would never be possible to obtain a set of 2 polynomials of degrees 1 and 3. Indeed, the 3×3 Vandermonde matrix K for $d = 1$ would be of full rank and hence no linear polynomial can be determined. It is only because there exists a linear relation between the x_1 and x_2 components of 2 of the fixed points in $\mathcal{Z}$ that K is rank-deficient for $d = 1$ and g_1 can be determined.

Suppose now that we perturb $\mathcal{Z}$ with normal distributed noise over the interval $[-10^{-5}, 10^{-5}]$. The exact linear relationship between x_1 and x_2 is then only approximate. The absolute errors are upper bounded by $\epsilon = 2 \times 10^{-6}$ for this particular example. At $d = 1$ during the execution of Algorithm 2 and when $B = \{1, x_1\}, a = x_2$ we have that $\tau = 4.000 \times 10^{-6}$ and

$\underline{\sigma} = 4.722 \times 10^{-7}$. The algorithm therefore correctly detects the approximate linear dependency of x_2 with respect to $1, x_1$. The algorithm stops at $d = 3$ and returns the following reduced Gröbner basis

$$\begin{cases} \hat{g}_1 &= 0.5774 - 0.5773\, x_1 - 0.5774\, x_2, \\ \hat{g}_2 &= -2.8244 \times 10^{-7} - 0.8944\, x_1 + 4.1284 \times 10^{-7}\, x_1^2 + 0.4472 x_1^3. \end{cases}$$

The noise has introduced some extra terms to $\hat{g}_2$. Since these extra terms have coefficients that are smaller than $\tau = 7.746 \times 10^{-6}$ at $d = 3$, they can be considered to be numerically zero. We have that $||g_1 - \hat{g}_1||_2 = 6.034 \times 10^{-7}$ and $||g_1 - \hat{g}_1||_2 = 4.142 \times 10^{-7}$.

6. Conclusions

In this article, the inverse problem of polynomial root-finding was solved for the case of a finite number of roots. Two SVD-based algorithms were presented that produce a minimal set of polynomials that vanish on a given set of roots. It was shown how by adding additional roots at infinity it is possible to determine a square system of n polynomials in n variables. The tolerance that needs to be used when the given roots are only known with a finite accuracy was derived. It was also shown how the smallest singular value is a measure of how well the resulting polynomials vanish on the given roots. The application of both algorithms on the problem of realizing a polynomial dynamical system from a set of given fixed points was illustrated by means of examples.

References

[1] W. Auzinger, H. J. Stetter, An elimination algorithm for the computation of all zeros of a system of multivariate polynomial equations, in: Int. Conf. on Numerical Mathematics, Singapore, 1988, pp. 11–30.

[2] H. M. Moller, H. J. Stetter, Multivariate Polynomial Equations with Multiple Zeros solved by Matrix Eigenproblems, Numer. Math. 70 (3) (1995) 311–329.

[3] H. J. Stetter, Numerical Polynomial Algebra, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2004.

[4] K. Batselier, P. Dreesen, B. De Moor, The Canonical Decomposition of $\mathcal{C}_d^n$ and Numerical Gröbner and Border Bases, SIAM J. Matrix Anal. Appl. 35 (4) (2014) 1242–1264.

[5] H. Möller, B. Buchberger, The construction of multivariate polynomials with preassigned zeros, in: J. Calmet (Ed.), Computer Algebra, Vol. 144 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 1982, pp. 24–31.

[6] CoCoATeam, CoCoa: a system for doing Computations in Commutative Algebra, Available at `http://cocoa.dima.unige.it`.

[7] J. Abbott, A. Bigatti, M. Kreuzer, L. Robbiano, Computing ideals of points, J. Symb. Comput. 30 (4) (2000) 341 – 356.

[8] D. J. Bates, J. D. Hauenstein, T. M. McCoy, C. Peterson, A. J. Sommese, Recovering exact results from inexact numerical data in algebraic geometry, Experimental Mathematics 22 (1) (2013) 38–50.

[9] D. Heldt, M. Kreuzer, S. Pokutta, H. Poulisse, Approximate Computation of Zero-dimensional Polynomial Ideals, J. Symb. Comput. 44 (11) (2009) 1566–1591.

[10] G. H. Golub, C. F. Van Loan, Matrix Computations, 3rd Edition, The Johns Hopkins University Press, 1996.

[11] M. R2012a, The Mathworks Inc., natick, Massachusetts (2012).

[12] J. W. Eaton, D. Bateman, S. Hauberg, GNU Octave Manual Version 3, Network Theory Ltd., 2008.

[13] K. Batselier, N. Wong, Computing the state recursion polynomials for discrete linear mD systems, Automatica 64 (2016) 254–261.

[14] D. A. Cox, J. B. Little, D. O'Shea, Using Algebraic Geometry, Graduate Texts in Mathematics, Springer-Verlag, 2005.

[15] D. A. Cox, J. B. Little, D. O'Shea, Ideals, Varieties and Algorithms, 3rd Edition, Springer-Verlag, 2007.

[16] M. Kreuzer, L. Robbiano, Computational Commutative Algebra 1, Computational Commutative Algebra, Springer, 2000.

[17] M. Kreuzer, L. Robbiano, Computational Commutative Algebra 2, Springer, 2005.

[18] K. Batselier, P. Dreesen, B. De Moor, The Geometry of Multivariate Polynomial Division and Elimination, SIAM J. Matrix Anal. Appl. 34 (1) (2013) 102–125.

[19] P. R. Halmos, Finite-Dimensional Vector Spaces, Undergraduate Texts in Mathematics, Springer, 1974.

[20] B. H. Dayton, Z. Zeng, Computing the multiplicity structure in solving polynomial systems, in: Proc. of ISSAC '05, ACM Press, 2005, pp. 116–123.

[21] W. Hao, A. J. Sommese, Z. Zeng, Algorithm 931: An Algorithm and Software for Computing Multiplicity Structures at Zeros of Nonlinear Systems, ACM Trans. Math. Softw. 40 (1) (2013) 5:1–5:16.

[22] Z. Zeng, The closedness subspace method for computing the multiplicity structure of a polynomial system, in: Interactions of Classical and Numerical Algebraic, 2009.

[23] D. Bates, C. Peterson, A. Sommese, A numerical-symbolic algorithm for computing the multiplicity of a component of an algebraic set, J. Complexity 22 (4) (2006) 475–489.

[24] G. W. Stewart, Perturbation Theory for the Singular Value Decomposition, in: SVD and Signal Processing, II: Algorithms, Analysis and Applications, Elsevier, 1990, pp. 99–109.

[25] R. Laubenbacher, B. Stigler, A computational algebra approach to the reverse engineering of gene regulatory networks, J. Theor. Biol. 229 (4) (2004) 523–537.

[26] B. Stigler, Polynomial dynamical systems in systems biology, Proceedings of Symposia in Applied Mathematics 64 (2007) 53.

[27] A. A. Berryman, The Origins and Evolution of Predator-Prey Theory, Ecology 73 (5) (1992) 1530–1535.

[28] R. M. Goodwin, A growth cycle, Socialism, capitalism and economic growth (1967) 54–58.

[29] R. FitzHugh, Impulses and physiological states in theoretical models of nerve membrane, Biophys J 1 (1961) 445–466.

[30] J. S. Nagumo, S. Arimato, S. Yoshizawa, An Active Pulse Transmission Line Simulating a Nerve Axon, Proceedings of IRE 50 (1962) 2061–2070.

[31] R. I. Epstein, J. A. Pojman, An Introduction to Nonlinear Chemical Dynamics : Oscillations, Waves, Patterns, and Chaos: Oscillations, Waves, Patterns, and Chaos, Topics in Physical Chemistry, Oxford University Press, USA, 1998.

[32] E. N. Lorenz, Deterministic Nonperiodic Flow, J. Atmos. Sci. 20 (2) (1963) 130–141.