Journal of Computational and Applied Mathematics 376 (2020) 112825

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

A new deflation method for verifying the isolated singular zeros of polynomial systems

Jin-San Cheng [a,c,*], Xiaojie Dou [b,*], Junyi Wen [a,c]

[a] KLMM, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, 100190, Beijing, China
[b] College of Science, Civil Aviation University of China, 300300, Tianjin, China
[c] University of Chinese Academy of Sciences, Beijing, China

ARTICLE INFO

Article history:
Received 2 June 2019
Received in revised form 2 December 2019

Keywords:
Polynomial system
Deflation method
Isolated singular zero
Interval verification

ABSTRACT

In this paper, we develop a new deflation technique for refining or verifying the isolated singular zeros of polynomial systems. Starting from a polynomial system with an isolated singular zero, by computing the derivatives of the input polynomials directly or the linear combinations of the related polynomials, we construct a new system, which can be used to refine or verify the isolated singular zero of the input system. In order to preserve the accuracy in numerical computation as much as possible, new variables are introduced to represent the coefficients of the linear combinations of the related polynomials. To our knowledge, it is the first time that considering the deflation problem of polynomial systems from the perspective of linear combinations. Some acceleration strategies are proposed to reduce the scale of the final system. We also give some further analysis of the tolerances we use, which can help us have a better understanding of our method. The experiments show that our method is effective and efficient. Especially, it works well for zeros with high multiplicities of large systems. It also works for isolated singular zeros of non-polynomial systems.

© 2020 Elsevier B.V. All rights reserved.

1. Introduction

Solving polynomial systems with singular zeros is always a challenge in algebraic and geometric computation. For an isolated simple zero of a polynomial system, the classical Newton's method is widely used and quadratic convergent. However, for singular zeros of a polynomial system, Newton's method is not fit for the original system directly because it converges slowly or even does not converge in a bad situation. What is more, it is an ill-posed problem to compute an isolated singular zero of a polynomial system or a nonlinear system, since a small perturbation of coefficients may transform an isolated singular zero into a cluster of simple zeros.

Therefore, finding methods to keep the quadratic convergence of Newton's method for singular zeros is a way to handle this problem. Given a polynomial system with an isolated singular zero, we can construct a new system owing the same singular zero as an isolated simple one. Based on this idea, in recent years, there are many symbolic or symbolic-numerical methods coming up to deal with this problem. The basic idea is the deflation techniques [1–8], which usually have two basic strategies: adding new equations only or both new equations and new variables to the original system.

Deflation for an isolated singular solution originated from the ideas of Ojika [9–11]. T. Ojika et al. present a deflation algorithm for determining the multiple zeros for a system of nonlinear equations. Through triangulating the Jacobian

[*] Corresponding author.
E-mail addresses: jcheng@amss.ac.cn (J.-S. Cheng), xjdou@cauc.edu.cn (X. Dou), wenjunyi15@mails.ucas.ac.cn (J. Wen).

https://doi.org/10.1016/j.cam.2020.112825
0377-0427/© 2020 Elsevier B.V. All rights reserved.

matrix of the original system at an approximate zero, new equations, which comes from the minors of the Jacobian matrix, are introduced to the original system to reduce the multiplicity until they get a system which is regular at the singular zero.

In [12], Giusti and Yakoubsohn propose a construction, which is based on two operations: deflating and kerneling, to determine a regular system without adding new variables. In the deflating, all the partial derivatives of the polynomials, which are zero at the multiple zero, are introduced to replace the corresponding polynomials. The kerneling operation consists of adding the polynomials given by the nonzero numerators of the coefficients of the Schur complement of the Jacobian matrix of the original system to the original system.

In [6], Hauenstein and Wampler define a strong deflation by only adding new equations coming from the one order differential of the Jacobian matrix of the original system to the original system. Different from [12], at each iteration step, both the number and the degree of the added equations are reduced.

In [13], Mourrain et al. give a method which uses a single linear differential form defined from the Jacobian matrix of the input system, and defines the deflated system by applying this differential form to the original system.

These above methods do introduce new equations and finally get a new system owing the isolated singular zero of the original system as a simple zero. In order to get the new polynomials, one needs to compute the determinant of some polynomial matrices. Thus the degree of the polynomials in the new system may be very high.

In the following, denote n as the number of both the variables and the equations in the original system, and μ as the multiplicity of the isolated singular zero of the original system.

In [14], Yamamoto introduces new equations and new variables to the original system simultaneously. New variables are used to bring some perturbations of the original system and the Jacobian matrix of the original system, which produce new equations.

In [15–17], Leykin et al. present an effective modification of Newton's method to restore quadratic convergence for isolated singular solutions of polynomial systems. Different from [14], new variables are only introduced to the Jacobian matrix of the original system, which produce new equations. Meanwhile, they also prove that the number of deflation stages is bounded by μ.

In [18], Dayton and Zeng modify the method in [15] and further prove that the number of deflation steps is bounded by the depth of the dual space. For the special case of breadth one, they also propose a modified deflation method, which is based on duality analysis, to reduce the final size $2^{\mu-1}n \times 2^{\mu-1}n$ of deflated system in [15] to $\mu n \times \mu n$.

In [19], by introducing a smoothing parameter to the original system and $n-1$ new variables to the Jacobian matrix of the original system, which produces new equations, Rump and Graillat consider the case of the double zero of the original system. In [20], based on the parameterized multiplicity structure, Li and Zhi generalize the algorithm in [19] to deflate the breadth-one isolated singular zero of the original system. Their final deflated regular system is of size $\mu n \times \mu n$.

In [21], based on the given multiplicity structure of the original system, which depends on the accuracy of the given approximate multiple zero, Mantzaflaris and Mourrain give a method to find a (small) perturbed system of the original system and then first compute a deflated system in one deflation step. The size of the final deflated system is equal to $\mu n \times \mu n$.

In [22], by lifting the independent perturbations in the first-order differential system appearing in [14] back to the original system, Li and Zhi modify the method in [14] and also prove that the modified deflation technique terminates after a finite number of steps bounded by the depth of the dual space. The size of the final modified regularized system is bounded by $2^{\mu-1}n \times 2^{\mu-1}n$.

In [13], by introducing some variables to represent the coefficients of the dual basis, Mourrain et al. give a method to deflate the original system and determine the multiplicity structure simultaneously. They also show that the number of variables and equations in this method is bounded by $n + n\mu(\mu-1)/2$ and $n\mu + n(n-1)(\mu-1)(\mu-2)/4$. However, one point worth noting is that this method needs to know the monomial basis of the original system first.

These methods introduce new variables and new equations to the original system simultaneously. By repeatedly using these deflation constructions, they will get an augmented system finally, which has an isolated simple zero, whose partial projection corresponds to the isolated singular zero of the original system.

Main contributions. In this paper, given a polynomial system $\mathbf{F} \subset \mathbb{C}[\mathbf{x}]$ with an isolated singular zero $\mathbf{p}$, by computing the derivatives of the input polynomials directly or the linear combinations of the related polynomials, we propose a new deflation method to construct a final deflated system $\widetilde{\mathbf{F}}'(\mathbf{x}, \boldsymbol{\alpha})$, which has an isolated simple zero $(\mathbf{p}, \hat{\boldsymbol{\alpha}})$, whose projection corresponds to the isolated singular zero $\mathbf{p}$ of the input system. New variables $\boldsymbol{\alpha}$ are introduced to represent the coefficients of the linear combinations of the related polynomials to ensure the accuracy of the numerical implementation. Moreover, we also prove that the size of our deflation system depends on the depth or the multiplicity of $\mathbf{p}$.

Compared to the previous methods, our method has the following differences:

1. For the input system $\mathbf{F}$, we can, if needed, compute the derivatives of every f_i to get the needed polynomials, which are regular at $\mathbf{p}$ at the beginning. Then, we put all these polynomials together to construct a system $\mathbf{F}_0$, such that the rank r of its Jacobian matrix at $\mathbf{p}$ is maximal. In some cases, we have $r = n$, which means that we need not introduce new variables.
2. We compute the derivatives of the linear combinations of the related polynomials to get some polynomials which are regular at $\mathbf{p}$. Here we introduce new variables to represent the coefficients of the linear combinations.

3. Considering that we know only the approximate zero $\tilde{\mathbf{p}}$ in actual computations, we use a tolerance θ to judge if a polynomial is θ-regular or θ-singular at $\tilde{\mathbf{p}}$ and another tolerance ε to judge the numerical rank of the Jacobian matrix. As long as the tolerance θ is chosen properly, we will get the same judgement in numerical case as in the exact case. Thus, our deflation system usually has the same isolated zero as the input system. Inspired by the work [15] of Leykin et al., we also give some further analysis on the tolerances θ and ε, which tells us that our final system is a perturbed system with a bounded perturbation in the worst case. To make our final system as accurate as possible, we also analyse the case that the tolerance θ is not introduced.

Thanks to the above acceleration strategies, the size of the final system in our actual computations is much less than that we give in theory.

Furthermore, we implement our method in Matlab. The experiments show that our method is effective and efficient, especially for large systems with singular zeros of high multiplicities. Besides, for the non-polynomial systems, our method is also applicable.

The paper is organized as below. We introduce some notations and preliminaries in Section 2. In Section 3, we give a new deflation idea to construct a deflated system from the input system with an isolated singular zero. Some analysis of the tolerances we use is given in Section 4. Numerical experiment results are given to demonstrate the performance of our algorithm in Section 5 and at last, we draw some conclusions in Section 6.

2. Notations and preliminaries

Let $\mathbb{C}$ be the complex field and $\mathbb{C}[\mathbf{x}] = \mathbb{C}[x_1, \ldots, x_n]$ be the polynomial ring. Denote $\mathbf{F} = \{f_1, f_2, \ldots, f_n\} \subset \mathbb{C}[\mathbf{x}]$ as a polynomial system and $\deg(f_i)$ as the degree of the polynomial f_i. Similarly, $\deg(\mathbf{F}) = \max\limits_{f_i \in \mathbf{F}} \deg(f_i)$. Let $\mathbf{p} = (p_1, \ldots, p_n) \in \mathbb{C}^n$. $\mathbf{F}(\mathbf{p}) = \mathbf{0}$ denotes that $\mathbf{p}$ is a zero of $\mathbf{F}(\mathbf{x}) = \mathbf{0}$.

Let $\mathbb{V}(\mathbf{F}) \subset \mathbb{C}^n$ denote the variety defined by $\mathbf{F}$ and $\dim \mathbb{V}(\mathbf{F})$ denote the dimension of $\mathbb{V}(\mathbf{F})$.

Let $\mathbf{d}_{\mathbf{x}}^{\boldsymbol{\gamma}} : \mathbb{C}[\mathbf{x}] \to \mathbb{C}[\mathbf{x}]$ denote the differential functional defined by

$$\mathbf{d}_{\mathbf{x}}^{\boldsymbol{\gamma}}(f) = \frac{1}{\gamma_1! \cdots \gamma_n!} \cdot \frac{\partial^{|\boldsymbol{\gamma}|} f}{\partial x_1^{\gamma_1} \cdots \partial x_n^{\gamma_n}}, \qquad \forall f \in \mathbb{C}[\mathbf{x}],$$

where $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_n) \in \mathbb{N}^n$ with $\mathbb{N} = \{0, 1, 2, \ldots\}$ and $|\boldsymbol{\gamma}| = \sum_{i=1}^{n} \gamma_i$.

Denote $\operatorname{rank}(A)$ as the rank of a matrix A. Denote $\mathbf{J}(\mathbf{F})$ as the Jacobian matrix of $\mathbf{F}$. That is,

$$\mathbf{J}(\mathbf{F}) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{pmatrix}.$$

For a polynomial $f \in \mathbb{C}[\mathbf{x}]$, let $\mathbf{J}(f)$ denote $(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n})$ and $\mathbf{J}_i(f) = \frac{\partial f}{\partial x_i}$. Let $\mathbf{J}(\mathbf{F})(\mathbf{p})$ denote the value of a function matrix $\mathbf{J}(\mathbf{F})$ at a point $\mathbf{p}$, similarly for $\mathbf{J}(f)(\mathbf{p})$.

Let $\mathbf{F} = \{f_1, \ldots, f_n\} \subset \mathbb{C}[\mathbf{x}]$ be a polynomial system. We have two following definitions:

Definition 1. An **isolated zero** of $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ is a point $\mathbf{p} \in \mathbb{C}^n$ which satisfies:

$$\exists\, \varepsilon > 0 : \{\mathbf{y} \in \mathbb{C}^n : \|\mathbf{y} - \mathbf{p}\| < \varepsilon\} \cap \mathbf{F}^{-1}(\mathbf{0}) = \{\mathbf{p}\},$$

where $\mathbf{F}^{-1}(\mathbf{0}) \triangleq \{\mathbf{p} \in \mathbb{C}^n : \mathbf{F}(\mathbf{p}) = \mathbf{0}\}$.

Definition 2. We call an isolated zero $\mathbf{p} \in \mathbb{C}^n$ of $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ an **isolated singular zero** if and only if

$$\operatorname{rank}(\mathbf{J}(\mathbf{F})(\mathbf{p})) < n.$$

Otherwise, $\mathbf{p}$ is an **isolated regular (simple) zero** of $\mathbf{F}(\mathbf{x}) = \mathbf{0}$.

The **Taylor series expansion** (Taylor expansion for short) of $f \in \mathbb{C}[\mathbf{x}]$ at $\mathbf{p} = (p_1, \ldots, p_n) \in \mathbb{C}^n$ is

$$f(\mathbf{x}) = f(\mathbf{p}) + \sum_{j=1}^{n} \frac{\partial f(\mathbf{p})}{\partial x_j}(x_j - p_j) + \sum_{1 \le i,j \le n} \frac{\partial^2 f(\mathbf{p})}{\partial x_i \partial x_j}(x_i - p_i)(x_j - p_j) + \cdots. \tag{1}$$

Definition 3. Let $\mathbf{p} \in \mathbb{C}^n$ and $f(\mathbf{p}) = 0$. We say $f \in \mathbb{C}[\mathbf{x}]$ is **singular** at $\mathbf{p}$ if

$$\frac{\partial f(\mathbf{p})}{\partial x_j} = 0, \forall 1 \le j \le n.$$

Otherwise, we say f is **regular** at $\mathbf{p}$.

Definition 4. Let $f \in \mathbb{C}[\mathbf{x}]$, $\tilde{\mathbf{p}} \in \mathbb{C}^n$ and a tolerance $\theta > 0$, s.t. $|f(\tilde{\mathbf{p}})| < \theta$. We say f is θ-**singular** at $\tilde{\mathbf{p}}$ if

$$\left|\frac{\partial f(\tilde{\mathbf{p}})}{\partial x_j}\right| < \theta, \forall 1 \leq j \leq n.$$

Otherwise, we say f is θ-**regular** at $\tilde{\mathbf{p}}$.

Lemma 1. *Let $f \in \mathbb{C}[\mathbf{x}] \setminus \mathbb{C}$, s.t. $f(\mathbf{p}) = 0$. Then there exists at least a $\boldsymbol{\gamma} \in \mathbb{N}^n$, s.t. $\mathbf{d}_{\mathbf{x}}^{\boldsymbol{\gamma}}(f)$ is regular at $\mathbf{p}$.*

Proof. Without loss of generality, we assume $\mathbf{p} = \mathbf{0}$. Then f can be rewritten as a sum of homogeneous polynomials as

$$f = \sum_{d=1}^{deg(f)} f_d.$$

Since $f \not\equiv 0$, there exists at least a $\boldsymbol{\gamma}' \in \mathbb{N}^n$ such that $\mathbf{d}_{\mathbf{x}}^{\boldsymbol{\gamma}'}(f)(\mathbf{p}) \neq 0$. Thus there exists at least a $\boldsymbol{\gamma} \in \mathbb{N}^n$ such that $\mathbf{d}_{\mathbf{x}}^{\boldsymbol{\gamma}}(f)$ is regular at $\mathbf{p}$.

Now, we give an example to explain Definitions 3 and 4, and Lemma 1.

Example 1. Let $f = x_1 + 3x_3 + 4x_4 - x_1^2 + x_3^2 - x_4^2 - x_2^3$. For the exact point $\mathbf{p} = (0, 0, 0, 0)$, we have the Taylor expansion of f at $\mathbf{p}$ is:

$$f(\mathbf{x}) = x_1 + 3x_3 + 4x_4 - x_1^2 + x_3^2 - x_4^2 - x_2^3.$$

Since

$$|f(\mathbf{p})| = 0, \ |\mathbf{J}_2(f)(\mathbf{p})| = 0, \ |\mathbf{J}_i(f)(\mathbf{p})| \neq 0, \ i = 1, 3, 4,$$

we know that f is regular at $\mathbf{p}$. So is $\mathbf{d}_{\mathbf{x}}^{(0,2,0,0)}(f) = -3x_2$.

Similarly, for the approximate point $\tilde{\mathbf{p}} = (0.001, -0.001, 0.002, -0.001)$ and a tolerance $\theta = 0.01$, we have:

$$\begin{aligned} f(\mathbf{x}) =& 0.003002001 + 0.998(x_1 - 0.001) - 3 \cdot 10^{-6}(x_2 + 0.001) + 3.004(x_3 - 0.002) \\ &+ 4.002(x_4 + 0.001) - (x_1 - 0.001)^2 + 3 \cdot 10^{-3}(x_2 + 0.001)^2 + (x_3 - 0.002)^2 \\ &- (x_4 + 0.001)^2 - (x_2 + 0.001)^3. \end{aligned}$$

Since $|f(\tilde{\mathbf{p}})| = 0.003002001 < \theta$, $|\frac{\partial f}{\partial x_2}(\tilde{\mathbf{p}})| = 3 \cdot 10^{-6} < \theta$, $|\frac{\partial f}{\partial x_1}(\tilde{\mathbf{p}})| = 0.998 > \theta$, $|\frac{\partial f}{\partial x_3}(\tilde{\mathbf{p}})| = 3.004 > \theta$, $|\frac{\partial f}{\partial x_4}(\tilde{\mathbf{p}})| = 4.002 > \theta$, thus f is θ-regular at $\tilde{\mathbf{p}}$.

From this example, it is easy to see that when compared with the exact case, the approximate zero $\tilde{\mathbf{p}}$ brings a small perturbation in the coefficients of the Taylor expansion of f at $\tilde{\mathbf{p}}$. However, once given a proper θ, we could acquire the same judging result as the exact case. For the above example, f is regular at $\mathbf{p}$ and it is also θ-regular at $\tilde{\mathbf{p}}$.

Definition 5. Denote the operation set $\Delta \triangleq \{+, \ \cdot, \ \partial\}$, where " $+$ " denotes the sum of two polynomials, " $\cdot$ " denotes scalar multiplication and " ∂ " the differential of a polynomial. Given a polynomial system $\mathbf{F} = \{f_1, \ldots, f_n\} \subset \mathbb{C}[\mathbf{x}]$ and $\mathbf{p} \in \mathbb{C}^n$ such that $\mathbf{F}(\mathbf{p}) = \mathbf{0}$, we define a polynomial set $\Delta_{\mathbf{p}}(\mathbf{F})$, which satisfies:

(1) $\mathbf{F} \subset \Delta_{\mathbf{p}}(\mathbf{F})$;
(2) $\{a\,h | h \in \Delta_{\mathbf{p}}(\mathbf{F}), a \in \mathbb{C}\backslash\{0\}\} \subset \Delta_{\mathbf{p}}(\mathbf{F})$;
(3) $\{h_1 + h_2 | h_1(\mathbf{p}) + h_2(\mathbf{p}) = 0, h_1, h_2 \in \Delta_{\mathbf{p}}(\mathbf{F})\} \subset \Delta_{\mathbf{p}}(\mathbf{F})$;
(4) $\{\frac{\partial h}{\partial x_i} | \frac{\partial h}{\partial x_i}(\mathbf{p}) = 0, i \in \{1, \ldots, n\}, h \in \Delta_{\mathbf{p}}(\mathbf{F})\} \subset \Delta_{\mathbf{p}}(\mathbf{F})$.

Especially, for one polynomial $f \in \mathbb{C}[\mathbf{x}]$, we have the corresponding set $\Delta_{\mathbf{p}}(f)$.

The following lemma shows the relationship between the polynomials in $\Delta_{\mathbf{p}}(\mathbf{F})$ and the polynomials in $\mathbf{F}$.

Lemma 2. *Let $\mathbf{F} = \{f_1, \ldots, f_n\} \subset \mathbb{C}[\mathbf{x}]$ and $\mathbf{p} \in \mathbb{C}^n$, s.t. $\mathbf{F}(\mathbf{p}) = \mathbf{0}$. $\forall g \in \Delta_{\mathbf{p}}(\mathbf{F})$, we have*

$$g = \sum_{i=1}^{n} \sum_{j} a_{i,j} \frac{\partial^{|\boldsymbol{\gamma}_{i,j}|} f_i}{\partial \mathbf{x}^{\boldsymbol{\gamma}_{i,j}}}, \tag{2}$$

where $a_{i,j} \in \mathbb{C}$ and $\boldsymbol{\gamma}_{i,j} \in \mathbb{N}^n$.

Proof. The proof is obvious.

We illustrate Definition 5 and Lemma 2 by the following example.

Example 2. Let $\mathbf{F} = \{f_1 = (x+y)^2+x^3, f_2 = x+y+y^3\}$. $\mathbf{p} = (0, 0)$ is an isolated zero of $\mathbf{F} = 0$. Let $h_1 = \frac{\partial f_1}{\partial x} = 2\,(x+y)+3\,x^2$, $h_2 = \frac{\partial f_1}{\partial y} = 2\,(x+y)$, $h_3 = h_1-2\,f_2 = 3\,x^2-2\,y^3$, $h_4 = \frac{\partial h_3}{\partial x} = 6\,x$, $h_5 = \frac{\partial^2 h_3}{\partial y^2} = -12\,y$. It is clear that $h_i \in \Delta_{\mathbf{p}}(\mathbf{F})$, $i = 1, \ldots, 5$ and h_i has the form as (2).

3. Deflation of polynomial systems

Given a polynomial system with a multiple zero, Newton-type method usually is not used directly on the input system since it converges slowly or even does not converge. Thus, deflation techniques are developed to transform the input system into another deflated system, which is regular at some zero whose certain projection is the given multiple zero. In the following section, we consider the deflation problem of polynomial systems from a new perspective: the linear combination.

3.1. Symbolic deflation system

In this section, given a polynomial system $\mathbf{F} \subset \mathbb{C}[\mathbf{x}]$ with an isolated singular zero $\mathbf{p} \in \mathbb{C}^n$, by employing some differential operations on the input polynomials or on the linear combinations of the related polynomials, we propose a new method to construct a new square system $\mathbf{F}' \subset \mathbb{C}[\mathbf{x}]$, which satisfies that $\mathbf{p}$ is a simple zero of $\mathbf{F}' = \mathbf{0}$. We also prove the existence of $\mathbf{F}'$ and show some properties of it.

First, let us see a simple example to explain our idea.

Example 3 (*Toy Example*). Let $\mathbf{F} = \{f_1 = x-y+x^2, f_2 = x-y+y^2\}$ with a 3-fold isolated zero $\mathbf{p} = (0, 0)$. Obviously, f_1 and f_2 are already regular at $\mathbf{p}$. However, it is easy to find that the terms with degree one of f_1 and f_2 are linearly dependent. Using $f_2 - f_1$ to eliminate these terms of degree one, we get the polynomial $h = y^2 - x^2$ and two new polynomials $\frac{\partial h}{\partial x} = -2\,x$, $\frac{\partial h}{\partial y} = 2\,y$, which are both regular at $\mathbf{p}$. Selecting the two polynomials f_1 and $\frac{\partial h}{\partial y}$, we get a new square system $\mathbf{F}' = \{x - y + x^2, 2\,y\}$, which has a regular zero $\mathbf{p} = (0, 0)$. Moreover, it is a system without perturbation.

Based on the idea in the above simple example, now we show our technique to construct a deflated square system below.

Assume that we have got the polynomials $g_1, \ldots, g_s$, which are regular at $\mathbf{p}$, from the input polynomials $f_1, \ldots, f_s$ such that

$$\operatorname{rank}(\mathbf{J}(g_1, \ldots, g_s)(\mathbf{p})) = s.$$

Given one more polynomial f_{s+1}, we want to compute another polynomial g_{s+1}, s.t.

$$\operatorname{rank}(\mathbf{J}(g_1, \ldots, g_s, g_{s+1})(\mathbf{p})) = s + 1.$$

Using only $g_1, \ldots, g_s$ and f_{s+1}, we may not get the suitable g_{s+1} if

$$\dim \mathbb{V}(g_1, \ldots, g_s, f_{s+1}) > \dim \mathbb{V}(f_1, \ldots, f_s, f_{s+1}).$$

The input polynomials are needed in this case. Thus, we use $\{g_1, \ldots, g_s\} \cup \{f_1, \ldots, f_{s+1}\}$ to compute g_{s+1}. We will show how to compute g_{s+1} in the following lemma.

Lemma 3. *Let* $\mathbf{F} = \{f_1, \ldots, f_s, f_{s+1}, \ldots, f_{s+k}\} \subset \mathbb{C}[x_1, \ldots, x_n](k \geq 1)$ *and* $\mathbf{p} \in \mathbb{C}^n$, *s.t.* $\mathbf{F}(\mathbf{p}) = \mathbf{0}$ *and* $\operatorname{rank}(\mathbf{J}(\mathbf{F})(\mathbf{p})) = s$. *Assume* $\dim \mathbb{V}(\mathbf{F}) \leq n - s - 1$ *and* $\deg(\mathbf{F}) = m (m > 1)$. *Then we can get a polynomial system* $\mathbf{F}' = \{f_1', \ldots, f_s', f_{s+1}'\}$, *which satisfies:*

1. $\operatorname{rank}(\mathbf{J}(\mathbf{F}')(\mathbf{p})) = s + 1$, *and* $f_j' \in \Delta_{\mathbf{p}}(\mathbf{F})(1 \leq j \leq s + 1)$;
2. $\deg(\mathbf{F}') \leq m$.

Proof. Without loss of generality, we assume that $\mathbf{p}$ is the origin and

$$\operatorname{rank}(\mathbf{J}(f_1, \ldots, f_s)(\mathbf{p})) = s. \tag{3}$$

In the following, we consider the case of $s > 0$, since if $s = 0$, we can use the operator ∂ on $f_i (1 \leq i \leq s + k)$ to get some polynomials, which are regular at $\mathbf{p}$.

To construct a polynomial system $\mathbf{F}'$, s.t. $\operatorname{rank}(\mathbf{J}(\mathbf{F}')(\mathbf{p})) = s + 1$, we consider the rest polynomials $\{f_{s+1}, \ldots, f_{s+k}\}$. Our proof is constructive.

First, $f_i (i = 1, \ldots, s)$ has the form:

$$f_i = \sum_{k=1}^{n} a_{ik} x_k + \mathrm{T}_i,$$

where $\mathrm{T}_i \in \mathbb{C}[\mathbf{x}]$ and $\deg(\mathrm{T}_i) = 0$ or $\deg(\mathrm{T}_i) \geq 2$. It is easy to know that the row vector $\mathbf{a}_i = (a_{i1}, \ldots, a_{in})(1 \leq i \leq s)$ of the Jacobian matrix of $(f_1, \ldots, f_s)$ at $\mathbf{p}$ is linearly independent since (3) holds.

Therefore, we can consider the following linear coordinate transformation L:

$$L : \begin{cases} y_i = \sum\limits_{k=1}^{n} a_{ik}x_k, \ 1 \leq i \leq s \\ y_i = x_i, \ \ i = s+1, \ldots, n. \end{cases}$$

With a realignment of the sequence of the variables $\{x_1, \ldots, x_n\}$, we can always have the first s columns of the coefficient matrix of L being linearly independent. Then L is invertible. Denote the inverse of L as L^{-1}. Let $\mathbf{p}' = L(\mathbf{p})$ and $F_i = L^{-1}(f_i) \in \mathbb{C}[y_1, \ldots, y_n]$. We have:

$$\begin{cases} F_i = y_i + L^{-1}(\mathrm{T}_i), \ \ i = 1, \ldots, s, \\ F_{s+i} = \sum\limits_{j=1}^{s} b_{i,j}y_j + L^{-1}(\mathrm{T}_{s+i}), i = 1, \ldots, k. \end{cases} \tag{4}$$

Since $\dim \mathbb{V}(\mathbf{F}) \leq n - s - 1$ and L^{-1} is invertible, it is obvious that

$$\dim \mathbb{V}(F_1, \ldots, F_{s+k}) \leq n - s - 1.$$

Therefore, noticing that the terms with degree one of all $F_i(i = 1, \ldots, s+k)$ in (4) contain only s variables, there must be at least one of $\{L^{-1}(\mathrm{T}_i), i = 1, \ldots, s+k\}$ containing at least one term, which has the form of $y_{s+1}^{d_{s+1}} y_{s+2}^{d_{s+2}} \cdots y_n^{d_n}$, such that $\sum\limits_{j=s+1}^{n} d_j > 1$.

It is easy to prove the claim. Suppose all $L^{-1}(\mathrm{T}_i)(1 \leq i \leq s+k)$ contain no terms of the form of $y_{s+1}^{d_{s+1}} y_{s+2}^{d_{s+2}} \cdots y_n^{d_n}$. Then, all the terms of $F_i(1 \leq i \leq s+k)$ have the form of $y_1^{d_1} \cdots y_s^{d_s} y_{s+1}^{d_{s+1}} \cdots y_n^{d_n}$, $\sum\limits_{j=1}^{s} d_j > 0$. In this case, the system $\{F_1, \ldots, F_{s+k}\}$ vanishes on $\{y_1 = 0, \ldots, y_s = 0\}$. Thus, we can verify easily that $\dim \mathbb{V}(F_1, \ldots, F_{s+k}) = n - s$, which contradicts with $\dim \mathbb{V}(F_1, \ldots, F_{s+k}) \leq n - s - 1$. Thus, the claim is true.

Without loss of generality, we suppose that $L^{-1}(\mathrm{T}_l)(l \in \{1, \ldots, s+k\})$ has the term with the form of $y_{s+1}^{d_{s+1}} y_{s+2}^{d_{s+2}} \cdots y_n^{d_n}$ and take the variable y_{s+1} for example, i.e. $d_{s+1} \neq 0$. Further, we ask for the term with a lowest degree among all this kind of terms and denote the lowest degree as d. Then, we have:

$$F'_{s+1} = \frac{\partial^{d-1} F_l}{\partial y_{s+1}^{d_{s+1}-1} y_{s+2}^{d_{s+2}} \cdots y_n^{d_n}} = \sum_{i=1}^{n} \gamma_i y_i + \mathrm{T}'_l, \ d = \sum_{j=s+1}^{n} d_j. \tag{5}$$

It is easy to see that $\gamma_{s+1} \neq 0, \ \deg(F'_{s+1}) < \deg(F_l)$.

Thus, we have a new system $\{F_1, \ldots, F_s, F'_{s+1}\}$. It is easy to check that

$$\mathrm{rank}(\mathbf{J}(F_1, \ldots, F_s, F'_{s+1})(\mathbf{p}')) = s + 1.$$

Finally, after doing the transformation L on $F_i(1 \leq i \leq s)$ and F'_{s+1}, we have the new system $\mathbf{F}' = \{f'_1, \ldots, f'_{s+1}\}$, where

$$f'_i = L(F_i) = f_i(i = 1, \ldots, s), f'_{s+1} = L(F'_{s+1}) \text{ with } \mathrm{rank}(\mathbf{J}(\mathbf{F}')(\mathbf{p})) = s + 1.$$

By the definition of $\Delta_{\mathbf{p}}(\mathbf{F})$ (see Definition 5), we can find that $f'_i \in \Delta_{\mathbf{p}}(\mathbf{F})(1 \leq i \leq s+1)$. Therefore, we finish the first part of the proof.

From Lemma 2 and (5), it is easy to know that the maximal degree of $f'_i(i = 1, \ldots, s+1)$ is no larger than m. That is, $\deg(\mathbf{F}') \leq m$. Thus, we complete the proof.

Now, we consider constructing a square system, which is regular at an isolated singular zero of the input system.

Theorem 1. *Let $\mathbf{F} = \{f_1, \ldots, f_N\} \subset \mathbb{C}[\mathbf{x}](N \geq n)$ be a polynomial system. $\mathbf{p} \in \mathbb{C}^n$ an isolated singular zero of $\mathbf{F} = \mathbf{0}$ and $\deg(\mathbf{F}) = m$. Then there exists a square polynomial system $\mathbf{F}' = \{f'_1, \ldots, f'_n\} \subset \Delta_{\mathbf{p}}(\mathbf{F})$, s.t.*

1. *$\mathbf{p}$ is an isolated regular zero of $\mathbf{F}' = \mathbf{0}$;*
2. *$\deg(\mathbf{F}') \leq m$.*

Proof. Without loss of generality, assume that $\mathbf{p}$ is the origin. In the following, we will construct a square system by the polynomials in $\Delta_{\mathbf{p}}(\mathbf{F})$.

First, we can choose a system $\mathbf{F}_0$ from $\mathbf{F}$, denoted as $\mathbf{F}_0 = \{f_1, \ldots, f_r\}$, whose Jacobian matrix at $\mathbf{p}$ has a maximal rank, s.t.

$$\mathrm{rank}(\mathbf{J}(f_1, \ldots, f_r)(\mathbf{p})) = \mathrm{rank}(\mathbf{J}(\mathbf{F})(\mathbf{p})) = r, \ \ 0 \leq r \leq n.$$

If $r = n$, we finish the proof. Noticing that when $r = 0$, we need only considering at least one of the polynomials in $f_1, \dots, f_N$ and can always get at least one polynomial, which is regular at $\mathbf{p}$ by Lemma 1. Thus, in the following, we consider the case of $1 \le r < n$.

First, considering the system $\{f_1, \dots, f_r, f_{r+1}, \dots, f_N\}$, by Lemma 3, we can get a system

$$\mathbf{F}_1 = \{f_1^{(1)}, \dots, f_r^{(1)}, f_{r+1}^{(1)}\},$$

s.t.

$$\mathbf{F}_1(\mathbf{p}) = \mathbf{0} \text{ and } \operatorname{rank}(\mathbf{J}(f_1^{(1)}, \dots, f_r^{(1)}, f_{r+1}^{(1)})(\mathbf{p})) = r + 1.$$

Using the technique in Lemma 3, by considering the system $\mathbf{F} \cup \{f_1^{(1)}, \dots, f_r^{(1)}, f_{r+1}^{(1)}\}$, we can get a system

$$\mathbf{F}_2 = \{f_1^{(2)}, \dots, f_{r+1}^{(2)}, f_{r+2}^{(2)}\},$$

s.t.

$$\mathbf{F}_2(\mathbf{p}) = \mathbf{0} \text{ and } \operatorname{rank}(\mathbf{J}(f_1^{(2)}, \dots, f_{r+1}^{(2)}, f_{r+2}^{(2)})(\mathbf{p})) = r + 2.$$

Repeat this process $n - r$ times and finally, we get a square system

$$\mathbf{F}_{n-r} = \{f_1^{(n-r)}, f_2^{(n-r)}, \dots, f_n^{(n-r)}\},$$

s.t.

$$\mathbf{F}_{n-r}(\mathbf{p}) = \mathbf{0} \text{ and } \operatorname{rank}(\mathbf{J}(f_1^{(n-r)}, f_2^{(n-r)}, \dots, f_n^{(n-r)})(\mathbf{p})) = n.$$

Thus, our final square system

$$\mathbf{F}' = \{f_1' = f_1^{(n-r)}, f_2' = f_2^{(n-r)}, \dots, f_n' = f_n^{(n-r)}\}.$$

By Lemma 3, it is obvious that the maximal degree of $f_i'(1 \le i \le n)$ is no larger than m. That is, $\deg(\mathbf{F}') \le m$.

Remark 1. 1. In the above construction process, we repeat $n - r$ times to get the deflated system $\mathbf{F}'$. If considering all the variables simultaneously, we get more than one eligible polynomial each time in (5). Thus, the number of times in actual computation is less than $n - r$.

2. In the beginning of our construction, we also can compute all the related polynomials of all the input polynomials, which are regular at $\mathbf{p}$. Then, we choose a system from these polynomials, whose Jacobian matrix at $\mathbf{p}$ has a maximal rank. That is to say that we make r as big as possible to reduce our repeating steps.

Theorem 1 tells us that given a polynomial system $\mathbf{F}$ with an isolated singular zero $\mathbf{p}$, we can construct a new square system $\mathbf{F}'$, which is regular at $\mathbf{p}$ and moreover, the degree of the polynomials in $\mathbf{F}'$ does not increase. We give an example to illustrate our method in the following example.

Example 4 (*DZ2 [18]*)**.** Let $\mathbf{F} = \{f_1 = x_1^4, f_2 = x_1^2 x_2 + x_2^4, f_3 = x_3 + x_3^2 - 7\,x_1^3 - 8\,x_1^2\}$, which has a 16-fold zero $\mathbf{p} = (0, 0, -1)$. The maximal degree of f_1, f_2, f_3 is 4. First, by the Taylor expansions of f_1, f_2, f_3 at $\mathbf{p}$, we have:

$$\begin{aligned} f_1 &= x_1^4, \\ f_2 &= x_1^2 x_2 + x_2^4, \\ f_3 &= -(x_3 + 1) - 8\,x_1^2 + (x_3 + 1)^2 - 7\,x_1^3. \end{aligned}$$

It is easy to find that only f_3 is regular at $\mathbf{p}$. Since $s = \operatorname{rank}(\mathbf{J}(\mathbf{F})(\mathbf{p})) = 1$ and $\dim \mathbb{V}(f_3, f_2) = 1$, we consider the system $\{f_3, f_2\}$ directly. By Lemma 3, we have a system

$$\{f_1^{(1)} = f_3, f_2^{(1)} = \mathbf{d}_{\mathbf{x}}^{(2,0,0)}(f_2) = x_2\},$$

which satisfies $\operatorname{rank}(\mathbf{J}(f_1^{(1)}, f_2^{(1)})(\mathbf{p})) = 2$.

Next, we consider the system $\{f_1^{(1)}, f_2^{(1)}\} \cup \mathbf{F}$. Since $\dim \mathbb{V}(f_1^{(1)}, f_2^{(1)}, \mathbf{F}) = 0$, by Lemma 3, we have a system

$$\{f_1^{(2)} = f_3, f_2^{(2)} = x_2, f_3^{(2)} = \mathbf{d}_{\mathbf{x}}^{(3,0,0)}(f_1) = 4\,x_1\},$$

which satisfies $\operatorname{rank}(\mathbf{J}(f_1^{(2)}, f_2^{(2)}, f_3^{(2)})(\mathbf{p})) = 3$.

Thus, we acquire the final square system $\mathbf{F}' = \{f_3, x_2, 4\,x_1\}$. It is easy to check that $\mathbf{p}$ is a simple zero of $\mathbf{F}' = \mathbf{0}$ and the degree of every polynomial in $\mathbf{F}'$ is no more than 4.

In this example, we repeat $n - s = 2$ times to acquire the final square system $\mathbf{F}'$. In fact, as what we say in Remark 1 of Theorem 1, computing twice is not necessary. Noticing that when computing $f_2^{(1)} = \mathbf{d}_{\mathbf{x}}^{(2,0,0)}(f_2) = x_2$, we also can get $\mathbf{d}_{\mathbf{x}}^{(1,1,0)}(f_2) = 2\,x_1$. They are both regular at $\mathbf{p}$. It is easy to check that

$$\operatorname{rank}(\mathbf{J}(f_1^{(1)}, f_2^{(1)}, \mathbf{d}_{\mathbf{x}}^{(1,1,0)}(f_2) = 2\,x_1)(\mathbf{p})) = 3.$$

Thus, we obtain another square system $\mathbf{F}' = \{f_3, x_2, 2\,x_1\}$.

3.2. Parametric deflation system

Given a polynomial system $\mathbf{F}$ with an isolated singular zero $\mathbf{p}$, by employing some differential operations on the input polynomials directly or on the linear combinations of the related polynomials, we give a method to construct a new polynomial system $\mathbf{F}'$ in Section 3.1, which satisfies that $\mathbf{p}$ is a simple zero of $\mathbf{F}' = \mathbf{0}$.

However, in practice, once given a polynomial system $\mathbf{F}$ with an isolated singular zero $\mathbf{p}$, we can just get an approximate zero $\tilde{\mathbf{p}}$ by some numerical methods [23]. As what we say in Example 1, the inexact value of $\tilde{\mathbf{p}}$ usually brings perturbations in the coefficients when doing the Taylor series expansions of the input polynomials at $\tilde{\mathbf{p}}$. Therefore, we cannot do exact computations when adding two or more polynomials together. The inexact computations would produce a perturbed system of $\mathbf{F}'$, which will lead to a bad final deflation result. We show an example to illustrate this case.

Example 5 (*Toy Example*)**.** Continue with Example 3. Given an approximate zero $\tilde{\mathbf{p}} = (0.0006721, 0.0008381)$. Using the method in Theorem 1, we have $\tilde{h} = f_2 + \tilde{\alpha} f_1$. By solving a Least Square problem, we can get $\tilde{\alpha} = -0.9984909264232$. Finally, we get an inexact system

$$\widetilde{\mathbf{F}}' = \{x - y + x^2, 2y - 0.0015090735767\}.$$

Obviously, we cannot get a good result by the system $\widetilde{\mathbf{F}}'$.

With a simple analysis, we can find that we could not get an exact coefficient α of the linear combination of the polynomials with an approximate zero.

In the following, by introducing some new variables to represent the coefficients of the linear combinations, we give an effective version of our deflation method. Finally, the effective version of our deflation method will usually produce an exact deflated system, which has a simple zero, whose partial projection corresponds to the isolated singular zero of the input system. Furthermore, we also provide the size bound of our method. To our knowledge, it is the first time that considering the deflation of the polynomial system from the perspective of linear combinations.

Similarly, before giving our theoretical results, we also show our main idea with a simple example first.

Example 6 (*Toy Example*)**.** Still consider Example 3. Once given an approximate zero of the input system: $\tilde{\mathbf{p}} = (0.0006721, 0.0008381)$, by Example 5, we know the coefficient $\tilde{\alpha}$ is inexact. Now we introduce a new variable α_1. Let $h = f_2 + \alpha_1 f_1$ and compute

$$\frac{\partial h}{\partial x} = 1 + \alpha_1(2x + 1), \frac{\partial h}{\partial y} = 2y - 1 - \alpha_1.$$

Similar as in Example 5, we have $\tilde{\boldsymbol{\alpha}}_1 = -0.9984909264232$. Given a tolerance $\varepsilon = 0.05$, we have

$$\operatorname{rank}(\mathbf{J}(f_1, \frac{\partial h}{\partial x}, \frac{\partial h}{\partial y})(\tilde{\mathbf{p}}, \tilde{\alpha}_1), \varepsilon) = 2 < 3.$$

Do once again this process and introduce two new variables α_2, α_3. Let

$$g = \frac{\partial h}{\partial y} + \alpha_2 f_1 + \alpha_3 \frac{\partial h}{\partial x}$$

and compute

$$\frac{\partial g}{\partial x} = 2\alpha_1\alpha_3 + \alpha_2(2x + 1), \frac{\partial g}{\partial y} = 2 - \alpha_2, \frac{\partial g}{\partial \alpha_1} = \alpha_3(2x + 1) - 1.$$

By solving another Least Square problem, we get the approximate values:

$$\tilde{\alpha}_2 = 1.9985955412653, \tilde{\alpha}_3 = 1.0014510032456.$$

Then, we have

$$\operatorname{rank}(\mathbf{J}(f_1, \frac{\partial f}{\partial x}, \frac{\partial g}{\partial x}, \frac{\partial g}{\partial y}, \frac{\partial g}{\partial \alpha_1})(\tilde{\mathbf{p}}, \tilde{\alpha}_1, \tilde{\alpha}_2, \tilde{\alpha}_3), \varepsilon) = 5.$$

Thus, we get a polynomial system

$$\widetilde{\mathbf{F}}'(\mathbf{x}, \boldsymbol{\alpha}) = \{f_1, \frac{\partial h}{\partial x}, \frac{\partial g}{\partial x}, \frac{\partial g}{\partial y}, \frac{\partial g}{\partial \alpha_1}\},$$

whose Jacobian matrix at $(\tilde{\mathbf{p}}, \tilde{\alpha}_1, \tilde{\alpha}_2, \tilde{\alpha}_3)$ has a full rank under the tolerance ε.

In fact, we can find that $(0, 0, -1, 2, 1)$ is a simple zero of $\widetilde{\mathbf{F}}'(\mathbf{x}, \boldsymbol{\alpha}) = \mathbf{0}$ and the partial projection $(0, 0)$ of $(0, 0, -1, 2, 1)$ corresponds to the isolated singular zero $\mathbf{p}$ of the input system $\mathbf{F}$.

Given a polynomial system with an isolated zero, we have the following lemma.

Lemma 4 ([15]). *Let* $\mathbf{F} = \{f_1, \ldots, f_n\} \subset \mathbb{C}[\mathbf{x}]$ *be a polynomial system.* $\mathbf{p} \in \mathbb{C}^n$ *is an isolated singular zero of* $\mathbf{F} = \mathbf{0}$. $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_n) \in \mathbb{C}^n$ *is a nonzero row vector, which satisfies* $\mathbf{J}(\mathbf{F})(\mathbf{p})\boldsymbol{\lambda}^{\mathrm{T}} = \mathbf{0}$. *For the new system*

$$\mathbf{G} = \{\lambda_1 \frac{\partial f_1}{\partial x_1} + \cdots + \lambda_n \frac{\partial f_1}{\partial x_n},\ \ldots,\ \lambda_1 \frac{\partial f_n}{\partial x_1} + \cdots + \lambda_n \frac{\partial f_n}{\partial x_n}\},$$

we have the multiplicity of $\mathbf{p}$ *in* $\{\mathbf{F}, \mathbf{G}\} = \mathbf{0}$ *is lower than the multiplicity of* $\mathbf{p}$ *in* $\mathbf{F} = \mathbf{0}$.

Remark 2. In Remark 2.1 of [6], the authors mentioned that deflation could also be constructed using the left null space. That is, we can replace $\mathbf{G}$ by the following system

$$\mathbf{G}' = \{\lambda_1 \frac{\partial f_1}{\partial x_1} + \cdots + \lambda_n \frac{\partial f_n}{\partial x_1},\ \ldots,\ \lambda_1 \frac{\partial f_1}{\partial x_n} + \cdots + \lambda_n \frac{\partial f_n}{\partial x_n}\}, \tag{6}$$

where $\boldsymbol{\lambda}\,\mathbf{J}(\mathbf{F})(\mathbf{p}) = \mathbf{0}$. Furthermore, we have the following lemma.

Lemma 5. *Let* $\mathbf{F} = \{f_1, \ldots, f_n\} \subset \mathbb{C}[\mathbf{x}]$ *be a polynomial system.* $\mathbf{p} \in \mathbb{C}^n$ *be an isolated singular zero of* $\mathbf{F} = \mathbf{0}$. *Assume* $\mathrm{rank}(\mathbf{J}(f_1, \ldots, f_s)(\mathbf{p})) = \mathrm{rank}(\mathbf{J}(\mathbf{F})(\mathbf{p})) = s$. *Consider the augmented system*

$$\mathbf{G} = \{f_1, \ldots, f_n, h_1, \ldots, h_n\} \subset \mathbb{C}[\mathbf{x}, \boldsymbol{\alpha}],$$

where

$$h_j = \alpha_1 \frac{\partial f_1}{\partial x_j} + \cdots + \alpha_s \frac{\partial f_s}{\partial x_j} + \frac{\partial f_{s+1}}{\partial x_j},\ j = 1, \ldots, n.$$

Then, we have:

1. *there exists a unique* $\hat{\boldsymbol{\alpha}} \in \mathbb{C}^s$ *such that the system* $\mathbf{G}$ *has an isolated zero at* $(\mathbf{p}, \hat{\boldsymbol{\alpha}})$.
2. *the multiplicity of* $\mathbf{G}$ *at* $(\mathbf{p}, \hat{\boldsymbol{\alpha}})$ *is lower than that of* $\mathbf{F}$ *at* $\mathbf{p}$.

Proof. Let

$$A_{ij}(\mathbf{x}) = \frac{\partial f_i}{\partial x_j} \in \mathbb{C}[\mathbf{x}],\ a_{ij} = \frac{\partial f_i(\mathbf{p})}{\partial x_j} \in \mathbb{C},\ i = 1, \ldots, s+1,\ j = 1, \ldots, n.$$

Denote the matrix $A = (a_{ij}), i = 1, \ldots, s, j = 1, \ldots, n$ and the row vector $\mathbf{b} = (a_{s+1,1}, \ldots, a_{s+1,n})$.

On one hand, when we fix $\mathbf{x} = \mathbf{p}$, the system

$$\mathbf{H}(\mathbf{p}, \boldsymbol{\alpha}) = \{h_j(\mathbf{p}, \boldsymbol{\alpha}) = a_{1j}\alpha_1 + \cdots + a_{sj}\alpha_s + a_{s+1,j},\ j = 1, \ldots, n\}$$

is a linear system with respect to the variables $\alpha_1, \ldots, \alpha_s$. Furthermore, it is easy to check that $\hat{\boldsymbol{\alpha}}$, which is determined by $AA^{\mathrm{T}}\hat{\boldsymbol{\alpha}} = -A\mathbf{b}^{\mathrm{T}}$, is the unique zero of $\mathbf{H}(\mathbf{p}, \boldsymbol{\alpha}) = \mathbf{0}$. That is, there exists a unique $\hat{\boldsymbol{\alpha}}$ such that the system $\mathbf{G}$ has an isolated zero at $(\mathbf{p}, \hat{\boldsymbol{\alpha}})$.

On the other hand, with the row operations, we could reduce the system $\mathbf{G}$ to the system

$$\{\alpha_1 = l_1(\mathbf{x}), \ldots, \alpha_s = l_s(\mathbf{x})\},$$

where $l_i(\mathbf{x})$ are rational expressions and $\hat{\alpha}_i = l_i(\mathbf{p})$. Thus, considering the multiplicity of $\mathbf{G}$ at $(\mathbf{p}, \hat{\boldsymbol{\alpha}})$ is equivalent to considering the multiplicity of $\mathbf{G}(\mathbf{x}, \hat{\boldsymbol{\alpha}})$ at $\mathbf{p}$. Note that $(\hat{\alpha}_1, \ldots, \hat{\alpha}_s, 1, 0, \ldots, 0)\mathbf{J}(\mathbf{F})(\mathbf{p}) = \mathbf{0}$. By Lemma 4 and (6), we know the second part holds. Thus, we finished the proof.

In the above lemma, we construct n new polynomials $h_1, \ldots, h_n$. In fact, we can get them from the following way. Note that

$$\mathrm{rank}(\mathbf{J}(f_1, \ldots, f_s)(\mathbf{p})) = \mathrm{rank}(\mathbf{J}(\mathbf{F})(\mathbf{p})) = s.$$

We know easily that $\mathbf{J}(f_{s+1})(\mathbf{p})$ and $\mathbf{J}(f_1)(\mathbf{p}), \ldots, \mathbf{J}(f_s)(\mathbf{p})$ are linearly dependent. Thus, we can do the linear combination between f_{s+1} and $f_1, \ldots, f_s$ to eliminate this linear relationship. Let

$$g = f_{s+1} + \sum_{i=1}^{s} \alpha_i f_i,$$

where new variables α_i are used to represent the coefficients of the linear combination. Compute all the derivatives of g with respect to the variables $x_1, \ldots, x_n$ and we get

$$h_j = \frac{\partial g}{\partial x_j} = \alpha_1 \frac{\partial f_1}{\partial x_j} + \cdots + \alpha_s \frac{\partial f_s}{\partial x_j} + \frac{\partial f_{s+1}}{\partial x_j},\ j = 1, \ldots, n.$$

Thus, the above lemma tells us that after doing the linear combination of polynomials between f_{s+1} and $f_1, \ldots, f_s$, we get an augmented system $\mathbf{G}$, which satisfies that the multiplicity of $\mathbf{G}$ at $(\mathbf{p}, \hat{\boldsymbol{\alpha}})$ is lower than that of $\mathbf{F}$ at $\mathbf{p}$. By repeating using the linear combination between polynomials in the original system and its related derivatives, we can construct a final deflated system, which processes an isolated simple zero. Denote μ be the multiplicity of $\mathbf{F}$ at $\mathbf{p}$. We do this repetitive process at most μ times.

Further, based on Lemma 5, we have the following theorem.

Theorem 2. *Let $\mathbf{F} = \{f_1, \ldots, f_n\} \subset \mathbb{C}[\mathbf{x}]$ be a polynomial system. $\mathbf{p} \in \mathbb{C}^n$ be an isolated singular zero of $\mathbf{F} = \mathbf{0}$. Denote $m = \deg(\mathbf{F})$. Then there exists a square polynomial system $\widetilde{\mathbf{F}}'(\mathbf{x}, \boldsymbol{\alpha}) = \{g_1, \ldots, g_t\} \subset \mathbb{C}[\mathbf{x}, \boldsymbol{\alpha}]$, s.t.*

1. *$(\mathbf{p}, \hat{\boldsymbol{\alpha}}) \in \mathbb{C}^t$ is an isolated simple zero of $\widetilde{\mathbf{F}}'(\mathbf{x}, \boldsymbol{\alpha}) = \mathbf{0}$;*
2. *t is bounded by $2^{\mu}\, n$, where μ is the multiplicity of $\mathbf{p}$ in $\mathbf{F}$;*
3. *$\deg(\widetilde{\mathbf{F}}'(\mathbf{x}, \boldsymbol{\alpha})) \leq m$.*

Next, based on Lemma 5 and Theorem 2, we give an effective Algorithm 1 to compute a deflated square system from the input system with an approximate isolated singular zero below. It is an effective version of Lemma 3. θ is a tolerance to detect the regularity of the polynomials and we will talk about it in next subsection. ε is another tolerance to judge the numerical rank of the Jacobian matrix at an approximate zero and we also talk about it in next section.

Algorithm 1 CDSS : Compute a deflated square system.

```
Input:
    a polynomial system $\mathbf{F} := \{f_1, \ldots, f_n\} \subset \mathbb{C}[\mathbf{x}]$, an approximate isolated singular solution $\tilde{\mathbf{p}} \in \mathbb{C}^n$, two tolerances $\theta$ and $\varepsilon$.
Output:
    a square polynomial system $\widetilde{\mathbf{F}}'(\mathbf{x}, \boldsymbol{\alpha}) := \{\tilde{f}_1, \ldots, \tilde{f}_t\} \subset \mathbb{C}[\mathbf{x}, \boldsymbol{\alpha}]$ and a point $\tilde{\boldsymbol{\alpha}}$, s.t. $(\tilde{\mathbf{p}}, \tilde{\boldsymbol{\alpha}})$ is an approximate regular zero of $\widetilde{\mathbf{F}}'(\mathbf{x}, \boldsymbol{\alpha}) = \mathbf{0}$.
Compute $\mathbf{G} = \{\mathbf{d}_{\mathbf{x}}^{\gamma}(f) | \mathbf{d}_{\mathbf{x}}^{\gamma}(f)$ is $\theta$-regular at $\tilde{\mathbf{p}}, f \in \mathbf{F}\}$;
Let $\mathbf{H} := \mathbf{F} \cup \mathbf{G}$, $\mathbf{X} := \mathbf{x}$;
while $\mathrm{rank}(\mathbf{J}(\mathbf{H})(\tilde{\mathbf{p}}), \varepsilon) \neq |\mathbf{X}|$ do
    Compute $r := \mathrm{rank}(\mathbf{J}(\mathbf{H})(\tilde{\mathbf{p}}), \varepsilon)$;
    Choose any $\mathbf{H}_1 := \{h_1, \ldots, h_r\} \subset \mathbf{H}$, s.t. $\mathrm{rank}(\mathbf{J}(\mathbf{H}_1)(\tilde{\mathbf{p}}), \varepsilon) = r$;
    Choose $h_{r+1} := \mathbf{H} \setminus \mathbf{H}_1$, s.t. $\dim \mathbb{V}(\mathbf{H}_1, h_{r+1}) = n - r - 1$;
    Let $g := h_{r+1} + \sum_{j=1}^{r} \alpha_j h_j$;
    Compute $\tilde{\boldsymbol{\alpha}} := \mathit{LeastSquares}((\mathbf{J}(\mathbf{H}_1, h_{r+1})(\tilde{\mathbf{p}}))^{\mathrm{T}}(\boldsymbol{\alpha}, 1)^{\mathrm{T}} = \mathbf{0})$;
    Compute $g_1 := \mathbf{J}_1(g), \ldots, g_n := \mathbf{J}_n(g)$;
    Set $\mathbf{H} := \{\mathbf{H}, g_1, \ldots, g_n\}$, $\mathbf{X} := \mathbf{x} \cup \boldsymbol{\alpha}$ and $\tilde{\mathbf{p}} := (\tilde{\mathbf{p}}, \tilde{\boldsymbol{\alpha}})$;
end while
Return: a square system $\widetilde{\mathbf{F}}'(\mathbf{x}, \boldsymbol{\alpha}) = \{\mathbf{H}_1, g_1, \ldots, g_n\}$ and a point $\tilde{\boldsymbol{\alpha}}$.
```

Remark 3. 1. The termination and correctness of the algorithm is guaranteed by Lemma 5 and Theorem 2.

2. In the above algorithm, we compute polynomials of every f_i, which are regular at $\mathbf{p}$ at the beginning. Then, we put all these polynomials together to compute a system $\mathbf{F}_0$, such that the rank of its Jacobian matrix at $\mathbf{p}$ is maximal. This operation can make r as big as possible. In some cases, we have $r = n$, which means we need not introduce new variables, such as Example 8. The aim of this preprocessing operation can speed up our algorithm.

Now, we give two examples to illustrate Algorithm 1.

Example 7 (*General Case*). Consider a polynomial system $\mathbf{F} = \{f_1 = -\frac{9}{4} + \frac{3}{2}x_1 + 2x_2 + 3x_3 + 4x_4 - \frac{1}{4}x_1^2, f_2 = x_1 - 2x_2 - 2x_3 - 4x_4 + 2x_1x_2 + 3x_1x_3 + 4x_1x_4, f_3 = 8 - 4x_1 - 8x_4 + 2x_4^2 + 4x_1x_4 - x_1x_4^2, f_4 = -3 + 3x_1 + 2x_2 + 4x_3 + 4x_4\}$. Given an approximate singular zero

$$\tilde{\mathbf{p}} = (\tilde{p}_1, \tilde{p}_2, \tilde{p}_3, \tilde{p}_4) = (1.00004659, -1.99995813, -0.99991547, 2.00005261)$$

of $\mathbf{F} = \mathbf{0}$ and the tolerance $\varepsilon = 0.005$.

First, we have the Taylor expansion of f_3 at $\tilde{\mathbf{p}}$:

$$\begin{aligned} f_3 = {} & 3 \cdot 10^{-9} - 3 \cdot 10^{-9}(x_1 - \tilde{p}_1) + 0.00010522(x_4 - \tilde{p}_4) + 0.99995341(x_4 - \tilde{p}_4)^2 \\ & - 0.00010522(x_1 - \tilde{p}_1)(x_4 - \tilde{p}_4) - (x_1 - \tilde{p}_1)(x_4 - \tilde{p}_4)^2. \end{aligned}$$

Consider the tolerance $\theta = 0.05$. Since

$$|f_3(\tilde{\mathbf{p}})| < \theta, \ \left|\frac{\partial f_3}{\partial x_i}(\tilde{\mathbf{p}})\right| < \theta (i = 1, 2, 3, 4), \ \left|\frac{\partial^2 f_3}{\partial x_4^2}(\tilde{\mathbf{p}})\right| > \theta,$$

we get a polynomial

$$\frac{\partial f_3}{\partial x_4} = -8 + 4x_1 + 4x_4 - 2x_1x_4,$$

which is θ-regular at $\tilde{\mathbf{p}}$. Similarly, by the Taylor expansion of f_1, f_2, f_4 at $\tilde{\mathbf{p}}$, we have that f_1, f_2, f_4 are all θ-regular at $\tilde{\mathbf{p}}$.

Thus, by Algorithm 1, we have $\mathbf{G} = \{f_1, f_2, -8 + 4x_1 + 4x_4 - 2x_1x_4, f_4\}$. Compute

$$r = \text{rank}(\mathbf{J}(\mathbf{G})(\tilde{\mathbf{p}}), \varepsilon) = 3.$$

We can choose $\mathbf{H}_1 = \{h_1 = f_1, h_2 = f_2, h_3 = -8 + 4x_1 + 4x_4 - 2x_1x_4\}$ from $\mathbf{H} = \mathbf{G} \cup \mathbf{F}$. To $h_4 = f_4 \in \mathbf{H} \setminus \mathbf{H}_1$, let

$$g = h_4 + \alpha_1 h_1 + \alpha_2 h_2 + \alpha_3 h_3.$$

First, by solving a Least Square problem:

$$\mathit{LeastSquares}((\mathbf{J}(\mathbf{H}_1, h_4)(\tilde{\mathbf{p}}))^T[\alpha_1, \alpha_2, \alpha_3, 1]^T = 0),$$

we get an approximate value:

$$(\tilde{\alpha}_1, \tilde{\alpha}_2, \tilde{\alpha}_3) = (-1.000006509, -0.9997557989, 0.000106178711).$$

Then, compute

$$\begin{cases} g_1 = \dfrac{\partial g}{\partial x_1} = 3 + \dfrac{3}{2}\alpha_1 + \alpha_2 + 4\alpha_3 - \dfrac{1}{2}\alpha_1 x_1 + 2\alpha_2 x_2 + 3\alpha_2 x_3 + 4\alpha_2 x_4 - 2\alpha_3 x_4, \\ g_2 = \dfrac{\partial g}{\partial x_2} = 2 + 2\alpha_1 - 2\alpha_2 + 2\alpha_2 x_1, \\ g_3 = \dfrac{\partial g}{\partial x_3} = 4 + 3\alpha_1 - 2\alpha_2 + 3\alpha_2 x_1, \\ g_4 = \dfrac{\partial g}{\partial x_4} = 4 + 4\alpha_1 - 4\alpha_2 + 4\alpha_3 + 4\alpha_2 x_1 - 2\alpha_3 x_1, \end{cases}$$

and we get the polynomial set

$$\mathbf{H}' = \{h_1, h_2, h_3, g_1, g_2, g_3, g_4\},$$

which satisfies

$$\text{rank}(\mathbf{J}(\mathbf{H}')(\tilde{\mathbf{p}}, \tilde{\alpha}_1, \tilde{\alpha}_2, \tilde{\alpha}_3), \varepsilon) = 7.$$

Thus, we get the final square system $\widetilde{\mathbf{F}}'(\mathbf{x}, \boldsymbol{\alpha}) = \mathbf{H}_1$ and the point $\tilde{\boldsymbol{\alpha}} = (\tilde{\alpha}_1, \tilde{\alpha}_2, \tilde{\alpha}_3) = (-1.000006509, -0.9997557989, 0.000106178711)$.

In this example, given the input polynomial system $\mathbf{F}$ with an approximate singular zero $\tilde{\mathbf{p}}$, we can get a final square system by Algorithm 1 with only one step. In fact, α_3 is not necessary to be introduced in this example by noticing that we can acquire a needed square system $\widetilde{\mathbf{F}}'(\mathbf{x}, \boldsymbol{\alpha})$ by using $F = f_4 + \alpha_1 f_1 + \alpha_2 f_2$. We give another example to illustrate the case that we do not introduce new variables.

Example 8 (*Special Case*)**.** (DZ2) Continue with Example 4. Given an approximate isolated singular zero

$$\tilde{\mathbf{p}} = (\tilde{p}_1, \tilde{p}_2, \tilde{p}_3) = (0.00006787, 0.00007577, -0.9999)$$

and a tolerance $\varepsilon = 0.005$, we use the Taylor series to expand $f_i (i = 1, 2, 3)$ at $\tilde{\mathbf{p}}$ and compare all the coefficients with a tolerance $\theta = \varepsilon$. For f_1, we have

$$f_1 = 2.121833963630161 \cdot 10^{-17} + 1.250528341612 \cdot 10^{-12}(x_1 - \tilde{p}_1) + 2.76380214 \cdot 10^{-8} \\ (x_1 - \tilde{p}_1)^2 + 0.27148 \cdot 10^{-3}(x_1 - \tilde{p}_1)^3 + (x_1 - \tilde{p}_1)^4.$$

It is obvious that only the absolute value of the coefficient of $(x_1 - \tilde{p}_1)^4$ is bigger than θ. Therefore, compute $\mathbf{d}^{(3,0,0)}_{(x_1,x_2,x_3)}(f_1) = 4x$, which is θ-regular at $\tilde{\mathbf{p}}$. Similarly, for f_2, f_3, we have the corresponding polynomial(s): $\{2x_1, x_2\}$ and f_3. Thus, we have $\mathbf{G} = \{4x_1, 2x_1, x_2, f_3\}$. It is easy to check that

$$r = \text{rank}(\mathbf{J}(\mathbf{G})(\tilde{\mathbf{p}}), \varepsilon) = \text{rank}(\mathbf{J}(4x_1, x_2, f_3)(\tilde{\mathbf{p}}), \varepsilon) = 3.$$

Thus, we get the needed square system $\widetilde{\mathbf{F}}'(\mathbf{x}) = \mathbf{G} = \{4x_1, x_2, f_3\}$.

In the above two examples, we assume that we have a right judgement on the tolerances θ and ε. In fact, the choice of the tolerances θ and ε is important to our algorithm. Next section, we give some analysis of them.

4. The analysis of θ and ε

As what we say in Example 1, θ is an important parameter in deciding if a polynomial is θ-regular at $\tilde{\mathbf{p}}$. The other important parameter involved in our actual computation is ε, which is used to judge the numerical rank of the Jacobian matrix. Therefore, in this section, we will give some analysis about the parameters θ and ε.

4.1. The analysis of θ

First, we point out that θ is related to the absolute values of the coefficients of the Taylor expansion of the polynomial at its approximate zero.

For example, given a polynomial $f = x^2 + 10000\,y^2$ with an approximate zero

$$\tilde{\mathbf{p}} = (\tilde{p}_1, \tilde{p}_2) = (0.0006851, -0.0004368),$$

we have the Taylor expansion of f at $\tilde{\mathbf{p}}$:

$$f = 0.001908411762 + 0.0013702(x - \tilde{p}_1) - 8.7360(y - \tilde{p}_2) + (x - \tilde{p}_1)^2 + 10000(y - \tilde{p}_2)^2.$$

Given $\theta = 0.5$, we have

$$|f(\tilde{\mathbf{p}})| < \theta, |\frac{\partial f}{\partial x}(\tilde{\mathbf{p}})| < \theta, |\frac{\partial f}{\partial y}(\tilde{\mathbf{p}})| > \theta.$$

Thus, we draw the conclusion that f is θ-regular at $\tilde{\mathbf{p}}$. However, considering that the lowest degree of f is 2, we know that f is singular at the exact zero $\mathbf{p} = (0, 0)$ actually, which is a different result from the case of $\tilde{\mathbf{p}}$. That means θ is not chosen properly. The main reason is that the coefficient of f has a great fluctuation or the accuracy of $\tilde{\mathbf{p}}$ is not high enough. If given another approximate zero $\tilde{\mathbf{q}} = (\tilde{q}_1, \tilde{q}_2) = (0.000006851, -0.000004368)$ with higher precision, we have:

$$f = 1.908411762 \cdot 10^{-7} + 0.000013702(x - \tilde{q}_1) - 0.087360(y - \tilde{q}_2) \\ + (x - \tilde{q}_1)^2 + 10000(y - \tilde{q}_2)^2.$$

By this time, using the same $\theta = 0.5$, we have f is θ-singular at $\tilde{\mathbf{q}}$, which is the same judgement as the exact case of $\mathbf{p}$.

In actual computation, to deal with this case, we give one solution: For a nonzero polynomial $f \in \mathbb{C}[\mathbf{x}]$, let Γ_f be a set of the absolute values of all the coefficients of f. We denote the maximal and minimal ones inside Γ_f as $M = \max(\Gamma_f)$ and $m = \min(\Gamma_f)$ respectively. If $m/M \le 10^{-a}$, we regard that the coefficients of f fluctuate a lot and take $\epsilon = (m + M)/2M$; Else, we take $\theta = (m + M)/(2M \times 10^a)$, where $a \in \mathbb{N}$ is related to the precision of the given approximate zero $\tilde{\mathbf{p}}$. For example, if the accuracy of the given approximate zero $\tilde{\mathbf{p}}$ has three significant digits, we can take $a = 3$. Of course, we can overcome this problem thoroughly by refining the approximate zero to a higher precision with the input system if the Jacobian matrix of the system at $\tilde{\mathbf{p}}$ is numerically nonsingular.

In summary, the reason for the above situation is that we judge a polynomial, which is singular at the exact zero $\mathbf{p}$, as a polynomial being θ-regular at the approximate zero $\tilde{\mathbf{p}}$.

The other situation is that a polynomial, which is regular at the exact zero $\mathbf{p}$, may be judged as a polynomial being θ-singular at the approximate zero $\tilde{\mathbf{p}}$.

For example, consider the polynomial $f = \frac{1}{20}x + x^2 + 10000y^2$ with the approximate zero $\tilde{\mathbf{q}} = (\tilde{q}_1, \tilde{q}_2) = (0.000006851, -0.000004368)$. We have:

$$f = 5.333911762 \cdot 10^{-7} + 0.05001370(x - \tilde{q}_1) - 0.087360(y - \tilde{q}_2) + (x - \tilde{q}_1)^2 + 10000(y - \tilde{q}_2)^2.$$

Still use $\theta = 0.5$ and we get the judgement that f is θ-singular at $\tilde{\mathbf{q}}$. In fact, f is regular at $\mathbf{p} = (0, 0)$. One way to deal with this case is that we can take a smaller θ. When we take $\theta = 0.05$, we will acquire the appropriate result.

From the above analysis about the tolerance θ, we know that the choice of θ is crucial to our method. We give a further theoretical analysis about the tolerance θ below. Here, we assume that the judgement of the other tolerance ε, which is used to decide the numerical rank of the Jacobian matrix at the approximate zero, is correct.

Let θ be a tolerance. Assume that we have computed an intermediate system $\mathbf{H} = \{h_1, \dots, h_s\} \subset \mathbb{C}[\mathbf{x}']$. Denote $\mathbf{x}' = (\mathbf{x}, \boldsymbol{\alpha})$. Assume that $\mathbf{p}$ is an isolated singular zero of the original system. The exact value of $\boldsymbol{\alpha}$ related to the coefficients of linear combinations is $\hat{\boldsymbol{\alpha}}$. Denote $\mathbf{p}' = (\mathbf{p}, \hat{\boldsymbol{\alpha}})$. Let $\tilde{\mathbf{p}}'$ be an approximate zero of $\mathbf{H}$ related to $\mathbf{p}'$ such that

$$\mathrm{rank}(\mathbf{J}(\mathbf{H})(\tilde{\mathbf{p}}')) = s.$$

Next, we consider one more polynomial $h \in \mathbb{C}[\mathbf{x}']$. If h, which is regular at $\mathbf{p}'$, is judged as being θ-singular at $\tilde{\mathbf{p}}'$, we may get a perturbed system finally. Specifically, compute the Taylor expansion of h at $\tilde{\mathbf{p}}'$:

$$h = h(\tilde{\mathbf{p}}') + \sum_j \frac{\partial h(\tilde{\mathbf{p}}')}{\partial x_j}(x_j - \tilde{p}'_j) + \sum_{i,j} \frac{\partial h^2(\tilde{\mathbf{p}}')}{\partial x_i \partial x_j}(x_i - \tilde{p}'_i)(x_j - \tilde{p}'_j) + \cdots.$$

Since h is θ-singular at $\tilde{\mathbf{p}}'$, we know that $|h(\tilde{\mathbf{p}}')| < \theta$ and all $|\frac{\partial h(\tilde{\mathbf{p}}')}{\partial x_j}| < \theta$. Thus, we compute

$$\frac{\partial h}{\partial x_j} = \frac{\partial h(\tilde{\mathbf{p}}')}{\partial x_j} + 2\sum_i \frac{\partial^2 h(\tilde{\mathbf{p}}')}{\partial x_i \partial x_j}(x_i - \tilde{\mathbf{p}}_i') + \cdots. \tag{7}$$

If there exists some j such that

$$\operatorname{rank}(\mathbf{J}(\mathbf{H}, \frac{\partial h}{\partial x_j})(\tilde{\mathbf{p}}')) = s + 1 \text{ and } \frac{\partial h(\mathbf{p}')}{\partial x_j} \neq 0,$$

we may derive a perturbed system in the end, where $\frac{\partial h}{\partial x_j}$ has and only has one perturbed term $\frac{\partial h(\mathbf{p}')}{\partial x_j}$ compared to the polynomial $\frac{\partial h}{\partial x_j} - \frac{\partial h(\mathbf{p}')}{\partial x_j}$ which vanishes at $\mathbf{p}'$.

For other cases, if

$$\operatorname{rank}(\mathbf{J}(\mathbf{H}, \frac{\partial h}{\partial x_j})(\tilde{\mathbf{p}}')) = s + 1 \text{ and } \frac{\partial h(\mathbf{p}')}{\partial x_j} = 0,$$

it is clear that $\frac{\partial h}{\partial x_j}$ vanishes at $\mathbf{p}'$. Thus it is exact. If

$$\operatorname{rank}(\mathbf{J}(\mathbf{H}, \frac{\partial h}{\partial x_j})(\tilde{\mathbf{p}}')) = s(\forall j),$$

according to our constructive method, we should do the linear combination

$$f = \frac{\partial h}{\partial x_j} + \sum_{i=1}^{s} \alpha_i h_i \text{ (for some } j)$$

and compute its derivatives. Thus the perturbed term $\frac{\partial h(\mathbf{p}')}{\partial x_j}$ disappears. We will get an exact polynomial which vanishes at $\mathbf{p}'$ in the end. Notice that if h_i's have perturbed terms, which are constants $h_i(\mathbf{p}')$. We know that if we compute the derivatives of f, these terms will disappear. Thus whether h_i's have perturbed terms or not, the polynomials in the final deflated system derived by the linear combinations vanish at the exact zero $\mathbf{p}'$.

Now we consider the case that h is regarded as θ-regular at $\tilde{\mathbf{p}}'$ while it is singular at $\mathbf{p}'$. If

$$\operatorname{rank}(\mathbf{J}(\mathbf{H}, h)(\tilde{\mathbf{p}}')) = s,$$

we will do the linear combination of h and $h_1, \ldots, h_s$ and compute its derivatives. It is obvious that this operation has no influence on our result. Usually the case

$$\operatorname{rank}(\mathbf{J}(\mathbf{H}, h)(\tilde{\mathbf{p}}')) = s + 1$$

will not happen. It is related to the numerical computation of the rank of the Jacobian matrix of $(\mathbf{H}, h)$ at $\tilde{\mathbf{p}}'$.

As a summary of the foregoing analysis, we have:

Let $\mathbf{F} = \{f_1, \ldots, f_n\} \subset \mathbb{C}[\mathbf{x}]$ be a polynomial system. $\tilde{\mathbf{p}} \in \mathbb{C}^n$ is an approximate zero of $\mathbf{F} = \mathbf{0}$ and θ is a tolerance. According to our method, we acquire a final system $\widetilde{\mathbf{F}}' \subset \mathbb{C}[\mathbf{x}, \boldsymbol{\alpha}]$. During we compute the final system $\widetilde{\mathbf{F}}'$,

1. if we judge a polynomial, which is singular at the exact zero $\mathbf{p}$, as being θ-regular at $\tilde{\mathbf{p}}$, the final system $\widetilde{\mathbf{F}}'$ is accurate.
2. if we judge a polynomial, which is regular at the exact zero $\mathbf{p}$, as being θ-singular at $\tilde{\mathbf{p}}$, the final system $\widetilde{\mathbf{F}}' = \widetilde{\mathbf{F}} + \boldsymbol{\vartheta}$, is a perturbed system, where $\widetilde{\mathbf{F}}$ is an accurate system and $\boldsymbol{\vartheta}$ is the perturbed term, which satisfies $\max_i |\vartheta_i| < \theta$.

In actual computation, to make our method as accurate as possible, we give an adaptive adjustment step at the end of our algorithm. To be specific, assume that the initial tolerance $\theta = \theta_1$. We use Newton-type methods [24–26] on the final system $\widetilde{\mathbf{F}}'$ to refine $\tilde{\mathbf{p}}$ to a higher accuracy. After the refining steps, denote the refined zero as $\bar{\mathbf{p}}$. We compute the Taylor expansions of all the related polynomials in computing the system $\widetilde{\mathbf{F}}'$ at $\bar{\mathbf{p}}$, including all the input polynomials. We denote the maximal absolute value of both the coefficients of the polynomials, which are judged as θ_1-singular at $\tilde{\mathbf{p}}$ and the polynomials, which are judged as θ_1-regular at $\tilde{\mathbf{p}}$, as θ_2. It is also the term named "Max err" in Tables 1 and 2 in the next section.

It is easy to imagine that $\theta_2 \leq \theta_1$ usually. If θ_2 has a very higher precision than θ_1, such as $\theta_1 = 10^{-2}$ and $\theta_2 = 10^{-13}$, we are sure that our conclusion is exact. If $\theta_2 > \theta_1$ or θ_2 still has a bad accuracy, such as $\theta_1 = 10^{-2}$ and $\theta_2 = 10^{-1}$ or $\theta_2 = 10^{-4}$, we will take a smaller $\theta < \min\{\theta_1, \theta_2\}$ and repeat our method again.

After repeating our method several times, if θ_2 is still bad, we will merely get a perturbed system.

Now, we give two examples to explain the above analysis of θ.

Example 9. Given a polynomial system $\mathbf{F} = \{f_1 = x + x^2 + 10000y^2, f_2 = x^2 + 10000y^2\}$ with an approximate zero

$$\tilde{\mathbf{p}} = (\tilde{p}_1, \tilde{p}_2) = (0.0006851, -0.0004368).$$

Consider the tolerances $\varepsilon = 0.05$ and $\theta = 0.5$. By the Taylor expansions of f_i at $\tilde{\mathbf{p}}$, we know that f_1, f_2 are both θ-regular at $\tilde{\mathbf{p}}$.

Next, according to Algorithm 1, we compute

$$\operatorname{rank}(\mathbf{J}(\mathbf{F})(\tilde{\mathbf{p}}), \varepsilon) = 2.$$

Thus, we can use Newton's method to refine $\tilde{\mathbf{p}}$ to a higher accuracy and get

$$\tilde{\mathbf{p}}' = (0.0000000001, -0.0000008533).$$

At this time, it is easy to check that f_1 is θ-regular at $\tilde{\mathbf{p}}'$ and f_2 is θ-singular at $\tilde{\mathbf{p}}'$. Therefore, for f_2, we have

$$\frac{\partial f_2}{\partial x} = 2x, \ \frac{\partial f_2}{\partial y} = 20000y,$$

which are both θ-regular at $\tilde{\mathbf{p}}'$. Furthermore,

$$\operatorname{rank}(\mathbf{J}(f_1, \frac{\partial f_2}{\partial y}), \varepsilon) = 2.$$

Thus, we get the final system $\widetilde{\mathbf{F}}' = \{f_1, \ 20000y\}$. After applying Newton's method, we get the refined zero $\bar{\mathbf{p}} = (\bar{p}_1, \bar{p}_2) = 10^{-16} \cdot (0.53016, 0)$.

At last, we check if our chosen θ is proper. We compute the Taylor expansion of all the polynomials, which is judged as θ-singular at $\tilde{\mathbf{p}}$, at the refined zero $\bar{\mathbf{p}}$ and get:

$$f_2 = 2.810696256 \cdot 10^{-33} + 1.060320 \cdot 10^{-16} \cdot (x - \bar{p}_1) + (x - \bar{p}_1)^2 + 20000 \cdot (y - \bar{p}_1)^2.$$

Thus, we have

$$\text{Max err} := \max\{2.810696256 \cdot 10^{-33}, 1.060320 \cdot 10^{-16}\} = 1.060320 \cdot 10^{-16} \ll \theta,$$

which means that our final system $\widetilde{\mathbf{F}}'$ is more accurate than before.

Example 10. Consider the system $\mathbf{F} = \{f_1 = x + x^2 + 2xy + 10000y^2, f_2 = \frac{1}{20}x + x^2 + 2xy + 10000y^2\}$ with an approximate zero

$$\tilde{\mathbf{p}} = (\tilde{p}_1, \tilde{p}_2) = (0.000006851, -0.000004368).$$

Let the tolerances $\varepsilon = 0.05$ and $\theta = 0.5$. Similarly, by the Taylor expansions of f_i at $\tilde{\mathbf{p}}$, we know that f_1 is θ-regular at $\tilde{\mathbf{p}}$ and f_2 is θ-singular at $\tilde{\mathbf{p}}$. Therefore, we have

$$\frac{\partial f_2}{\partial x} = \frac{1}{20} + 2x + 2y, \ \frac{\partial f_2}{\partial y} = 2x + 20000y.$$

Compute

$$\operatorname{rank}(\mathbf{J}(f_1, \frac{\partial f_2}{\partial x}), \varepsilon) = 2.$$

Thus, we get the final system

$$\widetilde{\mathbf{F}}'_1 = \{f_1, \ \frac{1}{20} + 2x + 2y\}.$$

Obviously, $\widetilde{\mathbf{F}}'_1$ is a perturbed system and $\vartheta_2 = \frac{1}{20}$ is the perturbed term, which satisfies $|\vartheta_2| < \theta$. It is easy to imagine that with $\widetilde{\mathbf{F}}'_1$, we could not get a good result. The main reason is that $\theta = 0.5$ is too big, which leads to a wrong judgement on whether f_2 is θ-regular at $\tilde{\mathbf{p}}$.

If given another smaller tolerance $\theta' = 0.05$, we will get a right judgement that f_2 is θ'-regular at $\tilde{\mathbf{p}}$. Thus, we consider the linear combination of f_1 and f_2. Let $f = f_2 + \alpha f_1$ and compute

$$\begin{aligned} g_1 = \frac{\partial f}{\partial x} &= \frac{1}{20} + 2x + 2y + \alpha(2x + 2y + 1), \\ g_2 = \frac{\partial f}{\partial y} &= 2x + 20000y + \alpha(2x + 20000y), \end{aligned}$$

where α is a new variable and its initial value $\tilde{\alpha} = -0.050076986$. Compute

$$\operatorname{rank}(\mathbf{J}(f_1, g_1, g_2), \varepsilon) = 3.$$

Thus, we get the final system $\widetilde{\mathbf{F}}' = \{f_1, g_1, g_2\}$. Similarly, we consider applying Newton's method on the final system $\widetilde{\mathbf{F}}'$ and get the refined zero:

$$\bar{\mathbf{p}} = (0.000000000000000, 0.000000000000000, -0.050000000000000).$$

Then, we check the coefficients of the terms with degree one of the Taylor expansion of f at $\bar{\mathbf{p}}$ and get

$$\text{Max err} := \{0, 0, 0\} = 0 \ll \theta' = 0.05.$$

Thus, we are sure that our final system $\widetilde{\mathbf{F}}'$ is accurate. Here, "0" is not exact zero but means in Matlab machine accuracy.

From the above two examples, we can see that once given an appropriate tolerance θ, we can make sure that our final system is accurate. Otherwise, what we acquired is just a perturbed system, such as the system $\widetilde{\mathbf{F}}'_1$ in Example 10.

4.2. The analysis of ε

We continue analysing the other tolerance ε, which is used to judge the numerical rank of a matrix. That is, we determine the numerical rank by comparing the absolute values of the singular values of the Jacobian matrix at approximate zero with the tolerance ε. Specifically, assume that we have computed an intermediate system $\mathbf{H} = \{h_1, \dots, h_s\} \subset \mathbb{C}[\mathbf{x}']$. Denote $\mathbf{x}' = (\mathbf{x}, \boldsymbol{\alpha})$. Assume that $\mathbf{p}$ is an isolated singular zero of the original system. The exact value of $\boldsymbol{\alpha}$ related to the coefficients of linear combinations is $\hat{\boldsymbol{\alpha}}$. Denote $\mathbf{p}' = (\mathbf{p}, \hat{\boldsymbol{\alpha}}) \in \mathbb{C}^t$. Let $\tilde{\mathbf{p}}' \in \mathbb{C}^t$ be an approximate zero of $\mathbf{H}$ related to $\mathbf{p}'$ such that

$$\text{rank}(\mathbf{J}(\mathbf{H})(\tilde{\mathbf{p}}'), \varepsilon) = s.$$

Next, we consider one more polynomial $h_{s+1} \in \mathbb{C}[\mathbf{x}']$. Given the tolerance θ, we can compute a polynomial h from h_{s+1}, which is θ-regular at $\tilde{\mathbf{p}}'$. Denote

$$\text{rank}(\mathbf{J}(h_1, \dots, h_s, h)(\mathbf{p}')) = r_1, \quad \text{rank}(\mathbf{J}(h_1, \dots, h_s, h)(\tilde{\mathbf{p}}'), \varepsilon) = r_2.$$

For simplicity, we denote the deflated system as $\mathbf{H}'$, which comes from $\{\mathbf{H}, h_{s+1}\}$ after one step deflation, and its corresponding exact zero as $\mathbf{q}$, whose partial projection is $\mathbf{p}'$.

According to the above analysis of θ, for h_{s+1}, we have the following cases:

1. if θ is chosen properly, that is, we judge h_{s+1}, which is regular or singular at $\mathbf{p}'$, as being θ-regular or θ-singular at $\tilde{\mathbf{p}}'$ respectively, we know that h is regular at $\mathbf{p}'$. Thus, we have:

   (a) if $r_2 = r_1$, we, of course, get an exact system $\mathbf{H}'$. That is, $\mathbf{H}'(\mathbf{q}) = \mathbf{0}$.
   (b) if $r_2 < r_1$, according to our algorithm, we consider doing the linear combination:

   $$g = h + \sum_{j=1}^{s} \alpha_j h_j$$

   and compute all the derivatives of g with respect to all variables: $g_i = \frac{\partial g}{\partial x'_i},\ i = 1, \dots, t$. Correspondingly, $\mathbf{H}' = \{h_1, \dots, h_s, g_1, \dots, g_t\}$. Note that $\mathbf{J}(h)(\mathbf{p}')$ and $\mathbf{J}(h_1)(\mathbf{p}'), \dots, \mathbf{J}(h_s)(\mathbf{p}')$ are actually linearly independent, which means that the equations $(\alpha_1, \dots, \alpha_s, 1)\mathbf{J}(\mathbf{H})(\mathbf{p}') = \mathbf{0}$ has no solution. Thus, although we can give the initial value $\tilde{\alpha}_j$ of α_j by solving a Least Squares problem, the linear independence will bring us some inexact polynomials g_i, which means $g_i(\mathbf{q}') \neq 0$. Further, we may get a perturbed system $\mathbf{H}'$. That is, $\mathbf{H}'(\mathbf{q}) \neq \mathbf{0}$.
   (c) if $r_1 < r_2$, we will add h to the system $\mathbf{H}$ directly and get an exact system $\mathbf{H}' = \{h_1, \dots, h_s, h\}$.

2. if θ is chosen too big, that is, we judge h_{s+1}, which is regular at $\mathbf{p}'$, as being θ-singular at $\tilde{\mathbf{p}}'$, we may get a perturbed polynomial h, which means that $h(\mathbf{p}') \neq 0$ and $h - h(\mathbf{p}')$ is regular at $\mathbf{p}'$. Thus, we have:

   (a) if $r_2 = r_1$, only the choice of θ affects our final result. Thus, we may get a perturbed system $\mathbf{H}'$ in this case.
   (b) if $r_2 < r_1$, with a similar discussion with the case of 1(b), we get a perturbed system $\mathbf{H}'$.
   (c) if $r_1 < r_2$, we add h to $\{h_1, \dots, h_s\}$ directly and get a perturbed system $\mathbf{H}'$.

3. if θ is chosen too small, that is, we judge h_{s+1}, which is singular at $\mathbf{p}'$, as being θ-regular at $\tilde{\mathbf{p}}'$, we know that $h = h_{s+1}$ is singular at $\mathbf{p}'$. Thus, we have:

   (a) if $r_2 = r_1$, only the choice of θ affects our result. Thus, the system $\mathbf{H}'$ is exact in this case.
   (b) if $r_2 < r_1$, similar to the case of 1(b), we consider doing the linear combination:

   $$g = h + \sum_{j=1}^{s} \alpha_j h_j.$$

   One difference from 1(b) is that h is singular at $\mathbf{p}'$. Thus, all the $g_i = \frac{\partial g}{\partial x'_i}$ are exactly vanished at $\mathbf{q}$. So, the system $\mathbf{H}'$ is also exact in this case.
   (c) if $r_1 < r_2$, we add h to $\{h_1, \dots, h_s\}$ directly and get an exact system $\mathbf{H}'$.

With the above analysis, we know that the choice of the tolerances θ and ε has an influence on our final deflated system: an exact deflated system or a perturbed system. To judge which case a final deflated system belongs to, we use the following judgement method:

Denote the input system as $\mathbf{F} = \{f_1, \dots, f_n\}$, the final deflated system $\widetilde{\mathbf{F}}'$. Noticing that we use Newton's method to refine the system $\widetilde{\mathbf{F}}'$, thus, we denote Newton's iteration sequence as $\{\tilde{\mathbf{p}}_l,\ l \geq 1\}$ and the final certified zero $\tilde{\mathbf{p}}'$.

- First, we check if Newton's iteration sequence $\{\tilde{\mathbf{p}}_l,\ l \geq 1\}$ is quadratic convergence. If not, we claim that our deflated system is a perturbed system.
- If it is, we compute

$$\Delta := \max\{|f_i(\tilde{\mathbf{p}}')|\ |f_i \in \mathbf{F},\ i = 1, \dots, n\}.$$

- Next, we give a tolerance θ', which is usually a very small value, and compare the magnitude of θ' and Δ. If $\Delta < \theta'$, we regard the final deflated system $\widetilde{\mathbf{F}}'$ as an exact system; otherwise, $\widetilde{\mathbf{F}}'$ is a perturbed system.

Of course, for the exact case, we are done. For the perturbed case, we hope to make our final deflated system as accurate as possible by adjusting the values of θ and ε. However, we still do not have a good idea on how to distinguish the effect of the two tolerances θ and ε on the final system. Fortunately, noting that the tolerance θ is used to accelerate our algorithm and is not necessary, therefore, according to the remark of Lemma 4, we can use the deflation construction (6) to compute the final system. In this case, we just need to consider the tolerance ε, which is used to judge the numerical rank of the Jacobian matrix. That is to say, even if the first two steps of Algorithm 1 are removed, our deflation construction process can still work well.

Considering the possible judgement, our final system can also be a perturbed system. Next, we give a possible modified method to overcome this case.

let $\mathbf{F} = \{f_1, \dots, f_n\}$ be the input system, $\tilde{\mathbf{p}} \in \mathbb{C}^n$ be the initial approximate zero. Let ε and θ' be the given tolerances.

- First, assume $\operatorname{rank}(\mathbf{J}(\mathbf{F})(\tilde{\mathbf{p}}), \varepsilon) = n$. We apply Newton's method on the system $\mathbf{F}$ and get the refined zero $\tilde{\mathbf{p}}'$. Next, we check if Newton's iteration sequence is quadratic convergence. If it is, we continue comparing the magnitude of θ' and Δ. If $\Delta < \theta'$, we regard $\mathbf{F}$ as a system with an isolated simple zero; otherwise, we know $\mathbf{F}$ is a system with a multiple zero and $\operatorname{rank}(\mathbf{J}(\mathbf{F})(\tilde{\mathbf{p}}), \varepsilon) < n$.
- Assume $\operatorname{rank}(\mathbf{J}(\mathbf{F})(\tilde{\mathbf{p}}), \varepsilon) = n - 1$. After using the deflation construction in Algorithm 1 once(from step 3 to step 10), we get a deflation system $\widetilde{\mathbf{F}}_1$ and an approximate zero $\tilde{\mathbf{p}}_1$. Then, we consider all the possibilities of $\operatorname{rank}(\mathbf{J}(\widetilde{\mathbf{F}}_1)(\tilde{\mathbf{p}}_1), \varepsilon)$. For every case, we go on our deflation construction in Algorithm 1 and use our mentioned judging method to check which case the final deflated system belongs to. As long as the final deflated system is judged to be an exact system, we will stop our deflation process; Otherwise, we know $\operatorname{rank}(\mathbf{J}(\mathbf{F})(\tilde{\mathbf{p}}), \varepsilon) < n - 1$.
- Assume $\operatorname{rank}(\mathbf{J}(\mathbf{F})(\tilde{\mathbf{p}}), \varepsilon) = n - 2$. We consider as the case of $n - 1$.

About the above judgement process, we have two things to say:

1. The above judgement process must terminate in finite steps considering that our deflation construction terminates in finite steps.
2. In the above judgement process, we traverse all the possibilities of the rank of the Jacobian matrix. Thus, there must be at least one case that we get an exact deflated system.

Now, we give an example below to illustrate our idea.

Example 11. Continue with Example 10. Here, we only use the tolerance $\varepsilon = 0.05$ to judge the numerical rank. First, we compute

$$\operatorname{rank}(\mathbf{J}(f_1, f_2)(\tilde{\mathbf{p}}), \varepsilon) = 2.$$

Thus, we consider using Newton's method on the system $\mathbf{F}$ directly. Given an iterative error 10^{-8}, we have the following Newton's iteration sequence:

$\mathbf{p}_i$	x	y
$\tilde{\mathbf{p}}_1$	0.000006851	−0.000004368
$\tilde{\mathbf{p}}_2$	0.0000000000000	−0.0000021841948
$\tilde{\mathbf{p}}_3$	−0.0000000000000	−0.0000010920974
$\tilde{\mathbf{p}}_4$	−0.0000000000000	−0.0000005460487
$\tilde{\mathbf{p}}_5$	−0.0000000000000	−0.0000002730243
$\tilde{\mathbf{p}}_6$	−0.0000000000000	−0.0000001365122
$\tilde{\mathbf{p}}_7$	−0.0000000000000	−0.0000000682561
$\tilde{\mathbf{p}}_8$	0.0000000000000	−0.0000000341280
$\tilde{\mathbf{p}}_9$	−0.0000000000000	−0.0000000170640
$\tilde{\mathbf{p}}_{10}$	−0.0000000000000	−0.0000000853201

We can check easily that Newton's iteration sequence $\{\tilde{\mathbf{p}}_j,\ j = 1, \dots, 10\}$ is linear convergence. According to our judging criteria, we know that

$$\mathrm{rank}(\mathbf{J}(f_1, f_2)(\tilde{\mathbf{p}}), \varepsilon) = 1.$$

Next, according to our construction process in Algorithm 1, we let

$$g = f_2 + \alpha f_1$$

and compute

$$g_1 = \mathbf{J}_1(g) = \alpha(2x + 2y + 1) + (1/20 + 2x + 2y),\ \ g_2 = \alpha(2x + 20000y) + (2x + 20000y).$$

We have $\tilde{\alpha} = -0.091484814324$.

Next, let $\widetilde{\mathbf{F}}_1 = \{f_1, g_1, g_2\}$ and $\tilde{\mathbf{p}}_1 = (\tilde{\mathbf{p}}, \tilde{\alpha})$. By our given revised method above, we continue considering all the possibilities of $\mathrm{rank}(\mathbf{J}(\widetilde{\mathbf{F}}_1)(\tilde{\mathbf{p}}_1), \varepsilon)$. For example, we consider the case of

$$\mathrm{rank}(\mathbf{J}(\widetilde{\mathbf{F}}_1)(\tilde{\mathbf{p}}_1), \varepsilon) = 3.$$

Similarly, given the iterative error 10^{-8}, by using Newton's method on the system $\widetilde{\mathbf{F}}_1$, we get the following iteration sequence:

$\mathbf{p}_i$	x	y	α
$\tilde{\mathbf{p}}_1$	0.000006851	−0.000004368	−0.091484814324
$\tilde{\mathbf{p}}_2$	0.0000002081968	0.0000001993959	−0.0500009466172
$\tilde{\mathbf{p}}_3$	0.0000000003977	−0.0000000000002	−0.0500000007560
$\tilde{\mathbf{p}}_4$	0.0000000000000	0.0000000000000	−0.0500000000000

It is easy to check that the iteration sequence $\{\tilde{\mathbf{p}}_j, j = 1, 2, 3, 4\}$ is quadratic convergence. Furthermore, given a tolerance $\theta' = 10^{-12}$, we can compute

$$\Delta := \max\{|f_1(\tilde{\mathbf{p}}_4)|,\ |f_2(\tilde{\mathbf{p}}_4)|\} = 0$$

and verify that $\Delta < \theta'$. Thus, we regard the final deflated system $\widetilde{\mathbf{F}}' = \widetilde{\mathbf{F}}_1$ as an exact system. At the same time, we stop our deflation process.

Until now, we have finish all the discussions about the tolerances θ and ε. Once given a polynomial system $\mathbf{F} \subset \mathbb{C}[\mathbf{x}]$ with an isolated singular zero, we use Algorithm 1 to compute a new system $\widetilde{\mathbf{F}}'(\mathbf{x}, \boldsymbol{\alpha})$, which has a simple zero. What is more, according to the analysis of the tolerances θ and ε, our final system $\widetilde{\mathbf{F}}'(\mathbf{x}, \boldsymbol{\alpha})$ is an accurate system usually. For the perturbed case, we also give one ergodic way to adjust our final result as accurate as possible. Thus, we can use the final system $\widetilde{\mathbf{F}}'(\mathbf{x}, \boldsymbol{\alpha})$ to certify the isolated zeros of the input system.

In the following, by using the algorithm **verifynlss** in INTLAB [27], we give Algorithm 2 to verify the isolated singular zeros of the input system heuristically. In the verification steps, we employ the algorithm **verifynlss** in INTLAB for computing two inclusions $\mathbf{X} = ([\underline{x}_1, \overline{x}_1], \dots, [\underline{x}_n, \overline{x}_n])$ and $\boldsymbol{\mathcal{A}} = ([\underline{\alpha}_1, \overline{\alpha}_1], \dots, [\underline{\alpha}_n, \overline{\alpha}_n])$ for the simple zero of the deflated system.

Algorithm 2 VDSS : Verifying the deflated square system.

```
Input:
    a polynomial system $\mathbf{F} := \{f_1, \dots, f_n\} \subset \mathbb{C}[\mathbf{x}]$, an approximate isolated zero $\tilde{\mathbf{p}} = (\tilde{p}_1, \dots, \tilde{p}_n) \in \mathbb{C}^n$, a tolerance $\varepsilon$.
Output:
    a deflated system: $\widetilde{\mathbf{F}}(\mathbf{x}, \boldsymbol{\alpha}) := \{\tilde{f}_1, \dots, \tilde{f}_t\} \subset \mathbb{C}[\mathbf{x}, \boldsymbol{\alpha}]$, two inclusions $\mathbf{X}$ and $\boldsymbol{\mathcal{A}}$;
$(\widetilde{\mathbf{F}}', \tilde{\boldsymbol{\alpha}}) :=$ CDSS$(\mathbf{F}, \tilde{\mathbf{p}}, \varepsilon)$ ;
$[\mathbf{X}, \boldsymbol{\mathcal{A}}] :=$ verifynlss$(\widetilde{\mathbf{F}}', (\tilde{\mathbf{p}}, \tilde{\boldsymbol{\alpha}}))$;
Return: a deflated system $\widetilde{\mathbf{F}}(\mathbf{x}, \boldsymbol{\alpha}) := \widetilde{\mathbf{F}}'$, two inclusions $\mathbf{X}$ and $\boldsymbol{\mathcal{A}}$.
```

Now, we give an example in the following to explain how we certify the isolated singular zero of the input system heuristically.

Example 12. Continue with Example 7. Applying Algorithm 2, we get the system:

$$\widetilde{\mathbf{F}}(\mathbf{x},\boldsymbol{\alpha}) = \begin{cases} \tilde{f}_1 = -\dfrac{9}{4} + \dfrac{3}{2}x_1 + 2x_2 + 3x_3 + 4x_4 - \dfrac{1}{4}x_1^2, \\ \tilde{f}_2 = x_1 - 2x_2 - 2x_3 - 4x_4 + 2x_1x_2 + 3x_1x_3 + 4x_1x_4, \\ \tilde{f}_3 = -8 + 4x_1 + 4x_4 - 2x_1x_4, \\ \tilde{f}_4 = 3 + \dfrac{3}{2}\alpha_1 + \alpha_2 + 4\alpha_3 - \dfrac{1}{2}\alpha_1x_1 + 2\alpha_2x_2 + 3\alpha_2x_3 + 4\alpha_2x_4 - 2\alpha_3x_4, \\ \tilde{f}_5 = 2 + 2\alpha_1 - 2\alpha_2 + 2\alpha_2x_1, \\ \tilde{f}_6 = 4 + 3\alpha_1 - 2\alpha_2 + 3\alpha_2x_1, \\ \tilde{f}_7 = 4 + 4\alpha_1 - 4\alpha_2 + 4\alpha_3 + 4\alpha_2x_1 - 2\alpha_3x_1, \end{cases}$$

and two verified inclusions

$$\mathbf{X} = \begin{bmatrix} [\ 0.99999999999999, & 1.00000000000001] \\ [-2.00000000000001, & -1.99999999999998] \\ [-1.00000000000001, & -0.99999999999999] \\ [\ 1.99999999999999, & 2.00000000000001] \end{bmatrix}$$

and

$$\boldsymbol{\mathcal{A}} = \begin{bmatrix} [-1.00000000000001, & -0.99999999999999] \\ [-1.00000000000001, & -0.99999999999999] \\ [-0.00000000000001, & -0.00000000000001] \end{bmatrix}.$$

For the deflated system $\widetilde{\mathbf{F}}(\mathbf{x},\boldsymbol{\alpha})$, we affirm that there is a unique isolated simple zero $(\hat{\mathbf{x}},\hat{\boldsymbol{\alpha}}) \in (\mathbf{X},\boldsymbol{\mathcal{A}})$, such that $\widetilde{\mathbf{F}}(\hat{\mathbf{x}},\hat{\boldsymbol{\alpha}}) = \mathbf{0}$. What is more, the projection $\hat{\mathbf{x}}$ of $(\hat{\mathbf{x}},\hat{\boldsymbol{\alpha}})$ corresponds to the isolated singular zero of the input system $\mathbf{F}$. That is to say, we certified the isolated singular zeros of the original system.

5. Experiments and results

We implement our method in Matlab of Algorithm 2. The code and some examples can be found in http://www.mmrc.iss.ac.cn/~jcheng/VDSS. In this section, we show the results of the experiment and the comparison of our method with some other methods. We do the experiments in Matlab R2012b with INTLAB-V5.5 on a computer with Windows 7, Intel *i*7 processor and 8 GB memory.

In [22], by modifying the method proposed by Yamamoto [14], they give a deflation method to compute a regular and square augmented system. Which can be used to prove the existence of an isolated singular solution of a slightly perturbed system. Moreover, by applying INTLAB function **verifynlss** [27], they also give an algorithm **viss** to compute verified error bounds. However, noticing that their method is essentially a deflation method. Thus, we also implement our algorithm based on INTLAB function **verifynlss**.

In Table 1, we compare our algorithm **VDSS** with the algorithm **viss**. These examples are relatively simple and small scale, which can be found in [18,22]. We also list them in http://www.mmrc.iss.ac.cn/~jcheng/VDSS/fun.m. We denote *var* the number of polynomials, *mul* the multiplicity and Verified acc the final verified accuracy, which is measured by the breath of the verified inclusion **X**. And Max err is δ_2 as mentioned in Section 4. We use a same initial accuracy 10^{-4} for all the examples. "true" means we get two same endpoints of the verified inclusion. When the term for Max err is "0", it does not mean Max err is exactly zero and only shows in Matlab machine precision.

From Table 1, we can see that our algorithm is effective. On one hand, the verified accuracy of our method is never worse than **viss** for all these examples. On the other hand, thanks to our acceleration strategies, our practical size and computing time are smaller than those of **viss** in most cases.

We also compare our method with **viss** for large-scale polynomial systems. All the examples in Table 2 can be found in http://www.mmrc.iss.ac.cn/~jcheng/VDSS/example.m. The example LZ2000 can be found in [28]. The example nonpoly3 is a non-polynomial nonlinear system. We construct the other examples as below: First, we produce some polynomials randomly to form a zero-dimensional system $\{f_1,\ldots,f_n\}$, which has a simple zero $\mathbf{p}$ and $\deg(f_i) \geq 2$ usually. The final systems have the form: $\mathbf{F} = \{f_i^{d_i} + g_i, 1 \leq i \leq n, g_i \in \{f_1^{d_1'},\ldots,f_n^{d_n'},0\}, d_i \geq 1, d_i' \geq 1, 1 \leq i \leq n\}$. The new systems are always dense polynomial systems. The examples named simple1, reduce3, big1, big2, big3, large3, large6, large8 are of the form that $g_i = 0 (1 \leq i \leq n)$; The examples named addvar3, unre3, unre5, rankone2, rankone3 are of the form that g_i are not all zeros. The ranks of the Jacobian matrices of the examples rankone2, rankone3 at the zeros both are one. In Table 2, "–" means there is no results with the code.

From Table 2, we can see that for the examples with more variables and high multiplicity, our method has a better result regardless of the verified accuracy, computing time or the final scale.

We also test the example: $\{x_1^3 - x_1^2 - x_2^2, x_2^3 + x_2^2 - x_3, \ldots, x_{n-1}^3 + x_{n-1}^2 - x_n, x_n^2\}$ in [29]. The example named breath2 in Table 2 has this form for $n = 5$. The method in [13] can compute this example for $n = 6$ and it takes 659.59 s with the

Table 1
Comparison of **VDSS** and **viss** for simple systems.

System	*var*	*mul*	Verified acc		Max err	Times		Final size	
			VDSS	viss		VDSS	viss	VDSS	viss
DZ1	4	131	True	e−322	0	0.3066	0.3337	4	16
DZ2	3	16	e−14	e−14	0	0.2989	0.7343	3	24
DZ3	2	4	e−14	e−15	e−14	0.8780	1.0093	3	10
cbms1	3	11	True	e−322	0	0.1851	0.1107	3	6
cbms2	3	8	True	e−322	0	0.2546	0.1271	3	6
mth191	3	4	e−14	e−14	e−32	0.3118	0.1221	4	6
KSS	10	638	e−14	e−14	0	8.2295	0.3036	19	20
RuGr09	2	4	e−323	e−14	0	0.1567	0.4955	2	8
LZ	100	3	e−320	e−14	0	2.0197	13.3068	100	300
Ojika1	2	3	e−14	e−14	0	0.7636	0.3447	5	6
Ojika2	3	2	e−14	e−14	e−16	0.3936	0.2942	5	6
Ojika3	3	2	e−14	e−14	0	0.3967	0.3427	4	6
Ojika4	3	3	e−14	e−14	0	0.1851	1.0621	3	9
Decker2	3	4	e−323	e−14	0	0.1752	0.4650	3	8
Caprasse	4	4	e−14	e−14	e−31	2.0180	0.5126	6	8
Cyclic9	9	4	e−14	e−14	e−15	5.9266	3.6878	12	18

Table 2
Comparison of **VDSS** and **viss** for large systems.

System	*var*	*mul*	Verified acc		Max err	Times		Final size	
			VDSS	viss		VDSS	viss	VDSS	viss
LZ2000	2000	3	e−319	–	0	448.07	–	2000	–
simple1	5	9	e−14	e−14	0	0.29	8.20	5	45
addvar2	4	12	e−14	e−13	e−14	11.10	250.67	6	32
reduce3	4	24	e−14	e−14	e−11	12.21	317.50	7	12
unre3	4	36	e−15	e−14	e−13	4.08	360.32	4	32
unre5	8	576	e−14	e−14	e−13	24.26	229.83	8	64
big1	20	512	e−14	e−15	e−12	29.92	1724.09	20	160
big2	20	8192	e−14	e−14	e−12	40.90	1751.61	20	160
big3	30	196 608	e−15	e−14	e−15	155.18	425.51	30	240
rankone2	6	32	e−15	e−15	e−15	6.8693	1.5199	11	12
rankone3	6	96	e−15	e−14	e−14	12.44	136.54	11	48
breadth2	5	2^5	e−322	–	0	0.20	–	5	–
large3	100	3^{100}	e−323	e−319	0	187.88	647.86	100	400
large6	500	4^{100}	e−321	e−34	0	905.00	3262.78	500	2000
large8	500	4^{300}	e−321	–	0	1745.85	–	500	–
nonpoly3	3	64	e−322	e−14	0	0.19	6.62	3	36

final size for 321 variables and 819 polynomials. We test the cases for $n = 6, n = 1000$ and $n = 2000$ with our code, it takes 0.228965 s, 165.274439 s and 1036.773847 s respectively without introducing new variables.

For our method, although we introduce new variables, the size of our final deflated system is small in experiments. And further, we also compare our method with the other four deflation methods appeared in [13] on the following four systems.

1. $\{x_1^4 - x_2x_3x_4,\ x_2^4 - x_1x_3x_4,\ x_3^4 - x_1x_2x_4,\ x_4^4 - x_1x_2x_3\}$ at $(0, 0, 0, 0)$ with $\mu = 131$;
2. $\{x^4,\ x^2y + y^4, z + z^2 - 7x^3 - 8x^2\}$ at $(0, 0, -1)$ with $\mu = 16$;
3. $\{14x + 33y - 3\sqrt{5}x^2 - 12\sqrt{5}xy - 12\sqrt{5}y^2 - 6\sqrt{5} + x^3 + 6x^2y + 12xy^2 + 8y^3 + \sqrt{7}, 41x - 18y - \sqrt{5} + 8x^3 - 12x^2y + 6xy^2 - y^3 + 12\sqrt{7}xy - 12\sqrt{7}x^2 - 3\sqrt{7}y^2 - 6\sqrt{7}\}$ at $\mathbf{p} \approx (1.5055, 0.36528)$ with $\mu = 5$;
4. $\{2x_1 + 2x_1^2 + 2x_2 + 2x_2^2 + x_3^2 - 1, (x_1 + x_2 - x_3 - 1)^3 - x_1^3, (2x_1^3 + 5x_2^2 + 10x_3 + 5x_3^2 + 5)^3 - 1000x_1^5\}$ at $(0, 0, -1)$ with $\mu = 18$.

The result (see also in [13]) is below, where method A is in [15,18], method B is in [6], method C is in [12], method D is in [13], method E is our method **VDSS**. In Table 3, we denote *Poly* the number of the polynomials of the final deflation system and *Var* the number of the variables in the final deflation system. Noting that our final system does not always contain all the polynomials of the input system, therefore, we will contain the number of the different polynomials in the input system, which is not contained in the final system, into *Poly*.

In Table 3, for system 1, 2 and 4, our method matches the best of the other four methods and simultaneously has a smallest deflated system in the five methods. For system 3, although our final system has one more variable than method D, we have less polynomials.

Table 3
Comparison of VDSS and other methods for four examples.

	Method A		Method B		Method C		Method D		Method E	
	Poly	*Var*	*Poly*	*Var*	*Poly*	*Var*	*Poly*	*Var*	*Poly*	*Var*
1	16	4	22	4	22	4	16	4	8	4
2	24	11	11	3	12	3	12	3	5	3
3	32	17	6	2	6	2	6	2	4	3
4	96	41	54	3	54	3	22	3	5	3

6. Conclusions

In this paper, we develop a new deflation method for refining or verifying the isolated singular zeros of polynomial systems. Given a polynomial system $\mathbf{F} \subset \mathbb{C}[\mathbf{x}]$ with an isolated singular zero $\mathbf{p}$, by computing the derivatives of the input polynomials directly or the linear combinations of the related polynomials, we prove constructively that there exists a final deflated system $\widetilde{\mathbf{F}}'(\mathbf{x}, \boldsymbol{\alpha})$, which has an isolated simple zero $(\mathbf{p}, \hat{\boldsymbol{\alpha}})$, whose partial projection corresponds to the isolated singular zero $\mathbf{p}$ of the input system $\mathbf{F}$. New variables $\boldsymbol{\alpha}$ are introduced to represent the coefficients of the linear combinations of the related polynomials to ensure the accuracy of the numerical implementation.

Compared to the previous deflation methods, on one hand, our method also has an output size depending on the depth or the multiplicity of $\mathbf{p}$ in theory. On the other hand, thanks to the acceleration strategies we proposed in the paper, the size of the final system in our actual computations is much less than that we give in theory. The results of the experiments we conduct give a very persuasive argument for this.

In order to essentially have a deeper understanding of our approach, we also give some further analysis of the tolerances θ and ε we use. The results of the analysis tells us that our final system is a perturbed system with a bounded perturbation in the worst case. To make our final system as accurate as possible, we also analyse the case that the tolerance θ is not introduced.

Acknowledgement

The work is partially supported by National Natural Science Foundation of China (NSFC) Grants 11471327.

References

[1] B. Dayton, T. Li, Z. Zeng, Multiple zeros of nonlinear systems, Math. Comp. 80 (2011) 2143–2168.
[2] M. Giusti, B. Salvy, G. Lecerf, J.-C. Yakoubsohn, On location and approximation of clusters of zeros of analytic functions, Found. Comput. Math. 5 (2005) 257–311.
[3] M. Giusti, G. Lecerf, B. Salvy, J.-C. Yakoubsohn, On location and approximation of clusters of zeros: case of embedding dimension one, Found. Comput. Math. 7 (2007) 1–58.
[4] G. Lecerf, Quadratic newton iteration for systems with multiplicity, Found. Comput. Math. 2 (2002) 247–293.
[5] 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, 16 pages.
[6] J.D. Hauenstein, C.W. Wampler, Isosingular sets and deflation, Found. Comput. Math. 13 (3) (2013) 371–403.
[7] F. Sottile, J.D. Hauenstein, Algorithm 921: alphacertified: Certifying solutions to polynomial systems, ACM Trans. Math. Software Volume 38 Issue 4 (2012).
[8] Z. Zeng, Computing multiple roots of inexact polynomials, Math. Comp. 74 (2005) 869–903.
[9] T. Ojika, Modified deflation algorithm for the solution of singular problems. i. a system of nonlinear algebraic equations, J. Math. Anal. Appl. 123 (1987) 199–221.
[10] T. Ojika, A numerical method for branch points of a system of nonlinear algebraic equations, Appl. Numer. Math. 4 (1988) 419–430.
[11] T. Ojika, S. Watanabe, T. Mitsui, Deflation algorithm for the multiple roots of a system of nonlinear equations, J. Math. Anal. Appl. 96 (1983) 463–479.
[12] M. Giusti, J.-C. Yakoubsohn, Multiplicity hunting and approximating multiple roots of polynomial systems, Contemp. Math. 604 (2013) 105–128.
[13] J.D. Hauenstein, B. Mourrain, A. Szanto, Certifying isolated singular points and their multiplicity structure, in: Proceedings of the International Symposium on Symbolic and Algebraic Computation, ISSAC '2015, ACM, New York, 2015, pp. 213–220.
[14] N. Yamamoto, Regularization of solutions of nonlinear equations with singular jacobian matries, J. Inf. Process. 7 (1984) 16–21.
[15] A. Leykin, J. Verschelde, A. Zhao, Newton's method with deflation for isolated singularities of polynomial systems, Theoret. Comput. Sci. 359 (2006) 111–122.
[16] A. Leykin, J. Verschelde, A. Zhao, Higher-order deflation for polynomial systems with isolated singular solutions, in: A. Dickenstein, F.-O. Schreyer, A. Sommese (Eds.), Algorithms in Algebraic Geometry, in: The IMA Volumes in Mathematics and its Applications, vol. 146, Springer, New York, 2008, pp. 79–97.
[17] J. Verschelde, A. Zhao, Newton's method with deflation for isolated singularities, in: Poster Presented at ISSAC'04, 2004.
[18] B. Dayton, Z. Zeng, Computing the multiplicity structure in solving polynomial systems, in: M. Kauers (Ed.), Proceedings of the 2005 International Symposium on Symbolic and Algebraic Computation, in: ISSAC '05, ACM, New York, NY, USA, 2005, pp. 116–123.
[19] S.M. Rump, S. Graillat, Verified error bounds for multiple roots of systems of nonlinear equations, Numer. Algorithms 54 (3) (2010) 359–377.
[20] N. Li, L. Zhi, Verified error bounds for isolated singular solutions of polynomial systems: case of breadth one, Theoret. Comput. Sci. 479 (2013) 163–173.
[21] A. Mantzaflaris, B. Mourrain, Deflation and certified isolation of singular zeros of polynomial systems, in: Proc. ISSAC 2011, 2011, pp. 249–256.
[22] N. Li, L. Zhi, Verified error bounds for isolated singular solutions of polynomial systems, SIAM J. Numer. Anal. 52 (4) (2014) 1623–1640.

[23] J. Verschelde, Algorithm 795: Phcpack: A general-purpose solver for polynomial systems by homotopy continuation, ACM Trans. Math. Software 25 (1999).

[24] D.W. Decker, C.T. Kelley, Newton's method at singular points i, SIAM J. Numer. Anal. 17 (1980) 66–70.

[25] D. Li, M. Fukushima, A globally and superlinearly convergent gauss-newton based bfgs method for symmetric nonlinear equations, SIAM J. Numer. Anal. 37 (1999) 152–172.

[26] W. Zhou, D. Li, On the q-linear convergence rate of a class of methods for monotone nonlinear equations, Pac. J. Optim. 14 (2018) 723–737.

[27] S.M. Rump, INTLAB - INTerval LABoratory, in: T. Csendes (Ed.), Developments in Reliable Computing, Kluwer Academic Publishers, Dordrecht, 1999, pp. 77–104.

[28] N. Li, L. Zhi, Compute the multiplicity structure of an isolated singular solution: case of breadth one, J. Symbolic Comput. 47 (2012) 700–710.

[29] A. Szanto, J.D. Hauenstein, B. Mourrain, On deflation and multiplicity structure, J. Symbolic Comput. 83 (2017) 228–253.