A LINEAR HOMOTOPY METHOD FOR COMPUTING GENERALIZED TENSOR EIGENPAIRS

By

Liping Chen

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Applied Mathematics — Doctor of Philosophy

2016

ABSTRACT

A LINEAR HOMOTOPY METHOD FOR COMPUTING GENERALIZED TENSOR EIGENPAIRS

By

Liping Chen

A tensor is a multidimensional array. In general, an mth-order and n-dimensional tensor can be indexed as $\mathcal{A} = (A_{i_1 i_2 \ldots i_m})$, where $A_{i_1 i_2 \ldots i_m} \in \mathbb{C}$ for $1 \leq i_1, i_2, \ldots, i_m \leq n$. Let $\mathcal{A}$ be an mth order n-dimensional tensor and $\mathcal{B}$ be an m'th order n-dimensional tensor. A scalar $\lambda \in \mathbb{C}$ and a vector $x \in \mathbb{C}^n \backslash \{0\}$ is called a generalized $\mathcal{B}$-eigenpair of $\mathcal{A}$ if $\mathcal{A}x^{m-1} = \lambda \mathcal{B}x^{m'-1}$ with $\mathcal{B}x^{m'} = 1$ when $m \neq m'$. Different choices of $\mathcal{B}$ yield different versions of the tensor eigenvalue problem.

As one can see, computing tensor eigenpairs amounts to solving a polynomial system. To find all solutions of a polynomial system, the homotopy continuation methods are very useful in terms of computational cost and storage space. By taking advantage of the solution structure of the tensor eigenproblem, two easy-to-implement linear homotopy methods which follow the optimal number of paths will be proposed to solve the generalized tensor eigenproblem when $m \neq m'$. With proper implementation, these methods can find all equivalence classes of isolated eigenpairs. A MATLAB software package `TenEig 2.0` has been developed to implement these methods. Numerical results are provided to show its efficiency and effectiveness.

To my family

ACKNOWLEDGMENTS

I would like to gratefully and sincerely thank my advisor Professor Tien-Yien Li for his constant guidance, support, understanding and encouragements during my years at Michigan State University. Especially at this time, although he was so sick (he had a bladder cancer and just finished a surgery about three weeks before my thesis defense) and could not stand up these days, he still supported me to defense in this semester so that I can head to work on time.

I would like to thank Dr. Chichia Chiu, Dr. Patricia Lamm, Dr. Gabriel Nagy and Dr. Moxun Tang, for serving as members of my thesis committee. I am so indebted to Dr. Chichia Chiu for supporting me as a research assistant during my first year PhD study. I am grateful to Dr. Patricia Lamm, Dr. Sheldon Newhouse and Dr. Gabriel Nagy for giving me the opportunity to work on WeBWorK and providing letters of reference. I am also grateful to Dr. Peiru Wu and Dr. Zhengfang Zhou for their help in these years. Special thanks to Professor Lixing Han for introducing the tensor eigen problems to us and giving me a lot of advice on this subject.

I also want to take this opportunity to thank my friends, my academic brothers and sisters. Special thanks to Dr. Jiu Ding and Dr. Xiaoshen Wang for recommending me to study aboard.

Finally, and most importantly, I would like to thank my family for their continuous and strong support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Tensor

The tensor considered here is a multidimensional array, instead of the tensor in physics and engineering [17] or *tensor fields* in mathematics [28].

A first-order, n-dimensional tensor is an n-vector. A second-order, n-dimensional tensor is an $n \times n$ matrix. In general, an m-th order n-dimensional tensor can be indexed as

$$\mathcal{A} = (A_{i_1 i_2 \ldots i_m}),$$

where $A_{i_1 i_2 \ldots i_m} \in \mathbb{C}$ for $1 \leq i_1, i_2, \ldots, i_m \leq n$. We denote the set of all mth-order, n-dimensional tensors on $\mathbb{C}$ by $\mathbb{C}^{[m,n]}$. See Figure 1.1 for a third-order two-dimensional tensor.

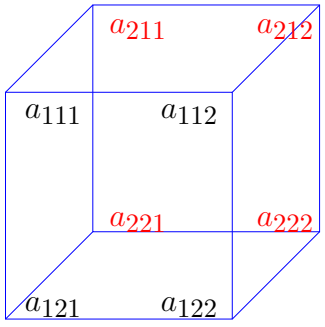

Figure 1.1: A third-order, two-dimensional tensor

A tensor $\mathcal{A}$ is symmetric if its entries are invariant under permutation. For example, a

third-order tensor $\mathcal{A} \in \mathbb{C}^{[3,n]}$ is symmetric if

$$A_{ijk} = A_{ikj} = A_{jik} = A_{jki} = A_{kij} = A_{kji}$$

for all $i, j, k = 1, \ldots, n$.

A tensor $\mathcal{A} \in \mathbb{C}^{[m,n]}$ is called a *diagonal tensor* if $A_{ii\ldots i} \neq 0$ and all other entries are zero. In particular, a diagonal tensor with ones along the diagonal is called an *identity tensor*.

1.2 Tensor eigenvalue and eigenvector problems

Let $\mathcal{A} \in \mathbb{C}^{[m,n]}$, $x := (x_1, \ldots, x_n)^T \in \mathbb{C}^n$, and $x^{[m]} := (x_1^m, x_2^m, \ldots, x_n^m)^T$. The tensor $\mathcal{A}$ induces an mth-degree homogeneous polynomial given by

$$\mathcal{A}x^m := \sum_{i_1, \cdots, i_m = 1}^{n} A_{i_1 \cdots i_m} x_{i_1} \cdots x_{i_m}$$

in $x_1, \ldots, x_n$. By the tensor product [23], $\mathcal{A}x^{m-1}$ denotes an n-vector whose jth entry is

$$(\mathcal{A}x^{m-1})_j = \sum_{i_2, \cdots, i_m = 1}^{n} A_{j, i_2 \cdots i_m} x_{i_2} \cdots x_{i_m}.$$

The following notion of eigenpairs for complex tensors was introduced by Qi [21] in 2005.

DEFINITION 1.2.1. *Suppose $\mathcal{A} \in \mathbb{C}^{[m,n]}$, where $m \geq 2$ and $n \geq 1$. We call a number $\lambda \in \mathbb{C}$ an eigenvalue of $\mathcal{A}$ if it and a nonzero vector $x \in \mathbb{C}^n$ are solutions of the following homogeneous polynomial equation:*

$$\mathcal{A}x^{m-1} = \lambda x^{[m-1]}, \tag{1.1}$$

and called the solution x an eigenvector of $\mathcal{A}$ associated with the eigenvalue λ.

At the same time, Lim [14] proposed a theory of eigenvalues, eigenvectors, singular values, and singular vectors for tensors from a variational approach. Since then, tensor eigenvalues have found applications in automatic control, diffusion tensor imaging, image authenticity verification, spectral hypergraph theory, and quantum entanglement, etc., see, for example, [3, 7, 10, 18, 22, 24, 25, 26].

In the following an application of the tensor eigenvalue in automatic control is described [18].

Consider an autonomous nonlinear dynamical system

$$\dot{x} = g(x),$$

where $x(t) \in \mathcal{D} \subset \mathbb{R}^n$ is the system state vector and $g : \mathcal{D} \to \mathbb{R}^n$ is continuous. It is well known that Lyapunovs method can be used to determine whether this system is asymptotically stable. More specifically, if a multivariate positive definite polynomial $f(x)$ can be found such that

$$\nabla f(x)^T g(x) < 0, \quad \forall x \in \mathbb{R}^n, x \neq 0$$

then the system $\dot{x} = g(x)$ is asymptotically stable. Here we call $f(x)$ positive definite if

$$f(x) > 0, \quad \forall x \in \mathbb{R}^n, x \neq 0. \tag{1.2}$$

To find a proper multivariate positive definite polynomial, let us consider the positive defi-

niteness of an even-degree homogeneous polynomial given by

$$f(x) = \sum_{i_1,\ldots,i_m=1}^{n} a_{i_1\ldots i_m} x_{i_1} \cdots x_{i_m}, \tag{1.3}$$

where m is even and $a_{i_1\ldots i_m}$'s are the entries of a symmetric tensor $\mathcal{A} \in \mathbb{R}^{[m,n]}$. Let

$$\mathcal{A}x^m := f(x).$$

By (1.2), the positive definiteness of $f(x)$ becomes

$$\mathcal{A}x^m > 0, \quad \forall x \in \mathbb{R}^n, x \neq 0. \tag{1.4}$$

It is equivalent to the positiveness of

$$\min\{\mathcal{A}x^m \,\big|\, \|x\|_m^m = 1\}, \tag{1.5}$$

where $\|x\|_m^m = x_1^m + \cdots + x_n^m$. To use the method of Lagrange multiplier, let

$$L(\lambda, x) := \mathcal{A}x^m - \lambda(\|x\|_m^m - 1).$$

Then the critical point must satisfy

$$\nabla_x L(\lambda, x) = 0, \quad \nabla_\lambda L(\lambda, x) = 0. \tag{1.6}$$

When $\mathcal{A}$ is symmetric, it was proved in [12]

$$\nabla_x(\mathcal{A}x^m) = m\mathcal{A}x^{m-1}.$$

Then (1.6) implies

$$\begin{aligned} m\mathcal{A}x^{m-1} - \lambda m x^{[m-1]} &= 0, \\ \|x\|_m^m - 1 &= 0, \end{aligned}$$

or,

$$\begin{aligned} \mathcal{A}x^{m-1} - \lambda x^{[m-1]} &= 0, \\ \|x\|_m^m &= 1, \end{aligned} \tag{1.7}$$

which is the eigenvalue problem defined in (1.1) with normalization condition $\|x\|_m^m = 1$. Multiplying both sides of (1.7) by x^T from the left yields

$$x^T\mathcal{A}x^{m-1} - \lambda x^T x^{[m-1]} = 0.$$

Clearly, $x^T\mathcal{A}x^{m-1} = \mathcal{A}x^m$ and $x^T x^{[m-1]} = \|x\|_m^m$. It follows that

$$\lambda = \mathcal{A}x^m.$$

Hence (1.5) can be replaced by

$$\min\{\lambda \,\big|\, \|x\|_m^m = 1\}.$$

Consequently $f(x)$ in (1.3) is positive definite if the smallest real eigenvalue of the corresponding tensor $\mathcal{A}$ is positive.

1.3 Generalized tensor eigenvalue and eigenvector problems

Various definitions of eigenvalues for tensors have been proposed in the literature, including E-eigenvalues and Qi-eigenvalues in the complex field, and Z-eigenvalues, H-eigenvalues, and D-eigenvalues in the real field [14, 21, 24]. In [6], Chang, Pearson, and Zhang introduced a notion of generalized eigenvalues for tensors that unifies several types of eigenvalues. Recently this definition has been further generalized by Cui, Dai, and Nie [9]. In this section, generalized tensor eigenvalue problems will be introduced.

Let $\mathcal{A} \in \mathbb{C}^{[m,n]}$ and $x := (x_1, \ldots, x_n)^T \in \mathbb{C}^n$. For $1 \leq k \leq m$, let $\mathcal{A}^{(k)}x^{m-1}$ be an n-vector whose jth entry is

$$(\mathcal{A}^{(k)}x^{m-1})_j = \sum_{i_1,\cdots,i_{k-1},i_{k+1},\cdots,i_m=1}^{n} A_{i_1\cdots i_{k-1}ji_{k+1}\cdots i_m} x_{i_1}\cdots x_{i_{k-1}} x_{i_{k+1}}\cdots x_{i_m}.$$

When $k = 1$, $\mathcal{A}^{(1)}x^{m-1}$ is the $\mathcal{A}x^{m-1}$ defined in the previous section.

The notion of mode-k generalized eigenpairs for complex tensors was defined by Chen, Han and Zhou [8] as follows.

DEFINITION 1.3.1. *Suppose $\mathcal{A} \in \mathbb{C}^{[m,n]}$ and $\mathcal{B} \in \mathbb{C}^{[m',n]}$, where $m \geq 2,\ m' \geq 2, n \geq 1$. Assume $\mathcal{B}x^{m'}$ as a function of x is not identically zero. For $1 \leq k \leq m$, $(\lambda, x) \in \mathbb{C}\times(\mathbb{C}^n\backslash\{0\})$ is a mode-k $\mathcal{B}$-eigenpair of $\mathcal{A}$ if*

- *when* $m = m'$,

$$\mathcal{A}^{(k)}x^{m-1} = \lambda\mathcal{B}x^{m-1}, \tag{1.8}$$

- *when* $m \neq m'$,

$$\mathcal{A}^{(k)}x^{m-1} = \lambda\mathcal{B}x^{m'-1}, \quad \mathcal{B}x^{m'} = 1, \tag{1.9}$$

For suitable $\mathcal{A}$, m' and $\mathcal{B}$, many different types of tensor eigenpairs defined in the literature were contained in (1.8) or (1.9). For instance,

- For $\mathcal{A} \in \mathbb{R}^{[m,n]}$, an E-eigenpair [14, 21] is defined as a pair $(\lambda, x) \in \mathbb{C} \times \mathbb{C}^n\backslash\{0\}$ such that

$$\mathcal{A}x^{m-1} = \lambda x, \quad x^T x = 1. \tag{1.10}$$

  This is a mode-1 $\mathcal{B}$-eigenpair with $m' = 2$ and $\mathcal{B}$ being the $n \times n$ identity matrix. A real E-eigenpair is called a Z-eigenpair [14, 21].

- For $\mathcal{A} \in \mathbb{R}^{[m,n]}$, let $D \in \mathbb{R}^{n\times n}$ be a symmetric positive definite matrix, $(\lambda, x) \in \mathbb{R} \times \mathbb{R}^n\backslash\{0\}$ is called a D-eigenpair [24] if

$$\mathcal{A}x^{m-1} = \lambda Dx, \quad x^T Dx = 1.$$

  This is a real mode-1 $\mathcal{B}$-eigenpair with $m' = 2$ and $\mathcal{B} = D$.

- For $\mathcal{A} \in \mathbb{R}^{[m,n]}$, a Qi-eigenpair [21] is defined as a pair $(\lambda, x) \in \mathbb{C} \times \mathbb{C}^n\backslash\{0\}$ such that

$$\mathcal{A}x^{m-1} = \lambda x^{[m-1]}, \tag{1.11}$$

  where $x^{[m-1]} := (x_1^{m-1}, x_2^{m-1}, \ldots, x_n^{m-1})^T$. This is a mode-1 $\mathcal{B}$-eigenpair with $m' = m$

and $\mathcal{B}$ being the identity tensor. A real Qi-eigenpair is called an H-eigenpair [21].

- For $\mathcal{A} \in \mathbb{R}^{[m,n]}$, a CPZ-eigenpair [6] is defined as a pair $(\lambda, x) \in \mathbb{C} \times \mathbb{C}^n \backslash \{0\}$ such that

$$\mathcal{A}x^{m-1} = \lambda \mathcal{B}x^{m-1}.$$

  This is a mode-1 $\mathcal{B}$-eigenpair with $m = m'$.

- For symmetric tensors $\mathcal{A} \in \mathbb{R}^{[m,n]}$ and $\mathcal{B} \in \mathbb{R}^{[m',n]}$, a CDN-eigenpair [9] is defined as a pair $(\lambda, x) \in \mathbb{C} \times \mathbb{C}^n \backslash \{0\}$ such that

$$\mathcal{A}x^{m-1} = \lambda \mathcal{B}x^{m'-1}, \quad \mathcal{B}x^{m'} = 1.$$

  This is a mode-1 $\mathcal{B}$-eigenpair with $\mathcal{A}$ and $\mathcal{B}$ being symmetric.

1.4 Problem formulation

Let $\mathcal{A} \in \mathbb{C}^{[m,n]}$ and $\mathcal{B} \in \mathbb{C}^{[m',n]}$. As one can see from Definition 1.3.1, computing mode-k $\mathcal{B}$-eigenpairs of $\mathcal{A}$ amounts to solving $\mathcal{A}^{(k)}x^{m-1} = \lambda \mathcal{B}x^{m'-1}$ followed by normalizing x for $\mathcal{B}x^{m'} = 1$ when $m \neq m'$.

As discussed in Remark 2.1 in [8], the solution set of $\mathcal{A}^{(k)}x^{m-1} = \lambda \mathcal{B}x^{m'-1}$ consists of different equivalence classes. If (λ, x) is a solution, we denote its corresponding equivalence class by

$$[(\lambda, x)] := \{(\lambda', x') \,|\, \lambda' = t^{m-m'}\lambda, x' = tx, t \in \mathbb{C}\backslash\{0\}\}.$$

When $m \neq m'$, imposing the normalization condition $\mathcal{B}x^{m'} = 1$, the problem (1.9) has m' eigenpairs from each equivalence class. According to this observation, the problem of solving

the eigenvalue problem (1.9) can be converted to first solving the following polynomial system

$$P(\lambda, x) = \begin{pmatrix} \mathcal{A}^{(k)} x^{m-1} - \lambda \mathcal{B} x^{m'-1} \\ \\ \eta^T x + \eta_0 \end{pmatrix} = 0, \tag{1.12}$$

where $\eta \in \mathbb{C}^n$ and $\eta_0 \in \mathbb{C}$ are randomly chosen. When $m \neq m'$, for each solution of (1.12) obtained we normalize it for an eigenpair (λ^*, x^*) satisfying (1.9), then find m' equivalent eigenpairs (λ', x') by setting $\lambda' = t^{m-m'}\lambda^*$ and $x' = tx^*$ with t being a root of $t^{m'} = 1$.

Note that (1.12) is a polynomial system. To find all solutions of a polynomial system, the homotopy continuation methods are very efficient in terms of computational cost and storage space (see, for example, [13], [29]). In [8], Chen, Han and Zhou proposed two homotopy continuation methods (Algorithms 3.1 and 3.2 in [8]) for computing all eigenpairs of a general real or complex tensor. Specifically, while Algorithm 3.1 is a linear homotopy method which handles the case $m = m'$; Algorithm 3.2, which handles the case $m \neq m'$, is based on a polyhedral homotopy method.

The most time consuming part of homotopy continuation methods is the path following step. The polyhedral homotopy methods have the advantage that in certain situations they can follow much fewer paths than continuation methods using other homotopies, such as the total degree homotopy (see, for example, [13]). As indicated in [1], however, the polyhedral homotopy methods involve a highly complicated combinatorial process for finding the maximal root count for a given sparse structure and setting up a compatible homotopy. This makes the implementation of a polyhedral homotopy method very sophisticated.

In this article, a linear homotopy method will be proposed to solve (1.12) when $m \neq m'$. It is shown that the method finds all isolated eigenpairs of problem (1.9). This method is

easy to implement. Moreover, the method follows an optimal number of paths by fully using the solution structure of problem (1.9), making it an efficient and competitive homotopy method for computing eigenpairs of general tensors.

This dissertation is organized as follows. We first describe the linear homotopy methods, showing these methods can find all isolated eigenpairs in Chapter 2. A detailed algorithm with some implementation tips will be given in Chapter 3. Finally, we give some numerical results in Chapter 4.

Chapter 2

Construct a linear homotopy to compute generalized tensor eigenpairs

Let $\mathcal{A} \in \mathbb{C}^{[m,n]}$ and $\mathcal{B} \in \mathbb{C}^{[m',n]}$. As discussed in Section 1.4, the problem of computing mode-k $\mathcal{B}$-eigenpairs in (1.9) is equivalent to solving (1.12), and when $m \neq m'$, we normalize it to satisfy $\mathcal{B}x^{m'} = 1$. In this chapter, we will construct two linear homotopy methods to compute all isolated zeros of the polynomial system (1.12) for $m > m'$ and $m < m'$ respectively, and thereby compute all the eigenpairs.

2.1 Preliminaries

The following two lemmas about coefficient-parameter homotopy play an essential role in our construction.

LEMMA 2.1.1 (Theorem 7.1.1 in [29]). *Let $F(z;c)$ be a system of polynomials in n variables and l parameters,*

$$F(z;c) : \mathbb{C}^n \times \mathbb{C}^l \to \mathbb{C}^n,$$

that is, $F(z;c) = (f_1(z;c), \ldots, f_n(z;c))^T$ and each $f_i(z;c)$ is polynomial in both z and c.

Furthermore, let $\mathcal{N}(c)$ denote the number of nonsingular solutions as a function of c:

$$\mathcal{N}(c) := \#\left\{ z \in \mathbb{C}^n \,\middle|\, F(z;c) = 0, \det\left(\frac{\partial F}{\partial z}(z;c)\right) \neq 0 \right\}.$$

Then

(1) $\mathcal{N}(c)$ is finite, and it is the same, say $\mathcal{N}_F$, for almost all $c \in \mathbb{C}^l$.

(2) for all $c \in \mathbb{C}^l$, $\mathcal{N}(c) \leq \mathcal{N}_F$.

Here is an example to illustrate Lemma 2.1.1. Let $F(z_1; q_1, q_2, q_3) : \mathbb{C} \times \mathbb{C}^3 \to \mathbb{C}$ be defined as

$$F(z_1; q_1, q_2, q_3) := q_1 z_1^2 + q_2 z_1 + q_3,$$

which is a polynomial both in the variable z_1 and in the parameters q_1, q_2, q_3. By the definition of $\mathcal{N}(q)$, we have

$$\mathcal{N}(q) = \#\left\{ z_1 \in \mathbb{C} \,\middle|\, F := q_1 z_1^2 + q_2 z_1 + q_3 = 0,\ \frac{\partial F}{\partial z_1} := 2q_1 z_1 + q_2 \neq 0 \right\}.$$

Therefore, $q_1 z_1^2 + q_2 z_1 + q_3 = 0$ always has two solutions

$$z_1 = \frac{-q_2 + \sqrt{q_2^2 - 4q_1 q_3}}{2q_1} \text{ or } \frac{-q_2 - \sqrt{q_2^2 - 4q_1 q_3}}{2q_1}$$

as long as $q_1 \neq 0$. And these solutions will satisfy $2q_1 z_1 + q_2 \neq 0$ as long as $q_2^2 - 4q_1 q_3 \neq 0$. Denote

$$Q_s := \{(q_1, q_2, q_3) \,|\, q_1 = 0 \text{ or } q_2^2 - 4q_1 q_3 = 0\}.$$

Then for any $q := (q_1, q_2, q_3) \in \mathbb{C}^3 \backslash Q_s$, $\mathcal{N}(q)$ is equal to 2. Note that Q_s has measure 0 in

$\mathbb{C}^3$. In this sense, we say for almost all $q \in \mathbb{C}^3$, $\mathcal{N}(q)$ is equal to the same number 2, which is denoted by $\mathcal{N}_F$ in Lemma 2.1.1. Moreover, for any $q := (q_1, q_2, q_3) \in Q_s$, we have the following three cases:

(i). If $q_1 = 0$ but $q_2^2 - 4q_1q_3 = q_2^2 \neq 0$, then $F = q_2z_1 + q_3$ is linear. So $\mathcal{N}(q) = 1$.

(ii). If $q_1 \neq 0$ but $q_2^2 - 4q_1q_3 = 0$, then $F = 0$ has no nonsingular solutions. So $\mathcal{N}(q) = 0$.

(iii). If $q_1 = 0$ and $q_2^2 - 4q_1q_3 = q_2^2 = 0$, then $F = q_3$. When $q_3 \neq 0$, $F = 0$ has no solutions. When $q_3 = 0$, $F = 0$ has no nonsingular solutions. So $\mathcal{N}(q) = 0$ in this case.

From the above discussion, we conclude that $\mathcal{N}(q)$ is equal to $\mathcal{N}_F$ for almost all $q \in \mathbb{C}^3$ and $\mathcal{N}(q) \leq \mathcal{N}_F$ for all $q \in \mathbb{C}^3$.

LEMMA 2.1.2 (Theorem 8.3.1 in [29]). *Let $F(z; c)$, $\mathcal{N}(c)$ and $\mathcal{N}_F$ be as in Lemma 2.1.1. Suppose $F(z; c)$ is linear in c. Let $f(z) = F(z; c_1)$ for some given $c_1 \in \mathbb{C}^l$. If $g(z) = F(z; c_0)$ for some $c_0 \in \mathbb{C}^l$ has $\mathcal{N}_F$ nonsingular zeros, then the linear homotopy*

$$h(z, t) := \gamma(1 - t)g(z) + tf(z) = 0$$

has $\mathcal{N}_F$ nonsingular solution paths on $t \in [0, 1)$ whose endpoints as $t \to 1$ include all of the isolated roots of $f(z) = 0$ in $\mathbb{C}^n$. Here γ is a randomly chosen nonzero complex number of absolute value 1.

Lemma 2.1.1 and Lemma 2.1.2 suggest the following steps to construct our linear homotopy for finding all isolated solutions of the system $P(\lambda, x) = 0$ in (1.12) when $m \neq m'$:

- **Step 1.** Construct a polynomial system $F(\lambda, x; c)$, where c denotes the set of parameters, making $F(\lambda, x; c)$ linear in c and $F(\lambda, x; c_1) = P(\lambda, x)$ for certain parameter set c_1.

- **Step 2.** Compute $\mathcal{N}_F$ as defined in Lemma 2.1.1 for $F(\lambda, x; c)$;

- **Step 3.** Find a parameter set c_0 such that $G(\lambda, x) := F(\lambda, x; c_0)$ has $\mathcal{N}_F$ nonsingular zeros.

Then by Lemma 2.1.2, all the isolated zeros of $P(\lambda, x)$ can be found by tracing the solution paths of the linear homotopy

$$H(\lambda, x, t) := \gamma(1-t)G(\lambda, x) + tP(\lambda, x)$$

from $t = 0$ to $t = 1$.

We shall follow the above three steps to construct our linear homotopy for the case of $m > m'$ and $m < m'$ separately.

Let $P(x) := (p_1(x), \ldots, p_n(x))^T$ be a polynomial system with $x := (x_1, \ldots, x_n)^T$. For $\alpha := (\alpha_1, \ldots, \alpha_n) \in (\mathbb{Z}^n_{\geq 0})^T$, write $x^\alpha := x_1^{\alpha_1} \cdots x_n^{\alpha_n}$ and $|\alpha| := \alpha_1 + \cdots + \alpha_n$. Then $P(x)$ can be written as

$$P(x) := \begin{pmatrix} p_1(x) := \sum\limits_{\alpha \in S_1} c_{1,\alpha} x^\alpha \\ \vdots \\ p_n(x) := \sum\limits_{\alpha \in S_n} c_{n,\alpha} x^\alpha \end{pmatrix}, \tag{2.1}$$

where $S_1, \ldots, S_n$ are finite subsets of $(\mathbb{Z}^n_{\geq 0})^T$ and $c_{i,\alpha} \in \mathbb{C}^* := \mathbb{C}\backslash\{0\}$ are the coefficients of the corresponding monomials. Here for each $i = 1, \ldots, n$, S_i is called the support of $p_i(x)$ and its convex hull $R_i := \text{conv}(S_i)$ in $\mathbb{R}^n$ is called the Newton polytope of $p_i(x)$. The n-tuple $(S_1, \ldots, S_n)$ is called the support of $P(x)$. For nonnegative variables $\lambda_1, \ldots, \lambda_n$, let

$\lambda_1 R_1 + \cdots + \lambda_n R_n$ be the *Minkowski* sum of $\lambda_1 R_1, \ldots, \lambda_n R_n$, i.e.,

$$\lambda_1 R_1 + \cdots + \lambda_n R_n := \{\lambda_1 r_1 + \cdots + \lambda_n r_n \,|\, r_i \in R_i, i = 1, \ldots, n\}.$$

The n-dimensional volume of $\lambda_1 R_1 + \cdots + \lambda_n R_n$, denoted by $\mathrm{Vol}_n(\lambda_1 R_1 + \cdots + \lambda_n R_n)$, is a homogeneous polynomial of degree n in $\lambda_1, \ldots, \lambda_n$ (See, for example, Proposition 4.9 of [4] for a proof). The coefficient of the monomial $\lambda_1 \lambda_2 \ldots \lambda_n$ in $\mathrm{Vol}_n(\lambda_1 R_1 + \cdots + \lambda_n R_n)$ is called the mixed volume of $R_1, \ldots, R_n$, denoted by $\mathrm{MV}_n(R_1, \ldots, R_n)$, or the mixed volume of the supports $S_1, \ldots, S_n$, denoted by $\mathrm{MV}_n(S_1, \ldots, S_n)$. It is also called the mixed volume of $P(x)$ if no ambiguity exists. The following theorem relates the number of solutions of a polynomial system to its mixed volume.

LEMMA 2.1.3. (**Bernstein's Theorem**) [2] *The number of isolated zeros in* $(\mathbb{C}^*)^n$*, counting multiplicities, of a polynomial system* $P(x) = (p_1(x), \ldots, p_n(x))^T$ *with supports* $S_1, \ldots, S_n$ *is bounded by the mixed volume* $\mathrm{MV}_n(S_1, \ldots, S_n)$*. Moreover, for generic choices of the coefficients in* p_i*, the number of isolated zeros is exactly* $\mathrm{MV}_n(S_1, \ldots, S_n)$*.*

An apparent limitation of Lemma 2.1.3 is that it only counts the isolated zeros of a polynomial system in $(\mathbb{C}^*)^n$ rather than $\mathbb{C}^n$. To deal with this issue, Li and Wang gave the following theorem.

LEMMA 2.1.4. [16] *The number of isolated zeros in* $\mathbb{C}^n$*, counting multiplicities, of a polynomial system* $P(x) = (p_1(x), \ldots, p_n(x))^T$ *with supports* $S_1, \ldots, S_n$ *is bounded by the mixed volume* $\mathrm{MV}_n(S_1 \cup \{0\}, \ldots, S_n \cup \{0\})$*.*

The following lemma was given as Exercise 7 on page 338 of [4].

LEMMA 2.1.5. *Consider a polynomial system* $P(x) = (p_1(x), \ldots, p_n(x))^T$ *with supports* $S_1 = S_2 = \cdots = S_n = S$. *Then*

$$\mathrm{MV}_n(S, \ldots, S) = n!\mathrm{Vol}_n(\mathrm{conv}(S)).$$

Recall that an n-simplex is defined to be the convex hull of $n+1$ points $z_1, \ldots, z_{n+1}$ such that $z_2 - z_1, \ldots, z_{n+1} - z_1$ are linearly independent in $(\mathbb{R}^n)^T$. It can be shown that for this simplex

$$\mathrm{Vol}_n(\mathrm{conv}(z_1, z_2, \ldots, z_{n+1})) = \frac{1}{n!}\left|\det\begin{pmatrix} z_2 - z_1 \\ \vdots \\ z_{n+1} - z_1 \end{pmatrix}\right|. \tag{2.2}$$

2.2 Construct a linear homotopy to compute generalized tensor eigenpairs when $m > m'$

Let $\mathcal{D} \in \mathbb{C}^{[m,n]}$, $\mathcal{E} \in \mathbb{C}^{[m',n]}$ and $\mathcal{L} \in \mathbb{C}^{[2,n]}$. Let $\mathrm{vdiag}(\mathcal{L}) := (L_{11}, \ldots, L_{nn})^T$. Write

$$c := \{\mathcal{D}, \mathcal{E}, \mathcal{L}\}. \tag{2.3}$$

Consider

$$F_1(\lambda, x; c) = \begin{pmatrix} \mathcal{D}x^{m-1} - \lambda\mathcal{E}x^{m'-1} - \mathcal{L}x^{[m-m']} + \lambda \cdot \mathrm{vdiag}(\mathcal{L}) \\ \eta^T x + \eta_0 \end{pmatrix}. \tag{2.4}$$

It is clear that F_1 is linear in c. Taking

$$c_1 := \{\mathcal{A}, \mathcal{B}, 0\}, \tag{2.5}$$

we obtain $F_1(\lambda, x; c_1) = P(\lambda, x)$, which is our target system (1.12).

Let $\mathcal{N}_{F_1}$ be defined in Lemma 2.1.1 for $F_1(\lambda, x; c)$. To compute $\mathcal{N}_{F_1}$, we first prove the following theorem.

THEOREM 2.2.1. *Let $F_1(\lambda, x; c)$ be as in (2.4) and c as in (2.3) be the set of parameters. Then the number of isolated zeros, counting multiplicities, of $F_1(\lambda, x; c)$ is bounded by*

$$\frac{(m-1)^n - (m'-1)^n}{m - m'}.$$

When c is generic, $F_1(\lambda, x; c)$ has exactly $((m-1)^n - (m'-1)^n)/(m-m')$ isolated zeros.

Proof: For the random hyperplane $\eta^T x + \eta_0 = 0$, i.e., $\eta_1 x_1 + \cdots + \eta_n x_n + \eta_0 = 0$, in (2.4), without loss, we suppose $\eta_n \neq 0$. Then

$$x_n = a_1 x_1 + \cdots + a_{n-1} x_{n-1} + b, \tag{2.6}$$

where $a_i = -\eta_i/\eta_n$ for $i = 1, \ldots, n-1$ and $b = -\eta_0/\eta_n$. Notice that the number of solutions of (2.4) in $\mathbb{C}^{n+1}$ is the same as the number of solutions in $\mathbb{C}^n$ of the resulting system $T^*(\lambda, x_1, \ldots, x_{n-1})$ by substituting (2.6) into the first n equations of (2.4). Denote the corresponding supports of T^* by $S_1, \ldots, S_n$. We claim that

$$\mathrm{MV}_n(S_1 \cup \{0\}, \ldots, S_n \cup \{0\}) = \frac{(m-1)^n - (m'-1)^n}{m - m'}. \tag{2.7}$$

If this is proved, let N denote the number of isolated zeros of (2.4) in $\mathbb{C}^n$. Then (2.7) implies

$$N \leq \frac{(m-1)^n - (m'-1)^n}{m - m'}.$$

When the parameter set $c = \{\mathcal{D}, \mathcal{E}, \mathcal{L}\}$ is generic, the equality in the above holds by using Lemma 2.1.3 and Lemma 2.1.4.

To prove (2.7), let $\bar{c} := \{\bar{\mathcal{D}}, \bar{\mathcal{E}}, \bar{\mathcal{L}}\}$ be generic. Similar to (2.4) the corresponding polynomial system being solved is

$$\bar{T}(\lambda, x) = \begin{pmatrix} \bar{\mathcal{D}}x^{m-1} - \lambda\bar{\mathcal{E}}x^{m'-1} - \bar{\mathcal{L}}x^{[m-m']} + \lambda \cdot \mathrm{vdiag}(\bar{\mathcal{L}}) \\ \eta^T x + \eta_0 \end{pmatrix} = 0. \tag{2.8}$$

Substituting (2.6) into the first n equations of (2.8) yields a new system $\bar{T}^*(\lambda, x_1, \ldots, x_{n-1})$. Let $\bar{S}_1, \ldots, \bar{S}_n$ be the corresponding supports of $\bar{T}^*$. Since $\bar{\mathcal{D}}$, $\bar{\mathcal{E}}$ and $\bar{\mathcal{L}}$ are generic, without loss of generality one can assume that all monomials $\lambda, x_1^{m-m'}, \ldots, x_n^{m-m'}$ with

$$\{x_1^{\alpha_1}x_2^{\alpha_2}\ldots x_n^{\alpha_n}\Big|\alpha_i \in \mathbb{Z}_{\geq 0},\, \alpha_1 + \alpha_2 + \cdots + \alpha_n = m-1\}$$

and

$$\{\lambda x_1^{\alpha_1}x_2^{\alpha_2}\ldots x_n^{\alpha_n}\Big|\alpha_i \in \mathbb{Z}_{\geq 0},\, \alpha_1 + \alpha_2 + \cdots + \alpha_n = m'-1\}$$

will appear in each of the first n equations in (2.8). It follows that all monomials

$$\{x_1^{\alpha_1}x_2^{\alpha_2}\ldots x_{n-1}^{\alpha_{n-1}}\Big|\alpha_i \in \mathbb{Z}_{\geq 0},\, \alpha_1 + \alpha_2 + \cdots + \alpha_{n-1} \leq m-1\}$$

and

$$\{\lambda x_1^{\alpha_1}x_2^{\alpha_2}\ldots x_{n-1}^{\alpha_{n-1}}\Big|\alpha_i \in \mathbb{Z}_{\geq 0},\, \alpha_1 + \alpha_2 + \cdots + \alpha_{n-1} \leq m'-1\}$$

will appear in each equation of $\bar{T}^*$. Consequently, $\bar{S}_1, \ldots, \bar{S}_n$ are all equal to

$$\bar{S} := \{(0,\alpha) \,\big|\, \alpha \in (\mathbb{Z}_{\geq 0}^{n-1})^T, |\alpha| \leq m-1\} \cup \{(1,\alpha) \,\big|\, \alpha \in (\mathbb{Z}_{\geq 0}^{n-1})^T, |\alpha| \leq m'-1\}.$$

Let $\bar{Q}$ be the convex hull of $\bar{S}$. Denote the i-th unit vector in $(\mathbb{R}^n)^T$ by e_i for $i = 1, \ldots, n$. Then vertices of $\bar{Q}$ are given by

$$z_i = \begin{cases} 0, & i = 0 \\ (m-1)e_{n+1-i}, & 1 \leq i \leq n-1 \\ e_1, & i = n \\ e_1 + (m'-1)e_{2n+1-i}, & n+1 \leq i \leq 2n-1. \end{cases} \tag{2.9}$$

From which,

$$z_{n+i} = e_1 + \frac{m'-1}{m-1} z_i, \quad 0 \leq i \leq n-1.$$

This indicates that $z_n, \ldots, z_{2n-1}$ are scaling of $z_0, \ldots, z_{n-1}$ respectively by a factor $(m'-1)/(m-1)$ followed by a shift. Actually, each z_{n+i} is obtained by moving z_i along the line

$$l_i(t) := (1-t)z_i + t z_{n+i}, \quad 0 \leq i \leq n-1 \tag{2.10}$$

as t changing from 0 to 1. Simple computation yields

$$l_i(t) = \begin{cases} te_1, & i = 0 \\ te_1 + (m-1+(m'-m)t)e_{n+1-i}, & 1 \leq i \leq n-1. \end{cases} \tag{2.11}$$

We now claim that

$$\bar{Q} = \bigcup_{t\in[0,1]} \mathrm{conv}(l_0(t), \ldots, l_{n-1}(t)). \tag{2.12}$$

To prove this, let $q \in \bar{Q}$, then there exist $\beta_i \geq 0$ $(i = 0, 1, \ldots, 2n-1)$ such that $\sum_{i=0}^{2n-1} \beta_i = 1$ and

$$q = \sum_{i=0}^{2n-1} \beta_i z_i. \tag{2.13}$$

Substituting (2.9) into (2.13) yields

$$q = \sum_{i=n}^{2n-1} \beta_i e_1 + \sum_{i=1}^{n-1} [\beta_i(m-1) + \beta_{n+i}(m'-1)] e_{n+1-i}. \tag{2.14}$$

To show $q \in \bigcup_{t\in[0,1]} \mathrm{conv}(l_0(t), \ldots, l_{n-1}(t))$, we need to find $t^* \in [0,1]$ and $\gamma_i \geq 0$ $(i = 0, \ldots, n-1)$ such that $\sum_{i=0}^{n-1} \gamma_i = 1$ and

$$q = \sum_{i=0}^{n-1} \gamma_i l_i(t^*). \tag{2.15}$$

Substituting (2.11) into (2.15), we have

$$\begin{aligned} q &= \gamma_0 t^* e_1 + \sum_{i=1}^{n-1} \gamma_i [t^* e_1 + (m - 1 + (m' - m)t^*) e_{n+1-i}] \\ &= t^* \left(\sum_{i=0}^{n-1} \gamma_i \right) e_1 + \sum_{i=1}^{n-1} \gamma_i [m - 1 + (m' - m)t^*] e_{n+1-i} \\ &= t^* e_1 + \sum_{i=1}^{n-1} \gamma_i [m - 1 + (m' - m)t^*] e_{n+1-i}. \end{aligned} \tag{2.16}$$

Comparing (2.14) with (2.16), if we choose

$$\begin{aligned} t^* &= \sum_{j=n}^{2n-1} \beta_j, \\ \gamma_i &= \frac{\beta_i(m-1)+\beta_{n+i}(m'-1)}{m-1+(m'-m)t^*}, \quad i=1,\ldots,n-1, \\ \gamma_0 &= 1-\sum_{i=1}^{n-1}\gamma_i, \end{aligned}$$

and rewrite the denominator of γ_i (for $i = 1, \cdots, n-1$) as

$$\begin{aligned} m-1+(m'-m)\sum_{j=n}^{2n-1}\beta_j &= m-1+(m'-1+1-m)\sum_{j=n}^{2n-1}\beta_j \\ &= (m-1)(1-\sum_{j=n}^{2n-1}\beta_j)+(m'-1)\sum_{j=n}^{2n-1}\beta_j \\ &= (m-1)\sum_{j=0}^{n-1}\beta_j+(m'-1)\sum_{j=n}^{2n-1}\beta_j, \end{aligned}$$

then we have $t^* \in [0,1]$, $\gamma_i \geq 0$ for $i = 0, \cdots, n-1$, and $\sum_{i=0}^{n-1}\gamma_i = 1$. Moreover, (2.15) holds. This shows $q \in \bigcup_{t\in[0,1]} \mathrm{conv}(l_0(t), \ldots, l_{n-1}(t))$. Therefore,

$$\bar{Q} \subset \bigcup_{t\in[0,1]} \mathrm{conv}(l_0(t), \ldots, l_{n-1}(t)).$$

On the other hand, let $q \in \bigcup_{t\in[0,1]} \mathrm{conv}(l_0(t), \ldots, l_{n-1}(t))$. Then there exist $t^* \in [0,1]$, $\gamma_i \geq 0$ $(i = 0, \ldots, n-1)$ such that $\sum_{i=0}^{n-1}\gamma_i = 1$ and (2.15) holds. By (2.10),

$$q = \sum_{i=0}^{n-1}\gamma_i((1-t^*)z_i + t^* z_{n+i}) = \sum_{i=0}^{n-1}\gamma_i(1-t^*)z_i + \sum_{i=0}^{n-1}\gamma_i t^* z_{n+i}.$$

Let

$$\beta_i = \begin{cases} \gamma_i(1-t^*), & i = 0, \ldots, n-1, \\ \gamma_{i-n}t^*, & i = n, \ldots, 2n-1. \end{cases}$$

Then

$$q = \sum_{i=0}^{2n-1} \beta_i z_i,$$

where $\beta_i \geq 0$ for each i and $\sum_{i=0}^{2n-1} \beta_i = 1$. Therefore, $q \in \bar{Q}$. We conclude that $\bigcup_{t\in[0,1]} \operatorname{conv}(l_0(t), \ldots, l_{n-1}(t)) \subset \bar{Q}$.

To compute the volume of $\bar{Q}$, let $q := (q_1, \ldots, q_n) \in \bar{Q}$, by (2.12) there exist $t \in [0,1]$ and $\gamma_i \geq 0$ $(i = 0, \ldots, n-1)$ such that $\sum_{i=0}^{n-1} \gamma_i = 1$ and

$$q = \sum_{i=0}^{n-1} \gamma_i l_i(t).$$

By (2.11) and $\sum_{i=0}^{n-1} \gamma_i = 1$,

$$\begin{aligned} q &= l_0(t)(1 - \sum_{i=1}^{n-1} \gamma_i) + \sum_{i=1}^{n-1} \gamma_i l_i(t) = l_0(t) + \sum_{i=1}^{n-1} \gamma_i (l_i(t) - l_0(t)) \\ &= te_1 + \sum_{i=1}^{n-1} \gamma_i (m - 1 + (m' - m)t) e_{n+1-i} \end{aligned}$$

with $t \in [0,1]$, $\gamma_i \geq 0$ and $\sum_{i=1}^{n-1} \gamma_i \leq 1$. Let $\partial(q_1, \ldots, q_n)/\partial(t, \gamma_1, \ldots, \gamma_{n-1})$ be the Jacobian matrix of $q_1, \ldots, q_n$ with respect to $t, \gamma_1, \ldots, \gamma_{n-1}$. Then

$$\left|\det\left(\frac{\partial(q_1, \ldots, q_n)}{\partial(t, \gamma_1, \ldots, \gamma_{n-1})}\right)\right| = \left|\det\begin{pmatrix} e_1 + \sum_{i=1}^{n-1} \gamma_i (m' - m) e_{n+1-i} \\ (m - 1 + (m' - m)t) e_n \\ \vdots \\ (m - 1 + (m' - m)t) e_2 \end{pmatrix}\right| = (m-1+(m'-m)t)^{n-1}.$$

By the change of variables (See, for example, [27]), the volume of $\bar{Q}$ is:

$$\begin{aligned}
\mathrm{Vol}_n(\bar{Q}) &= \int_{q\in\bar{Q}} dq = \int_0^1 \int_{\substack{\sum_{i=1}^{n-1}\gamma_i\le 1\\ \gamma_i\ge 0}} \left|\det\left(\frac{\partial(q_1,\ldots,q_n)}{\partial(t,\gamma_1,\ldots,\gamma_{n-1})}\right)\right| d\gamma_1\ldots d\gamma_{n-1}dt \\
&= \int_0^1 (m-1+(m'-m)t)^{n-1}dt \int_{\substack{\sum_{i=1}^{n-1}\gamma_i\le 1\\ \gamma_i\ge 0}} d\gamma_1\ldots d\gamma_{n-1} \\
&= \int_0^1 \frac{(m-1+(m'-m)t)^{n-1}}{(n-1)!}dt \qquad (2.17)\\
&= \frac{(m-1)^n-(m'-1)^n}{(m-m')n!},
\end{aligned}$$

where (2.17) holds because the region $\{(\gamma_1,\ldots,\gamma_{n-1}) \,\big|\, \sum_{i=1}^{n-1}\gamma_i \le 1, \gamma_i \ge 0\}$ is a standard $(n-1)$-simplex with volume $1/(n-1)!$ (See, for example, Exercise 2 and 3 on page 307 of [4])). Therefore, by Lemma 2.1.5,

$$\mathrm{MV}_n(\bar{S}_1,\ldots,\bar{S}_n) = n!\mathrm{Vol}_n(\bar{Q}) = \frac{(m-1)^n-(m'-1)^n}{m-m'}.$$

Noting that for $i=1,\ldots,n$, $S_i\cup\{0\}$ is a subset of $\bar{S}_i$. Hence

$$\mathrm{MV}_n(S_1\cup\{0\},\ldots,S_n\cup\{0\}) \le \mathrm{MV}_n(\bar{S}_1,\ldots,\bar{S}_n),$$

and therefore

$$\mathrm{MV}_n(S_1\cup\{0\},\ldots,S_n\cup\{0\}) \le \frac{(m-1)^n-(m'-1)^n}{m-m'}. \qquad (2.18)$$

On the other hand, consider the identity tensors $\mathcal{D}\in\mathbb{C}^{[m,n]}$ and $\mathcal{E}\in\mathbb{C}^{[m',n]}$ such that

$D_{ii\ldots i} = 1$, $E_{ii\ldots i} = 1$ and all other entries are zero. Let $\mathcal{L}$ be $n \times n$ zero matrix. Then (2.4) becomes

$$\begin{pmatrix} x_1^{m-1} - \lambda x_1^{m'-1} \\ \vdots \\ x_n^{m-1} - \lambda x_n^{m'-1} \\ \eta^T x + \eta_0 \end{pmatrix} = \begin{pmatrix} x_1^{m'-1}(x_1^{m-m'} - \lambda) \\ \vdots \\ x_n^{m'-1}(x_n^{m-m'} - \lambda) \\ \eta^T x + \eta_0 \end{pmatrix} = 0, \tag{2.19}$$

where $\eta = (\eta_1, \cdots, \eta_n) \in \mathbb{C}^n$ and η_i's are generic. Clearly, $x = 0$ cannot be a solution since generically $\eta_0 \neq 0$. Thus at least one of

$$x_1^{m-m'} - \lambda = 0, \quad \ldots, \quad x_n^{m-m'} - \lambda = 0 \tag{2.20}$$

must be valid. Assume i $(1 \leq i \leq n)$ equations of (2.20) hold. If the first i equations of (2.20) hold, then

$$\begin{aligned} x_j^{m-m'} - \lambda &= 0, \quad j = 1, \cdots, i, \\ x_j^{m'-1} &= 0, \quad j = i+1, \cdots, n. \end{aligned}$$

For $j = i+1, \cdots, n$, x_j can be 0 with multiplicity $m' - 1$. Also from the first i equations,

$$x_2^{m-m'} = x_1^{m-m'}, \ldots, x_i^{m-m'} = x_1^{m-m'}.$$

So each x_j $(j = 2, \ldots, n)$ can be expressed by x_1 in $m - m'$ ways. Correspondingly, x_1 can be determined uniquely under the choices of $x_{i+1}, \ldots, x_n$ and the last linear equation, so is λ. Therefore, there are $(m'-1)^{n-i}(m-m')^{i-1}$ solutions in total if the first i equations of (2.20) are valid. This argument holds for any i equations of (2.20) are chosen to be valid.

So there are

$$\binom{n}{i}(m'-1)^{n-i}(m-m')^{i-1}$$

solutions when i equations of (2.20) are true. Since i may be any one of $\{1,\cdots,n\}$, the number of zeros of (2.20) in total should be

$$\begin{aligned}\sum_{i=1}^{n}\binom{n}{i}(m'-1)^{n-i}(m-m')^{i-1} &= \frac{1}{m-m'}\sum_{i=1}^{n}\binom{n}{i}(m'-1)^{n-i}(m-m')^{i}\\ &= \frac{1}{m-m'}[(m'-1+m-m')^n-(m'-1)^n]\\ &= \frac{(m-1)^n-(m'-1)^n}{m-m'}.\end{aligned}$$

By Lemma 2.1.4,

$$\mathrm{MV}_n(S_1\cup\{0\},\ldots,S_n\cup\{0\})\geq\frac{(m-1)^n-(m'-1)^n}{m-m'}.$$

Combining the above inequality with (2.18), we have

$$\mathrm{MV}_n(S_1\cup\{0\},\ldots,S_n\cup\{0\})=\frac{(m-1)^n-(m'-1)^n}{m-m'}.$$

□

Let $\mathcal{N}_{F_1}$ be defined in Lemma 2.1.1 for $F_1(\lambda,x;c)$ in (2.4). In the following lemma, we compute $\mathcal{N}_{F_1}$.

LEMMA 2.2.1. *Let $\mathcal{N}_{F_1}$ be as in Lemma 2.1.1 for $F_1(\lambda, x; c)$ as in (2.4). Then*

$$\mathcal{N}_{F_1} = \frac{(m-1)^n - (m'-1)^n}{m - m'}. \tag{2.21}$$

Proof: By Lemma 2.1.1, $\mathcal{N}_{F_1}$ is the upper bound of the number of the nonsingular zeros of $F_1(\lambda, x; c)$ in (2.4). Since nonsingular zeros are isolated zeros, Theorem 2.2.1 implies that

$$\mathcal{N}_{F_1} \leq \frac{(m-1)^n - (m'-1)^n}{m - m'}.$$

On the other hand, $F_1(\lambda, x; c)$ in (2.4) has $((m-1)^n - (m'-1)^n)/(m - m')$ nonsingular zeros when c is generic. So by Lemma 2.1.1,

$$\frac{(m-1)^n - (m'-1)^n}{m - m'} \leq \mathcal{N}_{F_1}.$$

□

Now it is sufficient to find a parameter set c_0 such that $F_1(\lambda, x; c_0)$ defined in (2.4) has $\mathcal{N}_{F_1}$ nonsingular zeros. Let

$$c_0 := \{I^{[m,n]}, I^{[m',n]}, \mathcal{G}\}, \tag{2.22}$$

where $I^{[m,n]}$ is the m-th order n dimensional identity tensor, $\mathcal{G} \in \mathbb{C}^{[2,n]}$ is a diagonal matrix with $\mathcal{G}_{ii} = \alpha_i$ $(i = 1, \ldots, n)$ being randomly chosen nonzero complex numbers. Let

$$G_1(\lambda, x) := F_1(\lambda, x; c_0).$$

Then

$$G_1(\lambda, x) = \begin{pmatrix} x_1^{m-1} - \lambda x_1^{m'-1} - \alpha_1 x_1^{m-m'} + \alpha_1 \lambda \\ \vdots \\ x_n^{m-1} - \lambda x_n^{m'-1} - \alpha_n x_n^{m-m'} + \alpha_n \lambda \\ \eta^T x + \eta_0 \end{pmatrix} = \begin{pmatrix} (x_1^{m'-1} - \alpha_1)(x_1^{m-m'} - \lambda) \\ \vdots \\ (x_n^{m'-1} - \alpha_n)(x_n^{m-m'} - \lambda) \\ \eta^T x + \eta_0 \end{pmatrix}, \tag{2.23}$$

THEOREM 2.2.2. *Let $G_1(\lambda, x)$ and $\mathcal{N}_{F_1}$ be as in (2.23) and (2.21) respectively. Then $G_1(\lambda, x)$ has exactly $\mathcal{N}_{F_1}$, as given in (2.21), nonsingular zeros.*

Proof: It can be asserted that at least one of

$$\begin{aligned} x_1^{m-m'} - \lambda &= 0, \\ &\vdots \\ x_n^{m-m'} - \lambda &= 0 \end{aligned} \tag{2.24}$$

must be true. Otherwise system (2.23) is equivalent to an overdetermined system of $n+1$ equations in n unknowns, which has no solutions due to randomness. Assume that i $(1 \leq$

$i \leq n$) equations of (2.24) are true. If the first i equations of (2.24) hold, then

$$\begin{aligned} x_1^{m-m'} - \lambda &= 0, \\ &\vdots \\ x_i^{m-m'} - \lambda &= 0, \\ x_{i+1}^{m'-1} - \alpha_{i+1} &= 0, \\ &\vdots \\ x_n^{m'-1} - \alpha_n &= 0 \end{aligned}$$

From the $(i+1)$-th equation to the n-th equation, each x_j $(j = i+1, \ldots, n)$ can be any one of the $(m'-1)$-th root of α_j. Also from the first i equations,

$$\begin{aligned} x_2^{m-m'} &= x_1^{m-m'}, \\ &\vdots \\ x_i^{m-m'} &= x_1^{m-m'}. \end{aligned}$$

So each x_j $(j = 2, \ldots, n)$ can be expressed in x_1 by $m - m'$ ways. Correspondingly, x_1 can be determined uniquely by $x_{i+1}, \ldots, x_n$ and the last equation, and λ can also be determined uniquely. Therefore, there are $(m'-1)^{n-i}(m-m')^{i-1}$ solutions in total if the first i equations of (2.24) hold. This reasoning holds for any i equations of (2.24) are true. So there are

$$\binom{n}{i} (m'-1)^{n-i}(m-m')^{i-1}$$

solutions in this situation. Since i may be any one of $\{1,\ldots,n\}$, the number of isolated zeros of $G_1(\lambda,x)$ in total should be

$$\begin{aligned}\sum_{i=1}^{n}\binom{n}{i}(m'-1)^{n-i}(m-m')^{i-1} &= \frac{1}{m-m'}\sum_{i=1}^{n}\binom{n}{i}(m'-1)^{n-i}(m-m')^{i}\\ &= \frac{1}{m-m'}[(m'-1+m-m')^n-(m'-1)^n]\\ &= \frac{(m-1)^n-(m'-1)^n}{m-m'}.\end{aligned}$$

We now show that each zero of $G_1(\lambda,x)$ in (2.23) must be nonsingular. As discussed above, any zero (λ^*,x^*) of $G_1(\lambda,x)$ satisfies

$$\begin{aligned}(x_j^*)^{m-m'}-\lambda &= 0, \qquad j\in I_i\\ (x_j^*)^{m'-1}-\alpha_j &= 0, \qquad j\in\{1,\cdots,n\}\backslash I_i\\ \eta_1x_1^*+\cdots+\eta_nx_n^*+\eta_0 &= 0,\end{aligned}$$

where I_i is an index set which contains i elements of $\{1,\ldots,n\}$ for $1\le i\le n$. Without loss of generality, we assume $I_i=\{1,\ldots,i\}$. Then

$$\begin{aligned}(x_j^*)^{m-m'}-\lambda &= 0, \qquad j=1,\cdots,i,\\ (x_j^*)^{m'-1}-\alpha_j &= 0, \qquad j=i+1,\cdots,n,\\ \eta_1x_1^*+\cdots+\eta_nx_n^*+\eta_0 &= 0.\end{aligned}\tag{2.25}$$

Let $DG_1(\lambda,x)$ be the Jacobian of $G_1(\lambda,x)$ with respect to (λ,x). To show $DG_1(\lambda^*,x^*)$ is

nonsingular, let

$$\begin{aligned} A_j(\lambda, x) &:= -(x_j^{m'-1} - \alpha_j), \\ B_j(\lambda, x) &:= (m'-1)x_j^{m'-2}(x_j^{m-m'} - \lambda) + (m-m')x_j^{m-m'-1}(x_j^{m'-1} - \alpha_j) \end{aligned}$$

for $j = 1, \ldots, n$. Then

$$DG_1(\lambda, x) = \begin{pmatrix} A_1 & B_1 & & & & & \\ \vdots & & \ddots & & & & \\ A_i & & & B_i & & & \\ A_{i+1} & & & & B_{i+1} & & \\ \vdots & & & & & \ddots & \\ A_n & & & & & & B_n \\ 0 & \eta_1 & \ldots & \eta_i & \eta_{i+1} & \ldots & \eta_n \end{pmatrix}.$$

Note that

$$A_j(\lambda^*, x^*) = \begin{cases} -((x_j^*)^{m'-1} - \alpha_j), & j = 1, \ldots, i \\ 0, & j = i+1, \ldots, n \end{cases}$$

and by (2.25),

$$B_j(\lambda^*, x^*) = \begin{cases} (m-m')(x_j^*)^{m-m'-1}((x_j^*)^{m'-1} - \alpha_j), & j = 1, \ldots, i \\ (m'-1)(x_j^*)^{m'-2}((x_j^*)^{m-m'} - \lambda), & j = i+1, \ldots, n. \end{cases}$$

For simplicity, write $A_j^* := A_j(\lambda^*, x^*)$ and $B_j^* := B_j(\lambda^*, x^*)$. Then

$$DG_1(\lambda^*, x^*) = \begin{pmatrix} A_1^* & B_1^* & & & & & \\ \vdots & & \ddots & & & & \\ A_i^* & & & B_i^* & & & \\ 0 & & & & B_{i+1}^* & & \\ \vdots & & & & & \ddots & \\ 0 & & & & & & B_n^* \\ 0 & \eta_1 & \dots & \eta_i & \eta_{i+1} & \dots & \eta_n \end{pmatrix}.$$

So

$$\begin{aligned} & \det(DG_1(\lambda^*, x^*)) \\ = & \left(\prod_{j=i+1}^{n} (-1)^{(i+1)+(i+2)} B_j^* \right) \det \begin{pmatrix} A_1^* & B_1^* & & & & & & \\ \vdots & & \ddots & & & & & \\ A_{l-1}^* & & & B_{l-1}^* & & & & \\ A_l^* & & & & B_l^* & & & \\ A_{l+1}^* & & & & & B_{l+1}^* & & \\ \vdots & & & & & & \ddots & \\ A_i^* & & & & & & & B_i^* \\ 0 & \eta_1 & \dots & \eta_{l-1} & \eta_l & \eta_{l+1} & \dots & \eta_i \end{pmatrix} \end{aligned}$$

$$= \left(\prod_{j=i+1}^{n}(-1)B_j^*\right)\sum_{l=1}^{i}(-1)^{(i+1)+(l+1)}\eta_l\cdot\det\begin{pmatrix} A_1^* & B_1^* & & & & & \\ \vdots & & \ddots & & & & \\ A_{l-1}^* & & & B_{l-1}^* & & & \\ A_l^* & & & & 0 & & \\ A_{l+1}^* & & & & & B_{l+1}^* & \\ \vdots & & & & & & \ddots \\ A_i^* & & & & & & & B_i^* \end{pmatrix}.$$

Furthermore,

$$\begin{aligned}
&\det(DG_1(\lambda^*,x^*))\\
&= (-1)^{n-i}\left(\prod_{j=i+1}^{n}B_j^*\right)\sum_{l=1}^{i}(-1)^{(i+1)+(l+1)}\eta_l\cdot(-1)^{l+1}A_l^*\prod_{\substack{k=1\\k\neq l}}^{i}B_k^*\\
&= (-1)^{n+1}\left(\prod_{j=i+1}^{n}B_j^*\right)\sum_{l=1}^{i}\eta_l A_l^*\prod_{\substack{k=1\\k\neq l}}^{i}B_k^*\\
&= (-1)^{n+1}\left(\prod_{j=i+1}^{n}B_j^*\right)\sum_{l=1}^{i}\eta_l[-((x_l^*)^{m'-1}-\alpha_l)]\prod_{\substack{k=1\\k\neq l}}^{i}(m-m')(x_k^*)^{m-m'-1}((x_k^*)^{m'-1}-\alpha_k)\\
&= (-1)^n(m-m')^{i-1}\left(\prod_{j=i+1}^{n}B_j^*\right)\left(\prod_{k=1}^{i}((x_k^*)^{m'-1}-\alpha_k)\right)\sum_{l=1}^{i}\eta_l\left(\prod_{\substack{k=1\\k\neq l}}^{i}(x_k^*)^{m-m'-1}\right)\neq 0
\end{aligned}$$

by (2.25). □

By Lemmas 2.1.1, 2.1.2, Theorem 2.2.1, Lemma 2.2.1 and Theorem 2.2.2, we have proved the following theorem.

THEOREM 2.2.3. *Let $H_1(\lambda, x, t)$ be defined as*

$$H_1(\lambda, x, t) := (1-t)\gamma G_1(\lambda, x) + tP(\lambda, x), \quad t \in [0, 1] \tag{2.26}$$

where $G_1(\lambda, x)$ and $P(\lambda, x)$ are given by (2.23) and (1.12) respectively. Then following solution paths of $H_1(\lambda, x, t) = 0$ in (2.26) will give all isolated solutions of (1.12) for $m > m'$.

2.3 Construct a linear homotopy to compute generalized tensor eigenpairs when $m < m'$

Let $c = \{\mathcal{D}, \mathcal{E}, \mathcal{L}\}$ as defined in (2.3). Consider

$$F_2(\lambda, x; c) = \begin{pmatrix} \mathcal{D}x^{m-1} - \lambda \mathcal{E}x^{m'-1} - \text{vdiag}(\mathcal{L}) + \lambda \mathcal{L}x^{[m'-m]} \\ \eta^T x + \eta_0 \end{pmatrix}. \tag{2.27}$$

Clearly $F_2(\lambda, x; c)$ is linear in c. For $c_1 = \{\mathcal{A}, \mathcal{B}, 0\}$ defined in (2.5), $F_2(\lambda, x; c_1)$ agrees with our target system (1.12). Similar to Theorem 2.2.1 we have the following theorem.

THEOREM 2.3.1. *Let $F_2(\lambda, x; c)$ be given as in (2.27) with the set of parameters c being as in (2.3). Then the number of isolated zeros, counting multiplicities, of $F_2(\lambda, x; c)$ is bounded above by*

$$\frac{(m'-1)^n - (m-1)^n}{m'-m}.$$

When c is generic, $F_2(\lambda, x; c)$ has exactly $((m'-1)^n - (m-1)^n)/(m'-m)$ isolated zeros.

Proof: For the random hyperplane $\eta^T x + \eta_0 = 0$, i.e., $\eta_1 x_1 + \cdots + \eta_n x_n + \eta_0 = 0$, in

(2.27), we may suppose $\eta_n \neq 0$. Then

$$x_n = a_1 x_1 + \cdots + a_{n-1} x_{n-1} + b, \tag{2.28}$$

where $a_i = -\eta_i/\eta_n$ for $i = 1, \ldots, n-1$ and $b = -\eta_0/\eta_n$. Notice that the number of solutions of (2.27) in $\mathbb{C}^{n+1}$ is the same as the number of solutions in $\mathbb{C}^n$ of the system $T^*(\lambda, x_1, \ldots, x_{n-1})$ resulting from substituting (2.28) into the first n equations in (2.27). Denote the corresponding supports of T^* by $S_1, \ldots, S_n$. We claim that

$$\mathrm{MV}_n(S_1 \cup \{0\}, \ldots, S_n \cup \{0\}) = \frac{(m'-1)^n - (m-1)^n}{m' - m}. \tag{2.29}$$

Let N denote the number of isolated zeros of (2.27) over $\mathbb{C}$. Then (2.29), when it is proved, implies

$$N \leq \frac{(m'-1)^n - (m-1)^n}{m' - m}.$$

When the parameter set $c = \{\mathcal{D}, \mathcal{E}, \mathcal{L}\}$ is generic, the equality in the above holds from Lemma 2.1.3 and Lemma 2.1.4.

To prove (2.29), let $\bar{c} := \{\bar{\mathcal{D}}, \bar{\mathcal{E}}, \bar{\mathcal{L}}\}$ be generic. Similar to (2.27) the corresponding polynomial system is

$$\bar{T}(\lambda, x) = \begin{pmatrix} \bar{\mathcal{D}} x^{m-1} - \lambda \bar{\mathcal{E}} x^{m'-1} - \mathrm{vdiag}(\bar{\mathcal{L}}) + \lambda \bar{\mathcal{L}} x^{[m'-m]} \\ \eta^T x + \eta_0 \end{pmatrix} = 0. \tag{2.30}$$

Substituting (2.28) into the first n equations of (2.30) yields a new system $\bar{T}^*(\lambda, x_1, \ldots, x_{n-1})$. Let $\bar{S}_1, \ldots, \bar{S}_n$ be the corresponding supports of $\bar{T}^*$. Since $\bar{\mathcal{D}}$, $\bar{\mathcal{E}}$ and $\bar{\mathcal{L}}$ are generic, without

loss of generality one can assume that all monomials $1, \lambda x_1^{m'-m}, \ldots, \lambda x_n^{m'-m}$ with

$$\{x_1^{\alpha_1} x_2^{\alpha_2} \ldots x_n^{\alpha_n} \Big| \alpha_i \in \mathbb{Z}_{\geq 0},\, \alpha_1 + \alpha_2 + \cdots + \alpha_n = m - 1\}$$

and

$$\{\lambda x_1^{\alpha_1} x_2^{\alpha_2} \ldots x_n^{\alpha_n} \Big| \alpha_i \in \mathbb{Z}_{\geq 0},\, \alpha_1 + \alpha_2 + \cdots + \alpha_n = m' - 1\}$$

will appear in each of the first n equations in (2.8). Therefore, after substituting (2.6) into the first n equations of (2.8), all monomials

$$\{x_1^{\alpha_1} x_2^{\alpha_2} \ldots x_{n-1}^{\alpha_{n-1}} \Big| \alpha_i \in \mathbb{Z}_{\geq 0},\, \alpha_1 + \alpha_2 + \cdots + \alpha_{n-1} \leq m - 1\}$$

and

$$\{\lambda x_1^{\alpha_1} x_2^{\alpha_2} \ldots x_{n-1}^{\alpha_{n-1}} \Big| \alpha_i \in \mathbb{Z}_{\geq 0},\, \alpha_1 + \alpha_2 + \cdots + \alpha_{n-1} \leq m' - 1\}$$

are contained in each equation of $\bar{T}^*$. Consequently, $\bar{S}_1, \ldots, \bar{S}_n$ are all equal to

$$\bar{S} := \{(0, \alpha) \,\big|\, \alpha \in (\mathbb{Z}_{\geq 0}^{n-1})^T, |\alpha| \leq m - 1\} \cup \{(1, \alpha) \,\big|\, \alpha \in (\mathbb{Z}_{\geq 0}^{n-1})^T, |\alpha| \leq m' - 1\}.$$

Let $\bar{Q}$ be the convex hull of $\bar{S}$. To compute the volume of $\bar{Q}$, we employ a geometric approach here. As before, denote the i-th unit vector in $(\mathbb{R}^n)^T$ by e_i for $i = 1, \ldots, n$. Then the vertices of $\bar{Q}$ are

$$z_i = \begin{cases} 0, & i = 0 \\ (m-1)e_{n+1-i}, & 1 \leq i \leq n-1 \\ e_1, & i = n \\ e_1 + (m'-1)e_{2n+1-i}, & n+1 \leq i \leq 2n-1. \end{cases} \tag{2.31}$$

So,

$$z_{n+i} = e_1 + \frac{m'-1}{m-1} z_i, \quad 0 \le i \le n-1.$$

From which $z_n, \ldots, z_{2n-1}$ are scaling of $z_0, \ldots, z_{n-1}$ by a factor$(m'-1)/(m-1)$ followed by a shift respectively. In fact, each z_{n+i} is obtained by moving z_i along the line

$$l_i(t) := (1-t)z_i + tz_{n+i}, \quad 0 \le i \le n-1 \tag{2.32}$$

as t changing from 0 to 1. A simple computation yields

$$l_i(t) = \begin{cases} te_1, & i = 0 \\ te_1 + (m-1+(m'-m)t)e_{n+1-i}, & 1 \le i \le n-1. \end{cases} \tag{2.33}$$

Notice that when $m-1+(m'-m)t = 0$, i.e., $t = (1-m)/(m'-m)$, each l_i $(1 \le i \le n-1)$ will have the same intersection point

$$w := \frac{1-m}{m'-m} e_1$$

with the line $l_0(t)$. Moreover, all the first coordinates of $z_0, z_1, \ldots, z_{n-1}$ are 0 and all the first coordinates of $z_n, \ldots, z_{2n-1}$ are 1. Write

$$z_i := \begin{cases} (0, y_i), & i = 0, 1, \ldots, n-1 \\ (1, y_i), & i = n, \ldots, 2n-1. \end{cases}$$

Denote the i-th unit vector in $(\mathbb{R}^{n-1})^T$ by u_i for $i = 1, \ldots, n-1$. By (2.31),

$$y_i = \begin{cases} 0, & i = 0, n \\ (m-1)u_{n-i}, & i = 1, \ldots, n-1 \\ (m'-1)u_{2n-i}, & i = n+1, \ldots, 2n-1. \end{cases}$$

Let Δ_0 be the convex hull of $y_0, y_1, \ldots, y_{n-1}$ and Δ_1 be the convex hull of $y_n, \ldots, y_{2n-1}$. Then Δ_0 is a simplex with

$$\begin{aligned} \mathrm{vol}_{n-1}(\Delta_0) &= \frac{1}{(n-1)!} \left| \det \begin{pmatrix} y_1 - y_0 \\ y_2 - y_0 \\ \vdots \\ y_{n-1} - y_0 \end{pmatrix} \right| = \frac{1}{(n-1)!} \left| \det \begin{pmatrix} (m-1)u_{n-1} \\ (m-1)u_{n-2} \\ \vdots \\ (m-1)u_1 \end{pmatrix} \right| \\ &= \frac{(m-1)^{n-1}}{(n-1)!} \left| \det \begin{pmatrix} u_{n-1} \\ u_{n-2} \\ \vdots \\ u_1 \end{pmatrix} \right| = \frac{(m-1)^{n-1}}{(n-1)!}. \end{aligned}$$

Similarly, Δ_1 is also a simplex with

$$\mathrm{vol}_{n-1}(\Delta_1) = \frac{(m'-1)^{n-1}}{(n-1)!}.$$

Therefore, when $m < m'$, as t decreases from 1 to 0 to $(1-m)/(m'-m)$, Δ_1 shrinks to Δ_0 and then shrinks to one point w. See Figure 2.1 for the case of $m = 3$, $m' = 4$ and $n = 3$.

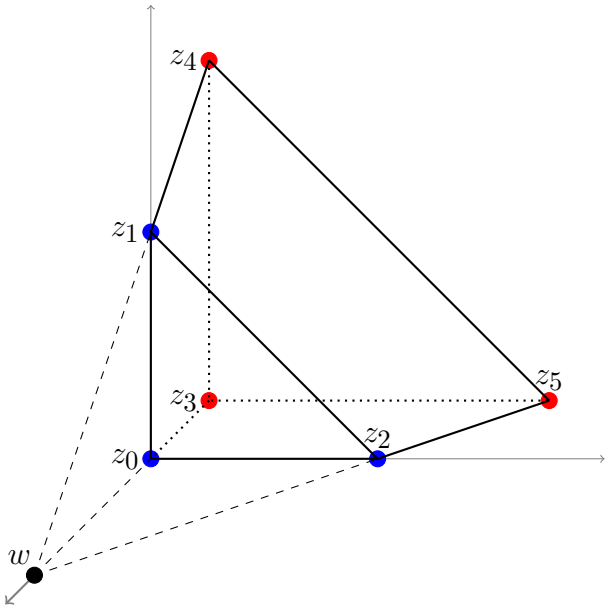

Figure 2.1: Diagram of $\bar{Q}$ when $m = 3$, $m' = 4$ and $n = 3$

Let

$$\begin{aligned} \bar{Q}_0 &:= \operatorname{conv}(z_0, z_1, \ldots, z_{n-1}, w), \\ \bar{Q}_1 &:= \operatorname{conv}(z_n, z_{n+1}, \ldots, z_{2n-1}, w). \end{aligned}$$

Then

$$\operatorname{vol}_n(\bar{Q}) = \operatorname{vol}_n(\bar{Q}_1) - \operatorname{vol}_n(\bar{Q}_0).$$

Note that

$$\begin{aligned}\mathrm{Vol}_n(\bar{Q}_0) &= \frac{1}{n!}\left|\det\begin{pmatrix} z_1-z_0 \\ z_2-z_0 \\ \vdots \\ z_{n-1}-z_0 \\ w-z_0 \end{pmatrix}\right| = \frac{1}{n!}\left|\det\begin{pmatrix} (m-1)e_n \\ (m-1)e_{n-1} \\ \vdots \\ (m-1)e_2 \\ \frac{1-m}{m'-m}e_1 \end{pmatrix}\right| = \frac{(m-1)^n}{(m'-m)n!}\left|\det\begin{pmatrix} e_n \\ e_{n-1} \\ \vdots \\ e_2 \\ e_1 \end{pmatrix}\right| \\ &= \frac{(m-1)^n}{(m'-m)n!}\left|\det\begin{pmatrix} e_1 \\ \vdots \\ e_n \end{pmatrix}\right| = \frac{(m-1)^n}{(m'-m)n!}\end{aligned}$$

and

$$\begin{aligned}\mathrm{Vol}_n(\bar{Q}_1) &= \frac{1}{n!}\left|\det\begin{pmatrix} z_{n+1}-z_n \\ z_{n+2}-z_n \\ \vdots \\ z_{2n-1}-z_n \\ w-z_n \end{pmatrix}\right| = \frac{1}{n!}\left|\det\begin{pmatrix} (m'-1)e_n \\ (m'-1)e_{n-1} \\ \vdots \\ (m'-1)e_2 \\ \frac{1-m'}{m'-m}e_1 \end{pmatrix}\right| = \frac{(m'-1)^n}{(m'-m)n!}\left|\det\begin{pmatrix} e_n \\ e_{n-1} \\ \vdots \\ e_2 \\ e_1 \end{pmatrix}\right| \\ &= \frac{(m'-1)^n}{(m'-m)n!}\left|\det\begin{pmatrix} e_1 \\ \vdots \\ e_n \end{pmatrix}\right| = \frac{(m'-1)^n}{(m'-m)n!}.\end{aligned}$$

Hence,

$$\mathrm{Vol}_n(\bar{Q}) = \frac{(m'-1)^n}{(m'-m)n!} - \frac{(m-1)^n}{(m'-m)n!} = \frac{(m'-1)^n-(m-1)^n}{(m'-m)n!}. \tag{2.34}$$

Therefore, by Lemma 2.1.5,

$$\mathrm{MV}_n(\bar{S}_1, \ldots, \bar{S}_n) = n!\mathrm{Vol}_n(\bar{Q}) = \frac{(m'-1)^n - (m-1)^n}{m'-m}.$$

Since $S_i \cup \{0\}$ is a subset of $\bar{S}_i$ for $i = 1, \ldots, n$, it follows that

$$\mathrm{MV}_n(S_1 \cup \{0\}, \ldots, S_n \cup \{0\}) \leq \mathrm{MV}_n(\bar{S}_1, \ldots, \bar{S}_n).$$

Consequently,

$$\mathrm{MV}_n(S_1 \cup \{0\}, \ldots, S_n \cup \{0\}) \leq \frac{(m'-1)^n - (m-1)^n}{m'-m}. \tag{2.35}$$

For the other direction, consider the identity tensors $\mathcal{D} \in \mathbb{C}^{[m,n]}$ and $\mathcal{E} \in \mathbb{C}^{[m',n]}$ in which $D_{ii\ldots i} = 1$, $E_{ii\ldots i} = 1$ and all other entries are zero. Let $\mathcal{L}$ be the $n \times n$ zero matrix. Then (2.27) becomes

$$\begin{pmatrix} x_1^{m-1} - \lambda x_1^{m'-1} \\ \vdots \\ x_n^{m-1} - \lambda x_n^{m'-1} \\ \eta^T x + \eta_0 \end{pmatrix} = \begin{pmatrix} x_1^{m-1}(1 - \lambda x_1^{m'-m}) \\ \vdots \\ x_n^{m-1}(1 - \lambda x_n^{m'-m}) \\ \eta^T x + \eta_0 \end{pmatrix} = 0, \tag{2.36}$$

where $\eta = (\eta_1, \cdots, \eta_n) \in \mathbb{C}^n$ are generic. Apparently $x = 0$ cannot be a solution since $\eta_0 \neq 0$ in the augmented random hyperplane. Hence at least one of

$$1 - \lambda x_1^{m'-m}, \quad \ldots, \quad 1 - \lambda x_n^{m'-m} \tag{2.37}$$

must be zero. Assume i $(1 \leq i \leq n)$ items of (2.37) are zero. If the first i items of (2.37) are

zero, then

$$\begin{aligned} 1 - \lambda x_j^{m'-m} &= 0, \quad j = 1, \cdots, i, \\ x_j^{m-1} &= 0, \quad j = i+1, \cdots, n. \end{aligned}$$

From the $(i+1)$-th equation to the n-th equation, each x_j $(j = i+1, \ldots, n)$ can be 0 with multiplicity $m-1$. But λ cannot be 0, otherwise the first n equations will result in $x_1 = \cdots = x_n = 0$ making the last equation of (2.36) invalid. Thus from the first i equations,

$$x_2^{m'-m} = x_1^{m'-m}, \ldots, x_i^{m'-m} = x_1^{m'-m}.$$

So each x_j $(j = 2, \ldots, n)$ can be expressed in x_1 by $m' - m$ ways. Correspondingly, x_1 can be determined uniquely by the choices of $x_{i+1}, \ldots, x_n$ and the last equation, and λ can also be determined uniquely. Therefore, there are $(m-1)^{n-i}(m'-m)^{i-1}$ solutions in total if the first i equations of (2.37) are valid. This argument holds for any i equations of (2.37) are valid. In this situation, there are

$$\binom{n}{i} (m-1)^{n-i}(m'-m)^{i-1}$$

solutions. Since i may be any one of $\{1, \ldots, n\}$, the number of zeros of (2.37) in total should

be

$$\begin{aligned}\sum_{i=1}^{n}\binom{n}{i}(m-1)^{n-i}(m'-m)^{i-1} &= \frac{1}{m'-m}\sum_{i=1}^{n}\binom{n}{i}(m-1)^{n-i}(m'-m)^{i}\\ &= \frac{1}{m'-m}[(m-1+m'-m)^n-(m-1)^n]\\ &= \frac{(m'-1)^n-(m-1)^n}{m'-m}.\end{aligned}$$

By Lemma 2.1.4,

$$\mathrm{MV}_n(S_1\cup\{0\},\ldots,S_n\cup\{0\})\geq\frac{(m'-1)^n-(m-1)^n}{m'-m}.$$

So,

$$\mathrm{MV}_n(S_1\cup\{0\},\ldots,S_n\cup\{0\})=\frac{(m'-1)^n-(m-1)^n}{m'-m}.$$

□

Let $\mathcal{N}_{F_2}$ be as in Lemma 2.1.1 with $F_2(\lambda,x;c)$ being as in (2.27). Similar to Lemma 2.2.1, we have the following lemma.

LEMMA 2.3.1. *Let $\mathcal{N}_{F_2}$ be as in Lemma 2.1.1 with $F_2(\lambda,x;c)$ being as in (2.27). Then*

$$\mathcal{N}_{F_2}=\frac{(m'-1)^n-(m-1)^n}{m'-m}. \tag{2.38}$$

Proof: By Lemma 2.1.1, $\mathcal{N}_{F_2}$ is an upper bound of the number of nonsingular zeros of

$F_2(\lambda, x; c)$ in (2.27). Since nonsingular zeros are isolated zeros, by Theorem 2.3.1,

$$\mathcal{N}_{F_2} \leq \frac{(m'-1)^n - (m-1)^n}{m'-m}.$$

On the other hand, $F_2(\lambda, x; c)$ in (2.27) has $((m'-1)^n - (m-1)^n)/(m'-m)$ nonsingular zeros when c is generic. So by Lemma 2.1.1,

$$\frac{(m'-1)^n - (m-1)^n}{m'-m} \leq \mathcal{N}_{F_2}.$$

□

Let $c_0 = \{I^{[m,n]}, I^{[m',n]}, \mathcal{G}\}$ be as in (2.22) and

$$G_2(\lambda, x) := F_2(\lambda, x; c_0).$$

Then,

$$G_2(\lambda, x) = \begin{pmatrix} x_1^{m-1} - \lambda x_1^{m'-1} - \alpha_1 + \lambda\alpha_1 x_1^{m'-m} \\ \vdots \\ x_n^{m-1} - \lambda x_n^{m'-1} - \alpha_n + \lambda\alpha_n x_n^{m'-m} \\ \eta^T x + \eta_0 \end{pmatrix} = \begin{pmatrix} (x_1^{m-1} - \alpha_1)(1 - \lambda x_1^{m'-m}) \\ \vdots \\ (x_n^{m-1} - \alpha_n)(1 - \lambda x_n^{m'-m}) \\ \eta^T x + \eta_0 \end{pmatrix}. \tag{2.39}$$

THEOREM 2.3.2. *Let $G_2(\lambda, x)$ and $\mathcal{N}_{F_2}$ be as in (2.39) and (2.38) respectively. Then $G_2(\lambda, x)$ has exactly $\mathcal{N}_{F_2}$ nonsingular zeros.*

Proof: At least one of the equations

$$\begin{aligned} 1-\lambda x_1^{m'-m} &= 0, \\ &\vdots \\ 1-\lambda x_n^{m'-m} &= 0 \end{aligned} \tag{2.40}$$

must hold, otherwise system (2.39) is equivalent to an overdetermined system of $n+1$ equations in n unknowns, which has no solutions due to randomness. Assume that i ($1 \leq i \leq n$) equations of (2.40) are true. Without loss, we may suppose they are the first i equations. Then

$$\begin{aligned} 1-\lambda x_1^{m'-m} &= 0, \\ &\vdots \\ 1-\lambda x_i^{m'-m} &= 0, \\ x_{i+1}^{m-1}-\alpha_{i+1} &= 0, \\ &\vdots \\ x_n^{m-1}-\alpha_n &= 0 \end{aligned}$$

From the $(i+1)$-th equation to the n-th equation, each x_j ($j = i+1, \ldots, n$) can be one of the $(m-1)$-th root of α_j. Also from the first i equations, $\lambda = 0$ cannot be a solution, since (2.39) will then be an overdetermined system with $n+1$ equations in n unknowns.

Consequently,

$$\begin{aligned} x_2^{m'-m} &= x_1^{m'-m}, \\ &\vdots \\ x_i^{m'-m} &= x_1^{m'-m}. \end{aligned}$$

So each x_j $(j = 2, \ldots, n)$ can be expressed in x_1 by $m'-m$ ways. Thus, x_1 can be determined uniquely by the choices of $x_{i+1}, \ldots, x_n$ and the last equation, and λ can also be determined uniquely. Therefore, there are $(m-1)^{n-i}(m'-m)^{i-1}$ solutions in total in this situation. This argument holds for any i equations of (2.24) are valid, and there are

$$\binom{n}{i}(m-1)^{n-i}(m'-m)^{i-1}$$

isolated solutions. Since i may be any one of $\{1, \ldots, n\}$, the number of zeros of $G_2(\lambda, x)$ in total should be

$$\begin{aligned} \sum_{i=1}^{n}\binom{n}{i}(m-1)^{n-i}(m'-m)^{i-1} &= \frac{1}{m'-m}\sum_{i=1}^{n}\binom{n}{i}(m-1)^{n-i}(m'-m)^{i} \\ &= \frac{1}{m'-m}[(m-1+m'-m)^n - (m-1)^n] \\ &= \frac{(m'-1)^n - (m-1)^n}{m'-m}. \end{aligned}$$

We now show that each isolated zero of $G_2(\lambda, x)$ must be nonsingular. As discussed

above, any zero (λ^*, x^*) of $G_2(\lambda, x)$ satisfies

$$\begin{aligned} 1 - \lambda(x_j^*)^{m-m'} &= 0, \qquad j \in I_i \\ (x_j^*)^{m-1} - \alpha_j &= 0, \qquad j \in \{1, \cdots, n\} \backslash I_i \\ \eta_1 x_1^* + \cdots + \eta_n x_n^* + \eta_0 &= 0, \end{aligned}$$

where I_i is an index set containing i distinct elements of $\{1, \ldots, n\}$ for some $1 \leq i \leq n$. Without loss of generality, we assume $I_i = \{1, \ldots, i\}$. Hence

$$\begin{aligned} 1 - \lambda(x_j^*)^{m-m'} &= 0, \qquad j = 1, \cdots, i, \\ (x_j^*)^{m-1} - \alpha_j &= 0, \qquad j = i+1, \cdots, n, \\ \eta_1 x_1^* + \cdots + \eta_n x_n^* + \eta_0 &= 0. \end{aligned} \tag{2.41}$$

for some $1 \leq i \leq n$. Let $DG_2(\lambda, x)$ be the Jacobian of $G_2(\lambda, x)$ with respect to (λ, x). To show $DG_2(\lambda^*, x^*)$ is nonsingular, let

$$\begin{aligned} A_j(\lambda, x) &:= -x_j^{m'-m}(x_j^{m-1} - \alpha_j), \\ B_j(\lambda, x) &:= (m-1)x_j^{m-2}(1 - \lambda x_j^{m-m'}) - (m'-m)\lambda x_j^{m'-m-1}(x_j^{m-1} - \alpha_j) \end{aligned}$$

for $j = 1, \ldots, n$. Then

$$DG_2(\lambda, x) = \begin{pmatrix} A_1 & B_1 & & & & & \\ \vdots & & \ddots & & & & \\ A_i & & & B_i & & & \\ A_{i+1} & & & & B_{i+1} & & \\ \vdots & & & & & \ddots & \\ A_n & & & & & & B_n \\ 0 & \eta_1 & \ldots & \eta_i & \eta_{i+1} & \ldots & \eta_n \end{pmatrix}.$$

Notice that

$$A_j(\lambda^*, x^*) = \begin{cases} -(x_j^*)^{m'-m}((x_j^*)^{m-1} - \alpha_j), & j = 1, \ldots, i \\ 0, & j = i+1, \ldots, n \end{cases}$$

and

$$B_j(\lambda^*, x^*) = \begin{cases} -(m'-m)\lambda^*(x_j^*)^{m'-m-1}((x_j^*)^{m-1} - \alpha_j), & j = 1, \ldots, i \\ (m-1)(x_j^*)^{m-2}(1 - \lambda(x_j^*)^{m'-m}), & j = i+1, \ldots, n \end{cases}$$

by (2.41). For simplicity, write $A_j^* := A_j(\lambda^*, x^*)$ and $B_j^* := B_j(\lambda^*, x^*)$. Then

$$DG_2(\lambda^*, x^*) = \begin{pmatrix} A_1^* & B_1^* & & & & & \\ \vdots & & \ddots & & & & \\ A_i^* & & & B_i^* & & & \\ 0 & & & & B_{i+1}^* & & \\ \vdots & & & & & \ddots & \\ 0 & & & & & & B_n^* \\ 0 & \eta_1 & \dots & \eta_i & \eta_{i+1} & \dots & \eta_n \end{pmatrix}.$$

So,

$$\begin{aligned} & \det(DG_2(\lambda^*, x^*)) \\ = & \left(\prod_{j=i+1}^{n} (-1)^{(i+1)+(i+2)} B_j^* \right) \det \begin{pmatrix} A_1^* & B_1^* & & & & & & \\ \vdots & & \ddots & & & & & \\ A_{l-1}^* & & & B_{l-1}^* & & & & \\ A_l^* & & & & B_l^* & & & \\ A_{l+1}^* & & & & & B_{l+1}^* & & \\ \vdots & & & & & & \ddots & \\ A_i^* & & & & & & & B_i^* \\ 0 & \eta_1 & \dots & \eta_{l-1} & \eta_l & \eta_{l+1} & \dots & \eta_i \end{pmatrix} \end{aligned}$$

$$= \left(\prod_{j=i+1}^{n}(-1)B_j^*\right)\sum_{l=1}^{i}(-1)^{(i+1)+(l+1)}\eta_l\cdot\det\begin{pmatrix} A_1^* & B_1^* & & & & & \\ \vdots & & \ddots & & & & \\ A_{l-1}^* & & & B_{l-1}^* & & & \\ A_l^* & & & & 0 & & \\ A_{l+1}^* & & & & & B_{l+1}^* & \\ \vdots & & & & & & \ddots \\ A_i^* & & & & & & & B_i^* \end{pmatrix}.$$

And,

$$\begin{aligned}
&\det(DG_2(\lambda^*,x^*))\\
&= (-1)^{n-i}\left(\prod_{j=i+1}^{n}B_j^*\right)\sum_{l=1}^{i}(-1)^{(i+1)+(l+1)}\eta_l\cdot(-1)^{l+1}A_l^*\prod_{\substack{k=1\\k\neq l}}^{i}B_k^*\\
&= (-1)^{n+1}\left(\prod_{j=i+1}^{n}B_j^*\right)\sum_{l=1}^{i}\eta_l A_l^*\prod_{\substack{k=1\\k\neq l}}^{i}B_k^*\\
&= (-1)^{n+1}\left(\prod_{j=i+1}^{n}B_j^*\right)\\
&\quad\cdot\sum_{l=1}^{i}\eta_l[-(x_l^*)^{m'-m}((x_l^*)^{m-1}-\alpha_l)]\prod_{\substack{k=1\\k\neq l}}^{i}[-(m'-m)\lambda^*(x_k^*)^{m'-m-1}((x_k^*)^{m-1}-\alpha_k)]\\
&= (-1)^n(m-m')^{i-1}\left(\prod_{j=i+1}^{n}B_j^*\right)\left(\prod_{k=1}^{i}((x_k^*)^{m-1}-\alpha_k)\right)\sum_{l=1}^{i}\eta_l x_l^*\left(\prod_{\substack{k=1\\k\neq l}}^{i}(x_k^*)^{m'-m-1}\right)\neq 0
\end{aligned}$$

by (2.41). □

By Lemmas 2.1.1, 2.1.2, Theorem 2.3.1, Lemma 2.3.1 and Theorem 2.3.2, we have proved the following theorem.

THEOREM 2.3.3. *Let*

$$H_2(\lambda, x, t) := (1-t)\gamma G_2(\lambda, x) + tP(\lambda, x), \quad t \in [0,1] \tag{2.42}$$

where $G_2(\lambda, x)$ *and* $P(\lambda, x)$ *are given by (2.39) and (1.12) respectively. Then following solution paths of* $H_2(\lambda, x, t) = 0$ *in (2.42) will give all isolated solutions of (1.12) for* $m < m'$.

Chapter 3

Implementation of the linear homotopy methods

Based on Theorem 2.2.3 and Theorem 2.3.3 in Chapter 2, a linear homotopy algorithm can be constructed to compute generalized tensor eigenpairs.

3.1 A linear homotopy algorithm to compute tensor eigenpairs

Suppose $m \neq m'$, $\mathcal{A} \in \mathbb{C}^{[m,n]}$ and $\mathcal{B} \in \mathbb{C}^{[m',n]}$. As discussed in Section 1.4, computing eigenpairs satisfying (1.9) is equivalent to solving (1.12) in the first place, followed by normalizing the corresponding solution for an eigenpair. According to Theorem 2.2.3 and Theorem 2.3.3 in Chapter 2, the following linear homotopy is useful to solve $P(\lambda, x)$ in (1.12):

$$H(\lambda, x, t) = (1-t)\gamma G(\lambda, x) + tP(\lambda, x) = 0, \quad t \in [0, 1), \tag{3.1}$$

where γ is a randomly chosen complex number on the unit circle, $G(\lambda, x)$ is given in (2.23) when $m > m'$ and (2.39) when $m < m'$.

We now present our linear homotopy algorithm for computing generalized tensor eigenpairs when $m \neq m'$.

ALGORITHM 3.1.1. (Compute mode-k $\mathcal{B}$-eigenpairs of $\mathcal{A}$, where $\mathcal{A} \in \mathbb{C}^{[m,n]}$, $\mathcal{B} \in \mathbb{C}^{[m',n]}$.)

Step 1. *Compute all solutions of* $G(\lambda, x) = 0$ *as given in (2.23) or (2.39).*

Step 2. *Compute all solutions* (λ, x) *of (1.9) by following the paths from* $t = 0$ *to* $t = 1$ *using the linear homotopy* $H(\lambda, x, t) = 0$ *defined in (3.1).*

Step 3. *Compute a representative from each equivalence solution class of* $\mathcal{A}^{(k)}x^{m-1} = \lambda\mathcal{B}x^{m'-1}$ *by normalizing each* (λ, x) *obtained in Step 2 for an eigenpair* (λ^*, x^*)*, i.e.,*

$$\lambda^* = \frac{\lambda}{(\mathcal{B}x^{m'})^{(m-m')/m'}}, \quad x^* = \frac{x}{(\mathcal{B}x^{m'})^{1/m'}}$$

to satisfy (1.9).

Step 4. *Compute* m' *equivalent eigenpairs* (λ', x') *for each* (λ^*, x^*) *obtained in Step 2 by* $\lambda' = t^{m-m'}\lambda^*$ *and* $x' = tx^*$ *with* t *being a root of* $t^{m'} = 1$.

REMARK 3.1.1. If only one representative from each equivalence class is required (see, for example, [5]), then Step 4 in the above Algorithm can be skipped.

REMARK 3.1.2. When $m \neq m'$, it was shown (Theorem 2.3 [8]) that if $\mathcal{A}$ has finitely many equivalence classes of $\mathcal{B}$ eigenpairs, then the number of equivalence classes of $\mathcal{B}$ eigenpairs, counting multiplicities, is bounded by

$$K(m, m', n) = \frac{(m-1)^n - (m'-1)^n}{m - m'}. \tag{3.2}$$

Furthermore, if $\mathcal{A}$ and $\mathcal{B}$ are generic tensors, then $\mathcal{A}$ has $K(m, m', n)$ equivalence classes of $\mathcal{B}$ eigenpairs, counting multiplicities. As a consequence, the optimal number of paths to follow for solving the system (1.12) is $K(m, m', n)$. Our starting system (2.23) or (2.39) has

exactly $K(m, m', n)$ nonsingular solutions. Therefore, Algorithm 3.1.1 follows the optimal number, making it an efficient homotopy method for computing generalized eigenpairs.

3.2 Algorithms to compute solutions of the starting system in the linear homotopy algorithm

In this section, two algorithms will be described to compute solutions of the starting system $G(\lambda, x)$ in (3.1), which is $G_1(\lambda, x)$ as defined in (2.23) when $m > m'$ and $G_2(\lambda, x)$ as defined in (2.39) when $m < m'$.

The following algorithm gives a method to compute all solutions of $G_1(\lambda, x)$.

ALGORITHM 3.2.1. (Compute all the solutions of $G_1(\lambda, x)$.)

Step 1. *For $i = 1, \ldots, n$, choose I_i to be an index set containing i elements of $\{1, \ldots, n\}$.*

Step 2. *For each I_i chosen in Step 1, let k be the smallest integer contained in I_i. For each $j \in \{1, \ldots, n\} \backslash \{k\}$, compute x_j by*

$$x_j = \begin{cases} e^{\frac{2\pi li}{m-m'}} x_k, & j \in I_i \backslash \{k\} \\ e^{\frac{2\pi pi}{m'-1}} \alpha_j, & j \in \{1, \ldots, n\} \backslash I_i \end{cases} \tag{3.3}$$

where l can be chosen from $\{0, 1, \ldots, m-m'-1\}$ and p can be chosen from $\{0, 1, \ldots, m'-2\}$.

Step 3. *For the chosen l and p from Step 2, substitute all x_j's (except x_k) given by (3.3) to $\eta^T x + \eta_0 = 0$ to compute x_k.*

Step 4. *For the chosen l and p from Step 2, substitute x_k back into (3.3), all the x_j's can be obtained, and $\lambda = x_k^{m-m'}$.*

The following algorithm is to compute all solutions of $G_2(\lambda, x)$.

ALGORITHM 3.2.2. (Compute all the solutions of $G_2(\lambda, x)$.)

Step 1. *For $i = 1, \ldots, n$, choose I_i to be an index set containing i elements of $\{1, \ldots, n\}$.*

Step 2. *For each I_i selected in Step 1, let k be the smallest integer contained in I_i. For each $j \in \{1, \ldots, n\} \backslash \{k\}$, compute x_j by*

$$x_j = \begin{cases} e^{\frac{2\pi l i}{m'-m}} x_k, & j \in I_i \backslash \{k\} \\ e^{\frac{2\pi p i}{m-1}} \alpha_j, & j \in \{1, \ldots, n\} \backslash I_i \end{cases} \tag{3.4}$$

where $l \in \{0, 1, \ldots, m' - m - 1\}$ and $p \in \{0, 1, \ldots, m - 2\}$.

Step 3. *For the chosen l and p in Step 2, substitute all x_j's (except x_k) given by (3.4) to $\eta^T x + \eta_0 = 0$ to compute x_k.*

Step 4. *For l and p selected in Step 2, substitute x_k back into (3.4), all the x_j's can be obtained. For a nonzero x_j, λ can be computed by*

$$\lambda = \frac{1}{x_j^{m'-m}}.$$

3.3 Algorithm to follow solution paths of the linear homotopy

In this section, we will propose an algorithm to follow the solution paths of the linear homotopy (3.1). Let $u := (\lambda, x)$. Then (3.1) becomes

$$H(u, t) = (1 - t)\gamma G(u) + tP(u) = 0, \quad t \in [0, 1]. \tag{3.5}$$

Denote the solution set of $G(u) = 0$ by Φ, which can be computed using Algorithm 3.2.1 or Algorithm 3.2.2.

ALGORITHM 3.3.1. (Follow solution paths of (3.5).)

Step 1. *Let* $(u_k, t_k) := (u(t_k), t_k)$. *Take* $t_0 = 0$, *and let* $u_0 \in \Phi$. *For finding the next point* (u_1, t_1) *on the solution path of*

$$H(u,t) = (1-t)\gamma G(u) + tP(u) = 0, \quad t \in [0,1]$$

in (3.5), the following steps are employed:

- *Prediction Step by Euler method: Compute the tangent vector* $\dfrac{du}{dt}$ *to a solution path* $u(t)$ *of* $H(u,t) = 0$ *at* (u_0, t_0) *by solving the linear system*

$$H_u(u_0, t_0)\frac{du}{dt} = -H_t(u_0, t_0)$$

  for $\dfrac{du}{dt}$. *Then compute the approximation* $\tilde{u}$ *to* u_1 *by*

$$\tilde{u} = u_0 + \Delta t \frac{du}{dt}, \quad t_1 = t_0 + \Delta t,$$

  where Δt *is the stepsize.*

- *Correction Step: Use Newton's iterations, i.e., for* $i = 0, 1, 2, \ldots,$ *compute*

$$v_{i+1} = v_i - [H_u(v_i, t_1)]^{-1} H(v_i, t_1) \quad \textit{with} \quad v_0 = \tilde{u}$$

  until $\|H(v_J, t_1)\|$ *is very small. Then let* $u_1 = v_J$. *When the iteration fails to converge,*

the Prediction Step will be repeat with $\Delta t = \frac{\Delta t}{2}$.

Step 2. *Path following: Follow the paths from* $t = t_1$ *to* $t = 1$ *using the prediction-correction strategy. Given* (u_k, t_k)*, to find the next point* (u_{k+1}, t_{k+1}) *on the solution path of* $H(u,t) = 0$ *as in (3.5), the following steps are employed:*

- *Prediction Step by the cubic Hermite interpolation: Compute the tangent vector* $\dfrac{du}{dt}$ *to a solution path* $u(t)$ *of* $H(u,t) = 0$ *at* (u_{k-1}, t_{k-1}) *and* (u_k, t_k) *by solving the linear system*

$$H_u(u_{k-1}, t_{k-1}) \left.\frac{du}{dt}\right|_{t=t_{k-1}} = -H_t(u_{k-1}, t_{k-1})$$

for $\left.\dfrac{du}{dt}\right|_{t=t_{k-1}}$ *and*

$$H_u(u_k, t_k) \left.\frac{du}{dt}\right|_{t=t_k} = -H_t(u_k, t_k)$$

for $\left.\dfrac{du}{dt}\right|_{t=t_k}$*. Let* $\tilde{u}(t)$ *be the cubic polynomial which interpolates* $u(t)$ *and* $u'(t)$ *at* $t = t_{k-1}$ *and* $t = t_k$*. Namely,*

$$\tilde{u}(t_{k-1}) = u_{k-1}, \quad \tilde{u}(t_k) = u_k$$

and

$$\tilde{u}'(t_{k-1}) = \left.\frac{du}{dt}\right|_{t=t_{k-1}}, \quad \tilde{u}'(t_k) = \left.\frac{du}{dt}\right|_{t=t_k}.$$

Then $\tilde{u}(t_{k+1})$ *can be taken as the prediction of* $u(t)$ *at* t_{k+1}*, i.e., an approximation to* u_{k+1}*. Here,* $t_{k+1} = t_k + \Delta t$ *with* Δt *being the stepsize.*

- *Correction Step: Use Newton's iterations, i.e., for* $i = 0, 1, 2, \ldots,$ *compute*

$$v_{i+1} = v_i - [H_u(v_i, t_{k+1})]^{-1} H(v_i, t_{k+1}) \quad \textit{with} \quad v_0 = \tilde{u}(t_{k+1})$$

until $\|H(v_J, t_{k+1})\|$ *is very small. Then let* $u_{k+1} = v_J$. *When the iteration fails to converge, the Prediction Step will be repeat with* $\Delta t = \frac{\Delta t}{2}$.

Step 3. *End game. When* t_N *is very close to 1, the corresponding* u_N *should be very close to a zero* u^* *of* $P(u) = P(\lambda, x)$. *So Newton's iterations*

$$u^{(k+1)} = u^{(k)} - [DP(u^{(k)})]^{-1} P(u^{(k)}), u^{(0)} = u_N, \quad k = 0, 1, \ldots$$

will be used again to refine our final approximation $\tilde{u}$ *to* u^*. *If* $DP(u^*)$ *is nonsingular, then* $\tilde{u}$ *will be a very good approximation of* u^* *with multiplicity 1. If* $DP(u^*)$ *is singular,* $\tilde{u}$ *is either an isolated singular zero of* $P(u)$ *with multiplicity* $l > 1$ *or in a positive dimensional solution component of* $P(u) = 0$. *We use a strategy suggested in Chapter VIII of [13] (see also [29]) to determine whether* $\tilde{u}$ *is an isolated zero with multiplicity bigger than* 1 *or in a positive dimensional solution component of* $P(u) = 0$.

By using random complex numbers in the formulation of homotopy, with probability one the solution paths do not intersect with each other or go to infinity for $0 < t < 1$ theoretically. Practically, however, two solution paths may become very close to each other and the magnitude of some components of a solution path may become very large during the procedure of path tracking. This causes various numerical difficulties such as missing solutions, losing efficiency or stability. In our implementation of Algorithms 3.3.1, we use the following strategies to address these issues.

When tracing two paths that are sufficiently close, it is possible for the path tracing algorithm to jump from one path to the other and thus result in missing of zeros. To minimize the chance for curve jumping and keep the efficiency, our *First Strategy* is: *The stepsize* Δt *in Step 1 and Step 2 of Algorithm 3.3.1 is chosen adaptively.* Initially, $\Delta t = 0.1/(n + 1)$,

where $n+1$ is the number of unknown variables $\lambda, x_1, \ldots, x_n$ in (1.12) or (1.9). Similar to [15], if more than three steps of Newton iterations were required to converge within the desired accuracy, Δt is halved. On the other hand, if several consecutive steps (the default being 2) were not cut, Δt is doubled, up to a prescribed maximum value (the default being $0.1/(n+1)$).

Although this adaptive approach can often reduce the possibility of curve jumping significantly, curve jumping can still occur in some occasions. Our *Second Strategy* is: *To check if there exist curve jumpings, all found solutions are stored in a binary search tree. Each time when a new solution is found, we can quickly find (with time complexity $O(\log N)$, where N is the number of solutions) whether there is any existing solution that is numerically identical to the new solution, that is, the difference between them is less than a threshold (the default being 10^{-6}). If two numerically identical solutions are detected and the condition numbers of their Jacobian matrices are less than a threshold (the default being 10^{10}), the curve jumping has likely occurred. We then retrace the two associated curves with more restricted parameters.*

When the magnitude of some components of certain solution curve become very large at $t_0 \in (0,1)$, tracing these paths may fail due to numerical instability. This issue can be largely resolved by following paths in the projective space (see, for example, [29]). However, empirically it is more time consuming to follow all paths in the projective space. In our implementation of Algorithms 3.3.1, our *Third Strategy* is: *To retrace solution curves in the projective space only for those paths that are detected to have very large solution components.*

For example, when $m > m'$, to trace a path in the projective space, we first homogenize

each polynomial equation of (2.26) in the variables $\lambda, x_1, \ldots, x_n$, namely

$$\begin{aligned}\hat{H}(\lambda,\hat{x},t) &= (1-t)\gamma \begin{pmatrix} (x_1^{m'-1} - \alpha_1 x_0^{m'-1})(x_1^{m-m'} - \lambda x_0^{m-m'-1}) \\ \vdots \\ (x_n^{m'-1} - \alpha_n x_0^{m'-1})(x_n^{m-m'} - \lambda x_0^{m-m'-1}) \\ \eta^T x + \eta_0 x_0 \end{pmatrix} \\ &\quad + t \begin{pmatrix} (\mathcal{A}^{(k)} x^{m-1})_1 - \lambda x_0^{m-m'-1} (\mathcal{B} x^{m'-1})_1 \\ \vdots \\ x_0 (\mathcal{A}^{(k)} x^{m-1})_n - \lambda x_0^{m-m'-1} (\mathcal{B} x^{m'-1})_n \\ a_1 x_1 + a_2 x_2 + \cdots + a_n x_n + b x_0 \end{pmatrix} = 0, \end{aligned} \tag{3.6}$$

where $\hat{x} = (x_0, x_1, \ldots, x_n)^T$. Then follow the solution curve of (3.6) in the projective space. Notice that in (3.6) if $(\lambda, \hat{x})$ is a solution, so is $(\alpha\lambda, \alpha\hat{x})$ for $\alpha \in \mathbb{C}\backslash\{0\}$. Thus along the path we can always scale $(\lambda, \hat{x})$ to keep each component's magnitude in a suitable finite range.

To the best of our knowledge, Strategies 2 and 3 have not been used in other implementations of homotopy methods, although some packages may trace all curves in the projective space.

3.4 Evaluating polynomials and derivatives

The prediction-correction process for following the homotopy paths of

$$H(\lambda, x, t) = (1-t)\gamma G(\lambda, x) + tP(\lambda, x) = 0$$

requires the computation of $H(\lambda, x, t)$, $H_t(\lambda, x, t)$, and the Jacobian matrix $DH(\lambda, x, t) = [H_\lambda(\lambda, x, t), H_x(\lambda, x, t)]$ for fixed $t \in [0, 1]$. What is essential in those evaluations is the evaluation of multivariate polynomials and their partial derivatives. In HOM4PS [15], a multivariate polynomial $g(x_1, \cdots, x_n)$ was evaluated via Horner's rule for univariate polynomials. The basic idea is to single out a variable, say x_1, and consider $g(x_1, \cdots, x_n)$ as a polynomial in x_1 with coefficients in $x_2, \cdots, x_n$. By the same approach, those coefficients, as polynomials in one less variable, were evaluated by singling out another variable. This may continue until the variables are exhausted.

In a single variable case, Horner's rule has been proved to be optimal [19, 20], i.e., any other algorithms to evaluate a polynomial must use at least as many operations (additions and multiplications) as Horner's method. However, in multivariate cases, it is not guaranteed. As pointed out in [15], the same powers of some variables will be computed repeatedly in this manner. To improve the efficiency, in HOM4PS 2.0, a table T of size $n \times M$ is precomputed to store all possible powers of $x_i, i = 1, \cdots, n$ where M is the maximum power of all the variables in the entire polynomial system. For example, for the following system [15]:

$$P(x_1, x_2, x_3) = \begin{pmatrix} 2x_1^6 + 3x_2^4 + 5x_3^5 - 1 \\ 3x_1^6x_2^4 + 2x_1^6x_3^5 + 4x_2^4x_3^5 - 5 \\ 5x_1^6x_2^4x_3^5 - 7 \end{pmatrix}, \tag{3.7}$$

the maximum degree of x_1 is 6, x_2 is 4, and x_3 is 5, so $M = 6$. We establish Table T in Table 3.1. With this table $T(i, j)$, the value of a monomial can be easily obtained. For instance, the quantity of $x_1^6x_2^4x_3^5$ is $T(1, 6) * T(2, 4) * T(3, 5)$. Since this method mainly bases on Table T, we call it *Table-T method*.

A big advantage of the Table-T method is that those powers of each variable involved

x_1	x_1^2	x_1^3	x_1^4	x_1^5	x_1^6
x_2	x_2^2	x_2^3	x_2^4		
x_3	x_3^2	x_3^3	x_3^4	x_3^5	

Table 3.1: Table T

in any monomial evaluations will only need to be computed once, no matter how often they appear. However, some shortcomings exist:

1. There are some redundant computations in Table 3.1. For example, if only the value of x_1^6 is required, the value of x_1^5 and x_1^4 are not needed since $x_1^6 = x_1^3 * x_1^3$, $x_1^3 = x_1^2 * x_1$, and $x_1^2 = x_1 * x_1$. A little more extreme example is that if only x^{100} is in demand, instead of computing from x^2, x^3 until x^{100}, one may compute it in the following way:

$$\begin{aligned} x^{100} &= x^{50} * x^{50}, \\ x^{50} &= x^{25} * x^{25}, \\ x^{25} &= x^{13} * x^{12}, \\ x^{13} &= x^{12} * x, \\ x^{12} &= x^6 * x^6, \\ x^6 &= x^3 * x^3, \\ x^3 &= x^2 * x, \\ x^2 &= x * x, \end{aligned}$$

where only 8 middle values need to be computed in comparison to computing 99 middle values previously.

2. The method can not take advantage of computing some higher degree monomial by the product of two lower degree monomials. For instance, in the polynomial system (3.7), the value of $x_1^6x_2^4x_3^5$ may set to be $x_1^6x_2^4 * x_3^5$ where values of $x_1^6x_2^4$ and x_3^5 are already computed in the first two polynomials. Comparing to the old way, i.e., $x_1^6x_2^4x_3^5 = T(1,6) * T(2,4) * T(3,5)$, one less multiplication is required.

3. If the same monomial appears in different polynomials or in its Jacobian matrix, the repeated computation of this monomial is inevitable in the above approach. For example, for the system

$$P(x_1, x_2, x_3) = (p_1(x_1, x_2, x_3), p_2(x_1, x_2, x_3), p_3(x_1, x_2, x_3)),$$

where

$$\begin{aligned} p_1 &= x_1^6x_3^5 + 3x_1^5x_2^4 + 5x_3^5 - 1 \\ p_2 &= 3x_1^6x_2^4 + 2x_1^6x_3^5 + 4x_2^4x_3^5 - 5 \\ p_3 &= 5x_1^6x_2^4x_3^5 - 7, \end{aligned} \tag{3.8}$$

the monomial $x_1^6x_3^5$ appears in both p_1 and p_2. And the monomial $x_1^5x_2^4$ appears in both p_1 and $\frac{\partial p_2}{\partial x_1} = 18x_1^5x_2^4 + 12x_1^5x_3^5$. In the above approach, those quantities will be computed repeatedly.

To resolve those issues, the following new method has been developed.

Step 1. Collect all the monomials from the given polynomials and their derivatives, and divide them into different groups according to their degrees. Since the coefficients of the monomials are not fixed in following the homotopy paths, only variable parts of the

monomials are selected, their coefficients will be calculated separately. For the convenience of future computations, each group is organized as a linked list with each node being a monomial and the nodes are sorted alphabetically and also by the powers of each variable. For duplicated monomials, they only need to appear once. This accounts for the problem stated in 3 above. For example, the Jacobian matrix of (3.8) is

$$\begin{pmatrix} 6x_1^5x_3^5 + 15x_1^4x_2^4 & 12x_1^5x_2^3 & 5x_1^6x_3^4 + 25x_3^4 \\ 18x_1^5x_2^4 + 12x_1^5x_3^5 & 12x_1^6x_2^3 + 16x_2^3x_3^5 & 10x_1^6x_3^4 + 20x_2^4x_3^4 \\ 30x_1^5x_2^4x_3^5 & 20x_1^6x_2^3x_3^5 & 25x_1^6x_2^4x_3^4 \end{pmatrix}. \tag{3.9}$$

Collecting all the monomials from (3.8) and (3.9) and putting them into different linked lists based on their degrees yields following structure:

$$\begin{aligned}
\boxed{1} &\rightarrow x_1 \rightarrow x_2 \rightarrow x_3 \\
\boxed{4} &\rightarrow x_3^4 \\
\boxed{5} &\rightarrow x_3^5 \\
\boxed{8} &\rightarrow x_1^5x_2^3 \rightarrow x_1^4x_2^4 \rightarrow x_2^4x_3^4 \rightarrow x_2^3x_3^5 \\
\boxed{9} &\rightarrow x_1^6x_2^3 \rightarrow x_1^5x_2^4 \rightarrow x_2^4x_3^5 \\
\boxed{10} &\rightarrow x_1^6x_2^4 \rightarrow x_1^6x_3^4 \rightarrow x_1^5x_3^5 \\
\boxed{11} &\rightarrow x_1^6x_3^5 \\
\boxed{14} &\rightarrow x_1^6x_2^4x_3^4 \rightarrow x_1^6x_2^3x_3^5 \rightarrow x_1^5x_2^4x_3^5 \\
\boxed{15} &\rightarrow x_1^6x_2^4x_3^5
\end{aligned} \tag{3.10}$$

Here x_1, x_2, and x_3 are also in the structure because they are the initial values needed to

evaluate the monomials.

Step 2. Starting from the monomials with the highest degree to degree 2, for each monomial x^α, search all the nodes in the linked lists to find two lower degree monomials x^β and x^γ such that their product is equal to x^α, i.e. $\alpha = \beta + \gamma$. Without loss of generality, let us assume that $|\beta| \geq |\gamma|$. Since $\left\lfloor \frac{|\alpha|}{2} \right\rfloor \leq |\beta| < |\alpha|$, we only need to search the monomials with degree between $\left\lfloor \frac{|\alpha|}{2} \right\rfloor$ and $|\alpha| - 1$ in order to find a possible monomial x^β with $\beta \leq \alpha$. For $\alpha := (\alpha_1, \ldots, \alpha_n)$ and $\beta := (\beta_1, \ldots, \beta_n)$, by $\beta \leq \alpha$ we mean $\beta_i \leq \alpha_i$ for $i = 1, \ldots, n$. There are three cases:

Case 1: If there exist two monomials x^β and x^γ in the linked lists such that $x^\alpha = x^\beta * x^\gamma$. Then label the relation in the structure. For example, as we can see from (3.10), $x_1^6 x_2^4 x_3^5$ is the product of $x_1^6 x_2^4 x_3^4$ and x_3, $x_1^6 x_2^4 x_3^4$ is the product of $x_1^6 x_2^4$ and x_3^4, then their relations will be marked in the structure as shown in the linked lists of degree 14 and 15 in Figure 3.1;

Case 2: If x^β is in the linked lists but $x^\gamma = x^{\alpha-\beta}$ is not in the lists, then add a new monomial x^γ into the linked lists and then label the relation. If there are several choices for x^β, choose the one with the largest degree. Take (3.10) for instance, x_3^4 is a factor of $x_2^4 x_3^4$ but x_2^4 is not in the linked lists. In this case, add x_2^4 into the linked list with degree 4;

Case 3: There is no monomial x^β satisfying the conditions. Let $\beta = \lceil \alpha/2 \rceil$ and $\gamma = \alpha - \beta$, add x^β and x^γ into the linked lists and label the relation. If $\beta = \gamma$, the monomial only needs to be added once. For example, there is no monomial with degree between 5 and 9 which is a factor of $x_1^6 x_3^4$ in (3.10). Let $\beta = \lceil \alpha/2 \rceil = \lceil [6\,0\,4]/2 \rceil = [3\,0\,2]$, then $\gamma = \alpha - \beta$ is equal to β, we only need to add the monomial $x_1^3 x_3^2$ to the linked lists.

Similarly, in order to evaluate $x_1^5 x_2^3$, two lower degree monomials $x_1^3 x_2^2$ and $x_1^2 x_2^1$ are added into the linked lists.

Applying the above rules to (3.10), results in Figure 3.1.

Step 3. Evaluate monomials from the lowest degree to the highest degree in the linked lists based on the evaluation diagram obtained in Step 2 (See, for example, Figure 3.1). Since our linked lists is built according to the criteria that every monomial of higher degree can be evaluated by computing the product of two monomials of lower degrees in Step 2, the values of all the monomials can be evaluated in this step.

Step 4. Compute the values of the polynomials and their derivatives by using the monomials' values. For this goal, there are two approaches.

On-line evaluation: we maintain a mapping between the monomials in the polynomials and their derivatives with the ones in the linked lists in the memory. To get the value of a polynomial, we replace the monomial with the values in the linked lists first, then sum up the values to obtain the polynomial's value;

Off-line evaluation: we create a new function which contains all the evaluation rules in the hard disk. So if we want to evaluate a polynomial system and its Jacobian matrix, all we need to do is to call that new function with the value of x and coefficients as inputs. See Figure 3.2 for an example of the off-line evaluation function.

The reason why we need two different approaches is that normally off-line evaluation is much more efficient than on-line evaluation. However, some languages, like C++ and Fortran, do not support off-line evaluation, i.e. one cannot dynamically create a new function and require the program to call it. For these languages, in order to use the new created function stored

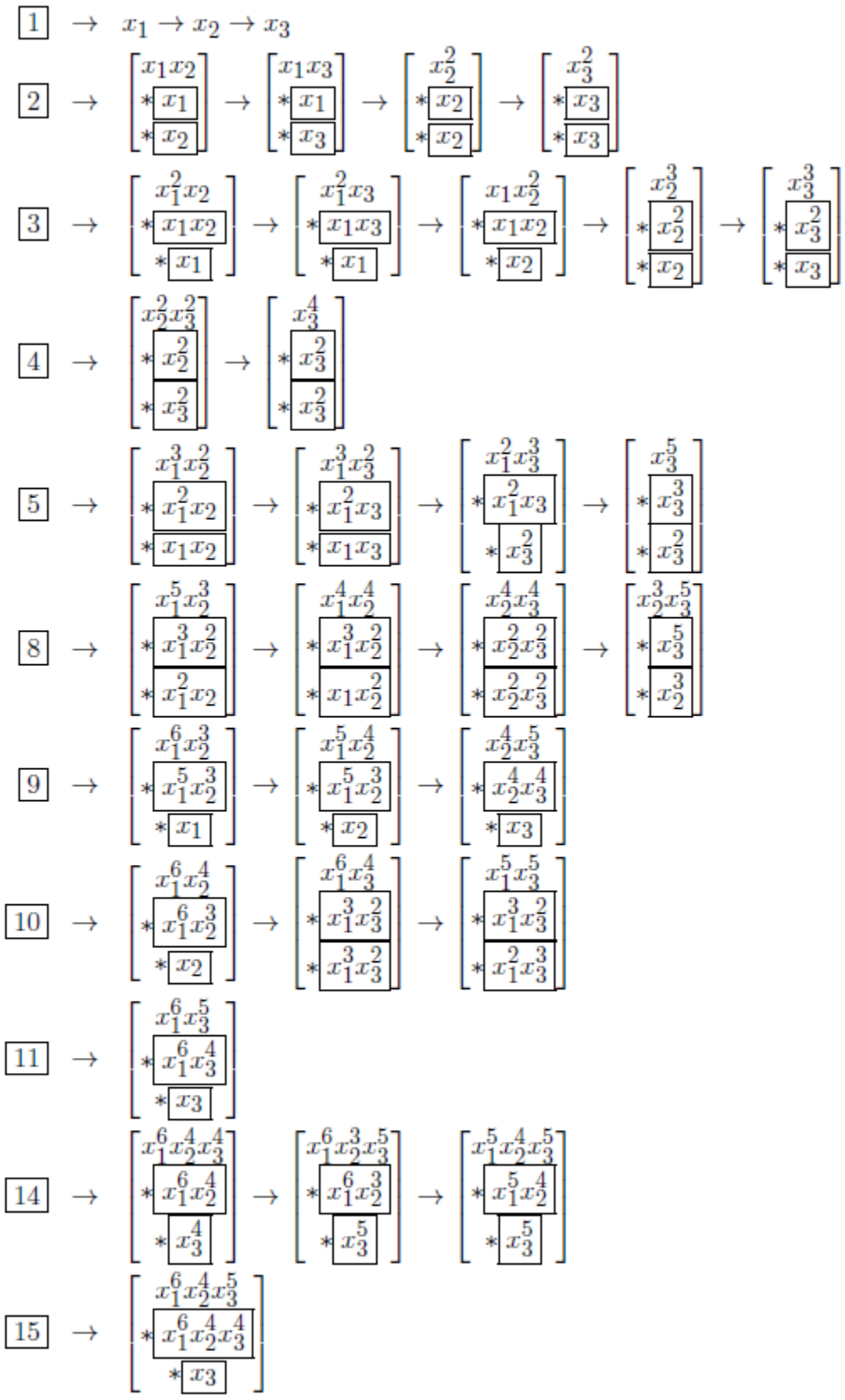

Figure 3.1: An example of the evaluation graph

Editor - D:\Dropbox\HOM4PS\psolve\tmpPDPEval.m

EDITOR | PUBLISH | VIEW

findFastEval.m | psolve.m | tmpPDPEval.m | tmpPsolveEval.m*

```
1   function [jacob, polyv] = tmpPDPEval(pCoef, x)
2
3   m = zeros(29, 1);
4   m(1) = x(1)*x(1);
5   m(2) = x(2)*x(2);
6   m(3) = x(2)*x(3);
7   m(4) = x(3)*x(3);
8   m(5) = m(1)*x(1);
9   m(6) = m(2)*x(1);
10  m(7) = m(1)*m(1);
11  m(8) = m(6)*x(2);
12  m(9) = m(2)*m(2);
13  m(10) = m(2)*m(3);
14  m(11) = m(4)*m(4);
15  m(12) = m(1)*m(5);
16  m(13) = m(11)*x(3);
17  m(14) = m(7)*m(1);
18  m(15) = m(8)*m(7);
19  m(16) = m(7)*m(9);
20  m(17) = m(9)*m(11);
21  m(18) = m(11)*m(10);
22  m(19) = m(12)*m(8);
23  m(20) = m(12)*m(9);
24  m(21) = m(13)*m(9);
25  m(22) = m(14)*m(9);
26  m(23) = m(14)*m(11);
27  m(24) = m(13)*m(12);
28  m(25) = m(13)*m(14);
29  m(26) = m(22)*m(11);
30  m(27) = m(19)*m(13);
31  m(28) = m(20)*m(13);
32  m(29) = m(22)*m(13);
33  polyv = zeros(3,1);
34  polyv(1) = pCoef(1)+pCoef(2)*m(20)+pCoef(3)*m(13)+pCoef(4)*m(25);
35  polyv(2) = pCoef(5)+pCoef(6)*m(22)+pCoef(7)*m(25)+pCoef(8)*m(21);
36  polyv(3) = pCoef(9)+pCoef(10)*m(29);
37  jacob = zeros(3);
38  jacob(1,1) = pCoef(2)*5*m(16) + pCoef(4)*6*m(24);
39  jacob(1,2) = pCoef(2)*4*m(15);
40  jacob(1,3) = pCoef(3)*5*m(11) + pCoef(4)*5*m(23);
41  jacob(2,1) = pCoef(6)*6*m(20) + pCoef(7)*6*m(24);
42  jacob(2,2) = pCoef(6)*4*m(19) + pCoef(8)*4*m(18);
43  jacob(2,3) = pCoef(7)*5*m(23) + pCoef(8)*5*m(17);
44  jacob(3,1) = pCoef(10)*6*m(28);
45  jacob(3,2) = pCoef(10)*4*m(27);
46  jacob(3,3) = pCoef(10)*5*m(26);
47
48  end
```

Figure 3.2: An example of the off-line evaluation function

in a separate file, the file should be complied before we can use it. In this case, one prefers an on-line evaluation.

Chapter 4

Numerical results

In this section, we present some numerical results to show the effectiveness and efficiency of Algorithm 3.1.1. The numerical experiments were carried out using MATLAB 2013a, on a Thinkpad T400 laptop computer with an Intel(R) dual core CPU at 2.80GHz and 2GB of RAM, running the Windows 7 operating system.

4.1 The efficiency of the new evaluation method

As discussed in Section 3.4, the most time-consuming procedure in Algorithm 3.1.1 is in fact evaluation of polynomials and their derivatives and new approaches have been proposed to execute these evaluations. In this section, we compare the efficiency of our new approaches with the Table-T method used in HOM4PS 2.0.

EXAMPLE 4.1.1. In this example, we intend to compare the efficiency of the on-line and off-line approaches described in Section 3.4 with the Table-T method. All these three approaches are used in Step 2 of Algorithm 3.1.1 compute $\mathcal{B}$-eigenpairs of a tensor $\mathcal{A}$, where $\mathcal{A} \in \mathbb{C}^{[m,n]}, \mathcal{B} \in \mathbb{C}^{[m',n]}$ are generic tensors with $m \neq m'$. Each tensor was generated using $randn(n, \cdots, n) + i * randn(n, \cdots, n)$ in MATLAB. The numerical results are reported in Table 4.1.

As it shows, the on-line and off-line approaches are considerably faster than Table-T method, and the speed-up ratio increases as the number of eigenpairs becomes bigger. The

(m, m', n)	Number of eigenpairs	Table-T method time(s)	On-line approach		Off-line approach		Speed-up ratio of off/on-line
			CPU time(s)	Speed-up ratio	CPU time(s)	Speed-up ratio	
$(3, 2, 5)$	31	38.7	2.3	16.8	1.2	32.3	1.9
$(4, 2, 5)$	121	605.0	21.8	27.8	12.3	49.2	1.8
$(4, 3, 5)$	211	735.1	24.4	30.1	10.1	72.8	2.4
$(5, 6, 4)$	369	1746.0	64.6	27.0	22.5	77.6	2.9
$(5, 3, 5)$	496	3215.2	95.1	33.8	37.0	86.9	2.6

Table 4.1: Comparison of on-line and off-line approaches with Table-T method

off-line approach is about 2-3 times faster than the on-line approach as tested in MATLAB.

4.2 Computing complex tensor eigenpairs

We compare the performance of Algorithm 3.1.1 with that of Algorithm 3.2 proposed in [8]. Algorithm 3.2 uses polyhedral homotopy methods. It has been implemented as a function `teneig` in the tensor eigenvalue package `TenEig1.1` ([8]). The function `teneig` is based on a sophisticated polyhedral homotopy polynomial system solver PSOLVE ([30]). Numerical results in [8] show that `teneig` is significantly more efficient than the popular NSolve in Mathematica for computing generalized eigenpairs defined in (1.9).

EXAMPLE 4.2.1. We intend to find all isolated $\mathcal{B}$-eigenpairs of a tensor $\mathcal{A}$, where $\mathcal{A} \in \mathbb{C}^{[m,n]}, \mathcal{B} \in \mathbb{C}^{[m',n]}$ are generic tensors with $m \neq m'$. Each tensor was generated using $randn(n, \cdots, n) + i * randn(n, \cdots, n)$ in MATLAB.

The numerical results for $m > m'$ and $m < m'$ are reported in Table 4.2 and Table 4.3 respectively. In these tables, $K(m, m', n)$ (defined in (3.2)) represents the theoretical bound of the number of equivalence classes of isolated $\mathcal{B}$-eigenpairs of $\mathcal{A}$ when $m \neq m'$. For generic tensors $\mathcal{A}$ and $\mathcal{B}$, this is the exact number of equivalence classes of $\mathcal{B}$-eigenpairs of $\mathcal{A}$. Note that by Remark 2.1 in [8] and Algorithm 3.1.1, the total number of eigenpairs is

$m'K(m,m',n)$. Let N denote the number of equivalence classes of $\mathcal{B}$-eigenpairs found by `Algorithm 3.1.1` or `teneig` in the tables.

(m,m',n)	$K(m,m',n)$	Method	N	CPU time (s)
$(3,2,7)$	127	`Algorithm 3.1.1`	127	5.9
		`teneig`	127	10.1
$(4,2,6)$	364	`Algorithm 3.1.1`	364	24.2
		`teneig`	364	29.3
$(4,3,5)$	211	`Algorithm 3.1.1`	211	10.1
		`teneig`	211	13.1
$(5,4,5)$	781	`Algorithm 3.1.1`	781	68.7
		`teneig`	781	82.6
$(6,5,4)$	369	`Algorithm 3.1.1`	369	27.7
		`teneig`	369	28.1
$(7,6,4)$	671	`Algorithm 3.1.1`	671	54.4
		`teneig`	671	74.0

Table 4.2: Comparison of `Algorithm 3.1.1` with `teneig` for $m > m'$

(m,m',n)	$K(m,m',n)$	Method	N	CPU time (s)
$(3,4,6)$	665	`Algorithm 3.1.1`	665	42.5
		`teneig`	665	62.7
$(3,5,5)$	496	`Algorithm 3.1.1`	496	37.2
		`teneig`	496	47.4
$(4,5,4)$	175	`Algorithm 3.1.1`	175	8.8
		`teneig`	175	9.6
$(4,5,5)$	781	`Algorithm 3.1.1`	781	72.9
		`teneig`	781	104.3
$(5,6,4)$	369	`Algorithm 3.1.1`	369	22.5
		`teneig`	369	31.8
$(7,8,3)$	127	`Algorithm 3.1.1`	127	8.4
		`teneig`	127	9.7

Table 4.3: Comparison of `Algorithm 3.1.1` with `teneig` for $m < m'$

EXAMPLE 4.2.2. Consider finding all E-eigenpairs of a generic tensor $\mathcal{A}$. Tensor $\mathcal{A} \in \mathbb{C}^{[m,n]}$ was generated using $randn(n,\cdots,n)+i*randn(n,\cdots,n)$ in MATLAB. In this case, tensor $\mathcal{B}$ is the $n \times n$ identity matrix.

The numerical results are reported in Table 4.4. In this table, $K(m, 2, n)$ (defined in (3.2)) represents the bound of the number of equivalence classes of isolated E-eigenpairs of $\mathcal{A}$. Since tensor $\mathcal{A}$ is generic, it has exactly $K(m, 2, n)$ equivalence classes of E-eigenpairs (see, [5]). N denotes the number of equivalence classes of E-eigenpairs found by `Algorithm 3.1.1` or `teneig`.

(m, n)	$K(m, 2, n)$	Method	N	CPU time (s)
$(3, 9)$	511	`Algorithm 3.1.1`	511	32.4
		`teneig`	511	47.3
$(3, 10)$	1023	`Algorithm 3.1.1`	1023	73.5
		`teneig`	1023	130.5
$(4, 7)$	1093	`Algorithm 3.1.1`	1093	69.8
		`teneig`	1093	117.5
$(4, 8)$	3280	`Algorithm 3.1.1`	3280	380.6
		`teneig`	3280	565.1
$(5, 5)$	341	`Algorithm 3.1.1`	341	18.8
		`teneig`	341	21.6
$(5, 6)$	1365	`Algorithm 3.1.1`	1365	117.8
		`teneig`	1365	167.9
$(6, 5)$	781	`Algorithm 3.1.1`	781	69.7
		`teneig`	781	99.3
$(6, 6)$	3906	`Algorithm 3.1.1`	3906	622.3
		`teneig`	3906	975.9
$(7, 4)$	259	`Algorithm 3.1.1`	259	15.2
		`teneig`	259	20.8
$(7, 5)$	1555	`Algorithm 3.1.1`	1555	211.3
		`teneig`	1555	230.8

Table 4.4: Comparison of `Algorithm 3.1.1` with `teneig` for computing E-eigenpairs

From Tables 1–3, while both Algorithm 3.1.1 and `teneig` find all $\mathcal{B}$-eigenpairs of a generic tensor $\mathcal{A}$, Algorithm 3.1.1 is more efficient in terms of CPU time. We believe the efficiency of Algorithm 3.1.1 is achieved because of its employment of the linear homotopy (3.1) with simple starting system (2.23) or (2.39). The algorithm `teneig`, on the other hand, uses a polyhedral homotopy in which mixed volume computation is required. The efficiency along

with easy implementation makes Algorithm 3.1.1 a competitive method for computing tensor eigenpairs when $m \neq m'$.

4.3 Computing real tensor eigenpairs

Sometimes, only real eigenpairs are of interest. As suggested in [8], one can compute all the complex eigenpairs first and use different approaches to extract real eigenpairs from complex eigenpairs. Here we use Algorithm 3.1.1 to compute complex eigenpairs and follow the same procedure indicated in Algorithm 4.1 in [8] to acquire real eigenpairs from these complex eigenpairs.

For computing all real eigenvalues of a symmetric tensor the only available methods at this time are Algorithm 3.6 in [9] and Algorithm 4.1 in [8]. In the following example, we compare the performance of our methods with theirs.

EXAMPLE 4.3.1. Consider the symmetric tensor $\mathcal{A} \in \mathbb{R}^{[4,n]}$ (Example 4.16 in [9]) in the polynomial form

$$\begin{aligned} \mathcal{A}x^4 &= (x_1 - x_2)^4 + \cdots + (x_1 - x_n)^4 + (x_2 - x_3)^4 + \cdots + (x_2 - x_n)^4 \\ &\quad + \cdots + (x_{n-1} - x_n)^4. \end{aligned}$$

Shown in Table 4.5, our new algorithm obtained all the Z-eigenvalues found by Algorithm 3.6 in [9] and Algorithm 4.1 in [8] for different n. Remarkably, when $n = 8, 9, 10$, our new algorithm and Algorithm 4.1 in [8] can find all the Z-eigenvalues in a reasonable amount of time, but [9] reports that Algorithm 3.6 can only find the first three largest Z-eigenvalues. The CPU times used by these three algorithms are listed in the table. (The CPU times

by Algorithm 3.6 ([9]) are from [9][1].) The corresponding Z-eigenvectors are not displayed. Apparently, as it shows, our new algorithm leads to a large speed-up, ranging up to 239s, over Algorithm 3.6 in [9] on all the systems, especially for large ones. Compared to Algorithm 4.1 in [8], the new algorithm is about 1.4 times faster.

n	λ	CPU time(s)		
		Alg 3.6 ([9])	Alg 4.1 ([8])	New Alg
4	0.0000 4.0000 5.0000 5.3333	3.6	1.7	1.1
5	0.0000, 4.1667, 4.2500, 5.5000, 6.2500	274.5	5.4	4.0
6	0.0000, 4.0000, 4.5000, 6.0000, 7.2000	280.2	15.5	11.8
7	0.0000, 4.0833, 4.1667, 4.7500, 4.8846, 4.9000, 6.5000, 8.1667	9565.6	58.3	40.1
8	0.0000, 4.0000, 4.2667, 4.2727, 4.3333, 5.0000, 5.2609, 5.3333, 7.0000, 9.1429	938.2*	244.1	165.3
9	0.0000, 4.0500, 4.1250, 4.5000, 5.2500, 5.6250, 5.7857, 7.5000, 10.1250	4173.8*	788.0	544.0
10	0.0000, 4.0000, 4.1667, 4.1818, 4.2500, 4.6667, 4.7500, 4.7593, 4.7619, 5.5000, 5.9808, 6.2500, 8.0000, 11.1111	15310.5*	2665.6	1893.6

Table 4.5: Z-eigenvalues of the tensor in Example 4.3.1 (* denotes that the CPU time used by Algorithm 3.6 ([9]) when it finds the first three largest Z-eigenvalues)

[1]It should be cautious when comparing the CPU times used by the two methods because of different computers were used.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] D.L. Bates, J.D. Hauenstein, A.J. Sommese and C.W. Wampler, *Numerically Solving Polynomial Systems with Bertini*, Society for Industrial and Applied Mathematics, Philadelphia, 2013.

[2] D. N. Bernstein, The number of the roots of a system of equations, *Funct. Anal. Appl.*, 1975, 9:183-185.

[3] J. Cooper and A. Duttle, Spectra of uniform hypergraphs, *Linear Algebra and its Applications*, 2012, 436: 3268–3292.

[4] D. A. Cox, J. Little, and D. O'Shea, *Using Algebraic Geometry*, 2nd ed., Springer-Verlag, New York, NY, 2005.

[5] D. Cartwright and B. Sturmfels, The number of eigenvalues of a tensor, *Linear Algebra and its Applications*, 2013, 438: 942–952.

[6] K.C. Chang, K. Pearson and T. Zhang, On eigenvalues of real symmetric tensors, *Journal of Mathematical Analysis and Applications*, 2009, 350: 416–422.

[7] K.C. Chang, L. Qi, and T. Zhang, A survey of the spectral theory of nonnegative tensors, *Numerical Linear Algebra with Applications*, 2013, 20: 891–912.

[8] L. Chen, L. Han and L. Zhou, Computing tensor eigenvalues via homotopy methods, *SIAM Journal on Matrix Analysis and Applications*, 2016, 37: 290–319.

[9] C. Cui, Y.-H. Dai and J. Nie, All real eigenvalues of symmetric tensors, *SIAM Journal on Matrix Analysis and Applications*, 2014, 35: 1582–1601.

[10] S. Hu, Li. Qi, and B. Zhang, The geometric measure of entanglement of pure states with nonnegative amplitudes and the spectral theory of nonnegative tensors, arXiv:1203.3675, 2012.

[11] T. Kolda and B. Bader, Tensor decompositions and applications, *SIAM Review*, 2009, 51(3): 455-500.

[12] T.G. Kolda and J.R. Mayo, Shifted power method for computing tensor eigenpairs, *SIAM Journal on Matrix Analysis and Applications*, 2011, 32: 1095-1124.

[13] T.Y. Li, Solving polynomial systems by the homotopy continuation method, *Handbook of Numerical Analysis*, XI, 2003, 209–304.

[14] L.-H. Lim, Singular values and eigenvalues of tensors: a variational approach, *Proceedings of the IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP'05)*, 2005, 1: 129–132.

[15] T.L. Lee, T.Y. Li, and C.H. Tsai, Hom4PS-2.0, a software package for solving polynomial systems by the polyhedral homotopy continuation method, *Computing*, 2008, 83:109–133.

[16] T.Y. Li and X. Wang, The BKK root count in $\mathbb{C}^n$, *Math. Comp.*, 1996, 65: 1477-1484.

[17] M. Narasimhan, *Principles of Continuum Mechanics*, John Wiley & Sons, New York, 1993.

[18] Q. Ni, L. Qi and F. Wang, An eigenvalue method for testing positive definiteness of a multivariate form, *Automatic Control, IEEE Transactions on*, 2008, 53: 1096-1197.

[19] A. M. Ostrowski, On two problems in abstract algebra connected with Horner's rule, *Studies in Math. Mech.*, 1954, 40-48.

[20] Y. Ja Pan, On means of calculating values of polynomials, *Russian Math. Surveys*, 1966, 21: 105-136.

[21] L. Qi, Eigenvalues of a real supersymmetric tensor, *Journal of Symbolic Computation*, 2005, 40: 1302–1324.

[22] L. Qi, W. Sun, and Y. Wang, Numerical multilinear algebra and its applications, *Frontiers of Mathematics in China*, 2007, 2: 501–526.

[23] L. Qi and K.L. Teo, Multivariate polynomial minimization and its application in signal processing, *Journal of Global Optim.*, 2003, 26: 419–433.

[24] L. Qi, Y. Wang, and E.X. Wu, D-eigenvalues of diffusion kurtosis tensors, *Journal of Computational and Applied Mathematics*, 2008, 221: 150–157.

[25] L. Qi, G. Yu, and E.X. Wu, Higher order positive semi-definite diffusion tensor imaging, *SIAM Journal on Imaging Sciences*, 2010, 3: 416–433.

[26] L. Qi, G. Yu, and Y. Xu, Nonnegative diffusion orientation distribution function, *Journal of Mathematical Imaging and Vision*, 2013, 45: 103-113.

[27] W. Rudin, *Principles of Mathematical Analysis*, 3rd edition, McGraw-Hill, New York, 2006.

[28] V. De Silva and L. Lim, Tensor rank and the ill-posedness of the best low-rank approximation problem, *SIAM J. Matrix Anal. Appl.*, 2008, 30: 1084-1127.

[29] A.J. Sommese and W.W. Wampler, *The Numerical Solution of Systems of Polynomials Arising in Engineering And Science*, World Scientific Pub Co Inc, 2005.

[30] Z. Zeng and T.Y. Li, NACLab, A Matlab Toolbox for Numerical Algebraic Computation, *ACM Communications in Computer Algebra*, 2013, 47: 170–173.