A linear homotopy method for computing generalized tensor eigenpairs

Liping Chen* Lixing Han† Liangmin Zhou‡

Abstract

Let m, m', n be positive integers such that $m \neq m'$. Let $\mathcal{A}$ be an mth order n-dimensional tensor and $\mathcal{B}$ be an m'th order n-dimensional tensor. $\lambda \in \mathbb{C}$ is called a $\mathcal{B}$-eigenvalue of $\mathcal{A}$ if $\mathcal{A}x^{m-1} = \lambda \mathcal{B}x^{m'-1}$, $\mathcal{B}x^{m'} = 1$ for some $x \in \mathbb{C}^n \backslash \{0\}$. In this paper, we propose a linear homotopy method for solving this eigenproblem. We prove that the method finds all isolated $\mathcal{B}$-eigenpairs. Moreover, it is easy to implement. Numerical results are provided to show the efficiency of the proposed method.

Key words. tensors, generalized eigenpairs, polynomial systems, linear homotopy.

AMS subject classification (2010). 15A18, 15A69, 65H10, 65H17, 65H20.

1 Introduction

Let $\mathbb{C}$ be the field of complex numbers. We denote the set of all mth-order, n-dimensional tensors on $\mathbb{C}$ by $\mathbb{C}^{[m,n]}$. For a tensor $\mathcal{A} \in \mathbb{C}^{[m,n]}$ and vector $x \in \mathbb{C}^n$, let $\mathcal{A}x^m$ denote the multilinear form

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

and $\mathcal{A}x^{m-1}$ denote a vector in $\mathbb{C}^n$ whose jth entry is

$$(\mathcal{A}x^{m-1})_j = \sum_{i_2,\cdots,i_m=1}^{n} A_{ji_2\cdots i_m} x_{i_2} \cdots x_{i_m}.$$

*Department of Mathematics, Michigan State University, East Lansing, MI 48824, USA. Email: chenlipi@msu.edu

†Department of Mathematics, University of Michigan-Flint, Flint, MI 48502, USA. Email: lxhan@umflint.edu

‡Department of Mathematics, Michigan State University, East Lansing, MI 48824, USA. Email: zhoulian@msu.edu

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 that $\mathcal{B}x^{m'}$ is not identically zero as a function of x. We say that $(\lambda, x) \in \mathbb{C} \times (\mathbb{C}^n \backslash \{0\})$ is a $\mathcal{B}$-eigenpair of $\mathcal{A}$ if

- when $m = m'$,
$$\mathcal{A}x^{m-1} = \lambda \mathcal{B}x^{m-1}, \tag{1.1}$$
- when $m \neq m'$,
$$\mathcal{A}x^{m-1} = \lambda \mathcal{B}x^{m'-1}, \quad \mathcal{B}x^{m'} = 1, \tag{1.2}$$

The tensor eigenvalue problems were first introduced by Lim [8] and Qi [10] in 2005. Since then, significant research activities have been devoted to the theory, computation, and applications of tensor eigenvalues. Different types of eigenvalues of tensors have been proposed in the literature. In order to handle various tensor eigenvalues in a unified way, Chang, Pearson, and Zhang [3] and Cui, Dai, and Nie [5] defined generalized eigenpairs for real symmetric tensors. Inspired by the definitions in [3, 5], the notion of generalized eigenpairs defined in (1.1) or (1.2) was introduced in [4] for general complex tensors. By choosing suitable m' and $\mathcal{B}$, many different types of tensor eigenpairs defined in the literature can be recovered by (1.1) or (1.2), including eigenpairs [8, 10], E-eigenpairs [8, 10], H-eigenpairs [10], Z-eigenpairs [8, 10], and D-eigenpairs [11].

Note that problem (1.1) or (1.2) is a polynomial system. To find all solutions of a polynomial system, the homotopy continuation methods are very attractive in terms of computational cost and space (see, for example, [7], [12]). In a previous paper [4], we proposed two homotopy continuation methods (Algorithms 3.1 and 3.2 in [4]) for computing all eigenpairs of a general real or complex tensor. Specifically, Algorithm 3.1 is a linear homotopy method which handles the case when $m = m'$. On the other hand, Algorithm 3.2, which handles the case $m \neq m'$, is based on a polyhedral homotopy method (see, for example, [7]).

The most time consuming part of homotopy continuation methods is the path following step. The polyhedral homotopy methods have the advantage that they can follow much fewer paths than a continuation using other homotopies, such as a total degree homotopy. As indicated in [1], however, the polyhedral homotopy methods involves 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 paper, we propose a linear homotopy method for computing all eigenpairs of a tensor in the case when $m \neq m'$. We will prove that the method finds all isolated eigenpairs of problem (1.2). This method is easy to implement. Moreover, it follows the optimal number of paths by fully using the solution structure of problem (1.2), making it an efficient and competitive homotopy method for computing eigenpairs of general tensors.

The paper is organized as follows. We first describe the linear homotopy method in Section 2. We then prove that the method finds all isolated eigenpairs in Section 3. Finally, we give some numerical results in Section 4.

2 A linear homotopy algorithm

Suppose that $m \neq m'$, $\mathcal{A} \in \mathbb{C}^{[m,n]}$ and $\mathcal{B} \in \mathbb{C}^{[m',n]}$. As discussed in Remark 2.1 in [4], the solution set of $\mathcal{A}x^{m-1} = \lambda \mathcal{B}x^{m'-1}$ consists of different equivalence classes. If (λ, x) is a solution, we denote its corresponding equivalence class by

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

Imposing the normalization condition $\mathcal{B}x^{m'} = 1$, the problem (1.2) has m' eigenpairs from each equivalence class. According to this observation, the problem of solving the eigenvalue problem (1.2) can be converted to first solving the following polynomial system

$$P(\lambda, x) = \begin{pmatrix} \mathcal{A}x^{m-1} - \lambda \mathcal{B}x^{m'-1} \\ \eta^T x + \eta_0 \end{pmatrix} = 0, \tag{2.3}$$

where $\eta \in \mathbb{C}^n$ and $\eta_0 \in \mathbb{C}$ are randomly chosen, and then for each obtained solution of (2.3) normalize it to get an eigenpair (λ^*, x^*) satisfying (1.2), find m' equivalent eigenpairs (λ', x') by setting $\lambda' = t^{m-m'}\lambda^*$ and $x' = tx^*$ with t being a root of $t^{m'} = 1$.

We shall construct a linear homotopy

$$H(\lambda, x, t) = (1-t)\gamma G(\lambda, x) + tP(\lambda, x) = 0, \quad t \in [0, 1), \tag{2.4}$$

and use a homotopy continuation method to solve problem (2.3), where γ is a randomly chosen complex number on the unit circle. The use of γ in (2.4) is the so-called "Gamma trick" which leads to finding all isolated solutions with probability one. A key step in the linear homotopy method is to choose a suitable starting polynomial system $G(\lambda, x)$.

The total degree homotopy (see [7, 9, 13]) provides an option to choose the starting system $G(\lambda, x)$ in (2.4). Denote $P(\lambda, x)$ in (2.3) by $P(\lambda, x) := (p_1(\lambda, x), \ldots, p_{n+1}(\lambda, x))$.

Let $d_1, \ldots, d_{n+1}$ be the degrees of the polynomials $p_1(\lambda, x), \ldots, p_{n+1}(\lambda, x)$, respectively. Then $d_1 = \cdots = d_n = \max(m-1, m')$ and $d_{n+1} = 1$. The number $d_1 \times \cdots \times d_{n+1} = \max((m-1)^n, (m')^n)$ is called the total degree of $P(\lambda, x)$. If the starting system $G(\lambda, x)$ in (2.4) is chosen to be

$$\begin{pmatrix} a_1 x_1^{d_1} - b_1 \\ \vdots \\ a_n x_n^{d_n} - b_n \\ a_{n+1}\lambda^{d_{n+1}} - b_{n+1} \end{pmatrix}, \tag{2.5}$$

where $a_1, \ldots, a_{n+1}, b_1, \ldots, b_{n+1}$ are randomly chosen nonzero complex numbers, then the resulting linear homotopy (2.4) is called the total degree homotopy. It can be proved (see, for example, [7, 9, 13]) that all the isolated zeros of $P(\lambda, x)$ in (2.3) can be found by tracing solution paths of the total degree homotopy (2.4). We observe that (2.5) is easy to solve and it has $\max((m-1)^n, (m')^n)$ zeros. Thus, to use the total degree homotopy one has to follow $\max((m-1)^n, (m')^n)$ paths. On the other hand, it has been shown in [4] that (2.3) has at most $((m-1)^n - (m'-1)^n)/(m-m')$ isolated zeros, which is far less than $\max((m-1)^n, (m')^n)$ when m, m', or n gets large. Therefore, instead of using the total degree homotopy, a more efficient linear homotopy is desired for solving (2.3).

In this paper, we propose to solve (2.3) by using the linear homotopy (2.4) with the following starting system $G(\lambda, x)$:

- if $m > m'$,

$$G(\lambda, x) = \begin{pmatrix} x_1^{m-1} - \lambda x_1^{m'-1} - \alpha_1 x_1^{m-m'} + \alpha_1\lambda \\ \vdots \\ x_n^{m-1} - \lambda x_n^{m'-1} - \alpha_n x_n^{m-m'} + \alpha_n\lambda \\ w^T x + w_0 \end{pmatrix} = \begin{pmatrix} (x_1^{m'-1} - \alpha_1)(x_1^{m-m'} - \lambda) \\ \vdots \\ (x_n^{m'-1} - \alpha_n)(x_n^{m-m'} - \lambda) \\ w^T x + w_0 \end{pmatrix} = 0, \tag{2.6}$$

- if $m < m'$,

$$G(\lambda, x) = \begin{pmatrix} x_1^{m-1} - \lambda x_1^{m'-1} - \alpha_1 + \lambda\alpha_1 x_1^{m'-m} \\ \vdots \\ x_n^{m-1} - \lambda x_n^{m'-1} - \alpha_n + \lambda\alpha_n x_n^{m'-m} \\ w^T x + w_0 \end{pmatrix} = \begin{pmatrix} (x_1^{m-1} - \alpha_1)(1 - \lambda x_1^{m'-m}) \\ \vdots \\ (x_n^{m-1} - \alpha_n)(1 - \lambda x_n^{m'-m}) \\ w^T x + w_0 \end{pmatrix} = 0, \tag{2.7}$$

where $\alpha_1, \ldots, \alpha_n, w_0$ are randomly chosen nonzero complex numbers and w is a randomly chosen nonzero complex vector. Apparently, the starting system is very easy to solve. We

will justify why the linear homotopy (2.4) works with this choice of the starting system in Sections 3.1 and 3.2 and why it follows the optimal possible number of paths (which is, $((m-1)^n-(m'-1)^n)/(m-m')$) in Section 3.3. We remark that for a general polynomial system, choosing a linear homotopy with a suitable starting system that can trace much fewer paths than the total degree homotopy is a very challenging or impossible task. Our choice of the starting system (2.6) or (2.7) for the linear homotopy (2.4) effectively utilizes the structure of the polynomial system (2.3) obtained from the generalized tensor eigenvalue problem.

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

Algorithm A: (Compute $\mathcal{B}$-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 defined in (2.6) or (2.7).

Step 2. Compute all solutions (λ, x) of (1.2) by following the paths from $t = 0$ to $t = 1$ using the linear homotopy $H(\lambda, x, t)$ defined in (2.4).

Step 3. For each (λ, x) obtained in Step 2, if $\mathcal{B}x^{m'} \neq 0$, normalize it to get 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 (1.2).

Step 4. Compute m' equivalent eigenpairs (λ', x') for each (λ^*, x^*) obtained in Step 3 by $\lambda' = t^{m-m'}\lambda^*$ and $x' = tx^*$ with t being a root of $t^{m'} = 1$.

REMARK 2.1 If we are only concerned to obtain a representative from each equivalence class (see, for example, [2]), we can skip Step 4 in the above Algorithm.

3 Analysis of Algorithm A

In this section, we will prove that Algorithm A finds all isolated generalized eigenpairs of problem (1.2). We will establish this result by treating the linear homotopy (2.4) in a coefficient-parameter homotopy framework. The following Lemmas 3.1 and 3.2 about parameter homotopy play an essential role in our analysis.

LEMMA 3.1 (Theorem 7.1.1 in [12]) *Let $F(z;q)$ be a system of polynomials in n variables and l parameters,*

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

that is, $F(z;q) = (f_1(z;q), \ldots, f_n(z;q))^T$ and each $f_i(z;q)$ is polynomial in both z and q. Furthermore, let $\mathcal{N}(q)$ denote the number of nonsingular solutions as a function of q:

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

Then

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

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

We give an example to illustrate Lemma 3.1 before presenting Lemma 3.2. 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 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 3.1. 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 $q_2^2 - 4q_1 q_3 = q_2^2 \neq 0$, then $F = q_2 z_1 + q_3$ 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 discussions, 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 3.2 (Theorem 8.3.1 in [12]) *Let $F(z;q)$, $\mathcal{N}(q)$ and $\mathcal{N}_F$ be defined as Lemma 3.1. Suppose $F(z;q)$ is linear in q. Let $f(z) = F(z;q_1)$ for some given $q_1 \in \mathbb{C}^l$. If $g(z) = F(z;q_0)$ for some $q_0 \in \mathbb{C}^l$ 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 3.1 and Lemma 3.2 suggest the following steps to prove that our linear homotopy (2.4) can be used to find all isolated solutions of the system (2.3) when $m \neq m'$:

(1). Construct a polynomial system $F(\lambda, x; q)$, where q denotes the set of parameters, to satisfy that $F(\lambda, x; q)$ is linear in q and $F(\lambda, x; q_1) = P(\lambda, x)$ for certain parameter set q_1.

(2). Compute $\mathcal{N}_F$ as defined in Lemma 3.1 for $F(\lambda, x; q)$;

(3). Find a parameter set q_0 such that $G(\lambda, x) = F(\lambda, x; q_0)$ has $\mathcal{N}_F$ nonsingular zeros, where $G(\lambda, x)$ is defined in (2.6) for $m > m'$ and (2.7) for $m < m'$.

We shall justify the above three steps for the cases of $m > m'$ and $m < m'$ separately.

3.1 The justification of using linear homotopy (2.4) to solve (2.3) when $m > m'$

Let $\mathcal{D} \in \mathbb{C}^{[m,n]}$, $\mathcal{E} \in \mathbb{C}^{[m',n]}$, $\mathcal{L} \in \mathbb{C}^{[2,n]}$, $\xi \in \mathbb{C}^n$ and $\xi_0 \in \mathbb{C}$. Let $\mathrm{vdiag}(\mathcal{L}) := (\mathcal{L}_{11}, \ldots, \mathcal{L}_{nn})^T$. Write

$$q := \{\mathcal{D}, \mathcal{E}, \mathcal{L}, \xi, \xi_0\}. \tag{3.1}$$

Consider

$$F(\lambda, x; q) = \begin{pmatrix} \mathcal{D}x^{m-1} - \lambda\mathcal{E}x^{m'-1} - \mathcal{L}x^{[m-m']} + \lambda \cdot \mathrm{vdiag}(\mathcal{L}) \\ \xi^T x + \xi_0 \end{pmatrix}. \tag{3.2}$$

It is clear that F is linear in q. Take

$$q_1 := \{\mathcal{A}, \mathcal{B}, 0, \eta, \eta_0\}, \tag{3.3}$$

we obtain $F(\lambda, x; q_1) = P(\lambda, x)$, which is our target system (2.3).

To compute $\mathcal{N}_F$ for $F(\lambda, x; q)$ defined in (3.2), we prove the following lemma first.

LEMMA 3.3 *Let $F(\lambda, x; q)$ be defined as (3.2) with q in (3.1) being the set of parameters. Then the number of isolated zeros, counting multiplicities, of $F(\lambda, x; q)$ is bounded by*

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

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

The proof of Lemma 3.3 follows exactly the same approach used in the proof of Theorem 2.3 in [4]. We omit it here.

By Lemma 3.1, $\mathcal{N}_F$ is the bound of nonsingular zeros of $F(\lambda, x; q)$ in (3.2). Since nonsingular zeros are isolated zeros, Lemma 3.3 implies that

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

On the other hand, $F(\lambda, x; q)$ in (3.2) has $((m-1)^n - (m'-1)^n)/(m-m')$ nonsingular zeros when q is generic. So by Lemma 3.1,

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

Therefore,

$$\mathcal{N}_F = \frac{(m-1)^n - (m'-1)^n}{m - m'}. \tag{3.4}$$

Now it is sufficient to find a parameter set q_0 such that $F(\lambda, x; q_0)$ with F defined in (3.2) has $\mathcal{N}_F$ nonsingular zeros. Let

$$q_0 := \{I^{[m,n]}, I^{[m',n]}, \mathcal{G}, w, w_0\}, \tag{3.5}$$

where $I^{[m,n]}$ is the m-th order n dimensional unit 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, $w \in \mathbb{C}^n$ and $w_0 \in \mathbb{C}$ are randomly chosen. One can verify that $F(\lambda, x; q_0)$ with F defined in (3.2) is equal to $G(\lambda, x)$ as defined in (2.6).

We now prove the following lemma.

LEMMA 3.4 *Let $G(\lambda, x)$ be defined as (2.6). Then $G(\lambda, x)$ has exactly $\mathcal{N}_F$, as given in (3.4), nonsingular zeros.*

Proof: It can be asserted that at least one of

$$x_1^{m-m'} - \lambda = 0, \quad \ldots, \quad x_n^{m-m'} - \lambda = 0 \tag{3.6}$$

must be true. Otherwise system (2.6) is equivalent to an overdetermined system of $n+1$ equations and n unknowns, which has no solutions due to randomness. Assume that i $(1 \le i \le n)$ equations of (3.6) are true. If the first i equations of (3.6) are chosen to be true, then

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

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

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

So each x_j $(j = 2, \ldots, n)$ can be expressed by x_1 using $m - m'$ ways. Correspondingly, x_1 can be determined uniquely by the choices of $x_{i+1}, \ldots, x_n$ and the last random linear 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 (3.6) are true. This argument holds for any i equations of (3.6) are chosen to be true. So there are

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

solutions when i equations of (3.6) are chosen to be true. Since i may be selected to be $1, \ldots, n$, the number of zeros of $G(\lambda, x)$ 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 $G(\lambda, x)$ in (2.6) must be nonsingular. As discussed above, any zero (λ^*, x^*) of $G(\lambda, x)$ satisfies

$$\begin{aligned} (x_j^*)^{m-m'} - \lambda = 0, \qquad & j = 1, \cdots, i, \\ (x_j^*)^{m'-1} - \alpha_j = 0, \qquad & j = i+1, \cdots, n, \\ w_1 x_1^* + \cdots + w_n x_n^* + w_0 = 0. & \end{aligned} \tag{3.7}$$

for some $1 \le i \le n$. Let $DG(\lambda, x)$ be the Jacobian of $G(\lambda, x)$ with respect to (λ, x). It is sufficient to show $DG(\lambda^*, x^*)$ 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

$$DG(\lambda, x) = \begin{pmatrix} A_1 & B_1 & & & & & \\ \vdots & & \ddots & & & & \\ A_i & & & B_i & & & \\ A_{i+1} & & & & B_{i+1} & & \\ \vdots & & & & & \ddots & \\ A_n & & & & & & B_n \\ 0 & w_1 & \ldots & w_i & w_{i+1} & \ldots & w_n \end{pmatrix}.$$

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

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

by (3.7). For simplicity, write $A_j^* := A(\lambda^*, x^*)$ and $B_j^* := B(\lambda^*, x^*)$. Then

$$DG(\lambda^*, x^*) = \begin{pmatrix} A_1^* & B_1^* & & & & & \\ \vdots & & \ddots & & & & \\ A_i^* & & & B_i^* & & & \\ 0 & & & & B_{i+1}^* & & \\ \vdots & & & & & \ddots & \\ 0 & & & & & & B_n^* \\ 0 & w_1 & \ldots & w_i & w_{i+1} & \ldots & w_n \end{pmatrix}.$$

So

$$
\begin{aligned}
&\det(DG(\lambda^*, x^*)) \\
&= \left(\prod_{j=i+1}^{n} (-1)^{(i+1)+(i+2)} B_j^*\right) \det \begin{pmatrix}
A_1^* & B_1^* & & & & & & \\
\vdots & & \ddots & & & & & \\
A_{l-1}^* & & & B_{l-1}^* & & & & \\
A_l^* & & & & B_l^* & & & \\
A_{l+1}^* & & & & & B_{l+1}^* & & \\
\vdots & & & & & & \ddots & \\
A_i^* & & & & & & & B_i^* \\
0 & w_1 & \dots & w_{l-1} & w_l & w_{l+1} & \dots & w_i
\end{pmatrix} \\
&= \left(\prod_{j=i+1}^{n} (-1) B_j^*\right) \sum_{l=1}^{i} (-1)^{(i+1)+(l+1)} w_l \cdot \det \begin{pmatrix}
A_1^* & B_1^* & & & & & \\
\vdots & & \ddots & & & & \\
A_{l-1}^* & & & B_{l-1}^* & & & \\
A_l^* & & & & 0 & & \\
A_{l+1}^* & & & & B_{l+1}^* & & \\
\vdots & & & & & \ddots & \\
A_i^* & & & & & & B_i^*
\end{pmatrix}.
\end{aligned}
$$

And further computation gives

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

by (3.7). □

By Lemmas 3.1, 3.2, 3.3, and 3.4, we have actually proved the following theorem.

THEOREM 3.1 *Let $H(\lambda, x, t)$ be defined as (2.4) with $G(\lambda, x)$ given by (2.6). Then following solution paths of (2.4) will give all isolated solutions of (2.3) for $m > m'$.*

This theorem implies that Algorithm A is guaranteed to find all isolated generalized eigenpairs of problem (1.2) when $m > m'$.

3.2 The justification of using linear homotopy (2.4) to solve (2.3) when $m < m'$

Let q be defined as (3.1). Consider

$$F(\lambda, x; q) = \begin{pmatrix} \mathcal{D}x^{m-1} - \lambda \mathcal{E} x^{m'-1} - \text{vdiag}(\mathcal{L}) + \lambda \mathcal{L} x^{[m'-m]} \\ \xi^T x + \xi_0 \end{pmatrix}. \tag{3.8}$$

It can be verified that F in (3.8) is linear in q. For q_1 defined in (3.3), we have $F(\lambda, x; q_1)$ is the same as our target system (2.3). Similar to Lemma 3.2 we can prove the following lemma.

LEMMA 3.5 *Let $F(\lambda, x; q)$ be defined as (3.8) with q in (3.1) being the set of parameters. Then the number of isolated zeros, counting multiplicities, of $F(\lambda, x; q)$ is bounded by*

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

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

Similar to the proof of (3.4), using Lemma 3.1 and Lemma 3.5 we can show

$$\mathcal{N}_F = \frac{(m-1)^n - (m'-1)^n}{m - m'} \tag{3.9}$$

for $F(\lambda, x; q)$ defined as (3.8).

Let q_0 be defined as (3.5). Then one can verify that $F(\lambda, x; q_0)$ with F defined in (3.8) equals $G(\lambda, x)$ given in (2.7). Note that in (2.7), λ cannot be zero. Otherwise we end up with an overdetermined system of $n+1$ equations and n unknowns, which has no solution due to randomness. Using this fact and following a similar approach used in the proof of Lemma 3.4, we can prove the following lemma.

LEMMA 3.6 *Let $G(\lambda, x)$ be defined as (2.7). Then $G(\lambda, x)$ has exactly $\mathcal{N}_F$, given in (3.9), nonsingular zeros.*

By Lemma 3.1, 3.2, 3.5, and 3.6, we have the following theorem.

THEOREM 3.2 *Let $H(\lambda, x, t)$ be defined as (2.4) with $G(\lambda, x)$ given by (2.7). Then following solution paths of (2.4) will give all isolated solutions of (2.3) for $m < m'$.*

This theorem guarantees that Algorithm A finds all isolated eigenpairs of problem (1.2) when $m < m'$.

3.3 Efficiency of Algorithm A

While different homotopy methods use different starting systems, almost all of them use the same strategy to follow paths from the solutions of the starting system to the solutions of the targeting system: A prediction-correction method (see, for example, [1, 7, 12]). The path tracking step is the most time consuming step in a homotopy method since often many paths need to be followed. The fewer the number of paths followed, the more efficient is the homotopy method.

Suppose that $m \neq m'$. In Theorem 2.3 ([4]), it is proved that if $\mathcal{A}$ has finitely many equivalence classes of $\mathcal{B}$ eigenpairs, then the number of equivalence classes of $\mathcal{B}$ eigenpairs, counting multiplicity, is bounded by

$$K(m, m', n) = \frac{(m-1)^n - (m'-1)^n}{m - m'}. \tag{3.10}$$

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 multiplicity.

This result implies that the optimal number of paths to follow in a homotopy method for solving the system (2.3) is $K(m, m', n)$. Our starting system (2.6) or (2.7) has exactly $K(m, m', n)$ nonsingular solutions. Therefore, Algorithm A follows the optimal number of paths for solving the system (2.3). This makes it an efficient homotopy method for computing generalized eigenpairs.

4 Numerical results

In this section, we present some numerical results to illustrate the effectiveness and efficiency of Algorithm A. The numerical experiments were done 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.

We compare the performance of Algorithm A with that of the following Algorithm 3.2 proposed in [4].

Algorithm: (Algorithm 3.2 ([4]))

Step 1. Use modified PSOLVE ([14]) to get all solutions (λ, x) of (2.3).

Step 2. Compute a representative from each equivalence solution class of $\mathcal{A}x^{m-1} = \lambda\mathcal{B}x^{m'-1}$ by normalizing each (λ, x) obtained in Step 1 to get 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 (1.2).

Step 3. 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 $t^{m'} = 1$.

Algorithm 3.2 depends on the polynomial system solver PSOLVE ([14]), which uses a sophisticated polyhedral homotopy method. PSOLVE solves a polynomial system in two stages: computing mixed volume and tracking paths. Mixed volume computation is a very complicated process. It is also expensive for large or dense polynomial systems. On the other hand, Algorithm A uses a linear homotopy whose starting system is very easy to solve. Therefore, although the numbers of paths to track for Algorithm A and for Algorithm 3.2 are the same, it is expected that Algorithm A is more efficient than Algorithm 3.2 as it avoids the mixed volume computation. We also wish to remind that the implementation of Algorithm A is much simpler.

Algorithm 3.2 has been implemented as a function `teneig` in the tensor eigenvalue package `TenEig1.1` ([4]). Numerical results in [4] show that `teneig` is significantly more efficient than the popular NSolve in Mathematica for computing generalized eigenpairs defined in (1.2). We will compare the performance of our implemented Algorithm A with that of `teneig` here.

EXAMPLE 4.1 We consider 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 1 and Table 2 respectively. In these tables, $K(m, m', n)$ (defined in (3.10)) 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 [4] and Algorithm A, the total number of eigenpairs is $m'K(m,m',n)$. We use N to denote the number of equivalence classes of $\mathcal{B}$-eigenpairs found by `Algorithm A` or `teneig` in the tables. We also report CPU time (in seconds) used.

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

Table 1: Comparison of `Algorithm A` with `teneig` for $m > m'$

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

Table 2: Comparison of `Algorithm A` with `teneig` for $m < m'$

EXAMPLE 4.2 We consider to find 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 3. In this table, $K(m, 2, n)$ (defined in (3.10)) 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, [2]). N denotes the number of equivalence classes of E-eigenpairs found by `Algorithm A` or `teneig`. The reported CPU time is in seconds.

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

Table 3: Comparison of `Algorithm A` with `teneig` for computing E-eigenpairs

From Tables 1–3, we observe that both Algorithm A and `teneig` successfully find all $\mathcal{B}$-eigenpairs of a generic tensor $\mathcal{A}$. Algorithm A is more efficient in terms of CPU time. We believe that the savings of Algorithm A in computational time are achieved due to its use of the linear homotopy (2.4) and the simple starting system (2.6) or (2.7). `teneig`, on the other hand, uses a polyhedral homotopy which needs to compute mixed volumes. The efficiency and easy implementation of Algorithm A makes it a competitive method for computing tensor eigenpairs when $m \neq m'$.

Acknowledgment. We would like to thank the reviewer for the very helpful comments and suggestions that have improved the manuscript.

References

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

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

[4] L. Chen, L. Han and L. Zhou, Computing tensor eigenvalues via homotopy methods, arXiv:1501.04201, 2015. *SIAM Journal on Matrix Analysis and Applications*, 2016, 37(1): 290–319.

[5] 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.

[6] B. Huber and B. Sturmfels, A polyhedral method for solving sparse polynomial systems, *Mathematics of Computation*, 1995, 64: 1541-1555.

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

[8] 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.

[9] A.P. Morgan, *Solving Polynomial Systems Using Continuation for Engineering and Scientific Problems*, Society for Industrial and Applied Mathematics, Philadelphia, 2009.

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

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

[12] 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.

[13] A.H. Wright, Finding all solutions to a system of a polynomial equations, *Mathematics of Computation*, 1985, 44: 125-133.

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

Submitted July 2015. Revised January 2016.