SIAM J. MATRIX ANAL. APPL.
Vol. 37, No. 1, pp. 290–319

COMPUTING TENSOR EIGENVALUES VIA HOMOTOPY METHODS*

LIPING CHEN†, LIXING HAN‡, AND LIANGMIN ZHOU†

Abstract. We introduce the concept of mode-k generalized eigenvalues and eigenvectors of a tensor and prove some properties of such eigenpairs. In particular, we derive an upper bound for the number of equivalence classes of generalized tensor eigenpairs using mixed volume. Based on this bound and the structures of tensor eigenvalue problems, we propose two homotopy continuation type algorithms to solve tensor eigenproblems. With proper implementation, these methods can find all equivalence classes of isolated generalized eigenpairs and some generalized eigenpairs contained in the positive dimensional components (if there are any). We also introduce an algorithm that combines a heuristic approach and a Newton homotopy method to extract real generalized eigenpairs from the found complex generalized eigenpairs. A MATLAB software package, `TenEig`, has been developed to implement these methods. Numerical results are presented to illustrate the effectiveness and efficiency of `TenEig` for computing complex or real generalized eigenpairs.

Key words. tensors, mode-k eigenvalues, polynomial systems, homotopy continuation, TenEig

AMS subject classifications. 15A18, 15A69, 65H10, 65H17, 65H20

DOI. 10.1137/15M1010725

1. Introduction.

Eigenvalues of tensors were first introduced by Lim [29] and Qi [37] in 2005. Since then, tensor eigenvalues have found applications in automatic control, statistical data analysis, diffusion tensor imaging, image authenticity verification, spectral hypergraph theory, quantum entanglement, etc.; see, for example, [8, 11, 21, 37, 38, 40, 41, 42] and the references therein. The tensor eigenvalue problem has become an important subject of numerical multilinear algebra.

Various definitions of eigenvalues for tensors have been proposed in the literature, including E-eigenvalues and eigenvalues in the complex field and Z-eigenvalues, H-eigenvalues, and D-eigenvalues in the real field [29, 37, 40]. In [7], 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 [12].

Unlike the matrix eigenvalue problem, computing eigenvalues of the third or higher order tensors is a difficult problem [19]. Nonetheless, several algorithms which aim at computing one or some eigenvalues of a tensor have been developed recently. These algorithms are designed for tensors of certain type, such as entrywise nonnegative or symmetric tensors.

For nonnegative tensors, Ng, Qi, and Zhou [33] proposed a power-type method for computing the largest H-eigenvalue of a nonnegative tensor. Modified versions of the Ng–Qi–Zhou method have been proposed in [31, 50, 51].

For real symmetric tensors, Hu, Huang, and Qi [20] proposed a sequential semidefinite programming method for computing extreme Z-eigenvalues. Hao, Cui, and Dai [16] proposed a sequential subspace projection method for a similar purpose. Kolda and Mayo [23] proposed a shifted power method (SSHOPM) for computing a

*Received by the editors March 2, 2015; accepted for publication (in revised form) by D. Kressner January 4, 2016; published electronically March 15, 2016.
http://www.siam.org/journals/simax/37-1/M101072.html
†Department of Mathematics, Michigan State University, East Lansing, MI 48824 (chenlipi@msu.edu, zhoulian@msu.edu).
‡Department of Mathematics, University of Michigan–Flint, Flint, MI 48502 (lxhan@umflint.edu).

Z-eigenvalue. They have improved SSHOPM in [24] by updating the shift parameter adaptively. The resulting method can be used to compute a real generalized eigenvalue. Han [15] proposed an unconstrained optimization method for computing a real generalized eigenvalue for even order real symmetric tensors. The methods in [15, 23, 24] can find more eigenvalues of a symmetric tensor if they are run multiple times using different starting points. Recently, Cui, Dai, and Nie [12] proposed a novel method for computing all real generalized eigenvalues.

In this paper, we are concerned with computing all eigenpairs of a general real or complex tensor. As can be seen from the next section, finding eigenpairs of a tensor amounts to solving a system of polynomials. Naturally one would consider use of algebraic geometry methods such as the Gröbner basis method and the resultant method [10] for this purpose. These methods can obtain symbolic solutions of a polynomial system, which are accurate. However, they are expensive in terms of computational cost and space. Moreover, they are difficult to parallelize. A class of numerical methods, the homotopy continuation methods, can overcome these shortcomings of the Gröbner basis and the resultant methods. During the past few decades, significant advances have been made on homotopy continuation methods for polynomial systems; see, for example, [3, 25, 26, 32, 43]. Recently, the homotopy techniques have been used to study tensor decomposition and perfect identification problems [17].

In this paper we investigate computing complex eigenpairs of general tensors using homotopy continuation methods. One attractive feature of the homotopy continuation methods is that they can find all isolated solutions of polynomial systems and some solutions in the positive dimensional solution components. We propose two homotopy-type algorithms for computing complex eigenpairs of a tensor. These algorithms allow us to find all equivalence classes of isolated eigenpairs of a general tensor and some eigenpairs in positive dimensional eigenspaces (if there are any). We also present an algortihm combining a heuristic approach and a Newton homotopy method to compute real eigenpairs based on the found complex eigenpairs. Numerical examples show that our methods are effective and efficient.

This paper is organized as follows. In section 2, we define mode-k generalized eigenvalues and eigenvectors which extend the matrix right eigenpairs and left eigenpairs to higher order tensors. Some properties of such eigenpairs are proved. An upper bound for the number of equivalence classes of generalized tensor eigenpairs using mixed volume is derived. In section 3, we consider computing mode-k generalized complex eigenpairs and present two algorithms. In section 4, we introduce a method to compute real mode-k generalized eigenpairs. Finally in section 5, some numerical results are provided.

2. Tensor eigenvalues and eigenvectors. Let $\mathbb{F} = \mathbb{C}$ or $\mathbb{R}$ be the complex field or the real field. Let $m \geq 2$, $m' \geq 2$, and n be positive integers. Denote the set of all mth order, n-dimensional tensors on the field $\mathbb{F}$ by $\mathbb{F}^{[m,n]}$. A tensor in $\mathbb{F}^{[m,n]}$ is indexed as

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

where $A_{i_1 i_2 \cdots i_m} \in \mathbb{F}$, for $1 \leq i_1, i_2, \ldots, i_m \leq n$.

For $x \in \mathbb{C}^n$, the tensor $\mathcal{A}$ defines a multilinear form

$$\mathcal{A}x^m = \sum_{i_1, \cdots, i_m = 1}^{n} A_{i_1 i_2 \cdots i_m} x_{i_1} x_{i_2} \cdots x_{i_m}. \tag{2.1}$$

For $1 \le k \le m$, $\mathcal{A}^{(k)}x^{m-1}$ is an n-vector whose jth entry is defined as

$$(\mathcal{A}^{(k)}x^{m-1})_j = \sum_{i_1,\cdots,i_{k-1},i_{k+1},\cdots,i_m=1}^{n} A_{i_1\cdots i_{k-1}ji_{k+1}\cdots i_m}x_{i_1}\cdots x_{i_{k-1}}x_{i_{k+1}}\cdots x_{i_m}. \tag{2.2}$$

When $k = 1$, the vector $\mathcal{A}^{(1)}x^{m-1}$ is denoted by $\mathcal{A}x^{m-1}$.

A real tensor $\mathcal{A} \in \mathbb{R}^{[m,n]}$ is *positive definite* if the multilinear form $\mathcal{A}x^m$ is positive for all $x \in \mathbb{R}^n\backslash\{0\}$. A tensor $\mathcal{A} \in \mathbb{F}^{[m,n]}$ is *symmetric* if its entries $A_{i_1i_2\cdots i_m}$ are invariant under any permutations of their indices $i_1, i_2, \ldots, i_m$.

We now introduce the following mode-k generalized eigenvalue definition for a general tensor $\mathcal{A}$.

DEFINITION 2.1. *Let $\mathcal{A} \in \mathbb{F}^{[m,n]}$ and $\mathcal{B} \in \mathbb{F}^{[m',n]}$. Assume that $\mathcal{B}x^{m'}$ is not identically zero as a function of x. For $1 \le k \le m$, if there exist a scalar $\lambda \in \mathbb{C}$ and a vector $x \in \mathbb{C}^n\backslash\{0\}$ such that*

(i) *when $m = m'$,*

$$\mathcal{A}^{(k)}x^{m-1} = \lambda\mathcal{B}x^{m-1}, \tag{2.3}$$

(ii) *when $m \ne m'$,*

$$\mathcal{A}^{(k)}x^{m-1} = \lambda\mathcal{B}x^{m'-1}, \quad \mathcal{B}x^{m'} = 1, \tag{2.4}$$

then λ is called a mode-k $\mathcal{B}$-eigenvalue of $\mathcal{A}$ and x a mode-k $\mathcal{B}$-eigenvector associated with λ. (λ, x) is called a mode-k $\mathcal{B}$-eigenpair of $\mathcal{A}$.

If $\lambda \in \mathbb{R}, x \in \mathbb{R}^n$, then λ is called a mode-k $\mathcal{B}_R$-eigenvalue of $\mathcal{A}$ and x a mode-k $\mathcal{B}_R$-eigenvector associated with λ, and (λ, x) a mode-k $\mathcal{B}_R$-eigenpair of $\mathcal{A}$.

Denote the set of all mode-k $\mathcal{B}$ eigenvalues of $\mathcal{A}$ by $\sigma_{\mathcal{B}}(\mathcal{A}^{(k)})$.

Remark 2.1. Let (λ, x) be a mode-k $\mathcal{B}$-eigenpair of $\mathcal{A}$. By (2.3) or (2.4), (λ, x) is a solution to $\mathcal{A}^{(k)}x^{m-1} = \lambda\mathcal{B}x^{m'-1}$. So is (λ', x') with $\lambda' = t^{m-m'}\lambda$ and $x' = tx$ for $t \in \mathbb{C}\backslash\{0\}$. From this point of view, the solution space of $\mathcal{A}^{(k)}x^{m-1} = \lambda\mathcal{B}x^{m'-1}$ consists of different equivalence classes. We denote such an equivalence class by

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

When $m \ne m'$, taking arbitrary $(\lambda', x') \in [(\lambda, x)]$ and substituting $x' = tx$ into $\mathcal{B}x^{m'} = 1$ in (2.4) yields $t^{m'} = 1$, which gives m' different values for t. This implies that the normalization $\mathcal{B}x^{m'} = 1$ in (2.4) restricts us to choose m' representative solutions from each equivalence class.

In our later discussions, we often choose only one representative from each equivalence class, and we often count the number of equivalence classes of mode-k $\mathcal{B}$-eigenpairs.

Remark 2.2. If only one representative is desirable from each equivalence class of eigenpairs, we can solve $\mathcal{A}^{(k)}x^{m-1} = \lambda\mathcal{B}x^{m'-1}$ augmented with an additional linear equation

$$a_1x_1 + a_2x_2 + \cdots + a_nx_n + b = 0, \tag{2.5}$$

where $a_1, \ldots, a_n, b$ are random complex numbers. Then we normalize the resulting solution to satisfy $\mathcal{B}x^{m'} = 1$ in the case $m \ne m'$.

In the matrix case when $m = m' = 2$ and $\mathcal{B} = I_n$ (the $n \times n$ identity matrix), the mode-1 eigenvectors are right eigenvectors and the mode-2 eigenvectors are left eigenvectors of $\mathcal{A}$, and the mode-1 and mode-2 eigenvalues are the eigenvalues of matrix $\mathcal{A}$, i.e., $\sigma_{\mathcal{B}}(\mathcal{A}^{(1)}) = \sigma_{\mathcal{B}}(\mathcal{A}^{(2)})$. However, when $m \geq 3$, $\sigma_{\mathcal{B}}(\mathcal{A}^{(k)}) = \sigma_{\mathcal{B}}(\mathcal{A}^{(l)})$ is generally not true when $k \neq l$, unless $\mathcal{A}$ has a certain type of symmetry. The following example illustrates this situation.

Example 2.1. Consider the tensor $\mathcal{A} \in \mathbb{R}^{[3,2]}$ whose entries are

$$A_{111} = 1, A_{121} = 2, A_{211} = 3, A_{221} = 4,$$
$$A_{112} = 5, A_{122} = 6, A_{212} = 7, A_{222} = 0.$$

Choose $m' = 2$ and $\mathcal{B} = I_2$ (the 2×2 identity matrix). Note that in this case, if (λ, x) is a $\mathcal{B}$-eigenpair of $\mathcal{A}$, so is $(-\lambda, -x)$. We follow [5], regarding (λ, x) and $(-\lambda, -x)$ as the same eigenpair. Then

$$\sigma_{\mathcal{B}}(\mathcal{A}^{(1)}) = \{0.4105, 4.3820, 9.8995\},$$

$$\sigma_{\mathcal{B}}(\mathcal{A}^{(2)}) = \{0.2851, 4.3536, 9.5652\},$$

$$\sigma_{\mathcal{B}}(\mathcal{A}^{(3)}) = \{0.2936, 4.3007, 9.4025\}.$$

Clearly, $\sigma_{\mathcal{B}}(\mathcal{A}^{(k)}) \neq \sigma_{\mathcal{B}}(\mathcal{A}^{(l)})$ when $k \neq l$.

PROPOSITION 2.1. *Suppose that (λ, x) is a mode-k $\mathcal{B}$-eigenpair, (μ, x) is a mode-l $\mathcal{B}$-eigenpair of $\mathcal{A}$, and when $m = m'$, $\mathcal{B}x^m \neq 0$. Then $\lambda = \mu$.*

Proof. Note that

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

This immediately implies that $\lambda = \mu$, since $\mathcal{B}x^{m'} = 1$ when $m \neq m'$ and $\mathcal{B}x^m \neq 0$ when $m = m'$. ∎

Let $\mathcal{A} \in \mathbb{F}^{[m,n]}$. For $1 \leq k < l \leq m$, tensor $\mathcal{G} \in \mathbb{F}^{[m,n]}$ is said to be the $\langle k, l \rangle$ *transpose* of $\mathcal{A}$ if

$$\mathcal{G}_{i_1 \cdots i_{k-1} i_l i_{k+1} \cdots i_{l-1} i_k i_{l+1} \cdots i_m} = \mathcal{A}_{i_1 \cdots i_{k-1} i_k i_{k+1} \cdots i_{l-1} i_l i_{l+1} \cdots i_m}$$

for all $1 \leq i_1, \ldots, i_m \leq m$. Denote the $\langle k, l \rangle$ *transpose* of $\mathcal{A}$ by $\mathcal{A}^{\langle k,l \rangle}$. We say that tensor $\mathcal{A}$ is $\langle k, l \rangle$ *partially symmetric* if

$$\mathcal{A}^{\langle k,l \rangle} = \mathcal{A}.$$

PROPOSITION 2.2. *Let $\mathcal{A} \in \mathbb{F}^{[m,n]}$ and $\mathcal{B} \in \mathbb{F}^{[m',n]}$. Assume that $\mathcal{B}x^{m'}$ is not identically zero as a function of x. Let k, l be integers such that $1 \leq k < l \leq m$. Then*

(i) *(λ, x) is a mode-k $\mathcal{B}$-eigenpair of $\mathcal{A}$ if and only if it is a mode-l $\mathcal{B}$-eigenpair of $\mathcal{A}^{\langle k,l \rangle}$.*

(ii) *The sets of mode-k $\mathcal{B}$-eigenpairs and mode-l $\mathcal{B}$-eigenpairs are the same if $\mathcal{A}$ is $\langle k, l \rangle$ partially symmetric.*

The eigenvalues/eigenvectors defined in [7, 12, 37, 40] are mode-1 eigenvalues/eigenvectors. The tensors considered in these papers are primarily real symmetric tensors. For symmetric tensors, the sets of mode-k $\mathcal{B}$-eigenpairs and mode-1 $\mathcal{B}$-eigenpairs are the same for any k. Therefore, mode-1 eigenvalues serve the purpose of those papers. On the other hand, nonsymmetric tensors arise from applications and theoretical

studies; see, for example, [5, 6, 13, 33, 47, 48]. In [29], Lim defined mode-k eigenvalues/eigenvectors for nonsymmetric real tensors $\mathcal{A}$ when $\mathcal{B}$ is the m'th order identity tensor for some $m' \geq 2$. Definition 2.1 considers more general $\mathcal{A}$ and $\mathcal{B}$.

As in [7, 12], Definition 2.1 adapts a unified approach to define tensor eigenvalues. It covers various types of tensor eigenvalues introduced in the literature, including the following:

(i) If $\mathcal{A} \in \mathbb{R}^{[m,n]}$, $m' = 2$, and $\mathcal{B}$ is the identity matrix $I_n \in \mathbb{R}^{n\times n}$, the mode-1 $\mathcal{B}$-eigenpairs are the E-eigenpairs and the mode-1 $\mathcal{B}_R$-eigenpairs are the Z-eigenpairs defined in [37], which satisfy

$$\mathcal{A}x^{m-1} = \lambda x, \quad x^T x = 1. \tag{2.6}$$

(ii) If $\mathcal{A} \in \mathbb{R}^{[m,n]}$, $m' = 2$ and $\mathcal{B} = D$, where $D \in \mathbb{R}^{n\times n}$ is a symmetric positive definite matrix, the $\mathcal{B}_R$-eigenpairs are the D-eigenpairs defined in [40], which satisfy

$$\mathcal{A}x^{m-1} = \lambda Dx, \quad x^T Dx = 1. \tag{2.7}$$

(iii) If $\mathcal{A} \in \mathbb{R}^{[m,n]}$, $m = m'$, and $\mathcal{B} = \mathcal{I}$ is the identity tensor, mode-1 $\mathcal{B}$-eigenpairs are the eigenpairs defined in [37], which satisfy

$$\mathcal{A}x^{m-1} = \lambda x^{[m-1]}, \tag{2.8}$$

where $x^{[m-1]} = [x_1^{m-1}, x_2^{m-1}, \ldots, x_n^{m-1}]^T$.

(iv) If $\mathcal{A} \in \mathbb{R}^{[m,n]}$, $m = m'$, and $\mathcal{B} = \mathcal{I}$ is the unit tensor, mode-1 $\mathcal{B}_R$-eigenvalues are the H-eigenvalues defined in [37].

Remark 2.3. Theoretical properties of mode-1 eigenvalues of tensors such as the Perron–Frobenius theory [6, 13, 47, 48] for nonnegative tensors can be parallelly developed for mode-k eigenvalues. However, as Horn and Johnson indicated in [18], "One should not dismiss left eigenvectors as merely a parallel theoretical alternative to right eigenvectors. Each type of eigenvector can convey different information about a matrix"; we believe that mode-1 through mode-m eigenpairs can convey different information about a general tensor of order $m \geq 3$.

In the rest of this section, we will obtain an upper bound for the number of equivalence classes of mode-k eigenpairs. As shown in Definition 2.1, Remark 2.1, and Remark 2.2, the number of equivalence classes of mode-k generalized eigenpairs for general tensors $\mathcal{A} \in \mathbb{C}^{[m,n]}$ and $\mathcal{B} \in \mathbb{C}^{[m',n]}$ is equivalent to the number of solutions to the following system of polynomials:

$$T(\lambda, x) = \begin{pmatrix} (\mathcal{A}^{(k)}x^{m-1})_1 - \lambda(\mathcal{B}x^{m'-1})_1 \\ \vdots \\ (\mathcal{A}^{(k)}x^{m-1})_n - \lambda(\mathcal{B}x^{m'-1})_n \\ a_1x_1 + a_2x_2 + \cdots + a_nx_n + b \end{pmatrix} = 0, \tag{2.9}$$

where λ and $x := (x_1, \ldots, x_n)^T$ are the unknowns, and $a_1, \ldots, a_n, b$ are random complex numbers. This motivates us to use Bernstein's theorem and its extensions in the field of solving polynomial systems (see [2, 30]) to study the number of equivalence classes of eigenpairs.

To initiate our discussion, we first introduce some commonly used notation and definitions. Let $P(x) := (p_1(x), \ldots, p_n(x))^T$ be a polynomial system with $x :=$

$(x_1,\dots,x_n)^T$. For $\alpha := (\alpha_1,\dots,\alpha_n) \in (\mathbb{Z}^n_{\geq 0})^T$, write $x^\alpha := x_1^{\alpha_1}\cdots x_n^{\alpha_n}$ and denote $|\alpha| = \alpha_1 + \cdots + \alpha_n$. Then $P(x)$ can be denoted by

$$P(x) := \begin{pmatrix} p_1(x) := \sum\limits_{\alpha\in S_1} c_{1,\alpha}x^\alpha \\ \vdots \\ p_n(x) := \sum\limits_{\alpha\in S_n} c_{n,\alpha}x^\alpha \end{pmatrix}, \tag{2.10}$$

where $S_1,\dots,S_n$ are given finite subsets of $(\mathbb{Z}^n_{\geq 0})^T$ and $c_{i,\alpha} \in \mathbb{C}^* := \mathbb{C}\backslash\{0\}$ are given coefficients of the corresponding monomials. Here for each $i = 1,\dots,n$, S_i is called the support of $p_i(x)$ and its convex hull $R_i := \mathrm{conv}(S_i)$ in $\mathbb{R}^n$ is called the Newton polytope of $p_i(x)$. $(S_1,\dots,S_n)$ is called the support of $P(x)$. For nonnegative variables $\lambda_1,\dots,\lambda_n$, let $\lambda_1R_1 + \cdots + \lambda_nR_n$ be the *Minkowski* sum of $\lambda_1R_1,\dots,\lambda_nR_n$, i.e.,

$$\lambda_1R_1 + \cdots + \lambda_nR_n := \{\lambda_1r_1 + \cdots + \lambda_nr_n \,|\, r_i \in R_i, i = 1,\dots,n\}.$$

The n-dimensional volume of $\lambda_1R_1+\cdots+\lambda_nR_n$, denoted by $\mathrm{Vol}_n(\lambda_1R_1+\cdots+\lambda_nR_n)$, is a homogeneous polynomial function of degree n in $\lambda_1,\dots,\lambda_n$ (see, for example, Proposition 4.9 of [9] for a proof). The coefficient of the monomial $\lambda_1\lambda_2\dots\lambda_n$ in $\mathrm{Vol}_n(\lambda_1R_1+\cdots+\lambda_nR_n)$ is called the mixed volume of $R_1,\dots,R_n$, denoted by $\mathrm{MV}_n(R_1,\dots,R_n)$, or the mixed volume of the supports $S_1,\dots,S_n$, denoted by $\mathrm{MV}_n(S_1,\dots,S_n)$. Sometimes 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.

THEOREM 2.1 (Bernstein's theorem [2]). *The number of isolated zeros in $(\mathbb{C}^*)^n$, counting multiplicities, of a polynomial system $P(x) = (p_1(x),\dots,p_n(x))^T$ with supports $S_1,\dots,S_n$ is bounded by the mixed volume $\mathrm{MV}_n(S_1,\dots,S_n)$. Moreover, for generic choices of the coefficients in p_i, the number of isolated zeros is exactly $\mathrm{MV}_n(S_1,\dots,S_n)$.*

An unexpected limitation of Theorem 2.1 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.

THEOREM 2.2 (see [30]). *The number of isolated zeros in $\mathbb{C}^n$, counting multiplicities, of a polynomial system $P(x) = (p_1(x),\dots,p_n(x))^T$ with supports $S_1,\dots,S_n$ is bounded by the mixed volume $\mathrm{MV}_n(S_1\cup\{0\},\dots,S_n\cup\{0\})$.*

The following lemma was given as Exercise 7 on p. 338 of [9].

LEMMA 2.1. *Consider a polynomial system $P(x) = (p_1(x),\dots,p_n(x))^T$ with supports $S_1 = S_2 = \cdots = S_n = S$. Then*

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

An upper bound for the number of equivalence classes of mode-k eigenpairs which generalizes results in [5, 35, 37] is given in the following theorem.

THEOREM 2.3. *Let $\mathcal{A} \in \mathbb{C}^{[m,n]}$ and $\mathcal{B} \in \mathbb{C}^{[m',n]}$. Assume that $\mathcal{B}x^{m'}$ is not identically zero as a function of x. Let k be an integer such that $1 \leq k \leq m$. Assume that $\mathcal{A}$ has finitely many equivalence classes of mode-k $\mathcal{B}$-eigenpairs over $\mathbb{C}$.*

(i) *If $m = m'$, then the number of equivalence classes of mode-k $\mathcal{B}$-eigenpairs, counting multiplicities, is bounded by*

$$n(m-1)^{n-1}.$$

If $\mathcal{A}$ and $\mathcal{B}$ are generic tensors, then $\mathcal{A}$ has exactly $n(m-1)^{n-1}$ equivalence classes of mode-k $\mathcal{B}$-eigenpairs.

(ii) *If $m \neq m'$, then the number of equivalence classes of mode-k $\mathcal{B}$-eigenpairs, counting multiplicities, is bounded by*

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

If $\mathcal{A}$ and $\mathcal{B}$ are generic tensors, then $\mathcal{A}$ has exactly $((m-1)^n-(m'-1)^n)/(m-m')$ equivalence classes of mode-k $\mathcal{B}$-eigenpairs.

Proof. Recall that the number of equivalence classes of mode-k $\mathcal{B}$-eigenpairs of $\mathcal{A}$ is equal to the number of solutions of (2.9). For the random hyperplane $a_1x_1+\cdots+a_nx_n+b=0$ in (2.9), without loss of generality, suppose that $a_n \neq 0$. Then x_n can be solved as

$$x_n = c_1x_1+\cdots+c_{n-1}x_{n-1}+d, \tag{2.11}$$

where $c_i=-a_i/a_n$ for $i=1,\ldots,n-1$ and $d=-b/a_n$. Notice that the number of solutions of (2.9) 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 (2.11) into the first n equations of (2.9). Denote the corresponding supports of T^* by $S_1,\ldots,S_n$. We claim that

$$\mathrm{MV}_n(S_1\cup\{0\},\ldots,S_n\cup\{0\}) \leq \begin{cases} n(m-1)^{n-1}, & m=m', \\ \dfrac{(m-1)^n-(m'-1)^n}{m-m'}, & m\neq m'. \end{cases} \tag{2.12}$$

Let N denote the number of equivalence classes of mode-k $\mathcal{B}$-eigenpairs of $\mathcal{A}$ over $\mathbb{C}$. Then (2.12) implies that

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

for $m=m'$ and

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

for $m\neq m'$. When $\mathcal{A}$ and $\mathcal{B}$ are generic, equality holds in the two inequalities above by using Theorems 2.1 and 2.2.

To prove (2.12), let $\bar{\mathcal{A}}\in\mathbb{C}^{[m,n]}$ and $\bar{\mathcal{B}}\in\mathbb{C}^{[m',n]}$ be generic tensors. Similar to (2.9) the corresponding polynomial system to solve is

$$\bar{T}(\lambda,x)=\begin{pmatrix} (\bar{\mathcal{A}}^{(k)}x^{m-1})_1-\lambda(\bar{\mathcal{B}}x^{m'-1})_1 \\ \vdots \\ (\bar{\mathcal{A}}^{(k)}x^{m-1})_n-\lambda(\bar{\mathcal{B}}x^{m'-1})_n \\ a_1x_1+a_2x_2+\cdots+a_nx_n+b \end{pmatrix}=0. \tag{2.13}$$

Substituting (2.11) into the first n equations of (2.13) yields a new system

$$\bar{T}^*(\lambda,x_1,\ldots,x_{n-1})=0.$$

The coefficients of the polynomials in the new system $\bar{T}^*$ are the sums of products of certain coefficients of the old system $\bar{T}$. For example, when $k=1$, the coefficient of x_1^{m-1} in the first polynomial of the new system $\bar{T}^*$ is

$$\sum_{i=0}^{m-1}\left(\sum_{\sigma\in I_i}a_{1,\sigma}\right)c_1^i, \tag{2.14}$$

where I_i is the set of all permutations of the set consisting of i numbers of n and $m-1-i$ numbers of 1. Since $a_{1,\sigma}$'s and c_1 are all generic, the coefficient (2.14) is nonzero. Similarly, all other coefficients in the new system are nonzero. Now letting $\bar{S}_1, \ldots, \bar{S}_n$ be the corresponding supports of $\bar{T}^*$, we can assume that all monomials

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

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 (2.13). Therefore, after substituting (2.11) into the first n equations of (2.13), 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 be contained in each equation of $\bar{T}^*$. This implies that $\bar{S}_1, \ldots, \bar{S}_n$ are all equal to

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

Notice that the convex hull of the set $\{\alpha \in (\mathbb{Z}_{\geq 0}^{n-1})^T \,\big|\, |\alpha| \leq m-1\}$ is the $(n-1)$-simplex in R^{n-1} with vertices $(0,0,\ldots,0), (m-1,0,\ldots,0), \ldots, (0,\ldots,0,m-1)$, and the convex hull of the set $\{\alpha \in (\mathbb{Z}_{\geq 0}^{n-1})^T \,\big|\, |\alpha| \leq m'-1\}$ is the $(n-1)$-simplex in R^{n-1} with vertices $(0,0,\ldots,0), (m'-1,\bar{0},\ldots,0), \ldots, (0,\ldots,0,m'-1)$. Thus, their volumes are $(m-1)^{n-1}/(n-1)!$ and $(m'-1)^{n-1}/(n-1)!$, respectively (see, for example, Exercises 2 and 3 on p. 307 of [9]). Let $\bar{Q}$ be the convex hull of $\bar{S}$. Then $\bar{Q}$ is the linear interpolation between the two aforementioned simplices with

$$\bar{Q} = \{(t, \mathrm{conv}(\alpha)) \,\big|\, t \in [0,1], \alpha \in (\mathbb{Z}_{\geq 0}^{n-1})^T, |\alpha| \leq m-1+t(m'-m)\}.$$

Since for fixed t, $\mathrm{conv}(\{\alpha \in (\mathbb{Z}_{\geq 0}^{n-1})^T \,\big|\, |\alpha| \leq m-1+t(m'-m)\})$ is a simplex with volume $(m-1+t(m'-m))^{n-1}/(n-1)!$, we have

$$\mathrm{Vol}_n(\bar{Q}) = \int_0^1 \frac{(m-1+(m'-m)t)^{n-1}}{(n-1)!} dt = \begin{cases} \dfrac{(m-1)^n - (m'-1)^n}{(m-m')n!}, & m \neq m', \\ \dfrac{n(m-1)^{n-1}}{n!}, & m = m'. \end{cases}$$

Therefore, by Lemma 2.1,

$$\mathrm{MV}_n(\bar{S}_1, \ldots, \bar{S}_n) = n! \mathrm{Vol}_n(\bar{Q}) = \begin{cases} \dfrac{(m-1)^n - (m'-1)^n}{m-m'}, & m \neq m', \\ n(m-1)^{n-1}, & m = m'. \end{cases}$$

When tensors $\mathcal{A}$ and $\mathcal{B}$ in (2.9) have some zero entries, each support S_i of T^* must be a subset of the support $\bar{S}_i$ of $\bar{T}^*$, i.e., $S_i \subset \bar{S}_i$. In addition, since each polynomial of $\bar{T}^*$ contains a constant term (which arises when substituting (2.11) into the term x_n^{m-1} in the first n polynomials of $\bar{T}$), $0 \in \bar{S}_i$. Thus we have $S_i \cup \{0\} \subset \bar{S}_i$. Since mixed volume is monotonic [4],

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

This implies that (2.12) holds. ∎

Remark 2.4. A few remarks about Theorem 2.3 follow:

(i) Theorem 2.3 provides an upper bound for the number of equivalence classes of $\mathcal{B}$-eigenpairs of tensor $\mathcal{A}$ if $\mathcal{A}$ has finitely many $\mathcal{B}$-eigenpairs. The bound is tight when $\mathcal{A}$ and $\mathcal{B}$ are generic. In the literature, the numbers of eigenvalues as defined in (2.8) and E-eigenpairs have been investigated.
When $m = m'$ and $\mathcal{B}$ is the mth order, n-dimensional identity tensor, Qi [37, Theorem 1] and Chang, Qi, and Zhang [8, Remark 1] proved that a tensor $\mathcal{A} \in \mathbb{R}^{[m,n]}$ has $n(m-1)^{n-1}$ eigenvalues (2.8). In this case, part (i) of Theorem 2.3 gives $n(m-1)^{n-1}$ as the upper bound for the number of equivalence classes of such eigenpairs.
When $m' = 2$ and $\mathcal{B}$ is the $n \times n$ identity matrix, Cartwright and Sturmfels [5, Theorem 1.2] prove that tensor $\mathcal{A}$ has $((m-1)^n - 1)/(m-2)$ equivalent classes of E-eigenpairs if it has finitely many E-eigenpairs. Part (ii) of our Theorem 2.3 gives $((m-1)^n - 1)/(m-2)$ as the upper bound of the number of equivalence classes of E-eigenpairs if $\mathcal{A}$ has finitely many E-eigenpairs.

(ii) The upper bound given in Theorem 2.3 can be highly useful in designing effective homotopy methods for computing mode-k generalized eigenpairs. In fact, the homotopy method described in Algorithm 3.1 for the case $m = m'$ relies on the bound $n(m-1)^{n-1}$.

3. Computing complex tensor eigenpairs via homotopy methods. Consider $\mathcal{A} \in \mathbb{C}^{[m,n]}$ and $\mathcal{B} \in \mathbb{C}^{[m',n]}$. As discussed in section 2, the problem of computing mode-k $\mathcal{B}$-eigenpairs of $\mathcal{A}$ in (2.3) or (2.4) is equivalent to the problem of solving (2.9), and if $m \neq m'$, normalize (λ, x) to satisfy that $\mathcal{B}x^{m'} = 1$. Since (2.9) is a polynomial system, we consider use of a homotopy continuation method to numerically solve it.

The basic idea of using a homotopy continuation method to solve a general polynomial system $P(x) = (p_1(x), \ldots, p_n(x))^T = 0$ as defined in (2.10) is to first deform $P(x) = 0$ to another polynomial system $Q(x) = 0$ that is easy to solve. Specifically, we construct a homotopy $H : \mathbb{C}^n \times [0,1] \to \mathbb{C}^n$ such that $H(x,0) = Q(x)$ and $H(x,1) = P(x)$. Then under certain conditions, the homotopy $H(x,t) = 0$ has smooth solution paths parameterized by t for $t \in [0,1)$ and all the isolated solutions of $P(x) = 0$ can be reached by tracing these paths.

A useful homotopy is the linear homotopy (see [26, 32, 43, 45]):

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

where γ is a generic nonzero complex number. It is very critical to choose a suitable $Q(x)$ such that the system $Q(x) = 0$ is easy to solve and all isolated solutions of $P(x) = 0$ can be found by tracing solution curves of $H(x,t) = 0$.

One choice of $Q(x)$ that always makes the linear homotopy (3.1) work is the so-called total degree homotopy, in which the starting system $Q(x) = 0$ has $deg = d_1 \times d_2 \times \cdots \times d_n$ solutions [26, 32, 45], where $d_1, \ldots, d_n$ are the degrees of polynomials $p_1(x), \ldots, p_n(x)$, respectively; deg is called the total degree or Bézout's number. By tracking deg number of solution paths of (3.1) we can find all the isolated solutions of $P(x) = 0$. However, most polynomial systems in applications usually have far fewer than deg solutions. In this case, many of the deg paths will diverge to infinity as $t \to 1$, resulting in huge wasteful computations.

The polyhedral homotopy [22] based on Bernstein's theorem [2] makes significant progress in this sense. In this method, the number of paths that need to be traced is the mixed volume of a polynomial system, which generally provides a much tighter bound than Bézout's number for the number of isolated zeros of a polynomial system.

Hence the new method reduces a significant amount of extraneous paths than the total degree homotopy in most occasions and thereby is much more efficient. However, the polyhedral homotopy includes two major stages: mixed volume computation and tracking paths. The computation of mixed volumes is a sophisticated procedure [3]. Moreover, mixed volume computation can be very expensive for large polynomial systems. Thus if the mixed volume is far less than the Bézout's number and an appropriate linear homotopy can be constructed so that only a mixed volume number of paths need to be traced, the system is better solved by using a linear homotopy instead of the polyhedral homotopy.

To compute tensor eigenpairs, one can certainly use the polyhedral homotopy implemented in HOM4PS [25], PHCpack [44], PHoM [14], PSOLVE [49] (which is a MATLAB implementation of HOM4PS), or the total degree homotopy implemented in Bertini [3]. However, using these methods to solve (2.9) or (2.4) directly does not take advantage of the special structures of a tensor eigenproblem. We will introduce two homotopy-type algorithms here that utilize such structures.

3.1. A linear homotopy method when $m = m'$. Theorem 2.3 gives us that the mixed volume of (2.9) when $m = m'$ is $n(m-1)^{n-1}$, which is far less than the Bézout's number, m^n. We consider constructing a linear homotopy in which only the mixed volume number of paths are traced.

For a polynomial system $P(x) = (p_1(x), \ldots, p_n(x))^T$ as defined in (2.10), where $x = (x_1, \ldots, x_n)$. Partition the variables $x_1, \ldots, x_n$ into k groups $y_1 = (x_1^{(1)}, \ldots, x_{l_1}^{(1)})$, $y_2 = (x_1^{(2)}, \ldots, x_{l_2}^{(2)})$, $\ldots$, $y_k = (x_1^{(k)}, \ldots, x_{l_k}^{(k)})$ with $l_1 + \cdots + l_k = n$. Let d_{ij} be the degree of p_i with respect to y_j for $i = 1, \ldots, n$ and $j = 1, \ldots, k$. Then the multihomogeneous Bézout's number of $P(x)$ with respect to $(y_1, \ldots, y_k)$ is the coefficient of $\alpha_1^{l_1}\alpha_2^{l_2}\ldots\alpha_k^{l_k}$ in the product

$$\prod_{i=1}^{n}(d_{i1}\alpha_1 + \cdots + d_{ik}\alpha_k).$$

The following theorem will play an important role in constructing a proper linear homotopy.

THEOREM 3.1 (see [43]). *Let $Q(x)$ be a system of polynomials chosen to have the same multihomogeneous form as $P(x)$ with respect to a certain partition of the variables $(x_1, \ldots, x_n)$. Assume $Q(x) = 0$ has exactly the multihomogeneous Bézout's number $\mathcal{N}$ of nonsingular solutions with respect to this partition. Then for almost all $\gamma \in \mathbb{C}^*$, the homotopy*

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

has $\mathcal{N}$ nonsingular solution paths on $t \in [0,1)$ whose endpoints as $t \to 1$ include all the isolated solutions of $P(x) = 0$.

For (2.9), when $m = m'$ the polynomial system

$$G(\lambda, x) = \begin{pmatrix} (\mathcal{A}^{(k)}x^{m-1})_1 - \lambda(\mathcal{B}x^{m-1})_1 \\ \vdots \\ (\mathcal{A}^{(k)}x^{m-1})_n - \lambda(\mathcal{B}x^{m-1})_n \\ a_1x_1 + a_2x_2 + \cdots + a_nx_n + b \end{pmatrix} = 0 \tag{3.2}$$

needs to be solved, where λ and $x := (x_1, \ldots, x_n)^T$ are the unknowns, and $a_1, \ldots, a_n, b$ are random complex numbers. Consider the starting system

$$Q(\lambda, x) = \begin{pmatrix} (\lambda - \mu_1)(x_1^{m-1} - \beta_1) \\ (\lambda - \mu_2)(x_2^{m-1} - \beta_2) \\ \vdots \\ (\lambda - \mu_n)(x_n^{m-1} - \beta_n) \\ c_1 x_1 + \ldots c_n x_n + d \end{pmatrix} = 0, \tag{3.3}$$

where μ_i, β_i, c_i for $i = 1, \ldots, n$ and d are random nonzero complex numbers.

THEOREM 3.2. *Let $G(\lambda, x)$ and $Q(\lambda, x)$ be defined as* (3.2) *and* (3.3)*, respectively. Then all the isolated zeros (λ, x) in $\mathbb{C}^{n+1}$ of $G(\lambda, x)$ can be found by using the homotopy*

$$H(\lambda, x, t) = (1 - t)\gamma Q(\lambda, x) + tG(\lambda, x) = 0, \quad t \in [0, 1], \tag{3.4}$$

for almost all $\gamma \in \mathbb{C}^*$.

Proof. It is sufficient to verify that $Q(\lambda, x)$ satisfies all the assumptions of Theorem 3.1. Partition the variables $(\lambda, x_1, \ldots, x_n)$ into two groups: (λ) and $(x_1, \ldots, x_n)$; we can easily see that each of the first polynomial equations in (3.2) and (3.3) has degree 1 in (λ) and degree $m - 1$ in $(x_1, \ldots, x_n)$, and the last equation in both (3.2) and (3.3) has degree 0 in (λ) and degree 1 in $(x_1, \ldots, x_n)$. Hence (3.2) and (3.3) have the same multihomogeneous Bézout's number as the coefficient of $\alpha_1 \alpha_2^n$ in the product

$$[1 \cdot \alpha_1 + (m - 1)\alpha_2]^n (0 \cdot \alpha_1 + 1 \cdot \alpha_2).$$

It can be easily computed that this coefficient is equal to

$$\binom{n}{1} (m - 1)^{n-1} = n(m - 1)^{n-1}.$$

Hence (3.2) and (3.3) have the same multihomogeneous Bézout's number $n(m-1)^{n-1}$ with respect to the partition (λ) and $(x_1, \ldots, x_n)$.

We now show that $Q(\lambda, x)$ in (3.3) has exactly $n(m - 1)^{n-1}$ zeros. Notice that if λ is equal to none of $\mu_1, \ldots, \mu_n$, then we end up with a system of $n + 1$ equations and n unknowns, which has no solutions. Thus λ must be equal to one of $\mu_1, \ldots, \mu_n$. Without loss of generality, assume that $\lambda = \mu_1$. Then $x_1, \ldots, x_n$ can be determined by

$$\begin{aligned} x_i^{m-1} - \beta_i = 0, \quad i = 2, \ldots, n, \\ c_1 x_1 + \ldots c_n x_n + d = 0. \end{aligned}$$

Obviously, each x_i for $i = 2, \ldots, n$ can be chosen as one of the $(m - 1)$th root of β_i and x_1 will be solved by substituting the chosen $x_2, \ldots, x_n$ into the last hyperplane equation. So there are $(m-1)^{n-1}$ solutions corresponding to $\lambda = \mu_1$. This argument holds for λ being any μ_i. Therefore, there are totally $n(m - 1)^{n-1}$ solutions.

It remains to prove that each solution of $Q(\lambda, x) = 0$ in (3.3) is nonsingular. As discussed above, any solution (λ^*, x^*) of (3.3) satisfies

$$\begin{aligned} \lambda^* = \mu_i, \\ (x_j^*)^{m-1} - \beta_j = 0, \quad j = 1, \ldots, i - 1, i + 1, \ldots, n, \\ c_1 x_1^* + \cdots + c_n x_n^* + d = 0. \end{aligned} \tag{3.5}$$

Let $DQ(\lambda, x)$ be the Jacobian of $Q(\lambda, x)$ with respect to (λ, x). It is sufficient to show that $DQ(\lambda^*, x^*)$ is nonsingular. Denote

$$A_j(\lambda, x) := x_j^{m-1} - \beta_j, \quad B_j(\lambda, x) := (\lambda - \mu_j)(m-1)x_j^{m-2}$$

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

$$DQ(\lambda, x) = \begin{pmatrix} A_1 & B_1 & & & & & & \\ \vdots & & \ddots & & & & & \\ A_{i-1} & & & B_{i-1} & & & & \\ A_i & & & & B_i & & & \\ A_{i+1} & & & & & B_{i+1} & & \\ \vdots & & & & & & \ddots & \\ A_n & & & & & & & B_n \\ 0 & c_1 & \ldots & c_{i-1} & c_i & c_{i+1} & \ldots & c_n \end{pmatrix}.$$

Note that $A_j(\lambda^*, x^*) = (x_j^*)^{m-1} - \beta_j = 0, \quad j \neq i$ and $B_i(\lambda^*, x^*) = (\lambda^* - \mu_i)(m-1)(x_i^*)^{m-2} = (\mu_i - \mu_i)(m-1)(x_i^*)^{m-2} = 0$ by (3.5). For simplicity, write $A_j^* := A_j(\lambda^*, x^*)$ and $B_j^* := B_j(\lambda^*, x^*)$. Then

$$DQ(\lambda^*, x^*) = \begin{pmatrix} 0 & B_1^* & & & & & & \\ \vdots & & \ddots & & & & & \\ 0 & & & B_{i-1}^* & & & & \\ A_i^* & & & & 0 & & & \\ 0 & & & & & B_{i+1}^* & & \\ \vdots & & & & & & \ddots & \\ 0 & & & & & & & B_n^* \\ 0 & c_1 & \ldots & c_{i-1} & c_i & c_{i+1} & \ldots & c_n \end{pmatrix}.$$

Then

$$\det(DQ(\lambda^*, x^*)) = (-1)^{i+1} A_i^* (-1)^{n+i} c_i \prod_{j \neq i} B_j^* \neq 0$$

by (3.5). ∎

Theorem 3.2 suggests that (3.4) can be used to solve (2.9) in the case of $m = m'$. For simplicity, write $u := (\lambda, x)$. In order to improve numerical stability, we apply the transformation $s = \ln t$ to (3.4) (a strategy first suggested in [14]) and obtain the new homotopy as

$$\bar{H}(u, s) = (1 - e^s)\gamma Q(u) + e^s G(u) = 0, \quad s \in [-\infty, 0], \tag{3.6}$$

where $Q(u) = Q(\lambda, x)$ and $G(u) = G(\lambda, x)$ are defined in (3.3) and (3.2), respectively.

We now introduce our linear homotopy method for computing mode-k generalized eigenpairs when $m = m'$.

ALGORITHM 3.1 (compute complex mode-k $\mathcal{B}$-eigenpairs of $\mathcal{A}$, where $\mathcal{A}, \mathcal{B} \in \mathbb{C}^{[m,n]}$).

Step 1. *Compute all solutions of $Q(u)$ as defined in* (3.3).

Step 2. *Path following. Follow the paths from $s = -\infty$ to $s = 0$. In reality, we certainly cannot start from $s = -\infty$. In this case, one can choose a very negative s_0 and obtain a starting point by using Newton's iterations:*

$$w^{(k+1)} = w^{(k)} - [\bar{H}_u(w^{(k)}, s_0)]^{-1}\bar{H}(w^{(k)}, s_0), \quad k = 0, 1, \ldots,$$

until $\|\bar{H}(w^{(N)}, s_0)\|$ is very small for some N. Here $w^{(0)}$ is a solution of $Q(u) = 0$. Let $u_0 := u(s_0)$ and take $u_0 = w^{(N)}$. Then path following can be triggered.

Path following is done using the prediction-correction method. Letting $(u_k, s_k) := (u(s_k), s_k)$, to find the next point on the path $\bar{H}(u, s) = 0$, we employ the following strategy:

(i) *Prediction step: Compute the tangent vector $\frac{du}{ds}$ to $\bar{H}(u, s) = 0$ at s_k by solving the linear system*

$$\bar{H}_u(u_k, s_k)\frac{du}{ds} = -\bar{H}_s(u_k, s_k)$$

for $\frac{du}{ds}$. Then compute the approximation $\tilde{u}$ to u_{k+1} by

$$\tilde{u} = u_k + \Delta s\frac{du}{ds}, \quad s_{k+1} = s_k + \Delta s,$$

where Δs is a stepsize.

(ii) *Correction step: Use Newton's iterations. Initialize $v_0 = \tilde{u}$. For $i = 0, 1, 2, \ldots$, compute*

$$v_{i+1} = v_i - [\bar{H}_u(v_i, s_{k+1})]^{-1}\bar{H}(v_i, s_{k+1})$$

until $\|H(v_J, s_J)\|$ is very small. Then let $u_{k+1} = v_J$.

Step 3. *End game. During Step 3 when s_N is very close to 0, the corresponding u_N should be very close to a zero u^* of $G(u) = G(\lambda, x)$. So Newton's iterations*

$$u^{(k+1)} = u^{(k)} - [DG(u^{(k)})]^{-1}G(u^{(k)}), \quad k = 0, 1, \ldots,$$

will be employed to refine our final approximation $\tilde{u}$ to u^*. If $DG(u^*)$ is nonsingular, then $\tilde{u}$ will be a very good approximation of u^* with multiplicity 1. If $DG(u^*)$ is singular, $\tilde{u}$ is either an isolated singular zero of $G(u)$ with some multiplicity $l > 1$ or in a positive dimensional solution component of $G(u) = 0$. We use a strategy provided in Chapter VIII *of* [26] *(see also* [43]*) to verify whether $\tilde{u}$ is an isolated zero with multiplicity $l > 1$ or in a positive dimensional solution component of $G(u) = 0$.*

Step 4. *For each solution $u = (\lambda, x)$ obtained in Step 3, normalize x in the following fashion to get a new eigenvector:*

$$y = \frac{x}{x_{i_0}}, \tag{3.7}$$

where $i_0 := \arg\max_{1 \le i \le n} |x_i|$. Hence (λ, y) is an eigenpair.

Remark 3.1. A few remarks about Algorithm 3.1 follow:

(i) As defined in (2.3) and Remark 2.1, if (λ, x) is an eigenpair, (λ, tx) for $t \neq 0$ is also an eigenpair. Therefore, Step 4 is well defined in this sense.

(ii) Notice that if x is a real eigenvector associated with a real eigenvalue λ, tx for any $t \in \mathbb{C}\backslash\{0\}$ will be a complex eigenvector associated with the same eigenvalue λ. If in any case a complex eigenvector like tx is obtained in Step 3 of Algorithm 3.1, applying (3.7) to tx will give us a new real eigenvector. In this sense, Step 4 is very helpful for us to detect real eigenpairs.

(iii) According to Theorem 2.3, 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 $n(m-1)^{n-1}$. Moreover, this bound is attained when $\mathcal{A}$ and $\mathcal{B}$ are generic. This result implies that the optimal number of paths to follow in a homotopy method for solving the system (3.2) is $n(m-1)^{n-1}$. Our starting system (3.3) has exactly $n(m-1)^{n-1}$ nonsingular solutions. In this sense, Algorithm 3.1 follows the optimal number of paths for solving the system (3.2).

3.2. A polyhedral homotopy method when $m \neq m'$. To compute mode-k generalized tensor eigenpairs when $m \neq m'$, we use the equivalence class structure of the eigenproblem as described in Remark 2.1. We first solve (2.9) to find a representative (λ, x) from each equivalence class and then find all m' eigenpairs from each equivalence class by simply using $\lambda' = t^{m-m'}\lambda, x' = tx$, where t is a root of $t^{m'} = 1$. We use a polyhedral homotopy method to solve the system (2.9). In our implementation, the polyhedral homotopy method is PSOLVE [49], with some modifications, as described in subsection 3.3. The modifications are based on strategies 2 and 3 introduced in the next subsection.

One may think of solving (2.4) directly to get m' eigenpairs from each equivalence class. However, this alternative method would have to follow m' times as many paths as the approach we described in the previous paragraph and therefore it would need much more computation.

Now we present our algorithm for computing mode-k generalized eigenpairs when $m \neq m'$.

ALGORITHM 3.2 (compute complex mode-k $\mathcal{B}$-eigenpairs of $\mathcal{A}$, where $\mathcal{A} \in \mathbb{C}^{[m,n]}$ and $\mathcal{B} \in \mathbb{C}^{[m',n]}$ with $m \neq m'$).

Step 1. *Use modified PSOLVE to get all solutions* (λ, x) *of* (2.9).

Step 2. *For each* (λ, x) *obtained in Step* 1*, if* $\mathcal{B}x^{m'} \neq 0$*, normalize it to get an eigenpair* (λ^*, x^*) *by*

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

to satisfy (2.4).

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

3.3. Implementation tips. By using random complex numbers in the formulation of homotopy, theoretically, with probability one the solution paths do not cross or go to infinity in the middle. In practice, however, two paths may become very close to each other and the magnitude of some components of a solution curve may become very large during the procedure of path tracking. This causes various numerical difficulties such as missing solutions or losing efficiency or stability. In our implementation of Algorithms 3.1 and 3.2, we use the following strategies to address these issues. We focus our discussion on Algorithm 3.1. Step 1 of Algorithm 3.2 is done similarly.

When tracing two paths that are sufficiently close, it is possible for the path tracing algorithm to jump from one path to the other path and thus result in the missing of zeros. To minimize the chance for curve jumping and keep the efficiency, our *first strategy* is as follows: *The stepsize* Δs *in Step* 2 *of* Algorithm 3.1 *is chosen*

adaptively. Initially, set $s_0 = -20(n+1)$, where $n+1$ is the number of unknown variables $\lambda, x_1, \ldots, x_n$ in (2.9) or (3.2), and $\Delta s = -s_0/3$. Similar to [25], if more than three steps of Newton iterations were required to converge within the desired accuracy, Δs is halved and the shorter step is attempted. On the other hand, if several (the default being 2) consecutive steps were not cut, Δs is doubled, up to a prescribed maximum value (the default being $-s_k/3$).

Although this adaptive approach can often significantly reduce the possibility of curve jumping, it can still occur in some cases. Our *second strategy* is as follows: *To check if there is curve jumping, we store all the found solutions in a binary search tree. Each time 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 greater than a threshold (the default being 10^{10}), we consider that curve jumping likely has occurred. We then retrace the two associated curves with more restrictively chosen parameters in the projective space, as described in the paragraph after the next one.*

When the magnitude of some components of certain solution curves become very large in the middle, 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, [43]). However, empirically it is more time-consuming to follow all paths in the projective space than in the complex space. In our implementation of Algorithms 3.1 and 3.2, our *third strategy* is as follows: *We retrace solution curves in the projective space only for those paths that are detected to have very large solution components.*

To trace a path in the projective space, we first homogenize each polynomial equation of (3.6) in the variables $\lambda, x_1, \ldots, x_n$ to get the homotopy

$$\hat{H}(\lambda, \hat{x}, s) = (1-e^s)\gamma \begin{pmatrix} (\lambda - \mu_1 x_0)(x_1^{m-1} - \beta_1 x_0^{m-1}) \\ \vdots \\ (\lambda - \mu_n x_0)(x_n^{m-1} - \beta_n x_0^{m-1}) \\ c_1 x_1 + \ldots c_n x_n + d x_0 \end{pmatrix} + e^s \begin{pmatrix} x_0(\mathcal{A}^{(k)} x^{m-1})_1 - \lambda(\mathcal{B} x^{m-1})_1 \\ \vdots \\ x_0(\mathcal{A}^{(k)} x^{m-1})_n - \lambda(\mathcal{B} x^{m-1})_n \\ a_1 x_1 + a_2 x_2 + \cdots + a_n x_n + b x_0 \end{pmatrix} = 0, \tag{3.8}$$

where $\hat{x} = (x_0, x_1, \ldots, x_n)^T$, and then follow the solution curve of (3.8) in the projective space. Notice that in (3.8) 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.

4. Computing real tensor eigenpairs via homotopy methods. In some applications, tensor $\mathcal{A}$ is real and only real eigenpairs (or real eigenvalues) of $\mathcal{A}$ are of interest [12, 37]. In this situation, only real zeros of the polynomial system (2.4) or (3.2) are needed. It is worth noting that there is currently no effective method to find all real zeros for a polynomial system directly. One may think to use a homotopy

continuation method to trace only real zeros from the start system to the target system. However, this approach generally does not guarantee a real zero at the end, because the homotopy methods inherently have to trace paths in the complex space in order to avoid the discriminant locus.

For a real tensor $\mathcal{A}$, a real eigenvalue may have complex eigenvectors. Sometimes identifying which eigenvalues are real is the only concern. In this case, we can first compute complex zeros (λ, x) of (3.2) by Algorithm 3.1 or (2.4) by Algorithm 3.2, then identify the real eigenvalues by checking the size of the imaginary parts of λ's. Specifically, let (λ^*, x^*) be a computed eigenpair. If

$$|\operatorname{Im}(\lambda^*)| < \delta_0,$$

where δ_0 is a threshold for the imaginary part (the default value $\delta_0 = 10^{-8}$), then we take $\operatorname{Re}(\lambda^*)$ as a real eigenvalue.

Note that if $\frac{m'}{m-m'}$ is a nonzero integer multiple of 4 (for example, $m = 5, m' = 4$ or $m = 10, m' = 8$,) and $\mathcal{A}$ has an eigenpair (λ^*, x^*) with a purely imaginary eigenvalue $\lambda^* = bi$, where $b \in \mathbb{R}$, then one can easily show that $(b, (-i)^{1/(m-m')}x^*)$ and $(-b, i^{1/(m-m')}x^*)$ are eigenpairs with real eigenvalues. Therefore, when $\frac{m'}{m-m'}$ is a nonzero integer multiple of 4, if (λ^*, x^*) is an eigenpair found by Algorithm 3.2 such that

$$|\operatorname{Re}(\lambda^*)| < \delta_0,$$

then we take $\operatorname{Im}(\lambda^*)$ and $-\operatorname{Im}(\lambda^*)$ as real eigenvalues, with corresponding eigenvectors $(-i)^{1/(m-m')}x^*$ and $i^{1/(m-m')}x^*$.

When looking for real tensor eigenpairs (i.e., both eigenvalues and eigenvectors being real), the situation becomes more complicated. We use a two-step procedure. The first step is to get complex zeros (λ, x) of (3.2) by Algorithm 3.1 or (2.4) by Algorithm 3.2. The second step is to extract all real eigenpairs (λ, x) from the complex zeros obtained in the first step.

To facilitate the discussion, the following notation is introduced. For a vector $a = (a_1, \ldots, a_n)^T \in \mathbb{C}^n$, let

$$\operatorname{Im}(a) = (\operatorname{Im}(a_1), \ldots, \operatorname{Im}(a_n))^T, \quad \operatorname{Re}(a) = (\operatorname{Re}(a_1), \ldots, \operatorname{Re}(a_n))^T.$$

Suppose that (λ^*, x^*) is an eigenpair found in the first step. Consider two cases: (i) (λ^*, x^*) is an isolated eigenpair; (ii) (λ^*, x^*) is an eigenpair contained in a positive dimensional solution component of system (3.2) or (2.4).

When (λ^*, x^*) is an isolated eigenpair, take $(\operatorname{Re}(\lambda^*), \operatorname{Re}(x^*))$ as a real eigenpair if

$$\|\operatorname{Im}(\lambda^*, x^*)\|_2 < \delta_0.$$

If (λ^*, x^*) is an eigenpair in a positive dimensional solution component of system (3.2) or (2.4), in general real eigenvectors are not guaranteed to be found by Algorithm 3.1 or Algorithm 3.2 even if the corresponding eigenvalue λ^* is real. In this case, we will construct the Newton homotopy [1]

$$H(\lambda, x, t) := P(\lambda, x) - (1-t)P(\lambda^*, \operatorname{Re}(x^*)), \quad t \in [0, 1], \tag{4.1}$$

to follow curves in $(\lambda, x) \in \mathbb{R}^{n+1}$ in order to get a real eigenpair. Notice that when following curves in the complex space it is proved in [26] that the solution curves of (4.1) can be parameterized by t, but the solution curves of (4.1) are not necessarily to be a function of t when restricted in the real space. So a different method to follow

curves is needed. In this case parameterizing the solution curves by the arc length s is suggested in [27]. Instead of following paths using (4.1), we will use the homotopy

$$H((\lambda(s), x(s), t(s)) = 0. \tag{4.2}$$

For a description of the Newton homotopy method, we refer to [27].

An interesting phenomenon we have observed in our experiments is that in some cases, the real eigenpairs can be obtained more straightforwardly from the complex eigenpairs found from Algorithm 3.1 or Algorithm 3.2: If (λ^*, x^*) is in a positive dimensional solution component of (2.4) and $\lambda^* \in \mathbb{R}$, then $(\lambda^*, \mathrm{Re}(x^*)/(\mathcal{B}\,\mathrm{Re}(x^*))^{1/m'})$ and $(\lambda^*, \mathrm{Im}(x^*)/(\mathcal{B}\,\mathrm{Im}(x^*))^{1/m'})$ can be mode-k $\mathcal{B}_R$ eigenpairs of $\mathcal{A}$. This gives us a heuristic approach to find real eigenpairs for eigenpairs belonging to positive dimensional components. We remark that this approach works well for all the examples (e.g., Examples 4.8, 4.11, 4.13, 4.14) in [12] when a real Z-eigenvalue has infinitely many real Z-eigenvectors. The following proposition gives a justification for the approach in special cases.

PROPOSITION 4.1. *Let $\mathcal{A} \in \mathbb{R}^{[m,n]}$ and $\mathcal{B} \in \mathbb{R}^{[m',n]}$. Let k be an integer such that $1 \le k \le m$. Let $\lambda \in \mathbb{R}$ be a real mode-k $\mathcal{B}$ eigenvalue of $\mathcal{A}$. If $U := \{x \in \mathbb{C}^n \,|\, \mathcal{A}^{(k)}x^{m-1} = \lambda\mathcal{B}x^{m'-1}\}$ contains a complex linear subspace V of $\mathbb{C}^n$ such that $y \in V$ implies $\bar{y} \in V$, then for any $x = \xi + i\eta \in V$ such that $\xi, \eta \in \mathbb{R}^n$ and $\xi \neq 0, \eta \neq 0$,*

(i) *when $m = m'$, ξ and η are both real mode-k $\mathcal{B}$ eigenvectors of $\mathcal{A}$ associated with λ;*

(ii) *when $m \neq m'$, $\mathcal{B}\xi^{m'} \neq 0$, and $\mathcal{B}\eta^{m'} \neq 0$, the normalized vectors*

$$v := \frac{\xi}{(B\xi^{m'})^{1/m'}}, \quad w := \frac{\eta}{(B\eta^{m'})^{1/m'}}$$

are real mode-k $\mathcal{B}$ eigenvectors of $\mathcal{A}$ associated with λ.

Proof. Let $x \in V$. Then $\bar{x} \in V$. Since V is a linear subspace, $\xi = (x + \bar{x})/2$ and $\eta = (x - \bar{x})/(2i)$ are also in V. Thus, when $m = m'$, ξ and η are both real mode-k $\mathcal{B}$ eigenvectors of $\mathcal{A}$ associated with λ. If $m \neq m'$, we have

$$\begin{aligned}
\mathcal{B}v^{m'} &= \sum_{i_1,\cdots,i_{m'}=1}^{n} B_{i_1 i_2 \cdots i_{m'}} v_{i_1} v_{i_2} \cdots v_{i_{m'}} \\
&= \sum_{i_1,\cdots,i_{m'}=1}^{n} B_{i_1 i_2 \cdots i_{m'}} \frac{\xi_{i_1}}{(B\xi^{m'})^{1/m'}} \frac{\xi_{i_2}}{(B\xi^{m'})^{1/m'}} \cdots \frac{\xi_{i_{m'}}}{(B\xi^{m'})^{1/m'}} \\
&= \frac{\sum_{i_1,\cdots,i_{m'}=1}^{n} B_{i_1 i_2 \cdots i_{m'}} \xi_{i_1} \xi_{i_2} \cdots \xi_{i_{m'}}}{B\xi^{m'}} \\
&= \frac{B\xi^{m'}}{B\xi^{m'}} = 1.
\end{aligned}$$

This implies that v is a real mode-k $\mathcal{B}$ eigenvector of $\mathcal{A}$ associated with λ. Similarly, $Bw^{m'} = 1$ can also be verified. Therefore, w is also a real mode-k $\mathcal{B}$ eigenvector of $\mathcal{A}$ associated with λ. ∎

Remark 4.1. A natural question is when U defined in Proposition 4.1 contains a linear subspace V. Consider the case when $\mathcal{A}$ is a symmetric tensor with a low rank decomposition. For simplicity, consider $m = 3$. For vectors $a, b, c \in \mathbb{C}^n$, define the

outer product tensor $a \circ b \circ c = (a_i b_j c_k)$. Suppose that a symmetric tensor $\mathcal{A} \in \mathbb{R}^{[m,n]}$ can be decomposed as

$$\mathcal{A} = y_1 \circ y_1 \circ y_1 + y_2 \circ y_2 \circ y_2 + \cdots + y_r \circ y_r \circ y_r,$$

where $r < n$, $y_k \in \mathbb{R}^n, k = 1, 2, \ldots, r$.

Let $W = \operatorname{span}(y_1, y_2, \ldots, y_r)$ and let V be the orthogonal complement of W. Then V is a linear subspace of $\mathbb{C}^n$. Clearly, $x \in V$ implies $\bar{x} \in V$. Moreover, for any $x \in V \backslash \{0\}$,

$$Ax^{m-1} = 0.$$

Thus x is an eigenvector of $\mathcal{A}$ corresponding to the eigenvalue 0. In this case, $\operatorname{Re}(x)/\|\operatorname{Re}(x)\|$ and $\operatorname{Im}(x)/\|\operatorname{Im}(x)\|$ are real Z-eigenvectors of $\mathcal{A}$ corresponding to the real Z-eigenvalue 0. Hence, the set V is a desired linear subspace of $\mathbb{C}^n$ contained in U.

Finally we present an algorithm for computing real eigenpairs that combines the heuristic approach and the Newton homotopy method.

ALGORITHM 4.1 (compute real mode-k $\mathcal{B}$-eigenpairs of $\mathcal{A}$, where $\mathcal{A} \in \mathbb{R}^{[m,n]}$, $\mathcal{B} \in \mathbb{R}^{[m',n]}$).

Step 1. *Compute all complex eigenpairs using Algorithm* 3.1 *or Algorithm* 3.2. *Let K denote the set of found eigenpairs (λ, x) such that $|\operatorname{Im}(\lambda)| < \delta_0$.*

Step 2. *For each eigenpair $(\lambda^*, x^*) \in K$, if (λ^*, x^*) is in a positive dimensional solution component of* (3.2) *or* (2.4)*, go to Step* 3. *Otherwise, (λ^*, x^*) is an isolated eigenpair. If $\|\operatorname{Im}(x^*)\|_2 < \delta_0$, then take $(\operatorname{Re}(\lambda^*), \operatorname{Re}(x^*))$ as a real eigenpair and stop.*

Step 3. *Set $\lambda = \operatorname{Re}(\lambda^*)$. If $m = m'$, set $v := \operatorname{Re}(x^*)$ (if $\operatorname{Re}(x^*) \neq 0$) and $w := \operatorname{Im}(x^*)$ (if $\operatorname{Im}(x^*) \neq 0$); otherwise, set*

$$v := \frac{\operatorname{Re}(x^*)}{(\mathcal{B}\operatorname{Re}(x^*)^{m'})^{1/m'}} \quad (\text{if } \mathcal{B}\operatorname{Re}(x^*)^{m'} \neq 0)$$

and

$$w := \frac{\operatorname{Im}(x^*)}{(\mathcal{B}\operatorname{Im}(x^*)^{m'})^{1/m'}} \quad (\text{if } \mathcal{B}\operatorname{Im}(x^*)^{m'} \neq 0).$$

If (λ, v) or (λ, w) is a mode-k $\mathcal{B}$-eigenpair of $\mathcal{A}$, then we have obtained a real eigenpair and stop. Otherwise, goto Step 4.

Step 4. *Starting from (λ^*, x^*), use the Newton homotopy method to follow curves of* (4.2) *to find a real eigenpair.*

5. Numerical results. Based on the algorithms introduced in sections 3 and 4, a MATLAB package `TenEig` has been developed. The current version is `TenEig` 1.1. The numerical results reported in this paper were obtained using this version. The package can be downloaded from http://www.math.msu.edu/~chenlipi/TenEig.html. Consider the tensors $\mathcal{A} \in \mathbb{C}^{[m,n]}$ and $\mathcal{B} \in \mathbb{C}^{[m',n]}$. In the `TenEig` package, function `teig` can be used to compute the general mode-k $\mathcal{B}$ eigenvalues and eigenvectors of a tensor $\mathcal{A}$ for $m = m'$. The input of this function is tensor $\mathcal{A}$ or the polynomial form (if $\mathcal{A}$ is symmetric) $\mathcal{A}x^m$, tensor $\mathcal{B}$ (the default is the identity tensor),

mode k (the default value is 1); the output is mode-k $\mathcal{B}$ eigenvalues and eigenvectors of $\mathcal{A}$. By default, `teig` finds eigenvalues and eigenvectors for the eigenproblem (2.8).

The function `teneig` computes the general mode-k $\mathcal{B}$ eigenvalues and eigenvectors of a tensor $\mathcal{A}$ for $m \neq m'$. The input of this function is tensor $\mathcal{A}$ or the polynomial form (if $\mathcal{A}$ is symmetric) $\mathcal{A}x^m$, tensor $\mathcal{B}$, mode k (the default value is 1); the output is mode-k $\mathcal{B}$ eigenvalues and eigenvectors of $\mathcal{A}$. If $\mathcal{B}$ is chosen as the identity matrix, `teneig` computes the E-eigenvalues and E-eigenvectors of $\mathcal{A}$ as defined in Qi [37].

Since E-eigenpairs of a tensor are frequently needed, our package includes a separate function `eeig`, which only computes E-eigenpairs of a tensor.

The package also includes two functions `heig` and `zeig` to compute real eigenpairs of a tensor: The first one computes H-eigenpairs and the second one computes Z-eigenpairs.

In the next two subsections, numerical results are reported to illustrate the effectiveness and efficiency of our methods for computing tensor eigenpairs. All the numerical experiments were done on a Thinkpad T400 laptop with an Intel dual core CPU at 2.80 GHz and 2 GB of RAM, running on a Windows 7 operating system. The package `TenEig` was run using MATLAB 2013a. In our examples, we used `teig` or `teneig` to compute generalized eigenpairs, `teig` to compute eigenpairs (2.8), `eeig` to compute E-eigenvalues, `heig` to compute (real) H-eigenpairs, and `zeig` to compute (real) Z-eigenpairs, respectively.

5.1. Examples for computing complex eigenpairs. In this subsection, some numerical examples illustrating the performance of `TenEig` for computing complex tensor $\mathcal{B}$-eigenpairs are provided.

A numerical solver `NSolve`, based on the Gröbner basis, for solving systems of algebraic equations is provided by Mathematica. We will compare the performance of `TenEig` and `NSolve` in computing complex tensor $\mathcal{B}$-eigenpairs. Denote

$$\begin{aligned} T(m,n) &:= n(m-1)^{n-1}, \\ E(m,n) &:= ((m-1)^n - 1)/(m-2), \\ G(m,m',n) &:= ((m-1)^n - (m'-1)^n)/(m-m'). \end{aligned}$$

Recall that Theorem 2.3 explains that for tensors $\mathcal{A} \in \mathbb{C}^{[m,n]}$ and $\mathcal{B} \in \mathbb{C}^{[m',n]}$, the number of equivalence classes of isolated $\mathcal{B}$-eigenpairs of $\mathcal{A}$ is bounded by $T(m,n)$ for $m = m'$ and $G(m,m',n)$ for $m \neq m'$. In particular, Remark 2.4 states that the number of equivalence classes of isolated eigenpairs of the eigenproblem (2.8) is bounded by $T(m,n)$ and the number of equivalence classes of isolated E-eigenpairs of $\mathcal{A}$ is bounded by $E(m,n)$.

Example 5.1. In this example, we compare the performance of our `TenEig` with `NSolve` and `PSOLVE`. `teig`, `eeig`, `NSolve`, and `PSOLVE` are used to compute eigenpairs (2.8) or E-eigenpairs of a generic tensor $\mathcal{A} \in \mathbb{C}^{[m,n]}$. We note the following:

(a) `teig` is based on Algorithm 3.1. The polynomial system solved is (3.2).

(b) `eeig` is based on Algorithm 3.2. The polynomial system solved is (2.9).

(c) When computing eigenpairs (2.8), `NSolve` and `PSOLVE` solve the polynomial system defined by (3.2).

(d) When computing E-eigenpairs, `NSolve` and `PSOLVE` solve the polynomial system defined by (2.4).

The tensors $\mathcal{A}$ were generated using $randn(n,\ldots,n)+i*randn(n,\ldots,n)$ in MATLAB. The results are given in Table 5.1. In this table, N denotes the number of equivalence classes of eigenpairs (2.8) or E-eigenpairs found by `teig`, `eeig`, `PSOLVE`,

or `NSolve`, the reported CPU times are in seconds, and "-" denotes that no results were returned after 12 hours.

TABLE 5.1
Comparison of `teig` *and* `eeig` *with* `PSOLVE` *and* `NSolve`.

(m,n)	$T(m,n)$	Alg.	N	Time (s)	$E(m,n)$	Alg.	N	Time (s)
$(4,5)$	405	`teig`	405	15.8	121	`eeig`	121	5.4
		`PSOLVE`	404	14.0		`PSOLVE`	121	9.5
		`NSolve`	405	3136.4		`NSolve`	121	486.6
$(5,5)$	1280	`teig`	1280	73.8	341	`eeig`	341	22.3
		`PSOLVE`	1280	65.5		`PSOLVE`	341	38.6
		`NSolve`	-	-		`NSolve`	341	9264.8
$(5,6)$	6144	`teig`	6144	606.5	1365	`eeig`	1365	166.5
		`PSOLVE`	6144	694.2		`PSOLVE`	1365	283.6
		`NSolve`	-	-		`NSolve`	-	-
$(6,6)$	18750	`teig`	18750	3721.3	3906	`eeig`	3906	990.2
		`PSOLVE`	18748	4636.0		`PSOLVE`	3905	1721.0
		`NSolve`	-	-		`NSolve`	-	-

From Table 5.1, we see that `teig` and `eeig` successfully find all equivalence classes of eigenpairs (2.8) or E-eigenpairs using a reasonable amount of time in all the cases. `NSolve` cannot get any results in 12 hours in some cases (we terminated it after 12 hours). Although `PSOLVE` successfully finds all equivalence classes in many cases, it does miss a few equivalence classes in some cases. We believe that the robustness of `teig` and `eeig` is due to their use of the retracing strategies and working in the projective space, as described in subsection 3.3. Regarding the CPU time usage, `PSOLVE` is comparable to `teig` when the number of eigenpairs (2.8) $T(m,n)$ is moderate. When $T(m,n)$ gets larger, `teig` uses less time. This is because the mixed volume computation in `PSOLVE` takes significantly more time when $T(m,n)$ becomes large. We also observe that `eeig` uses less time than `PSOLVE`. Note that by using the equivalent class structure discussed in Remark 2.1, `eeig` traces $E(m,n)$ paths. On the other hand, `PSOLVE` traces $2E(m,n)$ paths because here it solves the system (2.4) from Definition 2.1 directly.

Example 5.2. In this example we show the effectiveness and efficiency of `teig` for finding all equivalence classes of isolated eigenpairs as defined in (2.8) of a generic tensor $\mathcal{A} \in \mathbb{C}^{[m,n]}$. Each tensor was generated using $randn(n,\ldots,n)+i*randn(n,\ldots,n)$ in MATLAB. The results are reported in Table 5.2, in which N denotes the number of equivalence classes of isolated eigenpairs found by `teig` and $T(m,n)$ denotes the bound of the number of equivalence classes of isolated eigenpairs (see Remark 2.4(i)).

Example 5.3. In this example we show the effectiveness and efficiency of `eeig` for finding all equivalence classes of isolated E-eigenpairs of a generic tensor $\mathcal{A} \in \mathbb{C}^{[m,n]}$. Each generic tensor was generated using $randn(n,\ldots,n) + i * randn(n,\ldots,n)$ in MATLAB. The results are reported in Table 5.3, in which N denotes the number of equivalence classes of E-eigenpairs found by `eeig` and $E(m,n)$ denotes the bound of the number of equivalence classes of isolated E-eigenpairs (see Remark 2.4(i)).

According to [37], [8], and [5], for a randomly generated tensor $\mathcal{A} \in \mathbb{C}^{[m,n]}$, it has $T(m,n)$ equivalence classes of eigenpairs (2.8) and $E(m,n)$ equivalence classes of E-eigenpairs. Moreover, its eigenpairs and E-eigenpairs are isolated. From Tables 5.2 and 5.3, we observe that `teig` and `eeig` can find all equivalence classes of eigenpairs (2.8) and E-eigenpairs of such a tensor in the examples we tested.

TABLE 5.2
Performance of `teig` *on computing eigenpairs* (2.8) *of complex random tensors.*

(m,n)	$T(m,n)$	N	Time (s)	(m,n)	$T(m,n)$	N	Time (s)
$(3,5)$	80	80	2.4	$(3,6)$	192	192	6.8
$(3,7)$	448	448	18.3	$(3,8)$	1024	1024	53.0
$(3,9)$	2304	2304	145.9	$(3,10)$	5120	5120	409.2
$(4,3)$	27	27	0.7	$(4,4)$	108	108	2.9
$(4,5)$	405	405	15.8	$(4,6)$	1458	1458	80.0
$(4,7)$	5103	5103	385.9	$(4,8)$	17496	17496	2115.5
$(5,3)$	48	48	1.2	$(5,4)$	256	256	8.8
$(5,5)$	1280	1280	73.8	$(5,6)$	6144	6144	606.5
$(5,7)$	28672	28672	5394.2	$(6,3)$	75	75	2.3
$(6,4)$	500	500	21.0	$(6,5)$	3125	3125	287.7
$(6,6)$	18750	18750	3721.3	$(7,3)$	108	108	3.6
$(7,4)$	864	864	51.5	$(7,5)$	6480	6480	981.3

TABLE 5.3
Performance of `eeig` *on computing E-eigenpairs of complex random tensors.*

(m,n)	$E(m,n)$	N	Time (s)	(m,n)	$E(m,n)$	N	Time (s)
$(3,5)$	31	31	1.4	$(3,6)$	63	63	3.1
$(3,7)$	127	127	7.5	$(3,8)$	255	255	20.3
$(3,9)$	511	511	48.5	$(3,10)$	1023	1023	133.9
$(4,3)$	13	13	0.4	$(4,4)$	40	40	1.7
$(4,5)$	121	121	5.4	$(4,6)$	364	364	26.9
$(4,7)$	1093	1093	119.5	$(4,8)$	3280	3280	555.8
$(5,3)$	21	21	0.7	$(5,4)$	85	85	4.2
$(5,5)$	341	341	22.3	$(5,6)$	1365	1365	166.5
$(5,7)$	5461	5461	1330.7	$(6,3)$	31	31	1.2
$(6,4)$	156	156	9.5	$(6,5)$	781	781	100.4
$(6,6)$	3906	3906	990.2	$(7,3)$	43	43	1.9
$(7,4)$	259	259	21.3	$(7,5)$	1555	1555	245.0

Example 5.4. In this example we show the effectiveness and efficiency of `teig` and `teneig` for finding all equivalence classes of 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. Each generic tensor was generated using $randn(n,\ldots,n) + i * randn(n,\ldots,n)$ in MATLAB. The results are reported in Table 5.4, in which N denotes the number of equivalence classes of eigenpairs found by `teig` or `teneig`, $T(m,n)$ denotes the bound of the number of equivalence classes of isolated $\mathcal{B}$-eigenpairs for $m = m'$, and $G(m,m',n)$ denotes the bound of the number of equivalence classes of isolated $\mathcal{B}$-eigenpairs for $m \neq m'$ (see Theorem 2.3).

From Table 5.4 we see that our `teig` and `teneig` find all equivalence classes of isolated $\mathcal{B}$-eigenpairs of $\mathcal{A}$ for the generic tensors $\mathcal{A}$ and $\mathcal{B}$ we tested in a reasonable amount of time.

5.2. Examples for computing real eigenpairs. In this subsection, numerical examples are provided to illustrate the effectiveness and efficiency of `zeig` or `heig` for computing real Z-eigenpairs or H-eigenpairs of a tensor $\mathcal{A} \in \mathbb{R}^{[m,n]}$. By Definition 2.1, (λ, x) is a Z-eigenpair if and only if $((-1)^{m-2}\lambda, -x)$ is a Z-eigenpair, and (λ, x) is an H-eigenpair if and only if (λ, tx) is an H-eigenpair for any nonzero $t \in \mathbb{R}$. Only one representative from each equivalence class of eigenpairs will be listed in our examples. The notation $\lambda^{(l)}$ is used to denote l eigenvectors counting multiplicities are found for the eigenvalue λ. In the following tables, the multiplicity of an eigenpair means the multiplicity of this eigenpair as a zero of the corresponding defining polynomial

TABLE 5.4
Performance of `teig` *and* `teneig` *on computing generalized eigenpairs of complex random tensors.*

`teig` ($m = m'$)				`teneig` ($m \neq m'$)			
(m, n)	$T(m, n)$	N	Time (s)	(m, m', n)	$G(m, m', n)$	N	Time (s)
$(3, 7)$	448	448	23.7	$(3, 2, 7)$	127	127	10.3
$(3, 8)$	1024	1024	68.3	$(3, 4, 6)$	665	665	68.1
$(3, 9)$	2304	2304	210.3	$(3, 5, 5)$	496	496	49.5
$(4, 5)$	405	405	20.8	$(4, 2, 6)$	364	364	28.9
$(4, 6)$	1458	1458	110.4	$(4, 3, 5)$	211	211	13.2
$(4, 7)$	5103	5103	737.5	$(4, 5, 4)$	175	175	9.5
$(5, 5)$	1280	1280	97.9	$(5, 4, 5)$	781	781	83.9
$(5, 6)$	6144	6144	623.6	$(5, 6, 3)$	61	61	2.6
$(6, 4)$	500	500	29.9	$(6, 5, 4)$	369	369	30.7
$(6, 5)$	3125	3125	449.4	$(6, 7, 3)$	91	91	6.0
$(7, 3)$	108	108	4.4	$(7, 6, 4)$	671	671	77.1
$(7, 4)$	864	864	77.6	$(7, 8, 3)$	127	127	9.4

system. For the sake of conciseness, the polynomial system resulting from the tensor eigenvalue problem will be omitted.

Example 5.5. Consider the symmetric tensor $\mathcal{A} \in \mathbb{R}^{[6,3]}$ whose corresponding polynomial form is the Motzkin polynomial:

$$\mathcal{A}x^6 = x_3^6 + x_1^4x_2^2 + x_1^2x_2^4 - 3x_1^2x_2^2x_3^2.$$

Example 5.9 of [5] states that this tensor has 25 equivalence classes of Z-eigenpairs. Using `zeig` exactly 25 equivalence classes of Z-eigenpairs are found as shown in Table 5.5, which confirms the results of [5]. `zeig` takes about 0.9 seconds to carry out the entire computation.

TABLE 5.5
Z-eigenpairs of the tensor in Example 5.5.

λ	$0^{(14)}$			$0.0156^{(8)}$		$0.2500^{(2)}$	1
x_1	0.5774	1	0	0.8253	0.2623	0.7071	0
x_2	$\pm$0.5774	0	1	$\pm$0.2623	$\pm$0.8253	$\pm$0.7071	0
x_3	$\pm$0.5774	0	0	$\pm$0.5000	$\pm$0.5000	0	1
multiplicity	1	5	5	1	1	1	1

All the H-eigenpairs found by Example 4.10 of [12] are also found by `heig`, as shown in Table 5.6. `heig` takes about 1.7 seconds to carry out the entire computation.

TABLE 5.6
H-eigenpairs of the tensor in Example 5.5.

λ	$0^{(14)}$			$0.0555^{(8)}$		$1^{(15)}$	
x_1	$\pm$1	1	0	$\pm$0.4568	1	0	1
x_2	1	0	1	1	$\pm$0.4568	0	$\pm$1
x_3	$\pm$1	0	0	$\pm$0.6856	$\pm$0.6856	1	1
multiplicity	1	5	5	1	1	13	1

So far the only available method that can find all real eigenvalues of a symmetric tensor is Algorithm 3.6 in [12]. In the next two examples, we report our experiments on `zeig` for computing all Z-eigenvalues, using examples taken from [12].

Example 5.6. We use our `zeig` to compute the Z-eigenpairs of 12 symmetric tensors from [12]. The test problems and numerical results are given in Appendix A. From the numerical results we see that `zeig` finds all the Z-eigenvalues found by Algorithm 3.6 of [12] on this set of test problems. We now summarize the CPU time (in seconds) used by `zeig` in Table 5.7. Since the computer used in [12] is different from the computer used in this paper, the CPU time used by Algorithm 3.6 in [12] is not reported here, but we refer to [12].

TABLE 5.7
CPU time used by `zeig` *for computing the Z-eigenvalues of* 12 *symmetric tensors from* [12].

Problem	1	2	3	4	5	6	7	8	9	10	11	12
Time (s)	0.3	4.0	0.3–0.4	0.1	0.6	1.8	15.7	6.1	0.3	6.3	27.3	4.5

Example 5.7. Consider the symmetric tensor $\mathcal{A} \in \mathbb{R}^{[4,n]}$ (Example 4.16 in [12]) with the polynomial form

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

For different n, all the Z-eigenvalues found by Algorithm 3.6 in [12] are also found by `zeig`, which are given in Table 5.8. We remark that when $n = 8, 9, 10$, `zeig` can find all the Z-eigenvalues in a reasonable amount of time, but [12] reports that Algorithm 3.6 can only find the first three largest Z-eigenvalues. The CPU time used by `zeig` is reported in Table 5.8. Since different computers were used, we refer to [12] for the CPU time used by Algorithm 3.6 of [12]. For the sake of conciseness, the corresponding Z-eigenvectors are not displayed.

TABLE 5.8
CPU time of `zeig` *for computing Z-eigenvalues of the tensor in Example* 5.7 *(∗ denotes that Algorithm* 3.6 [12] *only finds the first three largest Z-eigenvalues).*

n	λ					Time (s)
4	0.0000	4.0000	5.0000	5.3333		1.7
5	0.0000,	4.1667,	4.2500,	5.5000,	6.2500	5.4
6	0.0000,	4.0000,	4.5000,	6.0000,	7.2000	15.5
7	0.0000, 4.9000,	4.0833, 6.5000,	4.1667, 8.1667	4.7500,	4.8846,	58.3
8*	0.0000, 5.0000,	4.0000, 5.2609,	4.2667, 5.3333,	4.2727, 7.0000,	4.3333, 9.1429	244.1
9*	0.0000, 5.6250,	4.0500, 5.7857,	4.1250, 7.5000,	4.5000, 10.1250	5.2500,	788.0
10*	0.0000, 4.6667, 5.9808,	4.0000, 4.7500, 6.2500,	4.1667, 4.7593, 8.0000,	4.1818, 4.7619, 11.1111	4.2500, 5.5000,	2665.6

Appendix.

Problem 1. Consider the symmetric tensor $\mathcal{A} \in \mathbb{R}^{[4,3]}$ (Example 4.1 in [12]; see also [37]) with the polynomial form

$$\mathcal{A}x^4 = x_1^4 + 2x_2^4 + 3x_3^4.$$

Using `zeig` all the Z-eigenpairs found in [12] are obtained (see Table A. 1). `zeig` takes about 0.3 seconds to carry out the entire computation.

TABLE A. 1
Z-eigenpairs of the tensor in Problem 1.

λ	$0.5455^{(4)}$	$0.6667^{(2)}$	$0.7500^{(2)}$	1	$1.2^{(2)}$	2	3
x_1	0.7385	0.8165	0.8660	1	0	0	0
x_2	± 0.5222	± 0.5774	0	0	0.7746	1	0
x_3	± 0.4264	0	± 0.5000	0	± 0.6325	0	1
multiplicity	1	1	1	1	1	1	1

Problem 2. For the diagonal tensor $\mathcal{D} \in \mathbb{R}^{[5,4]}$ (Example 4.2 in [12]) such that $\mathcal{D}x^5 = x_1^5 + 2x_2^5 - 3x_3^5 - 4x_4^5$. Consider the symmetric tensor $\mathcal{A} \in \mathbb{R}^{[5,4]}$ such that $\mathcal{A}x^5 = \mathcal{D}(Qx)^5$, where

$$Q = (I - 2w_1w_1^T)(I - 2w_2w_2^T)(I - 2w_3w_3^T)$$

and w_1, w_2, w_3 are randomly generated unit vectors. All 30 Z-eigenpairs found in [12] are also found by using `zeig`. The 15 nonnegative Z-eigenvalues are listed below:

0.2518, 0.3261, 0.3466, 0.3887, 0.4805, 0.5402, 0.5550, 0.6057,
0.8543, 0.9611, 1.0000, 1.2163, 2.0000, 3.0000, 4.0000.

For conciseness, the corresponding Z-eigenvectors are not displayed here. `zeig` takes about 4.0 seconds to do the entire computation.

Problem 3. Consider the symmetric tensor $\mathcal{A} \in \mathbb{R}^{[4,3]}$ (Example 4.3 in [12]; see also [37]) with the polynomial form

$$\mathcal{A}x^4 = 2x_1^4 + 3x_2^4 + 5x_3^4 + 4ax_1^2x_2x_3,$$

where a is a parameter. All Z-eigenvalues found in [12] are also found by `zeig` for different values of a, as shown in Table A. 2. For conciseness, the corresponding Z-eigenvectors are not displayed here. The CPU time used by `zeig` for each a is also reported in the table.

TABLE A. 2
Z-eigenvalues of the tensor in Problem 3.

a	λ	Time (s)
0	$0.9677^{(4)}$, $1.2000^{(2)}$, $1.4286^{(2)}$, $1.8750^{(2)}$, 2, 3, 5	0.4
0.25	$0.8464^{(2)}$, $1.0881^{(2)}$, $1.2150^{(2)}$, $1.4412^{(2)}$, $1.8750^{(2)}$, 2, 3, 5	0.4
0.5	$0.7243^{(2)}$, $1.2069^{(2)}$, $1.2593^{(2)}$, $1.4783^{(2)}$, $1.8750^{(2)}$, 2, 3, 5	0.4
1	$0.4787^{(2)}$, $1.6133^{(2)}$, $1.8750^{(2)}$, 2, 3, 5	0.3
3	$-0.5126^{(2)}$, $1.8750^{(2)}$, 2, $2.2147^{(2)}$, 3, 5	0.3

Problem 4. Consider the symmetric tensor $\mathcal{A} \in \mathbb{R}^{[4,2]}$ (Example 4.4 in [12]; see also [37]) with the polynomial form

$$\mathcal{A}x^4 = 3x_1^4 + x_2^4 + 6ax_1^2x_2^2,$$

where a is a parameter. All Z-eigenvalues found in [12] are also found by `zeig` for different values of a, which are listed in Table A. 3. The CPU time used by `zeig` for each a is also given in the table. For conciseness, the corresponding Z-eigenvectors are not displayed here.

TABLE A. 3
Z-eigenvalues of the tensor in Problem 4.

a	λ	Time (s)
-1	$-0.6000^{(2)}$, 1, 3	0.1
0	$0.7500^{(2)}$, 1, 3	0.1
0.25	$0.9750^{(2)}$, 1, 3	0.1
0.5	1, 3	0.1
2	1, 3, $4.1250^{(2)}$	0.1
3	1, 3, $5.5714^{(2)}$	0.1

Problem 5. Consider the symmetric tensor $\mathcal{A} \in \mathbb{R}^{[4,3]}$ (Example 4.5 in [12]; see also [23] or [34]) such that

$$A_{1111} = 0.2883, A_{1112} = -0.0031, A_{1113} = 0.1973, A_{1122} = -0.2485,$$
$$A_{1123} = -0.2939, A_{1133} = 0.3847, A_{1222} = 0.2972, A_{1223} = 0.1862,$$
$$A_{1233} = 0.0919, A_{1333} = -0.3619, A_{2222} = 0.1241, A_{2223} = -0.3420,$$
$$A_{2233} = 0.2127, A_{2333} = 0.2727, A_{3333} = -0.3054.$$

All the Z-eigenpairs found in [12] are also found by `zeig`, as given in Table A. 4. `zeig` takes about 0.6 seconds to do the entire computation.

TABLE A. 4
Z-eigenpairs of the tensor in Problem 5.

λ	x^T	λ	x^T
–1.0954	(–0.5915, 0.7467, 0.3043)	–0.5629	(–0.1762, 0.1796, –0.9678)
–0.0451	(0.7797, 0.6135, 0.1250)	0.1735	(0.3357, 0.9073, 0.2531)
0.2433	(–0.9895, –0.0947, 0.1088)	0.2628	(–0.1318, 0.4425, 0.8870)
0.2682	(0.6099, 0.4362, 0.6616)	0.3633	(0.2676, 0.6447, 0.7160)
0.5105	(–0.3598, 0.7780, –0.5150)	0.8169	(–0.8412, 0.2635, –0.4722)
0.8893	(–0.6672, –0.2471, 0.7027)		

Problem 6. Consider the symmetric tensor $\mathcal{A} \in \mathbb{R}^{[3,6]}$ (Example 4.6 in [12]; see also [39]) such that $A_{iii} = i$ for $i = 1, \ldots, 6$ and $A_{i,i,i+1} = 10$ for $i = 1, \ldots, 5$ and zero otherwise. All the Z-eigenvalues found in [12] are also found by `zeig`. The 19 nonnegative Z-eigenvalues are listed below:

3.9992, 4.0225, 4.2464, 4.3358, 5.1402, 5.4817, 5.5218, 5.5668,
5.5674, 6.0000, 7.2165, 8.1889, 8.5979, 8.6596, 8.7347, 10.9711,
15.4298, 15.4552, 16.2345.

For conciseness, the corresponding Z-eigenvectors are not displayed here. `zeig` takes about 1.8 seconds to carry out the entire computation.

Problem 7. Consider the symmetric tensor $\mathcal{A} \in \mathbb{R}^{[4,6]}$ (Example 4.7 in [12]; see also [28]) with the polynomial form

$$\begin{aligned}-\mathcal{A}x^4 = {} & (x_1 - x_2)^4 + (x_1 - x_3)^4 + (x_1 - x_4)^4 + (x_1 - x_5)^4 + (x_1 - x_6)^4 \\ & + (x_2 - x_3)^4 + (x_2 - x_4)^4 + (x_2 - x_5)^4 + (x_2 - x_6)^4 \\ & + (x_3 - x_4)^4 + (x_3 - x_5)^4 + (x_3 - x_6)^4 \\ & + (x_4 - x_5)^4 + (x_4 - x_6)^4 + (x_5 - x_6)^4.\end{aligned}$$

TABLE A. 5
Z-eigenpairs of the tensor in Problem 7.

λ	x^T	Multiplicity
$-7.2000^{(6)}$	$(0.1826, 0.1826, 0.1826, 0.1826, 0.1826, -0.9129)$	1
$-6.0000^{(15)}$	$(0.7071, 0, 0, 0, 0, -0.7071)$	1
$-4.5000^{(\star)}$	$(0.5774, 0.5774, -0.2887, -0.2887, -0.2887, -0.2887)$	-
$-4.0000^{(10)}$	$(0.4082, 0.4082, 0.4082, -0.4082, -0.4082, -0.4082)$	1
$0^{(\star)}$	$(0.4082, 0.4082, 0.4082, 0.4082, 0.4082, 0.4082)$	-

All the 5 Z-eigenvalues found in [12] are also found by `zeig`, which are given in Table A. 5. As pointed out in [12], every permutation of a Z-eigenvector is also a Z-eigenvector. Only the Z-eigenvector with $x_1 \geq x_2 \geq \cdots \geq x_6$ corresponding to one Z-eigenvalue is listed. We remark that the Z-eigenpairs corresponding to Z-eigenvalues 0 and -4.5 are in a positive dimensional solution component of the corresponding polynomial system. Therefore, there are infinitely many Z-eigenvectors associated with 0 and -4.5. `zeig` finds 484 Z-eigenvectors associated with 0 and 180 Z-eigenvectors associated with -4.5. Only one of these Z-eigenvectors for each case is listed in Table A. 6. `zeig` takes about 15.7 seconds to do the entire computation.

Problem 8. Consider the symmetric tensor $\mathcal{A} \in \mathbb{R}^{[4,5]}$ (Example 4.8 in [12]; see also [46]) with the polynomial form

$$\mathcal{A}x^4 = (x_1 + x_2 + x_3 + x_4)^4 + (x_2 + x_3 + x_4 + x_5)^4.$$

All 3 Z-eigenvalues found in [12] are also found by `zeig`, which are shown in Table A. 6.

TABLE A. 6
Z-eigenpairs of the tensor in Problem 8.

λ	x^T	Multiplicity
$0^{(\star)}$	$(0.3870, -0.1537, 0.4532, -0.6866, 0.3870)$	-
0.5000	$(0.7071, 0, 0, 0, -0.7071)$	1
24.5000	$(0.2673, 0.5345, 0.5345, 0.5345, 0.2673)$	1

We remark that the Z-eigenpairs corresponding to Z-eigenvalue 0 are in a positive dimensional solution component of the corresponding polynomial system. Thus, there are infinitely many Z-eigenvectors associated with Z-eigenvalue 0. `zeig` finds 234 of them. Only one of them is listed in Table A. 6. `zeig` uses about 6.1 seconds to do the entire computation.

Problem 9. Consider the symmetric tensor $\mathcal{A} \in \mathbb{R}^{[3,3]}$ (Example 4.9 in [12]; see also [5]) with the polynomial form

$$\mathcal{A}x^3 = 2x_1^3 + 3x_1x_2^2 + 3x_1x_3^2.$$

We remark that the Z-eigenpairs corresponding to Z-eigenvalue 2 are in a positive dimensional solution component of the corresponding polynomial system. Thus, there are infinitely many Z-eigenvectors associated with Z-eigenvalue 0. `zeig` finds 7 of them. Only one of them is listed in Table A. 7. `zeig` uses about 0.3 seconds to do the entire computation.

TABLE A. 7
Z-eigenpairs of the tensor in Problem 9.

λ	x^T	Multiplicity
$2^{(\star)}$	$(1,0,0)$	-

Problem 10. Consider the tensor $\mathcal{A} \in \mathbb{R}^{[4,n]}$ (Example 4.12 in [12]; see also [34]) such that

$$\mathcal{A}_{i_1,\ldots,i_4} = \sin(i_1 + i_2 + i_3 + i_4).$$

When $n = 5$, all 5 Z-eigenvalues found in [12] are also found by `zeig`, which are given in Table A. 8.

TABLE A. 8
Z-eigenpairs of the tensor in Problem 10.

λ	x^T	Multiplicity
–8.8463	$(0.5809, 0.3563, -0.1959, -0.5680, -0.4179)$	1
–3.9204	$(-0.1785, 0.4847, 0.7023, 0.2742, -0.4060)$	1
$0^{(\star)}$	$(-0.5213, 0.3748, -0.6608, 0.1824, -0.3433)$	-
4.6408	$(0.5055, -0.1228, -0.6382, -0.5669, 0.0256)$	1
7.2595	$(0.2686, 0.6150, 0.3959, -0.1872, -0.5982)$	1

We remark that the Z-eigenpairs corresponding to Z-eigenvalue 0 are in a positive dimensional solution component of the corresponding polynomial system. Thus, there are infinitely many Z-eigenvectors associated with 0. `zeig` finds 234 of them. Only one of them is listed in Table A. 8. `zeig` takes about 6.3 seconds to carry out the entire computation.

Problem 11. Consider the tensor $\mathcal{A} \in \mathbb{R}^{[4,n]}$ (Example 4.13 in [12]) such that

$$\mathcal{A}_{i_1,\ldots,i_4} = \tan(i_1) + \cdots + \tan(i_4).$$

When $n = 6$, all the 3 Z-eigenvalues found in [12] are also found by `zeig`, which are given in Table A. 9.

TABLE A. 9
Z-eigenpairs of the tensor in Problem 11.

λ	x^T	Multiplicity
–133.2871	$(0.1936, 0.5222, 0.3429, 0.2287, 0.6272, 0.3559)$	1
$0^{(\star)}$	$(-0.5840, -0.3454, 0.1784, 0.6773, 0.1892, -0.1156)$	-
45.5045	$(0.6281, 0.0717, 0.3754, 0.5687, -0.1060, 0.3533)$	1

We remark that the Z-eigenpairs corresponding to Z-eigenvalue 0 are in a positive dimensional solution component of the corresponding polynomial system. Thus, there are infinitely many Z-eigenvectors associated with 0. `zeig` finds 724 of them. Only one of them is listed in Table A. 9. It takes `zeig` about 27.3 seconds to carry out the entire computation.

Problem 12. Consider the tensor $\mathcal{A} \in \mathbb{R}^{[5,n]}$ (Example 4.14 in [12]) such that

$$\mathcal{A}_{i_1,\ldots,i_5} = \ln(i_1) + \cdots + \ln(i_5).$$

Table A. 10
Z-eigenpairs of the tensor in Problem 12.

λ	x^T	Multiplicity
$0^{(\star)}$	$(-0.4304, 0.8139, 0.0069, -0.3903)$	-
0.7074	$(-0.9054, -0.3082, 0.0411, 0.2890)$	1
132.3070	$(0.4040, 0.4844, 0.5319, 0.5657)$	1

For $n = 4$, all 3 Z-eigenvalues found in [12] are also found by `zeig`, which are shown in Table A. 10.

We remark that the Z-eigenpairs corresponding to Z-eigenvalue 0 are in a positive dimensional solution component of the corresponding polynomial system. Thus, there are infinitely many Z-eigenvectors associated with 0. `zeig` finds 166 of them. Only one of them is listed in Table A. 10. The entire computation takes `zeig` about 4.5 seconds.

Acknowledgments. We would like to thank Professor T. Y. Li, the two anonymous referees, and the associate editor for their insightful and constructive comments and suggestions, which significantly improved both the content and presentation of the paper.

REFERENCES

[1] E. L. Allgower and K. Georg, *Numerical Continuation Methods: An Introduction*, Springer-Verlag, Berlin, 1990.
[2] D. N. Bernstein, *The number of the roots of a system of equations*, Funct. Anal. Appl., 9(1975), pp. 183–185.
[3] D. L. Bates, J. D. Hauenstein, A. J. Sommese, and C. W. Wampler, *Numerically Solving Polynomial Systems with Bertini*, SIAM, Philadelphia, 2013.
[4] Yu. D. Burago, *Mixed volume theory*, in Encyclopedia of Mathematics, M. Hazewinkel, ed., Springer, New York, 2001.
[5] D. Cartwright and B. Sturmfels, *The number of eigenvalues of a tensor*, Linear Algebra Appl., 438 (2013), pp. 942–952.
[6] K. C. Chang, K. Pearson, and T. Zhang, *Perron-Frobenius theorem for nonnegative tensors*, Commun. Math. Sci., 6 (2008), pp. 507–520.
[7] K. C. Chang, K. Pearson, and T. Zhang, *On eigenvalues of real symmetric tensors*, J. Math. Anal. Appl., 350 (2009), pp. 416–422.
[8] K. C. Chang, L. Qi, and T. Zhang, *A survey of the spectral theory of nonnegative tensors*, Numer. Linear Algebra Appl., 20 (2013), pp. 891–912.
[9] D. A. Cox, J. Little, and D. O'Shea, *Using Algebraic Geometry*, Springer-Verlag, New York, 2005.
[10] D. A. Cox, J. Little, and D. O'Shea, *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*, Springer-Verlag, New York, 2007.
[11] J. Cooper and A. Duttle, *Spectra of uniform hypergrphs*, Linear Algebra Appl., 436 (2012), pp. 3268–3292.
[12] C. Cui, Y.-H. Dai, and J. Nie, *All real eigenvalues of symmetric tensors*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 1582–1601.
[13] S. Friedland, S. Gaubert, and L. Han, *Perron-Frobenius theorem for nonnegative multilinear forms and extensions*, Linear Algebra Appl., 438 (2013), pp. 738–749.
[14] T. Gunji, S. Kim, M. Kojima, A. Takeda, K. Fujisawa, and T. Mizutani, *PHoM: A polyhedral homotopy continuation method*, Computing, 73 (2004), pp. 53–57.
[15] L. Han, *An unconstrained optimization approach for finding real eigenvalues of even order symmetric tensors*, Numer. Algebra Control Optim., 3 (2013), pp. 583–599.
[16] C. Hao, C. Cui, and Y.-H. Dai, *A sequential subspace projection method for extreme Z-eigenvalues of supersymmetric tensors*, Numer. Linear Algebra Appl., 22 (2015), pp. 287–298.
[17] J, Hauenstein, L. Oeding, G. Ottaviani, and A. Sommese, *Homotopy Techniques for Tensor Decomposition and Perfect Identifiability*, arXiv:1501.00090, 2015.

[18] R. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, Cambridge, UK, 2013.
[19] C. Hillier and L.-H. Lim, *Most tensor problems are NP-hard*, J. ACM, 60 (2013).
[20] S. Hu, Z. Huang, and L. Qi, *Finding the extreme Z-eigenvalues of tensors via a sequential semidefinite programming method*, Numer. Linear Algebra Appl., 20 (2013), pp. 972–984.
[21] S. Hu, L. Qi, and G. Zhang, *The Geometric Measure of Entanglement of Pure States with Nonnegative Amplitudes and the Spectral Theory of Nonnegative Tensors*, arXiv:1203.3675, 2012.
[22] B. Huber and B. Sturmfels, *A polyhedral method for solving sparse polynomial systems*, Math. Comp., 64 (1995), pp. 1541–1555.
[23] T. G. Kolda and J. R. Mayo, *Shifted power method for computing tensor eigenpairs*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 1095–1124.
[24] T. G. Kolda and J. R. Mayo, *An adaptive shifted power method for computing generalized tensor eigenpairs*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 1563–1581.
[25] 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, 83 (2008), pp. 109–133.
[26] T. Y. Li, *Solving polynomial systems by the homotopy continuation method*, in Foundations of Computational Mathematics, F. Cucker, ed., Handb. Numer. Anal. 11, North-Holland, Amsterdam, 2003, pp. 209–304.
[27] T. Y. Li, *Homotopy methods*, in Encyclopedia of Applied and Computational Mathematics, B. Engquist, ed., Springer, Berlin, 2015, pp. 653–656.
[28] G. Li, L. Qi, and G. Yu, *The Z-eigenvalues of a symmetric tensor and its application to spectral hypergraph theory*, Numer. Linear Algebra Appl., 20 (2013), pp. 1001–1029.
[29] L.-H. Lim, *Singular values and eigenvalues of tensors: A variational approach*, in Proceedings of the IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP'05), 2005, pp. 129–132.
[30] T. Y. Li and X. Wang, *The BKK root count in* $\mathbb{C}^n$, Math. Comp., 65 (1996), pp. 1477–1484.
[31] Y. Liu, G, Zhou, and N. F. Ibrahim, *An always convergent algorithm for the largest eigenvalue of an irreducible nonnegative tensor*, J. Comput. Appl. Math., 235 (2010), pp. 286–292.
[32] A. P. Morgan, *Solving Polynomial Systems Using Continuation for Engineering and Scientific Problems*, SIAM, Philadelphia, 2009.
[33] M. Ng, L. Qi, and G. Zhou, *Finding the largest eigenvalue of a nonnegative tensor*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 1090–1099.
[34] J. Nie and L. Wang, *Semidefinite relaxations for best rank-1 tensor approximations*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 1155–1179.
[35] L. Oeding and G. Ottaviani, *Eigenvectors of tensors and algorithms for Waring decomposition*, J. Symbolic Comput., 54 (2013), pp. 9–35.
[36] W. Rudin, *Principles of Mathematical Analysis*, McGraw-Hill, New York, 2006.
[37] L. Qi, *Eigenvalues of a real supersymmetric tensor*, J. Symbolic Comput., 40 (2005), pp. 1302–1324.
[38] L. Qi, W. Sun, and Y. Wang, *Numerical multilinear algebra and its applications*, Front. Math. China, 2 (2007), pp. 501–526.
[39] L. Qi, F. Wang, and Y. Wang, *Z-eigenvalue methods for a global optimization polynomial optimization problem*, Math. Program., 118 (2009), pp. 301–306.
[40] L. Qi, Y. Wang, and E. X. Wu, *D-eigenvalues of diffusion kurtosis tensors*, J. Comput. Appl. Math., 221 (2008), pp. 150–157.
[41] L. Qi, G. Yu, and E. X. Wu, *Higher order positive semi-definite diffusion tensor imaging*, SIAM J. Imaging Sci., 3 (2010), pp. 416–433.
[42] L. Qi, G. Yu, and Y. Xu, *Nonnegative diffusion orientation distribution function*, J. Math. Imaging Vis., 45 (2013), pp. 103–113.
[43] A. J. Sommese and W. W. Wampler, *The Numerical Solution of Systems of Polynomials Arising in Engineering and Science*, World Scientific, Singapore, 2005.
[44] J. Verschelde, *Algorithm 795: PHCpack: A general-purpose solver for polynomial systems by homotopy continuation*, ACM Trans. Math. Software, 25 (1999), pp. 251–276.
[45] A. H. Wright, *Finding all solutions to a system of a polynomial equations*, Math. Comp., 44 (1985), pp. 125–133.
[46] J. Xie and A. Chang, *On the Z-eigenvalues of the signless Laplacian tensor for an even uniform hypergraph*, Numer. Linear Algebra Appl., 20 (2013), pp. 1030–1045.

[47] Y. Yang and Q. Yang, *Further results for Perron-Frobenius theorem for nonnegative tensors*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 2517–2530.
[48] Y. Yang and Q. Yang, *Further results for Perron-Frobenius theorem for nonnegative tensors* II, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 1236–1250.
[49] Z. Zeng and T. Li, *NACLab: A Matlab Toolbox for Numerical Algebraic Computation*, ACM Commun. Comput. Algebra, 47 (2013), pp. 170–173.
[50] L. Zhang, L. Qi, and Y. Xu, *Finding the largest eigenvalue of an irreducible tensor and linear convergence for weakly positive tensors*, J. Comput. Math., 30 (2012), pp. 24–33.
[51] G. Zhou, L. Qi, and S.-Y. Wu, *On the largest eigenvalue of a symmetric nonnegative tensor*, Numer. Linear Algebra Appl., 20 (2013), pp. 913–928.