Spectral projected gradient methods for generalized tensor eigenvalue complementarity problems

Gaohang Yu¹ · Yisheng Song²,³ · Yi Xu⁴ · Zefeng Yu⁵

Abstract This paper looks at the tensor eigenvalue complementarity problem (TEiCP) which arises from the stability analysis of finite dimensional mechanical systems and is closely related to the optimality conditions for polynomial optimization. We investigate two monotone ascent spectral projected gradient (SPG) methods for TEiCP. We also present a shifted scaling-and-projection algorithm (SPA), which is a great improvement of the original SPA method proposed by Ling et al. (Comput. Optim. Appl. 63, 143–168 2016). Numerical comparisons with some existing gradient methods in the literature are reported to illustrate the efficiency of the proposed methods.

© Springer Science+Business Media, LLC, part of Springer Nature 2018

Received: 28 December 2015 / Accepted: 2 April 2018 / Published online: 24 April 2018

Gaohang Yu
maghyu@163.com

Yisheng Song
songyisheng1@gmail.com

Yi Xu
yi.xu1983@gmail.com

Zefeng Yu
yzf_2000@sina.com

¹ Department of Mathematics, School of Science, Hangzhou Dianzi University, Hangzhou, China
² School of Mathematics and Information Science, Henan Normal University, Xinxiang County, China
³ Henan Engineering Laboratory for Big Data Statistical Analysis and Optimal Control, Henan Normal University, Xinxiang County, China
⁴ Department of Mathematics, Southeast University, Nanjing, China
⁵ School of Mathematics and Computer Sciences, Gannan Normal University, Ganzhou, China
Keywords  Tensor · Pareto eigenvalue · Pareto eigenvector · Projected gradient method · Eigenvalue complementarity problem

1 Introduction

An $m$th-order $n$-dimensional real tensor $A$ consists of $n^m$ entries in real numbers:

$$A = (a_{i_1 i_2 \ldots i_m}), \quad a_{i_1 i_2 \ldots i_m} \in \mathbb{R}, \quad \text{for any } i_1, i_2, \ldots, i_m \in [n],$$

where $[n] = \{1, 2, \ldots, n\}$. Denote the set of all real $m$th-order $n$-dimensional tensors by $\mathbb{T}^{[m,n]}$. $A$ is called symmetric if the value of $a_{i_1 i_2 \ldots i_m}$ is invariant under any permutation of its indices $i_1, i_2, \ldots, i_m$. Denote the set of all real symmetric $m$th-order $n$-dimensional tensors by $\mathbb{S}^{[m,n]}$. For any vector $x \in \mathbb{R}^n$, $Ax^{m-1}$ is a vector in $\mathbb{R}^n$ with its $i$th component as

$$(Ax^{m-1})_i = \sum_{i_2, \ldots, i_m = 1}^n a_{i_1 i_2 \ldots i_m} x_{i_2} \cdots x_{i_m}.$$

A real symmetric tensor $A$ of order $m$ dimension $n$ uniquely defines a $m$th degree homogeneous polynomial function $h$ with real coefficient by

$$h(x) := Ax^m = x^T (Ax^{m-1}) = \sum_{i_1, \ldots, i_m = 1}^n a_{i_1 \ldots i_m} x_{i_1} \cdots x_{i_m}.$$

We call that the tensor $A$ is positive definite if $Ax^m > 0$ for all $x \neq 0$.

In 2005, Qi [33] and Lim [27] proposed the definition of eigenvalues and eigenvectors for higher order tensors, independently. Furthermore, in [5], these definitions were unified by Chang, Person and Zhang. Let $A$ and $B$ be real-valued, $m$th-order $n$-dimensional symmetric tensors. Assume further that $m$ is even and $B$ is positive definite. We call $(\lambda, x) \in \mathbb{R} \times \mathbb{R}^n \setminus \{0\}$ a generalized eigenpair of $(A, B)$ if

$$Ax^{m-1} = \lambda Bx^{m-1}. \quad (1)$$

When the tensor $B$ is an identity tensor $\varepsilon$ such that $\varepsilon x^{m-1} = \|x\|^{m-2} x$ for all $x \in \mathbb{R}^n$ [5], the eigenpair reduces to $Z$-eigenpair [27, 33]. Another special case is that when $B = I$ with

$$(I)_{i_1 i_2 \ldots i_m} = \delta_{i_1 i_2 \ldots i_m} = \begin{cases} 1, & \text{if } i_1 = i_2 = \ldots = i_m, \\ 0, & \text{otherwise,} \end{cases}$$

the real scalar $\lambda$ is called an $H$-eigenvalue and the real vector $x$ is the associated $H$-eigenvector of the tensor $A$ [33]. In the last decade, tensor eigenproblem has received...
much attention in the literature [8, 10, 14, 18, 20, 21, 25, 26, 34, 43–45, 49, 52–54, 56], which has numerous applications [9, 36, 38, 39].

In this paper, we consider the tensor eigenvalue complementarity problem (TEiCP): finding a scalar $\lambda \in \mathbb{R}$, and $x \in \mathbb{R}^n \setminus \{0\}$ such that

\[
\begin{align*}
    x &\geq 0, \\
    (\lambda B - A)x^{m-1} &\geq 0, \\
    \langle x, (\lambda B - A)x^{m-1} \rangle &= 0,
\end{align*}
\]

where $A \in \mathbb{T}[m,n]$, and $B \in \mathbb{S}[m,n]$ is positive definite.

The solution of TEiCP $(\lambda, x)$ is called Pareto eigenpair of $(A, B)$. In some special case, we can call it Pareto H-eigenpair or Pareto Z-eigenpair [42] if the tensor $B$ has an special form as shown above in the generalized eigenpairs (1). Replacing the nonnegative cones in (2) by a closed convex cone and its dual cone, Ling, He and Qi investigated the cone eigenvalue complementarity problem for higher-order tensor in [28]. Moreover, in [29], they studied the high-degree eigenvalue complementarity problem for tensors as a natural extension of quadratic eigenvalue complementarity problem for matrices. TEiCP is also closely related to the optimality conditions for polynomial optimization [31, 42, 48, 50, 55], a class of differential inclusions with noncovex processes [28], and a kind of nonlinear differential dynamical system [13]. The properties of Pareto eigenvalues and their connection to polynomial optimization are studied in [42]. Recently, as a special type of nonlinear complementarity problems, the tensor complementarity problem is inspiring more and more research in the literature [2, 6, 7, 12, 13, 17, 19, 30, 43, 44, 46, 47]. A shifted projected power method for TEiCP was proposed in [13], in which they need an adaptive shift to force the objective to be (locally) convex to guarantee the convergence of power method. In [28], Ling, He and Qi presented a scaling-and-projection algorithm (SPA) for TEiCP. One main shortcoming of SPA is the stepsize may approach to zero as the sequence gets close to a solution of TEiCP [28]. Recently, by introducing an NCP-function, Chen and Qi [12] reformulated the TEiCP as a system of nonlinear equations. And then, they proposed a semismooth Newton method for solving the system of nonlinear equations [12].

In this paper, we will investigate two spectral projected gradient algorithms for TEiCP. The rest of this paper is organized as follows. In Section 2, some properties of the solutions of TEiCP and two optimization reformulations of TEiCP are presented. In Section 3, two spectral projected gradient algorithms are proposed. Global convergence results could be established under some suitable assumptions. We also present a shifted scaling-and-projection algorithm (SSPA) in Section 4, which is a great improvement of the original SPA method [28]. Numerical experiments are reported in Section 4 to show the efficiency of the proposed methods. Finally, we have a conclusion section.

Throughout this paper, let $\mathbb{R}^n_+ = \{ x \in \mathbb{R}^n : x \geq 0 \}$, and $\mathbb{R}^n_{++} = \{ x \in \mathbb{R}^n : x > 0 \}$. Given a set $J \subseteq [n]$, the principal sub-tensor of a tensor $A \in \mathbb{T}[m,n]$, denoted by $A_J$, is the tensor in $\mathbb{T}[m,|J|]$, such that $A_{IJ} = (a_{i_1...i_m})$ for all $i_1, \ldots, i_m \in J$. Here, the symbol $|J|$ denotes the cardinality of $J$. 
2 Some properties and reformulations of TEiCP

The following proposition shows the relationship between the solution of TEiCP (2) and the generalized eigenvalue problem (1).

**Proposition 1** \((\lambda, x)\) is a solution of TEiCP (2) if and only if there exists a subset \(I \subseteq [n]\), such that \(\lambda\) is a generalized eigenvalue of \((A_I, B_I)\) and \(x_I \in \mathbb{R}^{|I|}\) is a corresponding eigenvector, and

\[
\sum_{i_2, \ldots, i_m \in I} (\lambda b_{i_2 \ldots i_m} - a_{i_2 \ldots i_m}) x_i_2 \cdots x_i_m \geq 0, \forall i \in \bar{I} := [n] \setminus I.
\]

In such a case, the Pareto eigenvector \(x\) satisfies \(x_{\bar{I}} = 0\).

This proposition was firstly presented in [42] for Pareto H-eigenpair and Pareto Z-eigenpair, and then unified by Xu and Ling for TEiCP [51].

Denote the set of solutions of (2) by \(\sigma(A, B)\), i.e.,

\[
\sigma(A, B) = \{(\lambda, x) \in \mathbb{R} \times \mathbb{R}^n \setminus \{0\} : 0 \leq x \perp (\lambda B - A)x^{m-1} \geq 0\}.
\]

Given a tensor \(A \in T^{[m,n]}\), we know that there exists a unique semi-symmetric tensor [31] \(\bar{A}\) such that \(Ax^{m-1} = \bar{A}x^{m-1}\). It is clear that \(\sigma(A, B) = \sigma(\bar{A}, B)\).

Without loss of generality, we always assume that \(A \in S^{[m,n]}\). On the other hand, if \((\lambda, x) \in \sigma(A, B)\), then \((\lambda, sx) \in \sigma(A, B)\) for any \(s > 0\). So, we could restrict \(\|x\| = 1\) to replace \(\|x\| \neq 0\). Since \(x \geq 0\), we use the linear constraint \(\|x\|_1 = e^T x = 1\), in which \(e \in \mathbb{R}^n\) denotes all-ones vector.

**Proposition 2** The symmetric TEiCP (2) is equivalent to the following optimization problem

\[
\max \lambda(x) = \frac{Ax^m}{Bx^m} \text{ subject to } x \in \Delta^n := \{x \in \mathbb{R}^n : e^T x = 1, x \geq 0\},
\]

in the sense that any equilibrium solution \(x\) of (3) is a solution of the symmetric TEiCP.

**Proof** By some simple calculations, we can obtain its gradient and Hessian are as follows

\[
g(x) \equiv \nabla \lambda(x) = \frac{m}{Bx^m} (Ax^{m-1} - \frac{Ax^m}{Bx^m} Bx^{m-1}).
\]

and

\[
H(x) \equiv \nabla^2 \lambda(x) = \frac{m(m-1)Ax^{m-2}}{Bx^m} - \frac{m(m-1)Ax^m Bx^{m-2} + m^2 (Ax^{m-1} \otimes Bx^{m-1})}{(Bx^m)^2} + \frac{m^2 Ax^m (Bx^{m-1} \otimes Bx^{m-1})}{(Bx^m)^3},
\]
where $x \otimes y = xy^T + yx^T$, and $A^{m-2}$ is a matrix with its component as

$$(A^{m-2})_{i_1i_2} = \sum_{i_3, \ldots, i_m=1}^n a_{i_1i_2i_3 \cdots i_m} x_{i_3} \cdots x_{i_m} \text{ for all } i_1, i_2 \in [n].$$

According to (4), we can derive that

$$x^T g(x) = \frac{m}{Bx^m} (x^T A^{m-1} - A x^m) x^T B x^m - 1 = 0. \tag{5}$$

The Lagrangian function is $L(x, \mu, v) = \lambda(x) + \mu (1 - e^T x) + v^T x$, where $\mu \in \mathbb{R}$ and $v \in \mathbb{R}^n$ are the Lagrange multipliers. Any equilibrium solution of the nonlinear programming problem (3) satisfies the KKT conditions

$$\begin{cases} 
\nabla \lambda(x) - \mu e + v = 0, \\
v \geq 0, \\
v^T x = 0, \\
x \geq 0, \\
e^T x = 1.
\end{cases}$$

Using $v^T x = 0$ and $x^T g(x) = 0$, by taking the dot product with $x$ in the first equation, we get that $\mu = 0$. So, the first equation could be written as $v = -\nabla \lambda(x)$.

Since $v \geq 0$ and $B$ is positive definite, it follows that

$$(\lambda, x) \in [\mathbb{R} \times \mathbb{R}^n \setminus \{0\} : 0 \leq x \perp (\lambda B - A) x^{m-1} \geq 0],$$

i.e. any equilibrium solution $x$ of (3) is a solution of the symmetric TEiiCP (2).

Furthermore, the global maximum/minimum of $\lambda(x)$ in the canonical simplex $\Delta^n$ corresponds to the extreme value of Pareto eigenpair of $(A, B)$ [28, 42] if $B$ is strictly copositive, i.e., $Bx^m > 0$ for any $x \in \mathbb{R}^n_+ \setminus \{0\}$. The concept of copositive tensor is introduced by Qi [35]. A tensor $A$ is said copositive if $A x^m \geq 0$ for all $x \in \mathbb{R}^n_+ \setminus \{0\}$. $A$ is copositive (strictly copositive) if and only if all of its Pareto H-eigenvalues or Z-eigenvalues are nonnegative (positive, respectively) [42]. Since the set $\Delta^n$ is compact, without loss of generality, we may assume that there exists a vector $\hat{x} \in \Delta^n$ satisfying $\lambda(\hat{x}) = \max \{\lambda(x) : x \in \Delta^n\}$. Then, according to the Proposition 2, we know that $(\lambda(\hat{x}), \hat{x})$, which is an equilibrium solution of (3), is a solution of TEiiCP (2). So, we have $\lambda(\hat{x}) \leq \lambda_{TCP}^{max}$. On the other hand, according to the Proposition 1, any solution of TEiiCP (2) should be a generalized eigenpair of $(A_I, B_I)$. So, it is easy to show that $\lambda(\hat{x}) \geq \lambda_{TCP}^{max}$.

**Proposition 3** Let $A, B \in \mathbb{S}^{[m,n]}$, and $B$ is copositive. Let

$$\lambda_{TCP}^{max} = \max \{\lambda : \exists x \in \mathbb{R}^n \setminus \{0\} \text{ such that } (\lambda, x) \in \sigma (A, B)\},$$

and

$$\lambda_{TCP}^{min} = \min \{\lambda : \exists x \in \mathbb{R}^n \setminus \{0\} \text{ such that } (\lambda, x) \in \sigma (A, B)\}.$$ Then $\lambda_{TCP}^{max} = \max \{\lambda(x) : x \in \Delta^n\}$ and $\lambda_{TCP}^{min} = \min \{\lambda(x) : x \in \Delta^n\}$. 

\[ \square \]
If both $A$ and $B$ are symmetric and strictly copositive tensors, then we can use logarithmic function as the merit function in (3). In such a case, TEiCP (2) could be reformulated to the following nonlinear optimization problem:

$$\max f(x) = \ln(Ax^m) - \ln(Bx^m) \text{ subject to } x \in \Delta^n.$$  \hfill (6)

Its gradient and Hessian are respectively

$$\nabla f(x) = \frac{m(Ax^{m-1})}{Ax^m} - \frac{m(Bx^{m-1})}{Bx^m}. \hfill (7)$$

and

$$\nabla^2 f(x) = \frac{m(m-1)Ax^{m-2}}{Ax^m} - \frac{m(m-1)Bx^{m-2}}{Bx^m}$$

$$+ \frac{m^2Bx^{m-1}(Bx^{m-1})^T}{(Bx^m)^2} - \frac{m^2Ax^{m-1}(Ax^{m-1})^T}{(Ax^m)^2}.$$  

The Hessian is much simpler than that of Rayleigh quotient function in (3). If one need to use Hessian for computing Pareto eigenvalue, the logarithmic merit function may be a favorable choice.

3 Spectral projected gradient methods

In this section, the spectral projected gradient (SPG) method is applied to the nonlinear programming problem (3). One main feature of SPG is the spectral choice of the step length (also called BB stepsize) along the search direction, originally proposed by Barzilai and Borwein [3]. The Barzilai-Borwein method performs much better than the steepest descent gradient method or projected gradient method in practice [4, 15, 41]. Especially, when the objective function is a convex quadratic function and $n = 2$, a sequence generated by the BB method converges $R$-superlinearly to the global minimizer [3]. For any dimension convex quadratic function, it is still globally convergent [40] but the convergence is $R$-linear [16].

Let the projection $P_{\Omega}$ be the mapping $P_{\Omega} : \mathbb{R}^n \rightarrow \Omega$ defined by

$$P_{\Omega}(x) = \arg\min \{\|x - y\| : y \in \Omega\}.$$  

The general projected gradient method for problem (3) consists of the following iterations:

$$\left\{ \begin{array}{l} z_k = P_{\Delta^n}(x_k + \beta_k g_k), \beta_k > 0 \\ x_{k+1} = x_k + \alpha_k(z_k - x_k), 0 < \alpha_k \leq 1. \end{array} \right.$$  

If we use the BB-stepsize to define the parameter $\beta_k$, we can present the following spectral projected gradient method (with monotone line search).

Here, $\Delta^n$ is a closed convex set. By the projection operation and the convexity of $\Delta^n$, we know that for all $u \in \Delta^n$ and $\forall v \in \mathbb{R}^n$,

$$(v - P_{\Delta^n}(v))^T (u - P_{\Delta^n}(v)) \leq 0.$$  

Set $u = x$ and $v = x + \beta g(x)$ in the above inequality, then we have

$$\beta g(x)^T [x - P_{\Delta^n}(x + \beta g(x))] + \|x - P_{\Delta^n}(x + \beta g(x))\|^2 \leq 0.$$
Algorithm 1 Spectral projected gradient (SPG1) algorithm for TEiCP

Given tensors $A \in S^{m,n}$ and $B \in S^{m,n}_+$, an initial unit iterate $x_0 \geq 0$, parameter $\rho \in (0, 1)$. Let $\varepsilon > 0$ be the tolerance of termination. Calculate gradient $g(x_0)$, $\beta_0 = 1/\|g(x_0)\|$. Set $k=0$.

Step 1: Compute $z_k = P_{\Delta^n}(x_k + \beta_k g_k)$ and the direction $d_k = z_k - x_k$.

Step 2: If $\|d_k\| < \varepsilon$ then stop: $\lambda(x_k) = \frac{Ax_k}{Bx_k}$ is a Pareto eigenvalue, and $x_k$ is a corresponding Pareto eigenvector of TEiCP. Otherwise, set $\alpha \leftarrow 1$.

Step 3: If

$$f(x_k + \alpha d_k) \geq f(x_k) + \rho \alpha g_k^T d_k,$$

then define $x_{k+1} = x_k + \alpha d_k$, $s_k = x_{k+1} - x_k$, $y_k = g_{k+1} - g_k$. Otherwise, take $\alpha = -\frac{a^2 g_k^T d_k}{2(f(x_k + \alpha d_k) - f(x_k) - \alpha g_k^T d_k)}$ or $\alpha = 0.5\alpha$ when the minimum of the one-dimensional quadratic interpolation lies outside $[0.1\alpha, 0.9\alpha]$, and try again.

Step 4: Compute $b_k = \langle s_k, y_k \rangle$. If $b_k \leq 0$ set $\beta_{k+1} = \beta_{\text{max}}$; else, compute $\alpha_k = \langle s_k, s_k \rangle$ and $\beta_{k+1} = \max\{\beta_{\text{min}}, \min\{\beta_{\text{max}}, \frac{\alpha_k}{b_k}\}\}$. Set $k := k + 1$ and go to Step 1.

Let $d_\beta(x) = P_{\Delta^n}(x + \beta g(x)) - x$ with $\beta > 0$, then we have the following lemma.

Lemma 1 For all $x \in \Delta^n$, $\beta \in (0, \beta_{\text{max}}]$, we have

$$g(x)^T d_\beta(x) \geq \frac{1}{\beta} \|d_\beta(x)\|_2^2 \geq \frac{1}{\beta_{\text{max}}} \|d_\beta(x)\|_2^2.$$

From (9), we know that $d_k$ is an ascent direction. Hence, a stepsize satisfying (8) will be found after a finite number of trials, and the SPG algorithm is well defined. When $\beta = \frac{\langle s_k, y_k \rangle}{\langle s_k, s_k \rangle}$ in $d_\beta(x)$, we call it spectral projected gradient (SPG). The vector $d_\beta(x^*)$ vanishes if and only if $x^*$ is a constrained stationary point of optimization problem (3) or (6). The convergence of SPG method is established as follows. The proof is similar to that in [4].

Theorem 1 Let $\{x_k\}$ be generated by SPG1 Algorithm. If there is a vector $x_k$ such that $d_\beta(x_k) = 0$, then $(\lambda(x_k), x_k)$ is a solution of the symmetric TEiCP. Otherwise, any accumulation point of the sequence $\{x_k\}$ is a constrained stationary point, i.e., the sequence $(\lambda(x_k))$ converges to a Pareto eigenvalue of the symmetric TEiCP.

Proof Let $x^*$ be an accumulation point of $\{x_k\}$, and relabel $\{x_k\}$ a subsequence converging to $x^*$. According to the Proposition 2, we just need to show that $x^*$ is a constrained stationary point of the optimization problem. Let us suppose by way of contradiction that $x^*$ is not a constrained stationary point. So, by continuity and compactness, there exist $\delta > 0$ such that $\|d_\beta(x^*)\| \geq \delta > 0$ for all $\beta \in (0, \beta_{\text{max}}]$. Furthermore, using the Lemma 1, we have $g(x^*)^T d_\beta(x^*) \geq \frac{\delta^2}{\beta_{\text{max}}} g(x^*)^T d_\beta(x^*)$ for all $\beta \in (0, \beta_{\text{max}}]$, which implies that for $k$ larger enough on the subsequence that converges to $x^*$, $g(x_k)^T d_\beta(x_k) > c$ for all $\beta \in [\beta_{\text{min}}, \beta_{\text{max}}]$. Here, we can set $c = \frac{\delta^2}{2\beta_{\text{max}}} > 0$. We consider two cases.
Firstly, assume that \( \inf \alpha_k \geq \varepsilon > 0 \). By continuity, for sufficiently large \( k \), \( \| d_{\beta_k}(x_k) \| \geq \delta/2 \). From the line search condition (8), we have
\[
\frac{f(x_k + \alpha d_k) - f(x_k)}{2\alpha} \geq \frac{\rho \varepsilon \delta^2}{4\beta_{\text{max}}}.
\]
Clearly, when \( k \to \infty \), \( f(x_k) \to \infty \), which is a contradiction. In fact, \( f \) is a continuous function and so \( f(x_k) \to f(x^*) \).

Assume that \( \inf \alpha_k = 0 \). Since \( \inf \alpha_k = 0 \), there exists a subsequence \( \{x_k\}_K \) such that \( \lim_{k \in K} \alpha_k = 0 \). In such a case, from the way \( \alpha_k \) is chosen in (8), there exists an index \( \bar{k} \) sufficiently large such that for all \( k \geq \bar{k}, k \in K \), \( \alpha_k \neq 0 \). Then, condition (8) fails to satisfy, i.e.,
\[
\frac{f(x_k + 2\alpha d_k) - f(x_k)}{2\alpha} < \rho g_k^T d_k.
\]
By the mean value theorem, we can rewrite this relation as
\[
d_k^T g(x_k + t_k d_k) < \rho g_k^T d_k \quad \text{for all} \quad k \in K, \ k \geq \bar{k},
\]
where \( t_k \in [0, 2\alpha_k] \) that goes to zero as \( k \in K \) goes to infinity. Taking limits in the above inequality, we deduce that \( (1 - \rho)g(x^*)^T d(x^*) \leq 0 \). Since \( 1 - \rho > 0 \) and \( g_k^T d_k > 0 \) for all \( k \), then \( g(x^*)^T d(x^*) = 0 \). By continuity, this indicates that for \( k \) large enough on the subsequence we have that \( g_k^T d_k < c/2 \), which contradicts to \( g_k^T d_k > c \).

Therefore, any accumulation point of the sequence \( \{x_k\} \) is a constrained stationary point. By using the Proposition 2, it follows that the sequence \( \{\lambda(x_k)\} \) converges to a Pareto eigenvalue of the symmetric TEiCP. \( \square \)

Similarly, we would like to present the following SPG algorithm for TEiCP with curvilinear search.

4 Numerical experiments

In this section, we present some numerical results to illustrate the effectiveness of the spectral projected gradient (SPG) methods, which were compared with the scaling-and-projection algorithm (SPA) proposed by Ling, He and Qi [28] and the shifted projected power (SPP) method for TEiCP proposed in [13].

We firstly describe the SPP algorithm and the SPA method as follows.

In SPA method, \( \Phi(x) \) is a normalizing function for the closed convex cone \( \mathbb{R}_+^n \). In the implementation, we let \( \Phi(x) = \|x\| \), and the same as in SSPA method. Since the stepsize \( \alpha_k \) in SPA approaches to zero as the sequence \( \{x_k\} \) gets close to a solution of TEiCP, as shown in [28], the number of iterations will increase significantly. In order to improve the efficiency of SPA method, they try to amplify the stepsize and proposed a modification of SPA such as \( u_k = P_{\mathbb{R}_+^n}(x_k + s\alpha_k y_k) \) with \( s \in (1, 8) \) being a constant parameter. A suitable choice \( s \) will get an improvement. But, how to choose it? Anyway, the stepsize \( s\alpha_k \) also approaches to zero when the sequence \( \{x_k\} \) gets close to a solution of TEiCP. But this situation will be better if the merit function...
Algorithm 2 Spectral projected gradient (SPG2) algorithm for TEiCP

Given tensors $A \in S^{[m,n]}$ and $B \in S^{[m,n]}_+$, an initial unit iterate $x_0 \geq 0$, parameter $\rho \in (0, 1)$. Let $\epsilon > 0$ be the tolerance of termination. Calculate gradient $g(x_0)$, $\beta_0 = 1/\|g(x_0)\|$. Set $k=0$.

Step 1: If $\|P_{\Delta^n}(x_k + \beta_k g_k) - x_k\| < \epsilon$, stop, declaring $\lambda(x_k) = \frac{A_{x_k}^m}{Bx_k^n}$ is a Pareto eigenvalue, and $x_k$ is a corresponding Pareto eigenvector of TEiCP.

Step 2: Set $\alpha \leftarrow \beta_k$.

Step 3: Set $x_+ = P_{\Delta^n}(x_k + \alpha g_k)$. If

$$f(x_+) \geq f(x_k) + \rho \alpha g_k^T(x_+ - x_k),$$

then define $x_{k+1} = x_+, s_k = x_{k+1} - x_k, y_k = g_{k+1} - g_k$. Otherwise, take $\alpha = 0.5\alpha$ when the minimum of the one-dimensional quadratic interpolation lies outside $[0.1\alpha, 0.9\alpha]$ and try again.

Step 4: Compute $b_k = \langle s_k, y_k \rangle$. If $b_k \leq 0$ set $\beta_{k+1} = \beta_{\text{max}}$; else, compute $a_k = \langle s_k, s_k \rangle$ and

$$\beta_{k+1} = \max\{\beta_{\text{min}}, \min\{\beta_{\text{max}}, \frac{a_k}{b_k}\}\}.$$ Set $k := k + 1$ and go to Step 1.

Algorithm 3 Shifted Projected Power (SPP) algorithm [13]

Given tensors $A \in S^{[m,n]}$ and $B \in S^{[m,n]}_+$, an initial unit iterate $x_0 \geq 0$. Let $\epsilon > 0$ be the tolerance on termination. Let $\tau > 0$ be the tolerance on being positive definite.

for $k = 0, 1, \ldots$ do

1: Compute the gradient $g(x_k) = \nabla f(x_k)$ and the Hessian $H(x_k) = \nabla^2 f(x_k)$, respectively. Let $r_k \leftarrow \max\{0, (\tau - \lambda_{\text{min}}(H_k))/m, \nabla \hat{f}(x_k) = \nabla f(x_k) + r_k m x_k\}$.

2: Let $\nabla \hat{f}_+(x_k) = \begin{cases} 0, & \text{if } (\nabla \hat{f}(x_k))_i < 0, \\ (\nabla \hat{f}(x_k))_i, & \text{otherwise}, \end{cases}$

3: If $\|\nabla \hat{f}_+(x_k)\| \leq \epsilon$, stop. Otherwise, $x_{k+1} \leftarrow \nabla \hat{f}_+(x_k)/\|\nabla \hat{f}_+(x_k)\|$. Set $k=k+1$ and go back to Step 1.

End for

Algorithm 4 Scaling-and-Projection Algorithm (SPA) [28]

Given tensors $A \in S^{[m,n]}$ and $B \in S^{[m,n]}_+$. For an initial point $u_0 \geq 0$, define $x_0 = \frac{u_0}{\Phi(u_0)}$. Let $\epsilon > 0$ be the tolerance on termination.

for $k = 0, 1, \ldots$ do

1: Compute $x_k = \frac{A_{x_k}^m}{B_{x_k}^n}, y(x_k) = A(x_k)^{m-1} - \lambda_k B(x_k)^{m-1}$.

2: If $\|y(x_k)\| \leq \epsilon$, stop. Otherwise, let $\alpha_k = \|y\|$, compute $u_k = P_{\Delta^n}(x_k + \alpha_k y_k)$, and $x_{k+1} = \frac{u_k}{\Phi(u_k)}$. Set $k=k+1$ and go back to Step 1.

End for
$f(x)$ is (locally) convex. So, we present the following shifted SPA method, in which an adaptive shift could force the objective to be (locally) convex [24].

Algorithm 5 Shifted Scaling-and-Projection Algorithm (SSPA)

Given tensors $A \in S^{[m,n]}$ and $B \in S^{[m,n]}_+$. For an initial point $u_0 \geq 0$, define $x_0 = \frac{u_0}{\Phi(u_0)}$. Compute $\lambda(x_0) = \frac{A(x_0)^m}{B(x_0)^m}$. Let $\epsilon > 0$ be the tolerance on termination. Let $\tau > 0$ be the tolerance on being positive definite.

for $k = 0, 1, \ldots$ do

1: Compute $y(x_k) = A(x_k)^{m-1} - \lambda_k B(x_k)^{m-1}$, the Hessian $H(x_k) = \nabla^2 f(x_k)$, respectively. Let $r_k \leftarrow \max\{0, (\tau - \lambda_{min}(H_k))/m\}$, $\hat{g}(x_k) = y(x_k) + r_k m x_k$.

2: Let $\alpha_k = \| \hat{g}(x_k) \|$, compute $u_k = P_{\mathbb{R}^n} (x_k + \alpha_k \hat{g}_k)$, and $x_{k+1} = \frac{u_k}{\Phi(u_k)}$, $\lambda(x_{k+1}) = \frac{A(x_{k+1})^m}{B(x_{k+1})^m}$.

3: If $|\lambda(x_{k+1}) - \lambda(x_k)| \leq \epsilon$, stop. Otherwise, Set $k=k+1$ and go back to Step 1.

End for

Both SPG1 and SPG2 are monotone ascent methods. In general, the merit function $f(x)$ is chosen to be the Rayleigh quotient function in (3). In the implementation, the iterates will be terminated once $\|x_{k+1} - x_k\| \leq \epsilon$ or $|\lambda_{k+1} - \lambda_k| \leq \epsilon$. In addition, the stopping criterion for SPG1/SPG2 algorithm is $\| P_{\Delta^n}(x_k + \beta_k \hat{g}_k) - x_k \| < \epsilon$ while the SPP algorithm would be terminated once $\| \nabla f(x_k) \| \leq \epsilon$, and the other two algorithms would be stopped once $\| y(x_k) \| \leq \epsilon$ or $|\lambda(x_{k+1}) - \lambda(x_k)| \leq \epsilon$ respectively. We choose the tolerance on termination $\epsilon = 10^{-6}$ for all algorithms. We set the parameter $\rho = 0.05$, $\beta_{max} = \frac{1}{\| \hat{g} \|}$, $\beta_{min} = \| \hat{g} \|$ for SPG method, and $\tau = 0.01$ for SPP and SSPA methods. For the implementation of the orthogonal projection onto the simplex $P_{\Delta^n}$, we could refer to the algorithm proposed by Y. Chen, X. Ye in [11]. In all numerical experiments, the maximum iterations were set to be 500. The experiments were done on a laptop with Intel Core i5-5300U CPU with a 8GB RAM, using MATLAB R2016b, and the Tensor Toolbox [1].

4.1 Comparison for computing Pareto Z-eigenpairs

The following example is originally from [22] and was used in evaluating the SS-HOPM algorithm in [23] and the GEAP algorithm in [24] for computing Z-eigenpairs.

Example 1 (Kofidis and Regalia [22]) Let $A \in S^{[4,3]}$ be the symmetric tensor defined by

\[
\begin{align*}
    a_{111} &= 0.2883, & a_{112} &= -0.0031, & a_{113} &= 0.1973, & a_{122} &= -0.2485, & a_{123} &= 0.1862, \\
    a_{133} &= 0.3847, & a_{122} &= 0.2972, & a_{123} &= -0.2939, & a_{123} &= 0.0919, & a_{133} &= -0.3619, \\
    a_{222} &= 0.1241, & a_{222} &= -0.3420, & a_{223} &= 0.2127, & a_{233} &= 0.2727, & a_{333} &= -0.3054.
\end{align*}
\]

To compare the convergence in terms of the number of iterations. Figure 1 shows the results for computing Pareto Z-eigenvalues of $A$ from Example 1, when the starting point is $x_0 = [1.0; 1.0; 1.0]$. In this case, all of the SPG1, SPG2, SPP, SSPA
can reach the same Pareto Z-eigenvalue 0.3633. SPP is similar to SPG1 method in this case. As we can see, comparing with SPA method, SSPA method gets a great improvement (Tables 1, 2, and 3).

Example 2 Let $A \in S^{[4,n]}$ be the diagonal tensor defined by $a_{iiii} = \frac{i-1}{i}$, for $i = 1, \ldots, n$.

Figure 2 shows the results for computing Pareto Z-eigenvalues of $A$ from Example 2 with $n = 5$, when the starting point is $x_0 = [1.0; 1.0; 1.0; 1.0]$. In this case, all of the SPG1, SPG2, SPP, SPA, and SSPA can reach the largest Pareto Z-eigenvalue 0.8. Comparing with SPA method, SSPA method get a great improvement again. But, SSPA method is still slower than the other three methods in this case.

Example 3 Let $A \in S^{[4,3]}$ be the symmetric tensor defined by: Firstly, set $A = \text{tensor(zeros}(3, 3, 3, 3))$, and

$$
\begin{align*}
a_{1111} &= 1.00397, \quad a_{2222} = 0.99397, \quad a_{3333} = 1.00207, \\
a_{1222} &= 0.00401, \quad a_{3111} = 0.00788, \quad a_{3111} = 0.00001, \\
a_{3222} &= 0.00005, \quad a_{1333} = 0.99603, \quad a_{2333} = 1.00400,
\end{align*}
$$

and then using $A = \text{symmetrize}(A)$ to symmetrize it.

Figure 3 shows the results for computing Pareto Z-eigenvalues of $A$ from Example 3, when the starting point is $x_0 = [0.9015; 0.3183; 0.5970]$. In this case, both SPG1 and SPG2 reach the largest Pareto Z-eigenvalue $\lambda = 1.2048$ while the other three methods reach the Z-eigenvalue $\lambda = 1.0040$.

Table 1 Comparative results for computing Pareto Z-eigenvalues of $A$ from Example 1

| Alg. | $\lambda$ | Eigenvector | Its. | Error          | Time (sec.) |
|------|-----------|-------------|------|----------------|-------------|
| SPG1 | 0.3633    | [0.1647;0.3961;0.4391] | 12   | 6.95E-07       | 0.06        |
| SPG2 | 0.3633    | [0.1642;0.3961;0.4396] | 6    | 8.03E-08       | 0.20        |
| SPP  | 0.3633    | [0.2679;0.6448;0.7158] | 10   | 5.71E-07       | 0.06        |
| SPA  | 0.3632    | [0.2771;0.6461;0.7112] | 260  | 9.96E-07       | 0.58        |
| SSPA | 0.3633    | [0.2680;0.6448;0.7158] | 12   | 5.21E-07       | 0.03        |
Table 2  Comparative results for computing Pareto Z-eigenvalues of $\mathcal{A}$ from Example 2 with $n = 5$

| Alg.  | $\lambda$ | Eigenvector            | Its. | Error          | Time (sec.) |
|-------|------------|------------------------|------|----------------|-------------|
| SPG1  | 0.8        | [0;0;0;0;1]            | 3    | 0.0$^1$        | 0.02        |
| SPG2  | 0.8        | [0;0;0;0;1]            | 3    | 0.0$^1$        | 0.03        |
| SPP   | 0.8        | [0;0;0;0;1]            | 7    | 3.97E−09       | 0.02        |
| SPA   | 0.7999     | [0.0014;0.0024;0.0037;0.0063;1] | 286  | 9.87E−07       | 0.75        |
| SSPA  | 0.8        | [0;0;0;1.0000E−04;1]   | 19   | 8.92E−08       | 0.06        |

$^1$The error value exceeds the precision range.

We also used 100 random starting guesses, each entry selected uniformly at random from the interval [0, 1], to test SPG1, SPP, and SSPA, respectively. For each set of experiments, the same set of random starts were used. We listed the median number of iterations until convergence, and the average run time in the 100 experiments in Table 4. The computed Pareto Z-eigenvalues were listed in Figs. 4, 5 and 6 for Example 1, 2, and 3, respectively. As we can see, most of time, all of the three methods can reach the same Pareto Z-eigenvalue. But for Example 1, it seems that SPG1 method could reach the largest Pareto Z-eigenvalue with a higher probability.

4.2 Comparison with SPP for computing Pareto H-eigenpairs

In this subsection, we test SPG1, SPG2, and SPP method for finding Pareto H-eigenpairs of $\mathcal{A}$ from Examples 4-6 ($n = 5$):

**Example 4** (Nie and Wang [32]) Let $\mathcal{A} \in \mathbb{S}^{[4,n]}$ be the symmetric tensor defined by

$$a_{ijkl} = \sin(i + j + k + l) \quad (1 \leq i, j, k, l \leq n).$$

**Example 5** (Nie and Wang [32]) Let $\mathcal{A} \in \mathbb{S}^{[4,n]}$ be the symmetric tensor defined by

$$a_{ijkl} = \tan(i) + \tan(j) + \tan(k) + \tan(l) \quad (1 \leq i, j, k, l \leq n).$$

Table 3  Comparative results for computing Pareto Z-eigenvalues of $\mathcal{A}$ from Example 3

| Alg.  | $\lambda$ | Eigenvector            | Its. | Error          | Time (sec.) |
|-------|------------|------------------------|------|----------------|-------------|
| SPG1  | 1.2048     | [0.1414;0.1426;0.7160] | 9    | 1.19E−07       | 0.02        |
| SPG2  | 1.2048     | [0.1415;0.1426;0.7159] | 11   | 9.38E−08       | 0.22        |
| SPP   | 1.0040     | [1:0.0020;0]           | 9    | 6.90E−10       | 0.06        |
| SPA   | 1.0039     | [1:0.0026;0.0062]      | 210  | 9.82E−07       | 0.48        |
| SSPA  | 1.0040     | [1:0.0020;0]           | 12   | 3.90E−08       | 0.03        |
Fig. 2 Comparison for computing Pareto Z-eigenvalues of \( A \) from Example 2 with \( n = 5 \) and \( x_0 = [1.0; 1.0; 1.0; 1.0; 1.0] \)

Fig. 3 Comparison for computing Pareto Z-eigenvalues of \( A \) from Example 3 with \( x_0 = [0.9015; 0.3183; 0.5970] \)

Table 4 Comparative results for 100 random test on computing Pareto Z-eigenvalues of \( A \) from Ex.1, Ex.2(\( n = 5 \)) and Ex.3

| Algorithm | SPG1 | SPP | SSPA |
|-----------|------|-----|------|
| Example   | Its. | Time| Its. | Time | Its. | Time |
| Ex. 1     | 7.69 | 0.03| 7.55 | 0.02| 14.79 | 0.03 |
| Ex. 2     | 2.38 | 0.01| 5.11 | 0.01| 34.65 | 0.07 |
| Ex. 3     | 5.07 | 0.02| 5.11 | 0.01| 14.28 | 0.03 |
Fig. 4 The computed Pareto $Z$-eigenvalues by SPG1, SPP, and SSPA in the 100 runs on the $A$ from Example 1

Fig. 5 The computed Pareto $Z$-eigenvalues by SPG, SPP, and SSPA in the 100 runs on the $A$ from Example 2 with $n = 5$

Fig. 6 The computed Pareto $Z$-eigenvalues by SPG, SPP, and SSPA in the 100 runs on the $A$ from Example 3
Example 6 (Nie and Wang [32]) Let $A \in S^{[4,n]}$ be the tensor defined by

$$a_{ijkl} = \frac{(-1)^i}{i} + \frac{(-1)^j}{j} + \frac{(-1)^k}{k} + \frac{(-1)^l}{l}, \quad (1 \leq i, j, k, l \leq n).$$

To compare the convergence in terms of the number of iterations, Fig. 7 shows the results for computing Pareto H-eigenvalues of $A$ from Example 4, when the starting point is $x_0 = [0.3319; 0.8397; 0.3717; 0.8282; 0.1765]$. In this case, both of SPG method and SPP method can find the same Pareto H-eigenvalue 5.2664. SPG1 method needs 27 iterations in 0.44 seconds while SPP method needs to run 53 iterations in 0.21 seconds. SPG2 method just needs 17 iterations in 0.25 seconds.

Figure 8 shows the results for computing Pareto H-eigenvalues of $A$ from Example 5, when the starting point is $x_0 = [0.2291; 0.0922; 0.2409; 0.9025; 0.21734]$. In this case, all of the three methods can find the largest Pareto H-eigenvalue 97.2637. SPG1 method needs 20 iterations in 0.42 seconds while SPG2 method needs 27 iterations in 0.5 seconds. SPP method needs to run 24 iterations in 0.11 seconds.
Figure 9 shows the results for computing Pareto H-eigenvalues of $A$ from Example 6, when the starting point is $x_0 = [0.1846; 0.8337; 0.1696; 0.9532; 0.7225]$. In this case, all of the three methods can find the same Pareto H-eigenvalue 25.6537. SPG2 method needs 24 iterations in 0.22 seconds. And SPG1 method needs 30 iterations in 0.39 seconds while SPP method needs to run 27 iterations in 0.11 seconds.

We also used 100 random starting guesses for finding Pareto H-eigenvalue, to test SPG1, SPG2, and SPP, respectively. For each set of experiments, the same set of random starts was used. We listed the median number of iterations until convergence, and the average run time in the 100 experiments in Table 5. The computed Pareto H-eigenvalues were listed in Figs. 10, 11 and 12 for Examples 4, 5, and 6, respectively. As we can see from the Table 5, SPP method need much more iterations than SPG1, but SPG1 is slightly slower than SPP method. In general, SPG1 is faster than SPG2 for the test problems.
100 random starting points for Example5

The computed Pareto H-eigenvalues from 100 random starting points for Example5 (n=5)

4.3 Comparison with SPP for computing Pareto B-eigenpairs

Now, we compare how well different methods perform for finding Pareto B-eigenpairs of A from Examples 7-8:

Example 7 Let $A \in \mathbb{S}^{[4, n]}$ be the tensor defined in Example 4 and $B \in \mathbb{S}^{[4, n]}$ be the diagonal tensor with $b_{iiii} = \frac{1}{\sin(i)}$ $(1 \leq i \leq n)$ and the other elements are zero.

Example 8 Let $A \in \mathbb{S}^{[4, n]}$ be the symmetric tensor defined by

$$a_{ijkl} = \cos(i + j + k + l) \quad (1 \leq i, j, k, l \leq n),$$

and $B \in \mathbb{S}^{[4, n]}_+$ be the Hilbert tensor defined by

$$b_{ijkl} = \frac{1}{i + j + k + l - 3} \quad (1 \leq i, j, k, l \leq n).$$

The computed Pareto H-eigenvalues from 100 random starting points for Example6

Fig. 12 The computed Pareto H-eigenvalues by SPG1, SPG2, and SPP in the 100 runs on the A from Example 6 (n=5)
Table 5 Comparative results for 100 random test on computing Pareto H-eigenvalues of $A$ from Examples 4-6

| Algorithm | SPG1 | | | SPG2 | | | SPP | |
|-----------|------|---|---|------|---|---|------|---|
| Example   | Its. | Time | | Its. | Time | | Its. | Time |
| Ex. 4     | 33.05 | 0.14 | | 17.49 | 0.14 | | 40.1 | 0.09 |
| Ex. 5     | 20.73 | 0.09 | | 24.95 | 0.25 | | 25.84 | 0.06 |
| Ex. 6     | 12.88 | 0.05 | | 21.32 | 0.17 | | 25.38 | 0.06 |

Fig. 13 Comparison SPG with SPP algorithm for computing B-eigenvalues of $A$ from Example 7 ($n = 5$) with $x_0 = [0.6885; 0.6039; 0.3869; 0.0655; 0.9985]$

Fig. 14 Comparison SPG with SPP algorithm for computing B-eigenvalues of $A$ from Example 8 ($n = 5$) with $x_0 = [1.0; 1.0; 1.0; 1.0; 1.0]$
Table 6 Comparative results for 2500 random test on computing Pareto $B$-eigenvalues of 4-order $n$-dimensional tensor $A$ ($3 \leq n \leq 10$)

| $n$ | 3   | 4   | 5   | 6   | 7   | 8   | 9   | 10  |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|
|     |     |     |     |     |     |     |     |     |
| The median number of iterations |     |     |     |     |     |     |     |     |
| SPG1 | 17.46 | 10.76 | 10.85 | 12.90 | 11.27 | 13.48 | 19.63 | 16.52 |
| SPG2 | 8.33  | 12.88 | 11.73 | 15.47 | 12.92 | 15.11 | 19.48 | 16.89 |
| SPP  | 11.92 | 20.36 | 24.79 | 33.81 | 42.45 | 46.57 | 81.21 | 63.51 |
| The average running time |     |     |     |     |     |     |     |     |
| SPG1 | 0.08  | 0.04  | 0.04  | 0.04  | 0.03  | 0.04  | 0.13  | 0.10  |
| SPG2 | 0.08  | 0.06  | 0.05  | 0.06  | 0.05  | 0.05  | 0.16  | 0.12  |
| SPP  | 0.03  | 0.05  | 0.06  | 0.08  | 0.10  | 0.11  | 0.43  | 0.33  |

As shown in the Figs. 13 and 14, both SPG1 and SPG2 perform better than SPP method significantly. For the number of iterations, SPG1 and SPG2 are superior to SPP. Furthermore, we calculate the $B$-eigenpairs of $A$ for some general cases. Firstly, we randomly generate 50 4-order $n$-dimensional tensor pairs $(A, B)$ for each $n$ ($3 \leq n \leq 10$), where $A$ is generated by the MATLAB function `rand` and then symmetrized to a symmetric tensor, and $B$ is a randomly generated completely positive tensor (which is a class of symmetric positive definite tensors) [37]. And then, we use 50 random starting guesses for finding the $B$-eigenpairs of $A$, to test SPG1, SPG2, and SPP, respectively. The average running time and the median number of iterations are listed in Table 6. As we can see from Table 6, SPG1/SPG2 is more superior to SPP method in terms of running time as the dimension of the tensor increases.

5 Conclusion

In this paper, two monotone ascent spectral projected gradient algorithms were investigated for the tensor eigenvalue complementarity problem (TEiCP). We also presented a shifted scaling-and-projection algorithm, which is a great improvement of the original SPA method [28]. Numerical experiments show that spectral projected gradient methods are efficient and competitive to the shifted projected power method (SPP) for some general cases.

Acknowledgements The authors would like to thank two anonymous referees who have contributed to improve the quality of the paper.

Funding information This work was supported in part by the National Natural Science Foundation of China (No. 11571095, 11501100, 11661007), Program for Innovative Research Team in University of Henan Province (14IRTSTHN023).
References

1. Bader, B.W., Kolda, T.G., et al.: MATLAB Tensor Toolbox Version 2.6, Available online, February 2015. http://www.sandia.gov/~tgkolda/TensorToolbox/
2. Bai, X., Huang, Z., Wang, Y.: Global uniqueness and solvability for tensor complementarity problems. J. Optimiz. Theory App. 170, 72–84 (2016)
3. Barzilai, J., Borwein, J.M.: Two-point step size gradient methods. IMA J. Numer. Anal. 8, 141–148 (1988)
4. Birgin, E.G., Martinez, J.M., Raydan, M.: Nonmonotone spectral projected gradient methods on convex sets. SIAM J. Optim. 10, 1196–1211 (2000)
5. Chang, K.C., Pearson, K.J., Zhang, T.: On eigenvalue problems of real symmetric tensors. J. Math. Anal. Appl. 350, 416–422 (2009)
6. Che, M., Qi, L., Wei, Y.: Positive definite tensors to nonlinear complementarity problems. J. Optimiz. Theory App. 168, 475–487 (2016)
7. Chen, H., Qi, L., Song, Y.: Column sufficient tensors and tensor complementarity problems. Front. Math China 13, 255–276 (2018). https://doi.org/10.1007/s11464-018-0681-4
8. Chen, H., Wang, Y.J.: On computing minimal H-eigenvalue of sign-structured tensors. Front. Math China 12, 1289–1302 (2017)
9. Chen, Y., Dai, Y., Han, D., Sun, W.: Positive semidefinite generalized diffusion tensor imaging via quadratic semidefinite programming. SIAM J. Imaging Sci. 6, 1531–1552 (2013)
10. Chen, Y., Qi, L., Wang, Q.: Computing eigenvalues of large scale Hankel tensors. J. Sci. Comput. 68, 716–738 (2016)
11. Chen, Y., Ye, X.: Projection onto a simplex. arXiv:1101.6081v2 (2011)
12. Chen, Z., Qi, L.: A semismooth Newton method for tensor eigenvalue complementarity problem. Comput. Optim. Appl. 65, 109–126 (2016)
13. Chen, Z., Yang, Q., Ye, L.: Generalized eigenvalue complementarity problem for tensors. Pac. J. Optim. 13(3), 527–545 (2017)
14. Cui, C., Dai, Y., Nie, J.: All real eigenvalues of symmetric tensors. SIAM J. Matrix Anal. Appl. 35, 1582–1601 (2014)
15. Dai, Y.H., Fletcher, R.: Projected Barzilai-Borwein methods for large-scale box-constrained quadratic programming. Numer. Math. 100, 21–47 (2002)
16. Dai, Y.H., Liao, L.Z.: R-linear convergence of the Barzilai and Borwein gradient method. IMA J. Numer. Anal. 22, 1–10 (2002)
17. Ding, W., Luo, Z., Qi, L.: P-tensors, P0-tensors, and tensor complementarity problem, arXiv:1507.06731 (2015)
18. Ding, W., Wei, Y.: Generalized tensor eigenvalue problems. SIAM J. Matrix Anal. Appl. 36, 1073–1099 (2015)
19. Gowda, M.S., Luo, Z., Qi, L., Xiu, N.: Z-tensors and complementarity problems, arXiv:1510.07933 (2015)
20. Hao, C., Cui, C., Dai, Y.: A sequential subspace projection method for extreme Z-eigenvalues of supersymmetric tensors. Numer. Linear Algebr. Appl. 22, 283–298 (2015)
21. Hao, C., Cui, C., Dai, Y.: A feasible trust-region method for calculating extreme Z-eigenvalues of symmetric tensors. Pac. J. Optim. 11, 291–307 (2015)
22. Kofidis, E., Regalia, P.: On the best rank-1 approximation of higher-order supersymmetric tensors. SIAM J. Matrix Anal. Appl. 23, 863–884 (2002)
23. Kolda, T.G., Mayo, J.R.: Shifted power method for computing tensor eigenpairs. SIAM J. Matrix Anal. Appl. 32, 1095–1124 (2011)
24. Kolda, T.G., Mayo, J.R.: An adaptive shifted power method for computing generalized tensor eigenpairs. SIAM J. Matrix Anal. Appl. 35, 1563–1581 (2014)
25. Li, G., Qi, L., Yu, G.: The Z-eigenvalues of a symmetric tensor and its application to spectral hypergraph theory. Numer. Linear Algebr. 20, 1001–1029 (2013)
26. Li, G., Qi, L., Yu, G.: Semismoothness of the maximum eigenvalue function of a symmetric tensor and its application. Linear Algebra Appl. 438, 813–833 (2013)
27. Lim, L.H.: Singular values and eigenvalues of tensors, a variational approach. In: Proc. 1st IEEE International Workshop on Computational Advances of Multi-tensor Adaptive Processing, pp. 129–132 (2005)
28. Ling, C., He, H., Qi, L.: On the cone eigenvalue complementarity problem for higher-order tensors. Comput. Optim. Appl. 63, 143–168 (2016)
29. Ling, C., He, H., Qi, L.: Higher-degree eigenvalue complementarity problem for tensors. Comput. Optim. Appl. 64, 149–176 (2016)
30. Luo, Z., Qi, L., Xiu, N.: The sparsest solutions to Z-tensor complementarity problems. Optim. Lett. 11, 471–482 (2017)
31. Ni, Q., Qi, L.: A quadratically convergent algorithm for finding the largest eigenvalue of a nonnegative homogeneous polynomial map. J. Global Optim. 61, 627–641 (2015)
32. Nie, J., Wang, L.: Semidefinite relaxations for best rank-1 tensor approximations. SIAM J. Matrix Anal. A 35, 1155–1179 (2014)
33. Qi, L.: Eigenvalues of a real supersymmetric tensor. J. Symb. Comput. 40, 1302–1324 (2005)
34. Qi, L.: Eigenvalues and invariants of tensor. J. Math. Anal. Appl. 325, 1363–1377 (2007)
35. Qi, L.: Symmetric nonnegative tensors and copositive tensors. Linear Algebra Appl. 439, 228–238 (2013)
36. Qi, L., Wang, Y., Wu, E.X.: D-Eigenvalues of diffusion kurtosis tensors. J. Comput. Appl. Math. 221, 150–157 (2008)
37. Qi, L., Xu, C., Xu, Y.: Nonnegative tensor factorization, completely positive tensors and a hierarchical elimination algorithm. SIAM J. Matrix Anal. Appl. 35, 1227–1241 (2014)
38. Qi, L., Yu, G., Wu, E.X.: Higher order positive semi-definite diffusion tensor imaging. SIAM J. Imag. Sci. 3, 416–433 (2010)
39. Qi, L., Yu, G., Xu, Y.: Nonnegative diffusion orientation distribution function. J. Math. Imag. Vis. 45, 103–113 (2013)
40. Raydan, M.: On the Barzilai and Borwein choice of steplength for the gradient method. IMA J. Numer. Anal. 13, 321–326 (1993)
41. Raydan, M.: The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem. SIAM J. Optim. 7, 26–33 (1997)
42. Song, Y., Qi, L.: Eigenvalue analysis of constrained minimization problem for homogeneous polynomial. J. Global Optim. 64, 563–575 (2016)
43. Song, Y., Qi, L.: Tensor complementarity problem and semi-positive tensors. J. Optim. Theory Appl. 169, 1069–1078 (2016)
44. Song, Y., Yu, G.: Properties of solution set of tensor complementarity problem. J. Optim. Theory Appl. 170, 85–96 (2016)
45. Wang, G., Zhou, G.L., Caccetta, L.: Z-eigencvalue inclusion theorems for tensors. Discret. Cont. Dyn. Syst. - Series B 22, 187–198 (2017)
46. Wang, X., Chen, H., Wang, Y.: Solution structures of tensor complementarity problem. Front. Math. China (2018) https://doi.org/10.1007/s11446-018-0675-2
47. Wang, Y., Huang, Z., Bai, X.: Exceptionally regular tensors and tensor complementarity problems. Optim. Methods Softw. 31, 815–828 (2016)
48. Wang, Y.J., Caccetta, L., Zhou, G.L.: Convergence analysis of a block improvement method for polynomial optimization over unit spheres. Numer. Linear Algebra Appl. 22, 1059–1076 (2015)
49. Wang, Y.J., Qi, L., Zhang, X.: A practical method for computing the largest M-eigenvalue of a fourth-order partially symmetric tensor. Numer. Linear Algebra Appl. 16, 589–601 (2009)
50. Wang, Y.J., Zhou, G.L.: A hybrid second-order method for homogenous polynomial optimization over unit sphere. J. Oper. Res. Soc. China 5, 99–109 (2017)
51. Xu, F., Ling, C.: Some properties on Pareto-eigenvalues of higher-order tensors. Oper. Res. Trans. 19, 34–41 (2015)
52. Yang, L., Yang, Q., Zhao, X.: Quadratic third-order tensor optimization problem with quadratic constraints. Statist. Optim. Inf. Comput. 2, 130–146 (2014)
53. Yang, Y., Yang, Q.: Further results for Perron-Frobenius theorem for nonnegative tensors. SIAM J. Matrix Anal. Appl. 31, 2517–2530 (2010)
54. Yu, G., Yu, Z., Xu, Y., Song, Y., Zhou, Y.: An adaptive gradient method for computing generalized tensor eigenpairs. Comput. Optim. Appl. 65, 781–797 (2016)
55. Zhou, G., Caccetta, L., Teo, K., Wu, S.: Positive polynomial optimization over unit spheres and convex programming relaxations. SIAM J. Optim. 22, 987–1008 (2012)
56. Zhou, G., Wang, G., Qi, L., Alqahtani, M.: A fast algorithm for the spectral radii of weakly reducible nonnegative tensors. Numer. Linear Algebra Appl. https://doi.org/10.1002/nla.2134 (2018)