AIMS Mathematics

http://www.aimspress.com/journal/Math

AIMS Mathematics, 8(7): 15008–15023.
DOI: 10.3934/math.2023766
Received: 06 February 2023
Revised: 23 March 2023
Accepted: 30 March 2023
Published: 23 April 2023

Research article

An accelerated conjugate gradient method for the Z-eigenvalues of symmetric tensors

Mingyuan Cao[1], **Yueting Yang**[1,*], **Chaoqian Li**[2] **and Xiaowei Jiang**[1,*]

[1] School of Mathematics and Statistics, Beihua University, Jilin, Jilin 132013, China
[2] School of Mathematics and Statistics, Yunnan University, Kunming, Yunnan 650091, China

* **Correspondence:** Email: yyt2858@163.com; jxwbh@126.com.

Abstract: We transform the Z-eigenvalues of symmetric tensors into unconstrained optimization problems with a shifted parameter. An accelerated conjugate gradient method is proposed for solving these unconstrained optimization problems. If solving problem results in a nonzero critical point, then it is a Z-eigenvector corresponding to the Z-eigenvalue. Otherwise, we solve the shifted problem to find a Z-eigenvalue. In our method, the new conjugate gradient parameter is a modified CD conjugate gradient parameter, and an accelerated parameter is presented by using the quasi-Newton direction. The global convergence of new method is proved. Numerical experiments are listed to illustrate the efficiency of the proposed method.

Keywords: symmetric tensors; Z-eigenvalues; accelerated conjugate gradient; variational; global convergence
Mathematics Subject Classification: 15A18, 15A69, 90C55

1. Introduction

Since Lim [1] and Qi [2] presented some theories of the eigenvalues and eigenvectors for higher order tensors, the related research has received much more attention (see [3–13], etc). The eigenvalues of symmetric tensors have been applied in blind source separation [2] and hypergraph theory [9], statistical data analysis [14] and high order Markov chains [15], etc. Moreover, various definitions of eigenvalues and eigenvectors for tensors have been introduced [2, 10, 16].

There are many works for computing the eigenvalues of tensors, especially for Z-eigenvalues. Qi et al. [17] proposed an elimination method for finding all Z-eigenvalues, which is specific to the third-order tensors. Kolda and Mayo [6] presented a shifted power method (SPM) for calculating Z-eigenvalues, in which the shifted parameter is crucial. Han [18] provided an unconstrained optimization approach for even order symmetric tensors. Hao, Cui and Dai [4] found the extreme

Z-eigenvalues and corresponding Z-eigenvectors by the sequential subspace projection method. Under certain assumptions, the global convergence and linear convergence were established for symmetric tensors. Hao, Cui and Dai [19] proposed a feasible trust region method for finding the extreme Z-eigenvalues of symmetric tensors. The global convergence and local quadratic convergence were established.

Inspired by the idea of improved conjugate parameters proposed in the above works and the application of optimization methods in tensor eigenvalue calculation, our main work is to study the transformation of Z-eigenvalues of symmetric tensors into unconstrained optimization and to propose a new algorithm. The contributions of this article are listed as follows:

For the case of different critical point, we transform the Z-eigenvalues of symmetric tensors into different unconstrained optimization problems which include shifted problem.

We propose a new conjugate gradient method with a new conjugate gradient parameter and an accelerated parameter, which converges to a critical point. The found nonzero critical point is a Z-eigenvector associated with a Z-eigenvalue of symmetric tensors. When the zero critical point is obtained, a shifted problem is solved for finding a Z-eigenvalue.

The global convergence of new method is established. We compare our method with conjugate gradient methods proposed in [20, 21], for computing the Z-eigenvalues of symmetric tensors. The numerical results show that the proposed method is competitive.

The rest of this paper is organized as follows. In Section 2, we transform the Z-eigenvalues problem into unconstrained optimizations, and propose an accelerated conjugate gradient method for solving it. Global convergence result is established in Section 3. Numerical experiments are shown in Section 4.

2. New method for the Z-eigenvalues of symmetric tensors

Let $\mathbb{R}$ be the real field, m, n be positive integers and

$$\mathcal{A} = (a_{i_1 i_2 \cdots i_m}),\ a_{i_1 i_2 \cdots i_m} \in \mathbb{R}, 1 \le i_1, \cdots, i_m \le n$$

be an mth-order n-dimensional real tensor. The set of mth-order n-dimensional real tensor is denoted by $\mathbb{R}^{[m,n]}$. Tensor $\mathcal{A}$ is symmetric if its entries are invariant under any permutation of their indices. The set of mth-order n-dimensional real symmetric tensor is denoted by $\mathbb{S}^{[m,n]}$.

If $\mathcal{A} \in \mathbb{R}^{[m,n]}$, an mth-degree homogeneous polynomial function with real coefficients is uniquely determined by

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

For $x = (x_1, \cdots, x_n)^{\mathrm{T}} \in \mathbb{R}^n$, $\mathcal{A}x^{m-1}$ denotes a n-dimensional column vector, i.e.,

$$\mathcal{A}x^{m-1} := \left(\sum_{i_2, \cdots, i_m = 1}^{n} a_{i i_2 \cdots i_m} x_{i_2} \cdots x_{i_m} \right)_{1 \le i \le n}.$$

If $\mathcal{A}$ is symmetric, then the gradient of $\mathcal{A}x^m$ satisfies

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

for all $x \in \mathbb{R}^n$. In our work, we consider the following Z-eigenvalues of symmetric tensors.

Definition 1. [2] Let $\mathcal{A} \in \mathbb{R}^{[m,n]}$, if there exist $\lambda \in \mathbb{R}$ and a vector $x \in \mathbb{R}^n \setminus \{0\}$ satisfying

$$\begin{aligned} \mathcal{A}x^{m-1} &= \lambda x, \\ x^T x &= 1, \end{aligned} \tag{2.1}$$

then λ is called a Z-eigenvalue of $\mathcal{A}$ and x is called the corresponding Z-eigenvector.

Motivated by the work of Auchmuty [22], we generalize the unconstrained variational principles to Z-eigenvalues of $\mathcal{A}$. Consider the following unconstrained optimization

$$\min_{x\in\mathbb{R}^n} f(x) = \frac{1}{2m}(x^{\mathrm{T}}x)^m - \frac{1}{m}\mathcal{A}x^m. \tag{2.2}$$

The gradient and Hessian of $f(x)$ are listed as follows

$$g(x) := \nabla f(x) = (x^{\mathrm{T}}x)^{m-1}x - \mathcal{A}x^{m-1}, \tag{2.3}$$

$$G(x) := \nabla^2 f(x) = (x^{\mathrm{T}}x)^{m-1}I + 2(m-1)(x^{\mathrm{T}}x)^{m-2}xx^{\mathrm{T}} - (m-1)\mathcal{A}x^{m-2}, \tag{2.4}$$

where

$$\mathcal{A}x^{m-2} := \left(\sum_{i_3,\cdots,i_m=1}^{n} a_{iji_3\cdots i_m}x_{i_3}\cdots x_{i_m} \right)_{1\le i,j\le n}.$$

Obviously, $G(x)$ is a symmetric matrix. In order to research the properties of $f(x)$ in (2.2), we cite the following definition and a nice feature of it.

Definition 2. [23] A continuous function $h : \mathbb{R}^n \to \mathbb{R}$ is called coercive if it satisfies

$$\lim_{\|x\|\to\infty} h(x) = +\infty.$$

If x satisfies the equation $\nabla h(x) = 0$, then it is termed as a critical point of $h(x)$.

Theorem 1. [23] Let $h : \mathbb{R}^n \to \mathbb{R}$ be continuous. If h is coercive, then h has at least one global minimizer. In addition, if the first partial derivatives exist on $\mathbb{R}^n$, then h attains its global minimizers at its critical points.

Based on a similar argument, for the Z-eigenvalues of tensors, we have the following result.

Theorem 2. Let $\mathcal{A} \in \mathbb{R}^{[m,n]}$ be symmetric tensors. Assume that $\lambda_{\max}$ is the largest Z-eigenvalue of $\mathcal{A}$. Denote the Z-spectrum of $\mathcal{A}$ by $\sigma_Z(\mathcal{A}) := \{\lambda : \lambda \text{ is a Z-eigenvalue of } \mathcal{A}\}$. We have

(i) $f(x)$ is coercive on $\mathbb{R}^n$.

(ii) The critical points of $f(x)$ are at $x = 0$ and any Z-eigenvector $x \neq 0$ associated with a Z-eigenvalue $\lambda > 0$ of $\mathcal{A}$ satisfying $\lambda = (x^{\mathrm{T}}x)^{m-1}$.

(iii) If $\lambda_{\max} > 0$, then $f(x)$ attains its global minimal value

$$f_{\min} = -\frac{1}{2m}(\lambda_{\max})^{\frac{m}{m-1}}$$

at any Z-eigenvector associated with the Z-eigenvalue $\lambda_{\max}$ such that $\lambda_{\max} = (x^{\mathrm{T}}x)^{m-1}$.

(iv) If $\lambda_{\max} \le 0$, then $x = 0$ is the unique critical point of $f(x)$. Moreover, it is the unique global minimizer of $f(x)$ on $\mathbb{R}^n$.

Proof. (i) Since
$$f(x) = \frac{1}{2m}(x^{\mathrm{T}}x)^m - \frac{1}{m}\mathcal{A}x^m = \frac{1}{2m}\|x\|^{2m} - \frac{1}{m}\mathcal{A}x^m$$
and $\mathcal{A}x^m$ is an *mth*-degree homogeneous polynomial function with real coefficients, then

$$f(x) \to \infty \text{ as } \|x\| \to \infty.$$

That is, $f(x)$ is coercive on $\mathbb{R}^n$.

(ii) From the definition of the critical point of $f(x)$, we have

$$\mathcal{A}x^{m-1} = (x^{\mathrm{T}}x)^{m-1}x. \tag{2.5}$$

It is obvious that $x = 0$ is a critical point of $f(x)$ as $g(0) = 0$. The point $x \in \mathbb{R}^n\backslash\{0\}$ satisfying (2.5) is a Z-eigenvector corresponding to the Z-eigenvalue $\lambda = (x^{\mathrm{T}}x)^{m-1} > 0$, which is also a critical point of $f(x)$.

(iii) At the critical point $x \in \mathbb{R}^n\backslash\{0\}$, a Z-eigenvector x associated with a Z-eigenvalue λ satisfies $\lambda = (x^{\mathrm{T}}x)^{m-1}$ and $\mathcal{A}x^m = \lambda x^{\mathrm{T}}x$. Moreover,

$$f(x) \;=\; \frac{1}{2m}\lambda x^{\mathrm{T}}x - \frac{1}{m}\lambda x^{\mathrm{T}}x = -\frac{1}{2m}\lambda x^{\mathrm{T}}x = -\frac{1}{2m}\lambda^{\frac{m}{m-1}} \geq -\frac{1}{2m}(\lambda_{\max})^{\frac{m}{m-1}}.$$

By Theorem 2.1 and conclusion (ii), we get the global minimum value $-\frac{1}{2m}(\lambda_{\max})^{\frac{m}{m-1}}$ at any Z-eigenvector x corresponding to the Z-eigenvalue $\lambda_{\max}$ such that $\lambda_{\max} = (x^{\mathrm{T}}x)^{m-1}$.

(iv) Since $\lambda_{\max} \leq 0$ implies that $\lambda \leq 0$ for any $\lambda \in \sigma_Z(\mathcal{A})$, $\lambda = (x^{\mathrm{T}}x)^{m-1}$ does not hold for any Z-eigenvector x associated with a Z-eigenvalue λ, as $(x^{\mathrm{T}}x)^{m-1} > 0$ for any $x \in \mathbb{R}^n\backslash\{0\}$. Therefore, from Theorem 2.1, $x = 0$ is the unique critical point and the unique global minimize of $f(x)$.

Note that, if $\lambda_{\max} \leq 0$, then $x = 0$ is the unique critical point, but it does not result in a Z-eigenvalue. In this case, we solve a following shifted problem

$$\min_{x\in\mathbb{R}^n} f_t(x) = \frac{1}{2m}(x^{\mathrm{T}}x)^m - \frac{1}{m}\mathcal{A}x^m - \frac{t}{2}(x^{\mathrm{T}}x), \tag{2.6}$$

where $t > 0$ is a shifted parameter. It is obvious that, when t is sufficient large, for any $x \neq 0$, we have $f_t(x) < 0$. From $f_t(0) = 0$, we know that $x = 0$ is the unique maximizer of $f_t(x)$. Denote the gradient of $f_t(x)$ as
$$g_t(x) := \nabla f_t(x) = (x^{\mathrm{T}}x)^{m-1}x - \mathcal{A}x^{m-1} - tx. \tag{2.7}$$

Obviously, $x = 0$ is also a critical point of $f_t(x)$. The nonzero critical point of problem (2.6) is a Z-eigenvector corresponding to Z-eigenvalue $\lambda_t = (x_t^{\mathrm{T}}x_t)^{m-1} - t$. In this case, a suitable descent algorithm for solving shifted problem (2.6) should converge to a nonzero critical point. Therefore, we can get Z-eigenvalues and its associated Z-eigenvectors by solving problem (2.2) or (2.6). The algorithm is described as follows. □

Algorithm 1. *Step 0: Given* $\mathcal{A} \in \mathbb{S}^{[m,n]}, t \geq 1, \bar{\rho} > 1$.

Step 1: Solving problem (2.2) by using an algorithm to obtain x_k *and compute* $\lambda_k = (x_k^{\mathrm{T}}x_k)^{m-1}$.

Step 2: If $\|x_k\| \leq \varepsilon$*, stop, output* x_k *and compute* $\lambda_k = (x_k^{\mathrm{T}}x_k)^{m-1} - t$;

otherwise, let $t := \bar{\rho}t$*, go to Step 3.*

Step 3: Solve problem (2.6) by using an algorithm to obtain x_k*, go to Step 2.*

Remark 1. There is an inner loop between Step 2 and Step 3. Since descent algorithm for solving problem (2.6) will result in a nonzero critical point, then this inner loop can be terminated by finite iteration for sufficient large t. Therefore, Algorithm 1 is well defined.

Remark 2. When executing algorithm, it should use the same unconstrained optimization method to solve problems (2.2) and (2.6). We will propose a new accelerated conjugate gradient method, especially for solving problem (2.2) or (2.6).

3. An accelerated conjugate gradient algorithm and its Convergence

In this section, we devote to giving an accelerated conjugate gradient method for solving unconstrained optimization problem, such as (2.2) or (2.6). Firstly, we consider the iterative formula of nonlinear conjugate gradient algorithm

$$x_{k+1} = x_k + \alpha_k d_k. \tag{3.1}$$

The stepsize $\alpha_k > 0$ is determined by a line search and the direction d_k are computed by

$$d_{k+1} = -\theta_{k+1} g_{k+1} + \beta_{k+1} s_k,\ d_0 = -g_0, \tag{3.2}$$

where $g_{k+1} = g(x_{k+1})$, $s_k = \alpha_k d_k$.

We first introduce a new conjugate parameter of our conjugate gradient method based on CD nonlinear conjugate gradient method ([20]). The new conjugate parameter is

$$\beta_{k+1} = \frac{\|g_{k+1}\|^2 s_k^{\mathrm{T}} y_k}{(g_k^{\mathrm{T}} s_k)^2} \tag{3.3}$$

is introduced. When using the exact line search, β_{k+1} in (3.3) reduces to CD conjugate gradient parameter. An accelerated parameter θ_{k+1} is obtained by the quasi-Newton direction ([24, 25]). Let $d_{k+1} = -G_{k+1}^{-1} g_{k+1}$, namely

$$-G_{k+1}^{-1} g_{k+1} = -\theta_{k+1} g_{k+1} + \beta_{k+1} s_k, \tag{3.4}$$

where G_{k+1} satisfies the following secant equation

$$G_{k+1} s_k = y_k,\ y_k = g_{k+1} - g_k. \tag{3.5}$$

From (3.4), then

$$g_{k+1} = \theta_{k+1} G_{k+1} g_{k+1} - \beta_{k+1} G_{k+1} s_k.$$

Pre-multiplying at the both sides by s_k^{T}, we have

$$s_k^{\mathrm{T}} g_{k+1} = \theta_{k+1} s_k^{\mathrm{T}} G_{k+1} g_{k+1} - \beta_{k+1} s_k^{\mathrm{T}} G_{k+1} s_k. \tag{3.6}$$

Then, combined with (3.3)–(3.6), we have

$$\theta_{k+1} = \frac{1}{y_k^{\mathrm{T}} g_{k+1}} \left[\frac{\|g_{k+1}\|^2 (s_k^{\mathrm{T}} y_k)^2}{(g_k^{\mathrm{T}} s_k)^2} + s_k^{\mathrm{T}} g_{k+1} \right]. \tag{3.7}$$

The stepsize is generated by the strong Wolfe line search conditions

$$f(x_k + \alpha_k d_k) - f(x_k) \leq \rho \alpha_k g_k^{\mathrm{T}} d_k, \tag{3.8}$$

$$|g(x_k + \alpha_k d_k)^T d_k| \leq -\sigma g_k^{\mathrm{T}} d_k, \tag{3.9}$$

where $0 < \rho < \sigma < 1$. We prove the descent property of the direction (3.2) under (3.8) and (3.9) in the following. Multiply both sides of (3.9) by α_k, from $s_k = \alpha_k d_k$, we can easily obtain $(g_{k+1}^{\mathrm{T}} s_k)^2 \leq \sigma^2(-g_k^{\mathrm{T}} s_k)^2$.

Theorem 3. If $\theta_{k+1} \geq \frac{1}{2} + 2\sigma^2$, then the direction determined by (3.2) satisfies the sufficient descent condition

$$g_{k+1}^{\mathrm{T}} d_{k+1} \leq -\frac{1}{2}\|g_{k+1}\|^2. \tag{3.10}$$

Proof. Multiplying (3.2) by g_{k+1}^{T}, we obtain

$$g_{k+1}^{\mathrm{T}} d_{k+1} = -\theta_{k+1}\|g_{k+1}\|^2 + \frac{g_{k+1}^{\mathrm{T}} s_k}{-g_k^{\mathrm{T}} s_k}\|g_{k+1}\|^2 + \frac{(g_{k+1}^{\mathrm{T}} s_k)^2\|g_{k+1}\|^2}{(g_k^{\mathrm{T}} s_k)^2}. \tag{3.11}$$

Using the inequality $a^{\mathrm{T}} b \leq \frac{1}{2}(\|a\|^2 + \|b\|^2)$, where $a, b \in \mathbb{R}^n$, we have

$$\begin{aligned} \frac{g_{k+1}^{\mathrm{T}} s_k\|g_{k+1}\|^2}{-g_k^{\mathrm{T}} s_k} &= \frac{\left[(-g_k^{\mathrm{T}} s_k) g_{k+1}/\sqrt{2}\right]^{\mathrm{T}}\left[\sqrt{2}(g_{k+1}^{\mathrm{T}} s_k) g_{k+1}\right]}{(-g_k^{\mathrm{T}} s_k)^2} \\ &\leq \frac{\frac{1}{2}\left[\frac{1}{2}(-g_k^{\mathrm{T}} s_k)^2\|g_{k+1}\|^2 + 2(g_{k+1}^{\mathrm{T}} s_k)^2\|g_{k+1}\|^2\right]}{(-g_k^{\mathrm{T}} s_k)^2} \\ &= \frac{1}{4}\|g_{k+1}\|^2 + \frac{(g_{k+1}^{\mathrm{T}} s_k)^2\|g_{k+1}\|^2}{(-g_k^{\mathrm{T}} s_k)^2}. \end{aligned} \tag{3.12}$$

Substituted (3.12) into (3.11), we have

$$g_{k+1}^{\mathrm{T}} d_{k+1} \leq -\theta_{k+1}\|g_{k+1}\|^2 + \frac{1}{4}\|g_{k+1}\|^2 + 2\frac{(g_{k+1}^{\mathrm{T}} s_k)^2\|g_{k+1}\|^2}{(g_k^{\mathrm{T}} s_k)^2}. \tag{3.13}$$

Then from $(g_{k+1}^{\mathrm{T}} s_k)^2 \leq \sigma^2(-g_k^{\mathrm{T}} s_k)^2$ and $\theta_{k+1} \geq \frac{1}{2} + 2\sigma^2$, we have

$$\begin{aligned} g_{k+1}^{\mathrm{T}} d_{k+1} &\leq -\theta_{k+1}\|g_{k+1}\|^2 + \frac{1}{4}\|g_{k+1}\|^2 + 2\sigma^2\|g_{k+1}\|^2 \\ &= -\left(\theta_{k+1} - 2\sigma^2 - \frac{1}{4}\right)\|g_{k+1}\|^2 \leq -\frac{1}{2}\|g_{k+1}\|^2 < 0. \end{aligned} \tag{3.14}$$

The proof is completed. □

Now, we describe an accelerated conjugate gradient algorithm (ACG).

ACG algorithm

Step 0: Given $x_0 \in \mathbb{R}^n, \varepsilon \geq 0$ and $0 < \rho < \sigma < 1$. Compute g_0, let $d_0 = -g_0$. Set $k := 0$.

Step 1: If $\|g_k\| \leq \varepsilon$, stop, output x_k. Otherwise, calculate α_k from (3.8) and (3.9). Let $x_{k+1} = x_k + \alpha_k d_k$ and $s_k = \alpha_k d_k$.

Step 2: Compute $g_{k+1}, y_k, \beta_{k+1}$ by (3.3) and θ_{k+1} by (3.7). Let $\theta_{k+1} := \max\{\theta_{k+1}, 2\sigma^2 + \frac{1}{2}\}$.

Step 3: Using (3.2) to obtain d_{k+1}. Set $k := k + 1$ and go to step 1.

Now, we establish the convergence result of ACG algorithm. Let the level set $\Omega = \{x \in \mathbb{R}^n | f(x) \leq f(x_0)\}$ be a bounded closed set, i.e., there exists a constant $\gamma > 0$ such that $\|x\| \leq \gamma$ for all $x \in \Omega$. To facilitate analyzing, denote $\mathcal{A}_{i_1} = (a_{i_1 i_2 \cdots i_m})_{1 \leq i_2, i_3, \cdots, i_m \leq n}$ and $\mathcal{A}_{i_1 i_2} = (a_{i_1 i_2 \cdots i_m})_{1 \leq i_3, i_4, \cdots, i_m \leq n}$.

Lemma 1. Consider tensors $\mathcal{A} \in \mathcal{S}^{[m,n]}$, then $\mathcal{A}x^m$ is Lipschitz continuous on Ω.

Proof. Let $p(x) = \mathcal{A}x^m$, mathematical induction is adopted. If $m = 1$, $p(x) = \sum\limits_{i=1}^{n} a_i x_i$, utilizing equivalence of vector norm, for all $x, y \in \Omega$, we have

$$\begin{aligned} \|p(x) - p(y)\| &= \left| \sum_{i=1}^{n} a_i x_i - \sum_{i=1}^{n} a_i y_i \right| = \left| \sum_{i=1}^{n} a_i (x_i - y_i) \right| \leq \sum_{i=1}^{n} |a_i||x_i - y_i| \\ &\leq \max_{i=1,2,\cdots,n} |a_i| \left(\sum_{i=1}^{n} |x_i - y_i| \right) \leq \max_{i=1,2,\cdots,n} |a_i| \|x - y\|_\infty \leq P_2 \|x - y\|. \end{aligned}$$

Assume that the statements satisfy for all order $m \leq k - 1$. When $m = k$, we have

$$\begin{aligned} \|p(x) - p(y)\| &= \left| \mathcal{A}x^k - \mathcal{A}y^k \right| \\ &= \left| \sum_{i_1, i_2, \cdots, i_k = 1}^{n} a_{i_1 i_2 \cdots i_k} x_{i_1} x_{i_2} \cdots x_{i_k} - \sum_{i_1, i_2, \cdots, i_k = 1}^{n} a_{i_1 i_2 \cdots i_k} y_{i_1} y_{i_2} \cdots y_{i_k} \right| \\ &= \left| \sum_{i_1=1}^{n} (x_{i_1} \mathcal{A}_{i_1} x^{k-1} - y_{i_1} \mathcal{A}_{i_1} y^{k-1}) \right| \\ &= \left| \sum_{i_1=1}^{n} (x_{i_1} - y_{i_1}) \mathcal{A}_{i_1} x^{k-1} + y_{i_1} (\mathcal{A}_{i_1} x^{k-1} - \mathcal{A}_{i_1} y^{k-1}) \right| \\ &\leq \sum_{i_1=1}^{n} (|x_{i_1} - y_{i_1}| \|\mathcal{A}_{i_1} x^{k-1}\| + |y_{i_1}| \|\mathcal{A}_{i_1} x^{k-1} - \mathcal{A}_{i_1} y^{k-1}\|). \end{aligned}$$

Since Ω is a bounded close set, then $\|\mathcal{A}_{i_1} x^{k-1}\|$ is bounded on Ω and $\|z\| \leq P_1$. From $\sum\limits_{i_1=1}^{n} |x_{i_1} - y_{i_1}| \leq \|x - y\|$ and $\sum\limits_{i_1=1}^{n} |y_{i_1}| \leq \|z\|$, there exists a positive constant P_3 such that

$$\|p(x) - p(y)\| \leq P_3 \|x - y\|.$$

Namely, $\mathcal{A}x^m$ is Lipschitz continuous on Ω. The proof is completed. □

Lemma 2. If $\mathcal{A} \in \mathcal{S}^{[m,n]}$, then $\mathcal{A}x^{m-1}$ and $\mathcal{A}x^{m-2}$ are Lipschitz continuous on Ω.

Proof. For all $x, y \in \Omega$, using Lemma 3.1 and equivalence of norm, we have

$$\begin{aligned}
\|\mathcal{A}x^{m-1} - \mathcal{A}y^{m-1}\| &\leq M_1\|(\mathcal{A}_{i_1}x^{m-1} - \mathcal{A}_{i_1}x^{m-1})_{1\leq i_1\leq n}\|_\infty \\
&= M_1 \max_{1\leq i_1\leq n} \left|\mathcal{A}_{i_1}x^{m-1} - \mathcal{A}_{i_1}x^{m-1}\right| \\
&\leq M_1 \max_{1\leq i_1\leq n} P_{i_1}\|x - y\| \\
&= P_4\|x - y\|, \qquad (3.15)
\end{aligned}$$

where P_4 depends on tensor $\mathcal{A}$ and set Ω.

Similarly, for all $x, y \in \Omega$, using Lemma 3.1 and equivalence of norm, we have

$$\begin{aligned}
\|\mathcal{A}x^{m-2} - \mathcal{A}y^{m-2}\| &\leq M_2\|(\mathcal{A}_{i_1i_2}x^{m-2} - \mathcal{A}_{i_1i_2}y^{m-2})_{1\leq i_1,i_2\leq n}\|_\infty \\
&= M_2 \max_{1\leq i_1\leq n} \sum_{i_2=1}^{n} \left|\mathcal{A}_{i_1i_2}x^{m-2} - \mathcal{A}_{i_1i_2}y^{m-2}\right| \\
&\leq M_2 \max_{1\leq i_1,i_2\leq n} \sum_{i_2=1}^{n} P_{i_1i_2}\|x - y\| \\
&= P_5\|x - y\|, \qquad (3.16)
\end{aligned}$$

where P_5 depends on tensor $\mathcal{A}$ and set Ω.

□

Lemma 3. If $\mathcal{A} \in \mathcal{S}^{[m,n]}$, then $g(x)$ is Lipschitz continuous in a neighbourhood $\mathbb{N}$ of Ω, namely

$$\|g(x) - g(y)\| \leq L\|x - y\| \qquad (3.17)$$

holds for any $x, y \in \mathbb{N}$, where L is a positive number.

Proof. There are two cases of gradient $g(x)$ to consider. One case is computed by (2.3) and the other case is computed by (2.7).

For (2.3): since $\mathbb{N}$ is a bound closed set, from Lemma 3.1, for all $x, y \in \mathbb{N}$, we have

$$\begin{aligned}
\|g(x) - g(y)\| &= \|(x^{\mathrm{T}}x)^{m-1}x - \mathcal{A}x^{m-1} - (y^{\mathrm{T}}y)^{m-1}x + \mathcal{A}y^{m-1}\| \\
&\leq \|\mathcal{A}y^{m-1} - \mathcal{A}x^{m-1}\| + \|(x^{\mathrm{T}}x)^{m-1}x - (y^{\mathrm{T}}y)^{m-1}x\| \\
&\leq P_4\|x - y\| + P_6\|x - y\| = (P_4 + P_6)\|x - y\|. \qquad (3.18)
\end{aligned}$$

For (2.7): since $\mathbb{N}$ is a bound closed set, from Lemma 3.1, for all $x, y \in \mathbb{N}$, we have

$$\begin{aligned}
\|g(x) - g(y)\| &= \|(x^{\mathrm{T}}x)^{m-1}x - \mathcal{A}x^{m-1} - tx - (y^{\mathrm{T}}y)^{m-1}x + \mathcal{A}y^{m-1} + ty\| \\
&\leq \|\mathcal{A}y^{m-1} - \mathcal{A}x^{m-1}\| + \|(x^{\mathrm{T}}x)^{m-1}x - (y^{\mathrm{T}}y)^{m-1}x\| + \|ty - tx\| \\
&\leq P_4\|x - y\| + P_6\|x - y\| + P_7\|x - y\| = (P_4 + P_6 + P_7)\|x - y\|. \qquad (3.19)
\end{aligned}$$

The proof is completed. □

We can easily get that there exists a constant $M > 0$ such that $\|g(x)\| \leq M$. The following useful lemma was essentially which proved by Zoutendijk [26]. From Theorem 3.1, we can obtain that the sequence $\{d_k\}$ generated by ACG algorithm satisfies the following Lemmas.

Lemma 4. Let the sequences $\{x_k\}$ and $\{d_k\}$ be generated by ACG algorithm, we have

$$\sum_{k=0}^{\infty} \frac{(g_k^{\mathrm{T}} d_k)^2}{\|d_k\|^2} < \infty.$$

Lemma 5. Let the sequence $\{x_k\}$ be generated by ACG algorithm. If

$$\sum_{k\geq 0} \frac{1}{\|d_k\|^2} = \infty,$$

then

$$\liminf_{k\to\infty} \|g_k\| = 0.$$

From Theorem 3.1, (3.17) and Lemma 3.4, the result can be proved, which is omitted here.

Theorem 4. Let the sequence $\{x_k\}$ be generated by ACG algorithm. Then, we have

$$\liminf_{k\to\infty} \|g_k\| = 0. \tag{3.20}$$

Proof. From Theorem 3.1, there exists a constant $c < 0$ satisfying $g_k^{\mathrm{T}} s_k \leq c\|g_k\|\|s_k\| < 0$, i.e., $-g_k^{\mathrm{T}} s_k \geq -c\|g_k\|\|s_k\|$. Then, we have

$$\beta_{k+1} \leq \frac{\|g_{k+1}\|^2}{-g_k^{\mathrm{T}} s_k}(1+\sigma) \leq \frac{\|g_{k+1}\|^2}{-c\|g_k\|\|s_k\|}(1+\sigma) \leq \frac{M(1+\sigma)}{-c\|s_k\|} = \frac{\xi}{\|s_k\|},$$

where $\xi = \frac{M(1+\sigma)}{-c}$. According to $|g_{k+1}^{\mathrm{T}} d_k| \leq -\sigma g_k^{\mathrm{T}} d_k$ and $\|s_k\| \leq 2\gamma$, we have

$$\begin{aligned}
\theta_{k+1} &= \frac{1}{y_k^{\mathrm{T}} g_{k+1}} \left[\frac{\|g_{k+1}\|^2 (s_k^{\mathrm{T}} y_k)^2}{(g_k^{\mathrm{T}} s_k)^2} + s_k^{\mathrm{T}} g_{k+1} \right] \\
&= \frac{\|g_{k+1}\|^2 (s_k^{\mathrm{T}} y_k)^2}{(g_k^{\mathrm{T}} s_k)^2) y_k^{\mathrm{T}} g_{k+1}} + \frac{s_k^{\mathrm{T}} g_{k+1}}{y_k^{\mathrm{T}} g_{k+1}} \\
&= \frac{\|g_{k+1}\|^2 \left[s_k^{\mathrm{T}} (g_{k+1} - g_k) \right]^2}{(g_k^{\mathrm{T}} s_k)^2 y_k^{\mathrm{T}} g_{k+1}} + \frac{s_k^{\mathrm{T}} g_{k+1}}{y_k^{\mathrm{T}} g_{k+1}} \\
&\leq \frac{\|g_{k+1}\|^2 (-\sigma g_k^{\mathrm{T}} s_k - g_k^{\mathrm{T}} s_k)^2}{(g_k^{\mathrm{T}} s_k)^2 y_k^{\mathrm{T}} g_{k+1}} + \frac{s_k^{\mathrm{T}} g_{k+1}}{y_k^{\mathrm{T}} g_{k+1}} \\
&= \frac{(\sigma+1)^2 \|g_{k+1}\|^2 (g_k^{\mathrm{T}} s_k)^2}{(g_k^{\mathrm{T}} s_k)^2 y_k^{\mathrm{T}} g_{k+1}} + \frac{s_k^{\mathrm{T}} g_{k+1}}{y_k^{\mathrm{T}} g_{k+1}} \\
&= \frac{(\sigma+1)^2 \|g_{k+1}\|^2 + s_k^{\mathrm{T}} g_{k+1}}{y_k^{\mathrm{T}} g_{k+1}}.
\end{aligned}$$

Because of $\theta_{k+1} \geq 2\sigma^2 + \frac{1}{2}$, so $y_k^{\mathrm{T}} g_{k+1} > 0$. Without losing generality, let $y_k^{\mathrm{T}} g_{k+1} > \kappa > 0$, then we have

$$\theta_{k+1} < \frac{(\sigma+1)^2 \|g_{k+1}\|^2 + 2\gamma \|g_{k+1}\|}{\kappa} < \frac{(\sigma+1)^2 M^2 + 2\gamma M}{\kappa} \doteq \delta. \tag{3.21}$$

Therefore,

$$\|d_{k+1}\| \leq |\theta_{k+1}|\|g_{k+1}\| + |\beta_{k+1}|\|s_k\| \leq \delta M + \frac{|\xi|}{\|s_k\|}\|s_k\| = \delta M + |\xi|.$$

We have

$$\sum_{k\geq 0} \frac{1}{\|d_k\|^2} \geq \frac{1}{(\delta M + |\xi|)^2} \sum_{k\geq 0} 1 = \infty.$$

From Lemma 3.5, it follows that (3.20) is derived. The proof is completed. □

Theorem 5. Let problems (2.2) and (2.6) are solved by ACG algorithm. ACG algorithm is well defined.

Proof. In ACG algorithm, we obtain the Z-eigenvalues of symmetric tensors by solving problem (2.2) or (2.6). When problem (2.6) is solved, it means that solving problem (2.2) results in a zero critical point. Then ACG algorithm turn to solve problem (2.6) which converges to a nonzero critical point. Moreover, since the convergence of our algorithm has been guaranteed, so the termination criteria condition always holds. That is, ACG algorithm is well defined. The proof is completed. □

4. Numerical experiments

In this section, we report some numerical performance of ACG algorithm for solving problems (2.2) and (2.6). For convenience, we provide a table of abbreviations for the methods in Table 1.

Table 1. The abbreviations for methods.

Abbreviation	*Method*
ACG	*Accelerated conjugate gradient method*
HS	*Hestenes − Stiefel method*
PRP	*Polak − Ribiȑe − Polyak method*
SPM	*Shifted power method*
QN	*Quasi − Newton method*
ATTCG	*Accelerated three − term conjugategradient method*
ADL	*Accelerated Dai − Liao projection method*

We compare ACG with HS and PRP, which have been reported to be very efficient for unconstrained optimization. All experiments are done on a PC with CPU 2.40GHz and 2.00GB RAM using MATLAB R2013a. In the implementation of ACG algorithm, we set parameters $\varepsilon = 10^{-5}, \rho = 0.1, \sigma = 0.5, t = 1, \bar{\rho} = 2$. In Table 2, Ex is the number of example, n is the dimension, k is the number of iterations, CPU stands for the time costed by algorithms (in seconds), λ^* stands for Z-eigenvalue outputted by algorithms. All algorithms share the same start points and stopping criteria. In the following examples, the tensors $\mathcal{A}$ are originally from [27].

Table 2. Some numerical results of examples for Z-eigenvalues.

Ex	n	λ^*	$\|\|g_k\|\|$	$k/CPU(ACG)$	$k/CPU(SPM)$
1	$n = 5$	9.9873	$1.6731e-007$	6/0.2188	10/0.3178
1	$n = 10$	17.7657	$2.0318e-006$	4/0.3594	7/0.4815
1	$n = 50$	81.6402	$9.0295e-006$	6/1.7031	11/2.8750
1	$n = 100$	158.1895	$1.0076e-007$	4/12.9219	8/18.0159
1	$n = 200$	311.3130	$5.3215e-006$	7/98.8125	12/115.5250
2	$n = 100$	132.1072	$2.9073e-006$	9/30.0761	14/42.0918
2	$n = 200$	405.2981	$6.9826e-007$	15/123.0050	12/145.0629
3	$n = 5$	13.0791	$3.9828e-006$	4/0.2188	7/0.2983
3	$n = 10$	49.4905	$2.3016e-006$	4/0.5705	8/0.9417
3	$n = 50$	154.9351	$7.6239e-006$	3/31.6406	6/47.2781
4	$n = 3$	0.8893	$1.1075e-005$	7/0.2188	12/0.4106
5	$n = 10$	43.2760	$3.6133e-006$	5/0.2656	8/0.3875
5	$n = 30$	136.2817	$2.8531e-006$	3/3.0469	6/6.2769
6	$n = 5$	34.5317	$1.7063e-006$	6/0.2031	10/0.3385
6	$n = 30$	164.9089	$6.7357e-007$	6/10.7969	12/18.7291

Example 1. *Let* $\mathcal{A} \in S^{[3,n]}$ *defined by*

$$\mathcal{A}_{i_1,i_2,i_3} = \frac{(-1)^{i_1}}{i_1} + \frac{(-1)^{i_2}}{i_2} + \frac{(-1)^{i_3}}{i_3}.$$

Example 2. *Let* $\mathcal{A} \in S^{[3,n]}$ *defined by*

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

Example 3. *Let* $\mathcal{A} \in S^{[4,n]}$ *defined by*

$$\mathcal{A}_{i_1,i_2,i_3,i_4} = arctan\left((-1)^{i_1}\frac{i_1}{n}\right) + \cdots + arctan\left((-1)^{i_4}\frac{i_4}{n}\right).$$

Example 4. *Let* $\mathcal{A} \in S^{[4,3]}$ *defined by*

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

Example 5. *Let* $\mathcal{A} \in S^{[4,n]}$ *defined by*

$$\mathcal{A}_{i_1,i_2,i_3,i_4} = \frac{(-1)^{i_1}}{i_1} + \cdots + \frac{(-1)^{i_4}}{i_4}.$$

Example 6. *Let* $\mathcal{A} \in S^{[4,n]}$ *defined by*

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

In Tables 2 and 3, we compare the numerical results of ACG algorithm and SPM algorithm (in [6]), PRP, HS and CD methods. Although the computed Z-eigenvectors x^* associated to λ^* are not shown, the values of $||g_k||$ are listed which are satisfy the given precision. It implies that (λ^*, x^*) are considered as true solutions of problem (2.1). Two methods reach the same Z-eigenvalues for problems with same dimensions. ACG algorithm requires less iterations and CPU time than that of SPM algorithm. That is, we can see that ACG algorithm is competitive for computing Z-eigenvalues of symmetric tensors.

Table 3. Some numerical results of examples for Z-eigenvalues.

Ex	n	λ^*	$\lVert g_k \rVert$	$k/CPU(PRP)$	$k/CPU(HS)$	$k/CPU(QN)$
1	$n = 5$	9.9872	$2.1905e-006$	10/0.4713	8/0.2964	13/0.5025
1	$n = 10$	17.7657	$1.0948e-005$	6/0.3855	6/0.4311	8/0.4903
1	$n = 50$	81.6401	$6.0182e-007$	7/2.1065	7/2.0021	9/2.3250
1	$n = 100$	158.1896	$2.7328e-005$	11/25.4710	10/21.1207	18/30.6260
1	$n = 200$	311.3130	$1.0823e-007$	10/85.5025	8/70.1025	13/100.4183
2	$n = 100$	132.1072	$8.2012e-005$	11/45.7262	9/39.1840	11/50.2500
2	$n = 200$	405.2981	$3.2909e-006$	18/135.3550	16/123.0931	20/150.2125
3	$n = 5$	13.0792	$3.9828e-005$	6/0.1750	4/0.1558	5/0.2910
3	$n = 10$	49.4905	$1.7293e-006$	7/0.7023	5/0.6500	7/0.6250
3	$n = 50$	154.9351	$4.2950e-006$	5/35.9826	5/30.8674	7/43.0050
4	$n = 3$	0.8893	$9.0482e-007$	12/0.4587	10/0.6039	10/0.8214
5	$n = 10$	43.2760	$3.9281e-006$	8/0.4980	7/0.4058	11/0.5709
5	$n = 30$	136.2817	$2.4615e-006$	5/4.2160	6/4.0156	8/0.5096
6	$n = 5$	34.5315	$2.9053e-006$	7/0.3900	5/0.3215	9/0.5600
6	$n = 30$	164.9086	$7.3155e-007$	9/15.9872	5/12.0060	12/13.4900

To show the numerical performance of a given optimal method, that the number of iterations (k) and CPU time (CPU) are important factors. So, we employ the profiles introduced by Dolan and Moré [28] to analyze the efficiency of ACG, PRP, HS [20], QN [29] ATTCG and ADL [30, 31] methods, with the following conjugate gradient parameters, respectively,

$$\beta_{k+1}^{PRP} = \frac{g_{k+1}^T y_k}{||g_k||^2}, \beta_{k+1}^{HS} = \frac{g_{k+1}^T y_k}{d_k^T y_k}, \beta_{k+1}^{CD} = \frac{||g_{k+1}||^2}{-d_k^T g_k}.$$

Let Y and W be the set of methods and test problems, n_y, n_w be the number of methods and test problems, respectively. The performance profile $\psi : \mathbb{R} \to [0, 1]$ is for each $y \in Y$ and $w \in W$ defined that $a_{w,y} > 0$ is k or CPU required to solve problems w by method y. Furthermore, the performance profile is obtained by

$$\psi_y(\tau) = \frac{1}{n_w} size\{w \in W : r_{w,y} \leq \tau\},$$

where $\tau > 0$, $size\{\cdot\}$ is the number of the elements in a set, and $r_{w,y}$ is the performance ratio defined as

$$r_{w,y} = \frac{a_{w,y}}{min\{a_{w,y} : y \in Y\}}.$$

In a performance profile plot, the top curve is a method that solved most problems in a time that is within a factor of best time. The horizontal axis gives the percentage (τ) of the test problems for which a method is the fastest (efficiency), while the vertical side gives the percentage (ψ) of the test problems that are successfully solved by each of the methods. If program runs failure, or the number of iterations can reach more than 500, it is regarded as failed. And we denote the number of iterations by 500 and CPU time by 200 seconds. In this way, only ACG algorithm can solve all test problems.

As can be seen from Figure 1 shows the CPU time performance of the ACG algorithm and the other algorithms. It can be seen from the figure that when $\tau > 3$, the curves of the ACG algorithm and the ATTCG algorithm are similar, but when $\tau > 3.5$, both the ACG algorithm and ADL algorithm tend to be stable and coincide. Figure 2, the ACG algorithm is better than other algorithms in terms of the number of iterations, especially when $\tau > 2$, the curve of ACG algorithm becomes stable, which indicates that ACG algorithm can solve the problem only with fewer iterations. Therefore, Figures 1 and 2 show that the ACG algorithm proposed in this paper converge to the solution quickly.

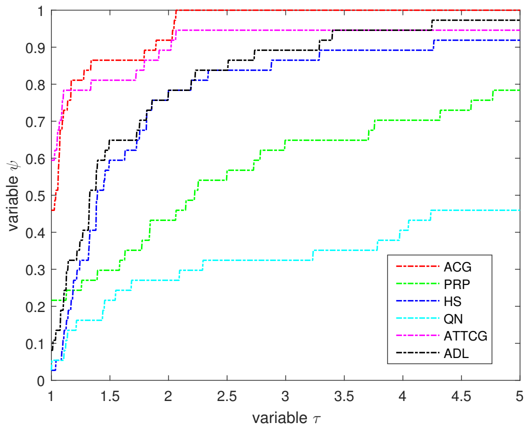

Figure 1. The performance profile for the CPU time.

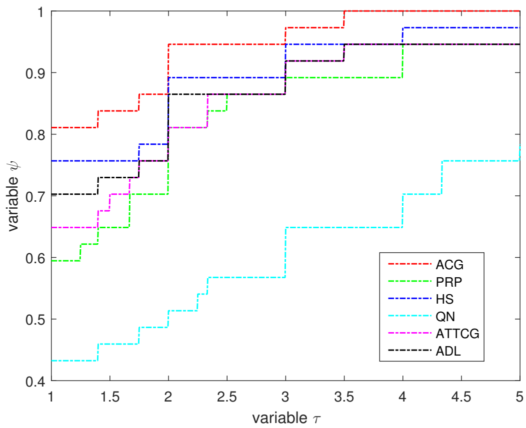

Figure 2. The performance profile for the number of iterations.

5. Conclusions

We constructed the unconstrained optimization problems with a shifted parameter. Based on the shifted unconstrained optimization problems, we presented an accelerated conjugate gradient method by using the quasi-Newton direction for solving them. Furthermore, we showed the global convergence analysis of the proposed algorithm. Numerical experiments demonstrated that our method has good numerical performance. We further highlight that the proposed algorithm can be used in other fields, such as the symmetric system of nonlinear equations. It is vital to note that some new methods with random technology will be taken into account in our future work.

Acknowledgments

National Natural Science Foundation of China (12061087), the Scientific and Technological Developing Scheme of Jilin Province (YDZJ202101ZYTS167, YDZJ202201ZYTS303), the Yunnan Provincial Ten Thousands Plan Young Top Talents, the Project of education department of Jilin Province (JJKH20210030KJ).

Conflict of interest

The authors declare no conflicts of interest.

References

1. L. Lim, Singular values and eigenvalues of tensors: a variational approach, In: *Computational Advances in Multi-Sensor Adaptive Processing*, 2005, 8912515. https://doi.org/10.1109/CAMAP.2005.1574201
2. L. Qi, Eigenvalues of a real supersymmetric tensor, *J. Symb. Comput.*, **40** (2005), 1302–1324. https://doi.org/10.1016/j.jsc.2005.05.007
3. M. Cao, Q. Huang, Y. Yang, A self-adaptive trust region method for extreme B-eigenvalues of symmetric tensors, *Numer. Algor.*, **81** (2019), 407–420. https://doi.org/10.1007/s11075-018-0554-7
4. C. Hao, C. Cui, Y. Dai, A sequential subspace projection method for extreme Z-eigenvalues of supersymmetric tensors, *Numer. Linear Algebra Appl.*, **22** (2015), 283–298. https://doi.org/10.1002/nla.1949
5. S. Aji, P. Kumam, A. Awwal, M. M. Yahaya, K. Sitthithakerngkiet, An efficient DY-type spectral conjugate gradient method for system of nonlinear monotone equations with application in signal recovery, *AIMS Math.*, **6** (2021), 8078–8106. http://dx.doi.org/10.3934/math.2021469
6. T. Kolda, J. Mayo, Shifted power method for computing tensor eigenpairs, *SIAM J. Matrix Anal. Appl.*, **32** (2011), 1095–1124. https://doi.org/10.1137/100801482
7. T. Kolda, J. Mayo, An adaptive shifted power methods for computing generalized tensor eigenpairs, *SIAM J. Matrix Anal. Appl.*, **35** (2014), 1563–1581. https://doi.org/10.1137/140951758

8. Y. Chen, M. Cao, Y. Yang, Q. Huang, An adaptive trust-region method for generalized eigenvalues of symmetric tensors, *J. Comput. Math.*, **39** (2021), 533–549.
9. G. Li, L. Qi, G. Yu, Semismoothness of the maximum eigenvalue function of a symmetric tensor and its application, *Linear Algebra Appl.*, **438** (2013), 813–833. https://doi.org/10.1016/j.laa.2011.10.043
10. M. Cao, Y. Yang, T. Hou, C. Li, A nonmonotone accelerated Levenberg-Marquardt method for the B-eigenvalues of symmetric tensors, *Inter. Trans. Oper. Res.*, **29** (2022), 113–129. https://doi.org/10.1111/itor.12954
11. M. Ng, L. Qi, G. Zhou, Finding the largest eigenvalue of a nonnegative tensor, *SIAM J. Matrix Anal. Appl.*, **31** (2009), 1090–1099. https://doi.org/10.1137/09074838X
12. J. Bai, W. Hager, H. Zhang, An inexact accelerated stochastic ADMM for separable composite convex optimization, *Comput. Optim. Appl.*, **81** (2022), 479–518. https://doi.org/10.1007/s10589-021-00338-8
13. J. Bai, D. Han, H. Sun, H. Zhang, Convergence on a symmetric accelerated stochastic ADMM with larger stepsizes, *CSIAM Trans. Appl. Math.*, **3** (2022), 448–479.
14. L. Qi, W. Sun, Y. Wang, Numerical multilinear algebra and its applications, *Front. Math. China*, **2** (2017), 501–526. https://doi.org/10.1007/s11464-007-0031-4
15. T. Wei, P. Goldbart, Geometric measure of entanglement and applications to bipartite and multipartite quantum states, *Phys. Rev. A*, **68** (2003), 042307. https://doi.org/10.1103/PhysRevA.68.042307
16. L. Qi, G. Yu, E. Wu, Higher order positive semi-definite diffusion tensor imaging, *SIAM J. Imaging Sci.*, **3** (2010), 416–433. https://doi.org/10.1137/090755138
17. L. Qi, F. Wang, Y. Wang, Z-eigenvalue methods for a global polynomial optimization problem, *Math. Program.*, **118** (2009), 301–316. https://doi.org/10.1007/s10107-007-0193-6
18. L. Han, An unconstrained optimization approach for finding real eigenvalues of even order symmetric tensors, *Numer. Algebr. Control Optim.*, **3** (2013), 583–599.
19. C. Hao, C. Cui, Y. Dai, A feasible trust-region method for calculating extreme Z-eigenvalues of symmetric tensors, *Pac. J. Optim.*, **11** (2015), 291–307.
20. J. Nocedal, S. J. Wright, *Numerical optimization*, New York: Springer, 1999.
21. Y. Yuan, W. Sun, *Optimization theory and methods*, Beijing: Science Press, 1997.
22. G. Auchmuty, Globally and rapidly convergent algorithm for symmetric eigenproblems, *SIAM J. Matrix Anal. Appl.*, **12** (1991), 690–706. https://doi.org/10.1137/0612053
23. A. Peressini, F. Sullivan, J. Uhl, *The mathematics of nonlinear programmming*, Berlin: Springer, 1988.
24. A. Neculai, A scaled BFGS preconditioned conjugate gradient algorithm for unconstrained optimization, *Appl. Math. Lett.*, **20** (2007), 645–650. https://doi.org/10.1016/j.aml.2006.06.015
25. A. Neculai, Scaled memoryless BFGS preconditioned conjugate gradient algorithm for unconstrained optimization, *Optim. Meth. Soft.*, **22** (2007), 561–571. https://doi.org/10.1080/10556780600822260

26. G. Zoutendijk, Nonlinear programming, computational method, *Integer Nonlinear Program.*, 1970, 37–86.

27. C. Cui, Y. Dai, J. Nie, All real eigenvalues of symmetric tensors, *SIAM J. Matrix Anal. Appl.*, **4** (2014), 1582–1601. https://doi.org/10.1137/140962292

28. E. Dolan, J. J. Moré, Benchmarking optimization software with performance profiles, *Math. Program.*, **91** (2002), 201–213. https://doi.org/10.1007/s101070100263

29. M. Zeng, Q. Ni, Quasi-Newton method for computing Z-eigenvalues of a symmetric tensor, *Pac. J. Optim.*, **2** (2015), 279–290.

30. X. Dong, D. Han, Z. Dai, L. Li, J. Zhu, An accelerated three-term conjugate gradient method with sufficient descent condition and conjugacy condition, *J. Optim. The. Appl.*, **179** (2018), 944–961. https://doi.org/10.1007/s10957-018-1377-3

31. B. Ivanov, G. Milovanović, P. Stanimirović, Accelerated Dai-Liao projection method for solving systems of monotone nonlinear equations with application to image deblurring, *J. Glob. Optim.*, **85** (2023), 377–420. https://doi.org/10.1007/s10898-022-01213-4

AIMS Press

© 2023 the Author(s), licensee AIMS Press. This is an open access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0)