Computing tensor Z-eigenpairs via an alternating direction method

Genjiao Zhou [1], **Shoushi Wang** [1], **Jinhong Huang** [Corresp. 1, 2]

[1] Gannan Normal University, Ganzhou, China

[2] Gannan Normal University, Key Laboratory of Jiangxi Province for Numerical Simulation and Emulation Techniques, Ganzhou, China

Corresponding Author: Jinhong Huang
Email address: hjhmaths@163.com

Tensor eigenproblems have wide applications in blind source separation, magnetic resonance imaging, and molecular conformation. In this study, we explore an alternating direction method for computing the largest or smallest Z-eigenvalue and corresponding eigenvector of an even-order symmetric tensor. The method decomposes a tensor Z-eigenproblem into a series of matrix eigenproblems that can be readily solved using off-the-shelf matrix eigenvalue algorithms. Our numerical results show that, in most cases, the proposed method converges over 2 times faster and could determine extreme Z-eigenvalues with 20-50% higher probability than a classical power method-based approach.

1 # Computing tensor Z-eigenpairs via an alternating direction
2 # method

4 Genjiao Zhou[1], Shoushi Wang[1], and Jinhong Huang[1, 2]

6 [1] School of Mathematics and Computer Science, Gannan Normal University, Ganzhou, China
7 [2] Key Laboratory of Jiangxi Province for Numerical Simulation and Emulation Techniques,
8 Gannan Normal University, Ganzhou, China.

10 Corresponding Author:
11 Jinhong Huang[1, 2]
12 Email address: hjhmaths@163.com

14 ## Abstract

15 Tensor eigenproblems have wide applications in blind source separation, magnetic resonance
16 imaging, and molecular conformation. In this study, we explore an alternating direction method
17 for computing the largest or smallest Z-eigenvalue and corresponding eigenvector of an even-order
18 symmetric tensor. The method decomposes a tensor Z-eigenproblem into a series of matrix
19 eigenproblems that can be readily solved using off-the-shelf matrix eigenvalue algorithms. Our
20 numerical results show that, in most cases, the proposed method converges over 2 times faster and
21 could determine extreme Z-eigenvalues with 20-50% higher probability than a classical power
22 method-based approach.
23 **Keywords:** higher-order tensor; Z-eigenvalues; power method; alternating direction method

25 ## 1 Introduction

26 The tensor eigenproblem has been of great interest since the seminal works of Qi (2005) and Lim
27 (2005). It has numerous applications in several areas, including automatic control (Ni et al., 2008),
28 magnetic resonance imaging (Qi et al., 2010; 2013; Schultz & Seidel, 2008), statistical data
29 analysis (Zhang & Golub, 2001), image analysis (Zhang et al., 2013), signal processing and
30 navigation (Kofidis & Regalia, 2001; Ashourian & Sharifi-Tehrani 2022; Sharifi-Tehrani &
31 Sabahi 2022; Sharifi-Tehrani et al. 2021), and higher-order Markov chains (Li & Ng, 2014).
32   Unlike for matrices, there are several definitions of tensor eigenvalues and corresponding
33 eigenvectors. For example, Qi (2005) proposed the definition of an H-eigenvalue and Z-eigenvalue
34 as being equivalent to the l^m-eigenvalue and l^2-eigenvalue in Lim (2005), respectively. In Chang
35 et al. (2009), these definitions were unified by employing a positive definite tensor $\mathcal{B}$ while m was
36 even. In this work, we mainly focus on computing Z-eigenvalues of symmetric tensors.
37   In general, the calculation of all eigenvalues of a higher-order tensor is very difficult due to
38 the NP-hardness of deciding tensor eigenvalues over $\mathbb{R}$ (Hillar & Lim, 2013). Fortunately, one

39 only needs to compute the largest or smallest eigenvalue of a tensor in certain scenarios. For
40 instance, to guarantee the positive definiteness of the diffusivity function in higher-order diffusion
41 tensor imaging, we just need to compute the smallest Z-eigenvalue of the tensor and make sure it
42 is nonnegative. In automatic control (Ni et al., 2008), the smallest Z-eigenvalue of a tensor is used
43 to determine whether a nonlinear autonomous system is stable or not. According to the Perron-
44 Frobenius theory, the spectral radius of a nonnegative tensor is the largest Z-eigenvalue of the
45 tensor (Chang et al., 2008).

46   To obtain the extreme eigenvalues of a symmetric tensor, De Lathauwer et al. (2000)
47 introduced a symmetric higher-order power method (S-HOMP). However, it was pointed out by
48 Kofidis & Regalia (2002) that the S-HOMP method is not guaranteed to converge while the
49 objective function is not convex. To address this problem, Kolda and Mayo (2011) presented a
50 shifted S-HOMP (SS-HOMP) for solving the eigenproblem, which is guaranteed to converge to a
51 tensor eigenpair. A major limitation of SS-HOMP is the difficulty in selecting an appropriate shift.
52 Hence, Kolda and Mayo (2014) further extended the SS-HOMP method to an adaptive version for
53 computing extreme eigenvalues, called GEAP, which chooses the shift automatically.

54   Over the past few years, there has been extensive work on handling the extreme eigenvalue
55 problem of symmetric tensors by solving different nonlinearly constrained models. Hu et al. (2013)
56 proposed a sequential semidefinite relaxations approach to compute extreme Z-eigenvalues. Han
57 (2012) employed the BFGS method to solve an unconstrained optimization problem for finding
58 real eigenvalues of even-order symmetric tensors. Cui et al. (2014) computed all of the real Z-
59 eigenvalues of symmetric tensors using a Jacobian semidefinite relaxation technique. Using the
60 method proposed by Qi et al. (2009) in which Z-eigenpairs are computed directly in a lower
61 dimensional case, a sequential subspace projection method (SSPM) (Hao et al., 2015) was
62 proposed to obtain the extreme Z-eigenvalues of symmetric tensors. All the methods mentioned
63 above converge linearly or superlinearly. To speed up convergence, Jaffe et al. (2018) presented a
64 fast iterative Newton-based method that converges at a locally quadratic rate. Based on the idea of
65 the SSPM method (Hao et al., 2015), Yu et al. (2016) proposed an adaptive gradient (AG) method
66 in which an inexact line search, rather than an optimal stepsize, was adopted. The experimental
67 results presented in Yu et al. (2016) showed that the AG method converges much faster and finds
68 the extreme eigenvalues with a higher probability than those methods using power algorithms. For
69 more related work, we refer readers to Benson & Gleich (2019), Chen et al. (2016), Sheng & Ni
70 (2021), Xiong et al. (2022), and references therein.

71   Despite the fact that the extreme eigenvalue problem has drawn a lot of attention in recent
72 years, there are still some issues to address. For example, all algorithms mentioned above are not
73 guaranteed to converge to the largest or smallest eigenvalue, which is exactly what we want to
74 obtain in some applications (Chang et al., 2008; Ni et al., 2008), and instead only converge to an
75 arbitrary eigenvalue of $\mathcal{A}$ depending on the initial conditions. However, in the case of symmetric
76 matrices, those counterpart algorithms can always converge to the largest or smallest eigenvalue.
77 Motivated by this, we propose to determine extreme eigenvalues by combining the method of
78 solving matrix eigenvalue problems and tensor optimization techniques. To this end, we develop

79 a novel method for computing the largest or smallest Z-eigenvalue of symmetric tensors using the
80 variable splitting method.
81   The main contributions of this work are listed as follows:
82   •   We reformulate a typical tensor Z-eigenvalue problem as an equivalent multi-variable
83   linearly constrained problem using the variable splitting method, which has a structure
84   similar to the matrix eigenvalue problem.
85   •   We design an efficient algorithm to solve the reformulated problem, in which the tensor
86   Z-eigenproblem is decomposed into a series of matrix eigenproblems that has been
87   extensively studied and can be readily solved using off-the-shelf matrix eigenvalue
88   algorithms.
89   The remainder of the paper is organized as follows. In the next section, we formulate the tensor
90 eigenproblem and introduce some classical methods for solving it. In Section 3, we propose a
91 simple and efficient algorithm for the problem and analyze its convergence property. Section 4
92 reports some experimental results to show the efficiency of our proposed method. Finally, we
93 conclude this paper in Section 5.
94

2 Problem Formulation and Related Work

2.1 Problem Formulation

97 An mth order n-dimensional real tensor consisting of n^m entries in $\mathbb{R}$:

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

98 $\mathcal{A}$ is called symmetric if the value of $a_{i_1 i_2 \cdots i_m}$ is invariant under any permutation of its indices i_1, i_2

99 $,\cdots, i_m$. For convenience, we use $\mathbb{S}^{[m,n]}$ to denote the set of all mth order n-dimensional real

100 symmetric tensors.

101   Throughout this paper, we use $\mathcal{A}x^{m-k}$ $(0 \le k \le m)$ to simply denote a kth order n-
102 dimensional tensor defined by

$$\left(\mathcal{A}x^{m-k}\right)_{i_1 i_2 \cdots i_k} = \sum_{i_{k+1}, \cdots, i_m = 1}^{n} a_{i_1 i_2 \cdots i_k i_{k+1} \cdots i_m} x_{i_{k+1}} \cdots x_{i_m},$$

(1)

$$for\ all\ 1 \le i_1, i_2, \cdots, i_k \le n\ and\ 1 \le k \le m.$$

103 Obviously, $\mathcal{A}x^m$ is a scalar, $\mathcal{A}x^{m-1}$ is a vector, and $\mathcal{A}x^{m-2}$ is a matrix. For brevity, let $\mathcal{A}x^p y^q$
104 denote the result of $\left(\mathcal{A}x^p\right)y^q = \left(\mathcal{A}x^{m-(m-p)}\right)y^{m-p-(m-p-q)}$, where $0 \le p,\ q,\ p+q \le m$, and

105 $\mathcal{A}x_1^{p_1} x_2^{p_2} \cdots x_2^{p_k}$ can be computed in a similar way, where $0 \le p_1,\ p_2, \cdots, p_k \le m$, and k is an
106 arbitrary integer such that $0 \le p_1 + p_2 + \cdots + p_k \le m$.

107   Using the definition of equation (1), an mth degree homogeneous polynomial function $f(x)$
108   with real coefficients can be represented by a symmetric tensor $\mathcal{A}$, i.e., $f(x) = \mathcal{A}x^m$. We call $\mathcal{A}$
109   positive definite tensor if $\mathcal{A}x^m > 0$ for all $x \in \mathbb{R}^n \backslash \{0\}$. It is easy to understand that m must be even
110   in this case.

111   As mentioned before, there are several definitions of tensor eigenvalues and corresponding
112   eigenvectors. In this work, we mainly focus on computing Z-eigenvalues of symmetric tensors
113   defined as follows.

114   **Definition 1 (Qi, 2005).** Let $\mathcal{A}$ be an mth order n-dimensional symmetric real tensor. If there
115   exists a nonzero vector $x \in \mathbb{R}^n$ and a scalar $\lambda \in \mathbb{R}$ satisfying
116   then
117   we
118   call

$$\begin{cases} \mathcal{A}x^{m-1} = \lambda x, \\ x^T x = 1, \end{cases} \tag{2}$$

119   the scalar λ a Z-eigenvalue of $\mathcal{A}$, and the vector x a Z-eigenvector associated with the Z-eigenvalue
120   λ. We also say the pair (λ, x) is a Z-eigenpair of $\mathcal{A}$.

121   Iterative algorithms to find the largest or smallest eigenvalues and corresponding eigenvectors
122   are usually designed to solve a nonlinearly constrained optimization problem
123   wher
124   e

$$\begin{aligned} \max\; & f(x) = \mathcal{A}x^m \\ s.t.\; & x \in \mathbb{S}^{n-1}, \end{aligned} \tag{3}$$

125      $\mathbb{S}^{n-1}$ denotes the unit sphere in the Euclidean norm, i.e., $\mathbb{S}^{n-1} = \{x \in \mathbb{R}^n \,|\, \|x\|^2 = 1\}$. We can
126   determine the gradient and Hessian of the objective function of (3) through some simple
127   calculations, as follows:

$$g(x) \equiv \nabla f(x) = m\mathcal{A}x^{m-1} \tag{4}$$

128   and

$$H(x) \equiv \nabla^2 f(x) = m(m-1)\,\mathcal{A}x^{m-2} \tag{5}$$

129   **2.2 Some existing methods for Z-eigenproblems**

130   In this subsection, we introduce some typical methods for computing Z-eigenpairs by solving the
131   problem (3) or its variants. From Theorem 3.2 of Kolda & Mayo (2011), we know that (λ, x) is a
132   Z-eigenpair of $\mathcal{A}$ if and only if x is a constrained stationary point of (3) and $\lambda = \mathcal{A}x^m/\|x\|$. Based
133   on the theorem, De Lathauwer et al. (2000) proposed the S-HOPM method for solving the problem
134   (3) to find the best symmetric rank-1 approximation of a symmetric tensor $\mathcal{A} \in \mathbb{S}^{[m,n]}$, which is
135   equivalent to finding the largest Z-eigenvalue of $\mathcal{A}$ (Qi 2005). The main step of the S-HOPM
136   algorithm is

$$x_{k+1} = \frac{\mathcal{A}x_k^{m-1}}{\left\|\mathcal{A}x_k^{m-1}\right\|}, \lambda_{k+1} = \mathcal{A}x_{k+1}^m. \tag{6}$$

137   Under the assumption of convexity on $\mathcal{A}x^m$, S-HOPM could be convergent for even-order
138 tensors. However, it has been pointed out that S-HOPM can not guarantee to converge globally
139 (Kofidis & Regalia, 2002). To address this issue, Kolda & Mayo (2011) modified the objective
140 function to

$$\hat{f}(x) = \mathcal{A}x^m + \alpha\|x\|^m, \tag{7}$$

141 and proposed the SS-HOPM for solving (3) with the objective function (7). SS- HOPM has a
142 similar iterative scheme to S-HOPM, but at the same time has a shortcoming in the choice of the
143 shift α. To overcome the limitation, the same authors proposed an adaptive method, called GEAP,
144 which is monotonically convergent and much faster than the SS-HOPM method due to its adaptive
145 shift choice of the shift. GEAP is originally designed to calculate generalized eigenvalues (Chang
146 et al., 2009) with a positive definite tensor $\mathcal{B}$. The authors also presented a specialization of the
147 method to the Z-eigenvalue problem, which is equivalent to SS-HOPM except for the adaptive
148 shift. The details of the GEAP specialization are briefly summarized in Algorithm 1.

Algorithm 1. GEAP method for the problem (3) with the objective function (7)

Initialization: Given a tensor $\mathcal{A} \in S^{[m,n]}$, an initial vector $x_0 \in \mathbb{R}^n$, and a tolerance $\epsilon > 0$. Let
$\beta = 1$ if we want to compute the largest Z-eigenvalue, and let $\beta = -1$ if we want to compute
the smallest Z-eigenvalue. Let τ be the tolerance on being positive/negative definite.
1: $x_0 \leftarrow x_0/\|x_0\|$, and $\lambda_0 \leftarrow \mathcal{A}x_0^m$
For k=0, 1, $\cdots$ do
    2: $H_k \leftarrow m(m-1)\,\mathcal{A}x_k^{m-2}$
    3: $\alpha_k \leftarrow \beta\max\{0, (\tau - \lambda_{min}(\beta H_k))/m\}$
    4: $x_{k+1} \leftarrow \beta(\mathcal{A}x_k^{m-1} + \alpha x_k)$
    5: $x_{k+1} = x_{k+1}/\|x_{k+1}\|$
    6: $\lambda_{k+1} \leftarrow \mathcal{A}x_{k+1}^m$
    7: Break if $|\lambda_{k+1} - \lambda_k| < \epsilon$
End for
Output: Z-eigenvalue λ and its associated Z-eigenvector x.

149
150   GEAP is a simple and effective approach for computing Z-eigenvalues of a symmetric tensor,
151 but it is not guaranteed to determine the largest eigenvalue or the smallest one, which is exactly
152 the goal in some applications. To obtain these extreme eigenvalues with a higher probability, we
153 propose to reformulate the problem (3) to make its structure similar to a matrix eigenproblem, so
154 that it can be solved by existing methods for matrices that always converge to extreme eigenvalues.

155   3 Proposed Method

156   3.1 An alternating direction method for Z-eigenproblems

157   Motivated by that algorithms for solving matrix eigenproblem can always converge to the
158   largest or smallest eigenvalue, we propose to compute extreme eigenvalues by combining the
159   method of solving the matrix eigenproblem and tensor optimization techniques. To this end, we
160   adopt a variable splitting strategy in which we introduce some superfluous variables and equality
161   constraints over these variables. Specifically, the term $\mathcal{A}x^m$ with even number m is rewritten as $\mathcal{A}$
162    $x_1^2 x_2^2 \cdots x_p^2$, where $p = m/2$, with the equality constraints $x_i = x_j$ $(i, j = 1, 2, \cdots, p)$. Therefore,
163   problem (3) is transformed into the following model:

$$\max \tilde{f}(x) = \mathcal{A}x_1^2 x_2^2 \cdots x_p^2$$
$$\text{s.t.} \quad x_i = x_j, \, i, j = 1, 2, \cdots, p \tag{8}$$
$$x_i \in \mathbb{S}^{n-1}, \, i = 1, 2, \cdots, p.$$

164   When $\mathcal{A}$ is symmetric and conditions $x_i = x_j = x(i, j = 1, 2, \cdots, p)$ hold, we can obtain $\mathcal{A}x_1^2 x_2^2 \cdots x_p^2$
165    $= \mathcal{A}x^m$. Using this fact, the equivalence between problems (3) and (8) can be easily checked. It
166   is also worthwhile to note that if all variables except x_i are available and those equality constraints
167   are not considered, the problem (8) reduces to the standard matrix eigenproblem for the matrix $\mathcal{A}x_1^2$
168    $\cdots x_{i-1}^2 x_{i+1}^2 \cdots x_p^2$.
169   Directly solving the problem (8) may be inefficient because its special structure is not
170   considered, and in doing so, it is easy to converge to a locally optimal point, thus the largest or
171   smallest eigenvalue could not be determined. On the other hand, it is comparatively easy to
172   compute extreme eigenvalues for the matrix cases. In (8), if all variables except x_i are known and
173   those equality constraints are not considered, solving (8) can exactly get the largest eigenvalue and
174   the corresponding eigenvector of the matrix $\mathcal{A}x_1^2 \cdots x_{i-1}^2 x_{i+1}^2 \cdots x_p^2$. Following this observation and
175   the fact that there are many efficient algorithms available for tackling matrix eigenproblems, we
176   propose a simple alternating direction scheme between solving different matrix eigenproblems for
177   the problem (8). The details of this method are given in Algorithm 2.
178

Algorithm 2 Alternating direction method (ADM) for (8)

Initialization: Given an even-order tensor $\mathcal{A} \in S^{[m,\,n]}$, initial unit vectors $x_i \in \mathbb{R}^n$, $i = 1, 2, \cdots, p$, where $p = m/2$, and $\epsilon > 0$ is the tolerance. Set $x = x_p$, $\lambda = \mathcal{A}x^m$, and δ as the absolute difference between successive values of λ.

While $\delta > \epsilon$

    For $i = 1, 2, \cdots, p$ do

        1: Compute the matrix $A = \mathcal{A}x_1^2 \cdots x_{i-1}^2 x_{i+1}^2 \cdots x_p^2$.

        2: Find the largest or smallest eigenvalue $\tilde{\lambda}$ and the corresponding unit eigenvector v
            of A using any eigenvalue algorithm for matrices.

        3: Update the variable $x_i = v$.

    End for

    4: Set $x = x_p$ and $\lambda = \mathcal{A}x^m$.

End while

Output: Z-eigenvalue λ and its associated Z-eigenvector x.

179

180   The main computational cost lies in tensor-vector multiplications $\mathcal{A}x_1^2 \cdots x_{i-1}^2 x_{i+1}^2 \cdots x_p^2$ and
181 matrix eigenvalue computations. For an mth order n-dimensional symmetric tensor $\mathcal{A}$, it costs $O(m$
182 $n^m)$ operations to compute the matrix $A = \mathcal{A}x_1^2 \cdots x_{i-1}^2 x_{i+1}^2 \cdots x_p^2$. The cost of computing the largest
183 or smallest eigenvalue of the matrix A is $(4/3)n^3$, which is much less than the products for large
184 n. Compared with GEAP, which has similar computations to our method but needs to compute
185 tensor multiplication at least twice in each iteration, our method can save about half of the time
186 because it only needs to calculate tensor multiplication once in each iteration. This can be verified
187 in the numerical experiments in Section 4.

188 ## 3.2 Specialization of ADM to fourth-order tensors

189   The proposed ADM transforms the tensor eigenvalue problem (3) into a series of matrix
190 eigenvalue problems that are easy to solve. For fourth-order tensors, there are two related variables
191 of x_1 and x_2, and the inner iteration can be omitted because $p = 1$. According to the symmetry
192 property of $\mathcal{A}$, we also have $\mathcal{A}x_1^2 x_2^2 = \mathcal{A}x_2^2 x_1^2$. Therefore, it is not necessary to explicitly write out
193 the variable x_2, and the procedure of Algorithm 2 can be simply described, as shown in Algorithm
194 3, for fourth-order tensors. To better describe the iterative steps, we use x_k to denote a kth iterate
195 in Algorithm 3, rather than the splitting variable as in Algorithm 2.
196

Algorithm 3 Specialization of the ADM to fourth-order tensors

---

Initialization: Given a tensor $\mathcal{A} \in S^{[4,\,n]}$, initial unit vectors $x_0 \in \mathbb{R}^n$, and $\epsilon > 0$ is the tolerance. Set $\lambda_0 = \mathcal{A}x_0^m$, $k := 0$, and δ as the absolute difference between successive values of λ.

For $k = 0, 1, 2 \cdots$ **do**
    1: Compute the matrix $A_k = \mathcal{A}x_k^2$.
    2: Find the largest or smallest eigenvalue $\tilde{\lambda}$ and the corresponding unit eigenvector v of A_k using any eigenvalue algorithm for matrices.
    3: Update the variable $x_{k+1} = v$ and the eigenvalue $\lambda_{k+1} = \tilde{\lambda}$.
    4: **Break if** $|\lambda_{k+1} - \lambda_k| < \epsilon$, set $k = k + 1$.
End for
Output: Z-eigenvalue λ_{k+1} and its associated Z-eigenvector x_{k+1}.

197

3.3 Convergence analysis

199   As shown in the main steps of Algorithm 2 and Algorithm 3, the equality constraints in (8) are
200 not considered in the process of calculation. A natural question arises about whether the algorithms
201 can converge, and furthermore, whether the algorithms can converge to a Z-eigenvalue of $\mathcal{A}$. In
202 this subsection, we handle these issues using properties of extreme eigenvalues and corresponding
203 eigenvectors of matrices. For simplicity, only the convergence property of Algorithm 3 is
204 analyzed. The convergence property of Algorithm 2 can be analyzed in a similar way.

205   Let x_k denote the kth iterate generated by Algorithm 3. According to Steps 2 and 3, in the case
206 of computing the largest Z-eigenvalue of $\mathcal{A}$, x_{k+1} is the largest eigenvalue of the matrix $\mathcal{A}x_k^2$.
207 Therefore, the quadratic function $q(y) = y^T(\mathcal{A}x_k^2)y = \mathcal{A}x_k^2 y^2$ reaches a maximum value λ_{k+1} at
208 the point $y = x_{k+1}$ over the unit sphere $\mathbb{S}^n$, i.e., $\lambda_{k+1} = \mathcal{A}x_k^2 x_{k+1}^2 \geq \mathcal{A}x_k^2 y^2$ for all $y \in \mathbb{S}^n$. At the
209 same time, x_k is the largest eigenvalue of the matrix $\mathcal{A}x_{k-1}^2$. These results give

210 Her
$$\lambda_{k+1} = \mathcal{A}x_k^2 x_{k+1}^2 \geq \mathcal{A}x_k^2 x_{k-1}^2 = \mathcal{A}x_{k-1}^2 x_k^2 = \lambda_k. \tag{9}$$
211 e,
212 the second equality holds because of the symmetric property of $\mathcal{A}$. From (9), we know that the
213 sequence $\{\lambda_k\}$ generated by Algorithm 3 is nondecreasing. On the other side, λ_k is computed by λ_k
214 $= \mathcal{A}x_{k-1}^2 x_k^2$, where $x_k \in \mathbb{S}^n$. Due to the compactness of the unit sphere $\mathbb{S}^n$, we also know that the
215 sequence $\{\lambda_k\}$ is bounded above. Consequently, $\{\lambda_k\}$ has a unique limit, and we can readily
216 conclude by posing this as a theorem.

217   **Theorem 1.** Let $\{\lambda_k\}$ be a sequence generated by Algorithm 3. Then the sequence $\{\lambda_k\}$ is
218 nonincreasing and there exists λ^* such that $\lambda_k \rightarrow \lambda^*$.

219   While Theorem 1 ensures that Algorithm 3 always terminates in finitely many iterations,
220 theoretically, it cannot ensure that the sequence $\{\lambda_k\}$ converges to a Z-eigenvalue of $\mathcal{A}$ because

221   the equality constraints in (8) are omitted in the implementation of Algorithm 3. One possible
222   result is the occurrence of cyclic solutions, that is, two consecutive iterates x_k and x_{k+1} that satisfy
223    $\mathcal{A}x_k^2 x_{k+1} = \lambda_k x_{k+1}$, $\mathcal{A}x_{k+1}^2 x_k = \lambda_{k+1} x_k$, and $\lambda_k = \lambda_{k+1}$. However, this situation is rarely
224   encountered in the numerical experiments presented in the next section. Additionally, how to
225   theoretically avoid this situation is the subject for future research.

226   4 Numerical experiments

227   In this section, we present some numerical results of the ADM for computing the largest or
228   smallest Z-eigenvalues of tensors. The proposed ADM is compared with the GEAP method, which
229   is an adaptive shifted power method first proposed by Kolda and Mayo (2014), and the AG method
230   (Yu et al., 2016), which is an adaptive gradient method with inexact stepsize. All experiments are
231   performed in MATLAB R2017a and the Tensor Toolbox (Bader et al., 2012) under a Windows 10
232   operating system on a laptop with an Intel(R) Core (TM) i7-10510U CPU and 12 GB RAM. In all
233   numerical experiments, we terminate the computation when the absolute difference between
234   successive eigenvalues is less than 10^{-10}, i.e., $|\lambda_{k+1} - \lambda_k| \leq 10^{-10}$, or the number of iterations
235   exceeded the maximum number 500.
236   In our experiments, we use some typical examples from references (Cui et al., 2014; Kofidis
237   & Regalia, 2002; Nie & Wang, 2014) to assess the performance of the proposed method in finding
238   the largest or smallest Z-eigenvalue of a symmetric tensor. All of the largest or smallest Z-
239   eigenvalues in these examples are given in the original literature. Therefore, those desired values
240   are known in advance.
241   ***Example 1*** (Kofidis & Regalia, 2002) Let $\mathcal{A} \in \mathbb{S}^{[4,3]}$ be the symmetric tensor with entries

$a_{1111} = 0.2883,$ $a_{1112} = -0.0031,$ $a_{1113} = 0.1973,$ $a_{1122} = -0.2485,$

$a_{1223} = 0.1862,$ $a_{1133} = 0.3847,$ $a_{1222} = 0.2972,$ $a_{1123} = -0.2939,$

$a_{1233} = 0.0919,$ $a_{1333} = -0.3619,$ $a_{2222} = 0.1241,$ $a_{2223} = -0.3420,$

$a_{2233} = 0.2127,$ $a_{2333} = 0.2727,$ $a_{3333} = -0.3054.$

242   The largest and smallest Z-eigenvalue of the tensor $\mathcal{A}$ are respectively

$$\lambda_{max} = 0.8893, \, v_{max} = (0.6672, 0.7160, 0.9073)^T;$$

$$\lambda_{min} = -1.0954, \, v_{min} = (-0.6447, -0.3357, 0.3043)^T.$$

243   We first test the convergence performance of the proposed ADM in comparison to GEAP.
244   Figure 1 shows the convergence trajectories of the two methods for computing extreme Z-
245   eigenvalues of $\mathcal{A}$ from *Example 1*, with the starting point $x_0 = (0.0417, -0.5618, 0.6848)^T$. As
246   shown on the left in Figure 1, both GEAP and ADM can find the largest Z-eigenvalue 0.8893.
247   Until the stopping criterion is met, GEAP runs 63 iterations in 0.2188 seconds, while the proposed
248   ADM runs only 26 iterations in 0.0313 seconds. When computing the smallest Z-eigenvalue with

249   the same starting point (right in Figure 1), although ADM runs longer than GEAP, the ADM find
250   the desired value of -1.0954, while GEAP fail.
251   To test the performance of the proposed algorithm in finding extreme eigenvalues, 1,000
252   random starting guesses drawn uniformly from [-1, 1] are employed in the experiments. All three
253   methods are implemented 1,000 times, each with the same initial point. We list the number of
254   occurrences of extreme eigenvalues (#Occu.), the average number of iterations (Iter$_{ave}$), and the
255   average running time in seconds (CPU$_{ave}$) for the two types of Z-eigenvalues in Tables 1-6. As
256   shown in Tables 1-6, for the cases of computing the largest eigenvalues, the proposed ADM runs
257   a similar or slightly larger number of iterations but in a similar or shorter time compared to both
258   GEAP and AG. For the case of computing the smallest Z-eigenvalue, ADM runs slower for
259   *Example 1* but runs much faster for the other examples. It can also be seen from the fourth columns
260   of Tables 1-6 that GEAP can only obtain the largest Z-eigenvalues with a probability of about
261   0.55, and the smallest Z-eigenvalue with a probability of about 0.65. AG performs clearly better
262   than GEAP. The proposed ADM performs best in almost all cases, which can reach the largest Z-
263   eigenvalues for Examples 2-5 and the smallest Z-eigenvalues for all examples with a probability
264   of 1.
265   ***Example 2*** (Nie & Wang, 2014). Consider the symmetric tensor $\mathcal{A} \in \mathbb{S}^{[4,n]}$ such that

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

266   In the case of $n = 5$, the largest and smallest Z-eigenvalues of the tensor $\mathcal{A}$ are respectively

$$\lambda_{max} = 7.2595, v_{max} = (0.2686, 0.6150, 0.3959, -0.1872, -0.5982)^T;$$
$$\lambda_{min} = -8.8463, v_{min} = (-0.5809, -0.3563, 0.1959, 0.5680, 0.4179)^T.$$

267

268   ***Example 3*** (Cui et al., 2014). Consider the symmetric tensor $\mathcal{A} \in \mathbb{S}^{[4,n]}$ such that

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

269   In the case of $n = 5$, we can obtain the largest and smallest Z-eigenvalues of the tensor $\mathcal{A}$ from
270   the reference as follows.

$$\lambda_{max} = 34.5304, v_{max} = (0.6665, 0.1089, 0.4132, 0.6070, -0.0692)^T;$$
$$\lambda_{min} = -101.1994, v_{min} = (0.2248, 0.5541, 0.3744, 0.2600, 0.6953)^T.$$

271

272   ***Example 4*** (Nie & Wang, 2014). Let $\mathcal{A} \in \mathbb{S}^{[4,n]}$ be a symmetric tensor with

$$a_{ijkl} = \arctan\left((-1)^i \frac{i}{n}\right) + \arctan\left((-1)^j \frac{j}{n}\right) + \arctan\left((-1)^k \frac{k}{n}\right) + \arctan\left((-1)^l \frac{l}{n}\right).$$

273   In the case of $n = 5$, the largest and smallest Z-eigenvalues of the tensor $\mathcal{A}$ are respectively

$$\lambda_{max} = 13.0779, v_{max} = (0.3174, 0.5881, 0.1566, 0.7260, 0.0418)^T;$$

$$\lambda_{min} = -23.5740, \, v_{min} = (0.4403, 0.2382, 0.5602, 0.1354, 0.6459)^T.$$

274   ***Example 5*** (Nie & Wang, 2014). Let $\mathcal{A} \in \mathbb{S}^{[4,n]}$ be a symmetric tensor with

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

275   For $n = 5$, we can get the largest and smallest Z-eigenvalue of the tensor $\mathcal{A}$ with

$$\lambda_{max} = 9.5821, \, v_{max} = (-0.1125, 0.7048, 0.2507, 0.5685, 0.3233)^T;$$

$$\lambda_{min} = -27.0429, \, v_{min} = (-0.6900, -0.1987, -0.4717, -0.2806, -0.4280)^T.$$

276   Besides the point $\lambda_1 = 9.5821$ of the largest Z-eigenvalue, the tensor $\mathcal{A}$ has another stable
277   eigenvalue $\lambda_2 = 0$. As shown in Figure 2 and Table 5, GEAP falls into the latter point with a
278   probability of around 0.4, while ADM can always converge to the previous point.
279   ***Example 6*** (Sheng & Ni, 2021). Let $\mathcal{A} \in \mathbb{S}^{[6,3]}$ be a tensor with

280                $$a_{111122} = \frac{1}{15}, \, a_{112222} = \frac{1}{15}, \, a_{112233} = -\frac{1}{30}, \, a_{333333} = 1,$$

281   and $a_{i_1 \cdots i_6} = 0$ if $(i_1 \cdots i_6)$ is not a permutation of an index in the above. We can get the largest Z-
282   eigenvalue $\lambda_{max} = 1$ and smallest Z-eigenvalue $\lambda_{min} = 0$, and these corresponding eigenvectors
283   are not unique. The comparison results are shown in Table 6, from which we find that GEAP
284   converges very slowly when computing the smallest eigenvalue of $\mathcal{A}$. In contrast, ADM reaches
285   the smallest eigenvalue with only about 12 iterations for each execution.

286   **5 Discussion**

287   In general, algorithms for computing the largest or smallest eigenvalue of a higher-order tensor are
288   prone to getting stuck in local extrema and then converging to an arbitrary eigenvalue of the tensor
289   depending on the initial conditions. However, the counterpart for symmetric matrices can always
290   converge to the largest or smallest one. Motivated by this, we propose combining algorithms for
291   matrix eigenproblem and tensor optimization techniques in order to obtain extreme eigenvalues.
292   Specifically, the tensor eigenproblem is split into a series of matrix eigenvalue problems using a
293   variable splitting method, and then an alternating scheme is proposed to solve the problem.
294   To solve the tensor eigenproblems, many algorithms for matrix eigenproblems are extended to
295   the tensor case. However, these generalizations cannot guarantee the low complexity of these
296   algorithms, and the global convergence to the extreme eigenvalues is also not ensured. In this
297   paper, the tensor eigenvalue problem is directly transformed into a series of matrix eigenvalue
298   problems so that its algorithms can be directly used to solve the original tensor eigenvalue problem.

299   This method not only overcomes local minima problems existing in direct generalizations, but also
300   has great potential to speed up the convergence. The experimental results verify the effectiveness
301   and advancement of the proposed algorithm, which converges rapidly in most cases and reaches
302   extreme Z-eigenvalues with a significantly higher probability. In many cases, we determine the
303   extreme eigenvalue with a probability of 1, indicating that we can obtain the extreme eigenvalue
304   under any given initial value in these cases. This demonstrates the significant robustness of the
305   proposed method.
306   However, the proposed algorithm cannot guarantee global convergence for each type of tensor.
307   For Examples 1 and 6, we could only obtain the largest eigenvalue with a probability of about 0.6.
308   In future research, the question of why this kind of tensor cannot obtain the extreme eigenvalue
309   under any initial point will be discussed in more detail.

310   6 Conclusion

311   In this paper, we transform a tensor Z-eigenvalue problem into a series of matrix eigenvalue
312   problems using a variable splitting method and propose an alternating scheme for computing the
313   largest or smallest Z-eigenvalue of symmetric tensors. Just like the classical power method, which
314   constantly uses the intermediate iterates to construct a vector, the proposed algorithm uses them to
315   construct a matrix and computes the eigenvalues and corresponding eigenvectors of the matrix.
316   We analyze the convergence properties of the proposed method which is verified in the numerical
317   experiments. The limitations of this work are twofold. First, as the authors note themselves, it can
318   only ensure the convergence of eigenvalues, but not the convergence of eigenvectors due to the
319   possible existence of cyclic solutions. Second, as can be seen from the numerical examples, it
320   cannot obtain the largest eigenvalues with a 100% probability. The numerical results are reported
321   for some testing examples, which showed that the proposed method converged much faster than
322   both GEAP and AG in most cases and could reach the extreme Z-eigenvalues with a significantly
323   higher probability than GEAP.
324
325

326   **Acknowledgements**
327   We thank the anonymous reviewers for their insightful comments. This work was supported in
328   part by the National Natural Science Foundation of China (Grant No. 82060328 and 61863001).

329
330
331

332   References

333   Ashourian M, and Sharifi-Tehrani O. 2022. Application of semi-circle law and Wigner spiked-
334   model in GPS jamming confronting. Signal, Image and Video Processing:1-8.
335   Bader BW, Kolda TG, and others. 2012. MATLAB Tensor Toolbox Version 2.5.
336   http://wwwsandiagov/~tgkolda/TensorToolbox/index-25html.

337   Benson AR, and Gleich DF. 2019. Computing tensor Z-eigenvectors with dynamical systems.
338   SIAM Journal on Matrix Analysis and Applications 40:1311-1324.
339   Chang K-C, Pearson K, and Zhang T. 2008. Perron-Frobenius theorem for nonnegative tensors.
340   Communications in Mathematical Sciences 6:507-520.
341   Chang K, Pearson K, and Zhang T. 2009. On eigenvalue problems of real symmetric tensors.
342   Journal of Mathematical Analysis and Applications 350:416-422.
343   Chen L, Han L, and Zhou L. 2016. Computing tensor eigenvalues via homotopy methods. SIAM
344   Journal on Matrix Analysis and Applications 37:290-319.
345   Cui C-F, Dai Y-H, and Nie J. 2014. All real eigenvalues of symmetric tensors. SIAM Journal on
346   Matrix Analysis and Applications 35:1582-1601.
347   De Lathauwer L, De Moor B, and Vandewalle J. 2000. On the best rank-1 and rank-(R 1, R 2,...,
348   Rn) approximation of higher-order tensors. SIAM Journal on Matrix Analysis and Applications
349   21:1324-1342.
350   Han L. 2012. An unconstrained optimization approach for finding real eigenvalues of even order
351   symmetric tensors. arXiv preprint arXiv:12035150.
352   Hillar CJ, and Lim L-H. 2013. Most tensor problems are NP-hard. Journal of the ACM (JACM)
353   60:45.
354   Hu S, Huang ZH, and Qi L. 2013. Finding the extreme Z-eigenvalues of tensors via a sequential
355   semidefinite programming method. Numerical Linear Algebra with Applications 20:972-984.
356   Jaffe A, Weiss R, and Nadler B. 2018. Newton correction methods for computing real eigenpairs
357   of symmetric tensors. SIAM Journal on Matrix Analysis and Applications 39:1071-1094.
358   Kofidis E, and Regalia PA. 2001. Tensor approximation and signal processing applications.
359   Contemporary Mathematics 280:103-134.
360   Kofidis E, and Regalia PA. 2002. On the best rank-1 approximation of higher-order
361   supersymmetric tensors. SIAM Journal on Matrix Analysis and Applications 23:863-884.
362   Kolda TG, and Mayo JR. 2011. Shifted power method for computing tensor eigenpairs. SIAM
363   Journal on Matrix Analysis and Applications 32:1095-1124.
364   Kolda TG, and Mayo JR. 2014. An adaptive shifted power method for computing generalized
365   tensor eigenpairs. SIAM Journal on Matrix Analysis and Applications 35:1563-1581.
366   L. Hao C, F. Cui C, and H. Dai Y. 2015. A sequential subspace projection method for extreme
367   Z‐eigenvalues of supersymmetric tensors. Numerical Linear Algebra with Applications
368   22:283-298.
369   Li W, and Ng MK. 2014. On the limiting probability distribution of a transition probability tensor.
370   Linear and Multilinear Algebra 62:362-385.
371   Lim L-H. 2005. Singular values and eigenvalues of tensors: a variational approach. IEEE
372   International Workshop on Computational Advances in Multi-Sensor Adaptive Processing
373   IEEE:129-132.

374 Ni Q, Qi L, and Wang F. 2008. An eigenvalue method for testing positive definiteness of a
375 multivariate form. IEEE Transactions on Automatic Control 53:1096-1107.

376 Nie J, and Wang L. 2014. Semidefinite relaxations for best rank-1 tensor approximations. SIAM
377 Journal on Matrix Analysis and Applications 35:1155-1179.

378 Qi L. 2005. Eigenvalues of a real supersymmetric tensor. Journal of Symbolic Computation
379 40:1302-1324.

380 Qi L, Wang F, and Wang Y. 2009. Z-eigenvalue methods for a global polynomial optimization
381 problem. Mathematical Programming 118:301-316.

382 Qi L, Yu G, and Wu EX. 2010. Higher order positive semidefinite diffusion tensor imaging. SIAM
383 Journal on Imaging Sciences 3:416-433.

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

386 Schultz T, and Seidel H-P. 2008. Estimating crossing fibers: A tensor decomposition approach.
387 IEEE Transactions on Visualization and Computer Graphics 14:1635-1642.

388 Sharifi-Tehrani O, and Sabahi MF. 2022. Eigen analysis of flipped Toeplitz covariance matrix for
389 very low SNR sinusoidal signals detection and estimation. Digital Signal Processing
390 129:103677.

391 Sharifi-Tehrani O, Sabahi MF, and Danaee MR. 2021. Efficient GNSS Jamming Mitigation Using
392 the MarcenkoPastur Law and Karhunen-Loeve Decomposition. IEEE Transactions on
393 Aerospace and Electronic Systems.

394 Sheng Z, and Ni Q. 2021. Computing tensor Z-eigenvalues via shifted inverse power method.
395 Journal of Computational and Applied Mathematics 398:113717.

396 Xiong L, Jiang Z, Liu J, and Qin Q. 2022. New Z-Eigenvalue Localization Set for T ensor and Its
397 Application in Entanglement of Multipartite Quantum States. Mathematics 10:2624.

398 Yu G, Yu Z, Xu Y, Song Y, and Zhou Y. 2016. An adaptive gradient method for computing
399 generalized tensor eigenpairs. Computational Optimization and Applications 65:781-797.

400 Zhang F, Zhou B, and Peng L. 2013. Gradient skewness tensors and local illumination detection
401 for images. Journal of Computational and Applied Mathematics 237:663-671.

402 Zhang T, and Golub GH. 2001. Rank-one approximation to high order tensors. SIAM Journal on
403 Matrix Analysis and Applications 23:534-550.

404

Figure 1

Convergence comparison of the GEAP method and the proposed method for computing the largest and smallest Z-eigenvalue of A from Example *1.*

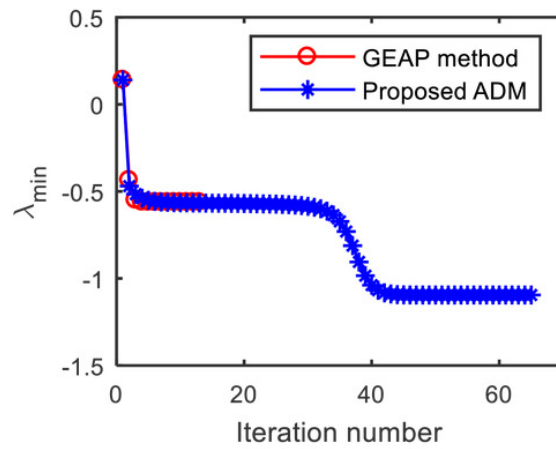
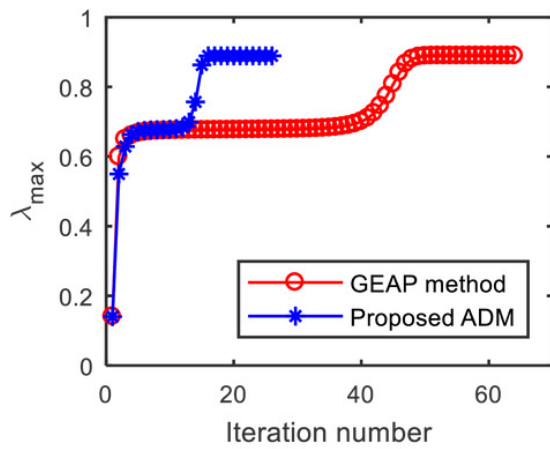



Figure 2

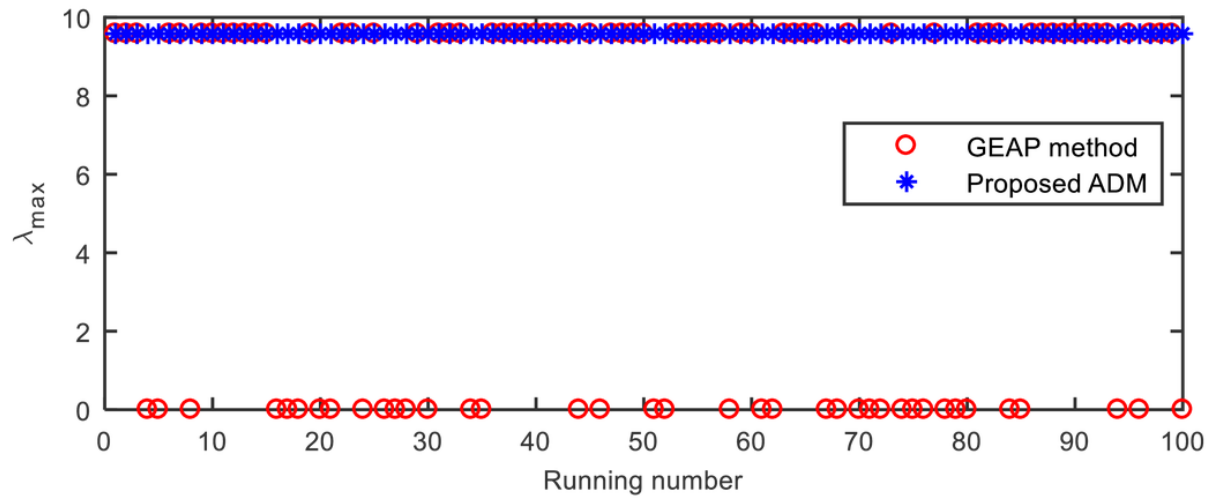
The largest Z-eigenvalues computed by GEAP and ADM in the 100 runs on the tensor A from Example 5.

Comparison results for computing extreme Z-eigenvalues of A from Example 1.

Table 1. Comparison results for computing extreme Z-eigenvalues of $\mathcal{A}$ from Example 1.

Type of Z-eigenvalue	Method	λ	#Occu.	Iter$_{ave}$	CPU$_{ave}$
λ_{max}	GEAP	0.8893	51.00%	27.59	0.0356
	AG	0.8893	**57.60%**	12.97	0.0202
	ADM	0.8893	57.10%	28.96	**0.0165**
λ_{min}	GEAP	-1.0953	41.10%	12.18	**0.0121**
	AG	-1.0953	52.50%	8.24	0.0185
	ADM	-1.0953	**100.00%**	43.52	0.0258

Comparison results for computing the extreme Z-eigenvalues of A from Example 2 (n=5).

2   **Table 2**. Comparison results for computing the extreme Z-eigenvalues of $\mathcal{A}$ from Example 2 ($n = 5$).

Type of Z-eigenvalue	Method	λ	#Occu.	Iter$_{ave}$	CPU$_{ave}$
λ_{max}	GEAP	7.2595	49.80%	49.30	0.0701
	AG	7.2595	60.90%	19.91	**0.0334**
	ADM	7.2595	**100.00%**	117.20	0.0806
λ_{min}	GEAP	-8.8463	51.60%	49.85	0.0692
	AG	-8.8463	77.80%	23.29	**0.0385**
	ADM	-8.8463	**100.00%**	57.41	0.0389

Comparison results for computing the extreme Z-eigenvalues of A from Example 3 (n=5).

2   **Table 3**. Comparison results for computing the extreme Z-eigenvalues of $\mathcal{A}$ from Example 3 ($n = 5$).

Type of Z-eigenvalue	Method	λ	#Occu.	Iter$_{\text{ave}}$	CPU$_{\text{ave}}$
	GEAP	34.5304	62.30%	27.52	**0.0341**
λ_{max}	AG	34.5304	92.10%	14.54	0.0350
	ADM	34.5304	**100.00%**	56.26	0.0362
	GEAP	-101.1994	72.00%	14.00	0.0184
λ_{min}	AG	-101.1994	98.90%	9.37	0.0149
	ADM	-101.1994	**100.00%**	13.16	**0.0057**

Comparison results for computing the extreme Z-eigenvalues of A from Example 4 (n=5).

1   Table 4. Comparison results for computing the extreme Z-eigenvalues of $\mathcal{A}$ from Example 4 ($n = 5$).

Type of Z-eigenvalue	Method	λ	#Occu.	Iter$_{ave}$	CPU$_{ave}$
λ_{max}	GEAP	13.0779	63.20%	22.01	0.0263
	AG	13.0779	93.00%	12.80	0.0209
	ADM	13.0779	**100.00%**	30.72	**0.0166**
λ_{min}	GEAP	-23.5741	69.10%	15.21	0.0161
	AG	-23.5741	97.20%	10.09	0.0193
	ADM	-23.5741	**100.00%**	15.32	**0.0030**

Comparison results for computing the extreme Z-eigenvalues of A from Example 5 (n=5).

2   **Table 5.** Comparison results for computing the extreme Z-eigenvalues of $\mathcal{A}$ from *Example 5* ($n = 5$).

Type of Z-eigenvalue	Method	λ	#Occu.	Iter$_{ave}$	CPU$_{ave}$
	GEAP	9.5821	61.00%	25.59	0.0309
λ_{max}	AG	9.5821	91.70%	14.03	**0.0285**
	ADM	9.5821	**100.00%**	50.69	0.0295
	GEAP	-27.0429	71.60%	13.47	0.0153
λ_{min}	AG	-27.0429	98.40%	9.13	0.0159
	ADM	-27.0429	**100.00%**	12.86	**0.0050**

Comparison results for computing the extreme Z-eigenvalues of A from Example 6.

1   **Table 6.** Comparison results for computing the extreme Z-eigenvalues of $\mathcal{A}$ from *Example 6*.

Type of Z-eigenvalue	Method	λ	#Occu.	Iter$_{ave}$	CPU$_{ave}$
λ_{max}	GEAP	1	51.50%	7.56	0.0077
	AG	1	**82.80%**	6.28	0.0054
	ADM	1	64.70%	12.40	**0.0066**
λ_{min}	GEAP	0	99.90%	231.32	0.3365
	AG	0	**100.00%**	22.73	0.0595
	ADM	0	**100.00%**	12.15	**0.0027**