

EfficientMaxEigenpairTridiagonal Vignette

Yue-Shuang Li and Xiao-Jun Mao

(Beijing Normal University)

October 29, 2018

EfficientMaxEigenpairTridiagonal is a package for computing the maximal eigenpair of Hermitizable tridiagonal matrices. This vignette is a simple guide to using the package. All the algorithms and examples provided in this vignette are available in the paper [4] (2018) by Mu-Fa Chen and Yue-Shuang Li. The algorithm is reformulated based on the papers [1] (2016)~[3] (2018) by Mu-Fa Chen.

Let us install and require the package EfficientMaxEigenpairTridiagonal first. We now introduce the package in details. It offers three main algorithms:

- `eff_ini_maxeig_tri` ($a, b, c, \text{digit_thresh}$): calculate the maximal eigenpair for Hermitizable tridiagonal matrix based on [1] (2016)~[3] (2018).

- The precise level used for output results is set to be `digit_thresh=6` which implies $e-6$ without any special requirement. The parameter a, b, c are just the same as those in the paper [3] (2018), we now state it explicitly. Given a tridiagonal matrix $T \sim (a_k, -c_k, b_k)$:

$$T = \begin{pmatrix} -c_0 & b_0 & & & \\ a_1 & -c_1 & b_1 & & \\ & a_2 & -c_2 & b_2 & \\ & & \ddots & \ddots & \ddots \\ & & & \ddots & \ddots & b_{N-1} \\ & & & & a_N & -c_N \end{pmatrix},$$

where a and b should be complex satisfying that $a_{k+1}b_k > 0$, let $a = (a_1, a_2, \dots, a_N)$, $b = (b_0, b_1, \dots, b_{N-1})$ and $c = (c_0, c_1, \dots, c_N)$. Same for the following two algorithms and the examples shown in this vignette. For `eff_ini_maxeig_tri` ($a, b, c, \text{digit_thresh}$) and `eff_ini_mineig_tri` ($a, b, -c, \text{digit_thresh}$), c is free. For `eff_ini_seceig_tri` ($a, b, \text{digit_thresh}$), the matrix $T \sim (a_k, -c_k, b_k)$ should satisfy that $b_0 = c_0$, $a_j + b_j - c_j \equiv 0$, $1 \leq j \leq N-1$, and $a_N = c_N$.

- `eff_ini_mineig_tri` ($a, b, -c, \text{digit_thresh}$): calculate the minimal eigenpair for Hermitizable tridiagonal matrix $T \sim (a_k, -c_k, b_k)$ based on [1] (2016) ~ [3] (2018).

- `eff_ini_seceig_tri` ($a, b, \text{digit_thresh}$): calculate the sub-maximal eigenpair for the tridiagonal matrix $T \sim (a_k, -c_k, b_k)$. As mentioned in [4] (2018), for simplicity, here we assume that the sums of each row of the input tridiagonal matrix should be 0.

There are four auxiliary functions:

•**eff_ini_tri**: generate the efficient initial eigenpair (v_0, z_0) and the upper triangular matrix M for the tridiagonal matrix $\tilde{Q} \sim (\tilde{a}, \tilde{b}, \tilde{c})$: $\tilde{b}_0 = \tilde{c}_0$, $\tilde{c}_k = \tilde{a}_k + \tilde{b}_k$, $1 \leq k \leq N-1$, and $\tilde{c}_N > \tilde{a}_N$ based on two input vectors $\tilde{a}, \tilde{b}$.

•**thoma**: explicit representation of the solution w to equation $(-T - zI)w = v$ based on $T \sim (a_k, -c_k, b_k)$, v, z .

•**shif_inv_delta**: shifted inverse iteration algorithm to compute the maximal eigenpair of symmetric matrix Q^{sym} , based on $Q^{\text{sym}} \sim (as, -cs, bs)$, the upper triangular matrix M , the initial eigenpair (v_0, z_0) and $\tilde{a}, \tilde{b}$.

•**testing**: main function including the following examples.

Examples

In this vignette, we show how to apply the MATLAB procedure to the tridiagonal examples in [4] (2018).

The computational efficiency and results of the algorithm are studied as follows. The numerical experiments are fulfilled on a PC with Intel(R) Core(TM) i5-5200 CPU @2.20 GHz and 4.00 GB RAM. These algorithms are implemented in MATLAB(R2014a).

The maximal eigenpair

Example 3 in [4] (2018)

```
n = 8; digit_thresh= 6;
a = ones(1, n - 1); b = 2 * ones(1, n - 1); c = 3 * ones(1, n);
[z, ~] = eff_ini_max eig_tri (a, b, c, digit_thresh)
When n = 8; 16; 50; 100; 500, the outputs are as follows
m = 0
```

n	8	16	50	100	500
$z \approx \lambda_{\max}$	-0.342148	-0.219732	-0.176937	-0.172941	-0.171628

Example 5 in [4] (2018) and Example A3 in [2] (2017)

```
a = [0.5142, 0.2115, 0.8442, 0.2347, 0.9837];
b = [0.9962, 0.1111, 0.1405, 0.7595, 0.0781];
c = [-2.334, -2.6725, -2.263, -2.8457, -2.2257, -2.1582];
[z, v] = eff_ini_max eig_tri (a, b, c, 6)
The outputs are as follows
m = 4.449400e + 00
z = 3.26753
v = [0.5551, 0.3737, 0.2150, 1.8648, 0.2502, 0.7872]*
```

Example 6 in [4] (2018)

```

a = [8 - 4 * i, 12 - 4 * i, 1 - 2 * i, 2 - 3 * i, 4 + 2 * i];
b = [2 + i, 3 + i, 1 + 2 i, 2 + 3 i, 2 - i];
c = [2, 1, 3, 2, 4, -3];
[z, v] = eff_ini_maxeig_tri (a, b, c, 6)
The outputs are as follows
m = 1.188518e + 01
z = 6.1964
v = [0.3888 + 0.0000 i, 1.7102 + 2.2803 i, -0.0000 + 8.5745 i,
      -2.4128 - 1.8096 i, 1.3255 - 0.7811 i, 0.3372 - 3.0255 i]*

```

Example 7 in [4] (2018) and Example 19 in [3](2018)

```

b = [2 + i, (2^4) * (2 + i), (3^4) * (2 + i), (4^4) * (2 + i)];
a = [(2^2) * (2 - i), (6^2) * (2 - i), (12^2) * (2 - i), (20^2) * (2 - i)];
c = [1, 1, 1, 1, 1];
[z, v] = eff_ini_maxeig_tri (a, b, c, 6)
The outputs are as follows
m = 8.934272e + 02
z = 754.391
v = [0.0001 + 0.0000 i, 0.0387 - 0.0516 i, -0.5724 - 1.9625 i,
      -10.5869 - 3.9814 i, -14.1158 + 8.9998 i]*

```

Example 8 in [4] (2018)

```

n = 32;
a = zeros(1, n - 1); b = zeros(1, n - 1);
c = zeros(1, n); c(1) = 2;
for = 1 : (n - 1)
b(k) = 2 * k;
a(k) = k;
c(k + 1) = k + 2 * (k + 1);
end
[z, ~] = eff_ini_maxeig_tri (a, b, c, 6)
When n = 8; 16; 32; 50; 100, the outputs are as follows
m = 0

```

n	8	16	32	50	100
$z \approx \lambda_{\max}$	-1.01355	-1.00011	-1.0	-1.0	-1.0

Example 9 in [4] (2018)

```

a = zeros(1, n - 1); b = zeros(1, n - 1);
c = ones(1, n);
for k = 1 : (n - 1)

```

$b(k) = k^4$;
 $a(k) = k * (k - 1/2) * (k^2 + 3 * k + 3)$;
 $c(k + 1) = (k + 1)^4 + a(k)$;
 end
 $[z, \sim] = \text{eff_ini_maxeig_tri}(a, b, c, 6)$
 When $n = 8; 50; 100; 500; 1000; 5000; 10000$, the outputs are as follows
 $m = 0$

n	8	50	100	500
$z \approx \lambda_{\max}$	-0.633466	-0.548169	-0.533353	-0.514498
n	1000	5000	10000	
$z \approx \lambda_{\max}$	-0.510182	-0.504512	-0.503184	

Example 10 in [4] (2018)

$a = (1 : n - 1) \wedge 2$; $b = (1 : n - 1) \wedge 2$; $c = \text{ones}(1, n)$;
 $c(2 : n - 1) = a(1 : n - 2) + b(2 : n - 1)$; $c(n) = a(n - 1) + n^2$;
 $[z, \sim] = \text{eff_ini_maxeig_tri}(a, b, c, 6)$
 When $n = 8; 100; 500; 1000; 5000; 7500; 10000$, the outputs are as follows
 $m = 0$

n	8	100	500	1000
$z \approx \lambda_{\max}$	-0.525268	-0.376383	-0.338329	-0.32724
n	5000	7500	10000	
$z \approx \lambda_{\max}$	-0.308529	-0.308529	-0.302561	

Example 11 in [4] (2018)

$b4 = 10^6$;
 $b = [5, 4, 1, 6]$; $a = [3, 2, 10, 11]$; $c = [5, 7, 3, 16, 11 + b4]$;
 $[z, \sim] = \text{eff_ini_maxeig_tri}(a, b, c, 6)$
 When $b4 = 0.01; 1; 100; 10^6$, the outputs are as follows
 $m = 0$

$b4$	0.01	1	100	10^6
$z \approx \lambda_{\max}$	-0.000278686	-0.0245175	-0.182819	-0.195145

The minimal eigenpair

Example 5 in [4] (2018) and Example A3 in [2] (2017)

$a = [0.5142, 0.2115, 0.8442, 0.2347, 0.9837]$;
 $b = [0.9962, 0.1111, 0.1405, 0.7595, 0.0781]$;
 $c = [-2.334, -2.6725, -2.263, -2.8457, -2.2257, -2.1582]$;
 $[z, v] = \text{eff_ini_mineig_tri}(a, b, -c, 6)$
 The outputs are as follows

$m = 0$
 $z = 1.73679$
 $v = [0.6801, 0.2929, 0.2837, 1.1127, 0.4735, 3.9227]^*$

Example 6 in [4] (2018)

$a = [8 - 4 * i, 12 - 4 * i, 1 - 2 * i, 2 - 3 * i, 4 + 2 * i];$
 $b = [2 + i, 3 + i, 1 + 2 * i, 2 + 3 * i, 2 - i];$
 $c = [2, 1, 3, 2, 4, -3];$
 $[z, v] = \text{eff_ini_mineig_tri}(a, b, -c, 6)$
 The outputs are as follows
 $m = 1.788518e + 01$
 $z = -10.0244$
 $v = [0.3432 + 0.0000 * i, 1.4781 - 1.9707 * i, -0.0000 - 10.1769 * i,$
 $-3.2794 + 2.4596 * i, 2.4224 + 1.4275 * i, 0.1512 + 1.3570 * i]^*$

Example 7 in [4] (2018) and Example 19 in [3](2018)

$b = [2 + i, (2^4) * (2 + i), (3^4) * (2 + i), (4^4) * (2 + i)];$
 $a = [(2^2) * (2 - i), (6^2) * (2 - i), (12^2) * (2 - i), (20^2) * (2 - i)];$
 $c = [1, 1, 1, 1, 1];$
 $[z, v] = \text{eff_ini_mineig_tri}(a, b, -c, 6)$
 The outputs are as follows
 $m = 8.954272e + 02$
 $z = -756.391$
 $v = [0.0001 + 0.0000 * i, 0.0387 - 0.0516 * i, -0.5724 - 1.9625 * i,$
 $-10.5869 - 3.9814 * i, -14.1158 + 8.9998 * i]^*$

The sub-maximal eigenpair

Example 14 in [4] (2018)

$n = 16;$
 $b = 2. * \text{ones}(1, n - 1);$
 $a = \text{ones}(1, n - 1);$
 $c = 3. * \text{ones}(1, n); c(1) = 2; c(n) = 1;$
 $[z, \sim] = \text{eff_ini_seceig_tri}(a, b, 6)$
 When $n = 8; 16; 50; 100$, the outputs are as follows

n	8	16	50	100
z	-0.386874	-0.22592	-0.177154	-0.172969

Example 15 in [4] (2018)

$n = 16;$
 $b = (1 : n - 1).^2;$

$a = (1 : n - 1) \wedge 2;$
 $c(2 : n - 1) = b(2 : n - 1) + a(1 : n - 2); c(1) = 1; c(n) = a(n - 1);$
 $[z, \sim] = \text{eff_ini_seceig_tri}(a, b, 6)$
 When $n = 8; 16; 50; 100$, the outputs are as follows

n	8	16	50	100
z	-0.820539	-0.650141	-0.503596	-0.451977

Example 16 in [4] (2018)

$b = [5, 4, 1, 6];$
 $a = [3, 2, 10, 11];$
 $c = [5, 7, 3, 16, 11];$
 $[z, v] = \text{eff_ini_seceig_tri}(a, b, 6)$
 The outputs are as follows
 $z = -3.03673$
 $v = [0, 0.5382, 0.9955, 0.5431, 0.0953]^*$

Acknowledgments The algorithm is reformulated based on a series of recent papers by Mu-Fa Chen. The research project is supported in part by the National Natural Science Foundation of China (Nos. 11131003, 11626245, 11771046) and the Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions.

References

- [1] Chen, M.F. (2016). *Efficient initials for computing the maximal eigenpair*. Front. Math. China 11(6), 1379–1418.
- [2] Chen, M.F. (2017). *Global algorithms for maximal eigenpair*. Frontiers of Mathematics in China, 1–21.
- [3] Chen, M.F. (2018). *Hermitizable, Isospectral Complex Matrices or Differential Operators*. Front. Math. China.
- [4] Chen, M.F. and Li, Y.S. (2018). *Development of powerful algorithm for maximal eigenpair*. Preprint.

Additional references can be found from volumes 1-4 in the middle of Chen's homepage:

<http://math0.bnu.edu.cn/~chenmf>

Maintainer Yue-Shuang Li, email: liyueshuang at mail.bnu.edu.cn