CENG 371 - Scientific Computing
Fall 2022-2023
Homework 3

KILIÇ, DEKAN
dekan.kilic@metu.edu.tr

December 25, 2022

---

1. (a) I implemented the power method in *power_method.m* file.

   (b) I implemented the shifted inverse power method in *inverse_power* file.

   (c) In order to find the largest (in magnitude) eigenvalue and the corresponding eigenvector, I have run the power method as follows

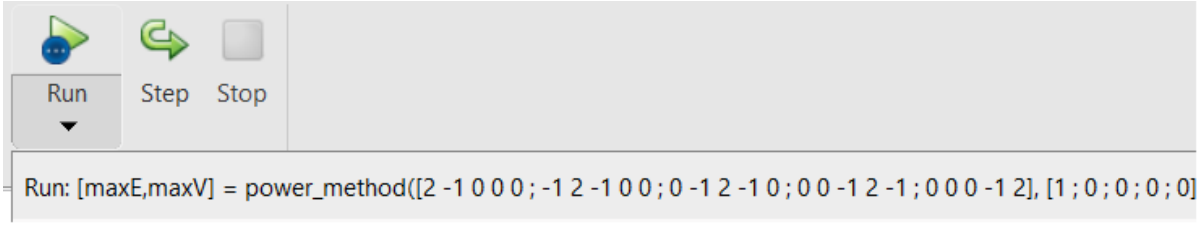

Figure 1: Running the power method for the given matrix in 1c

After this run, it returns the largest eigenvalue and the corresponding eigenvector as follows

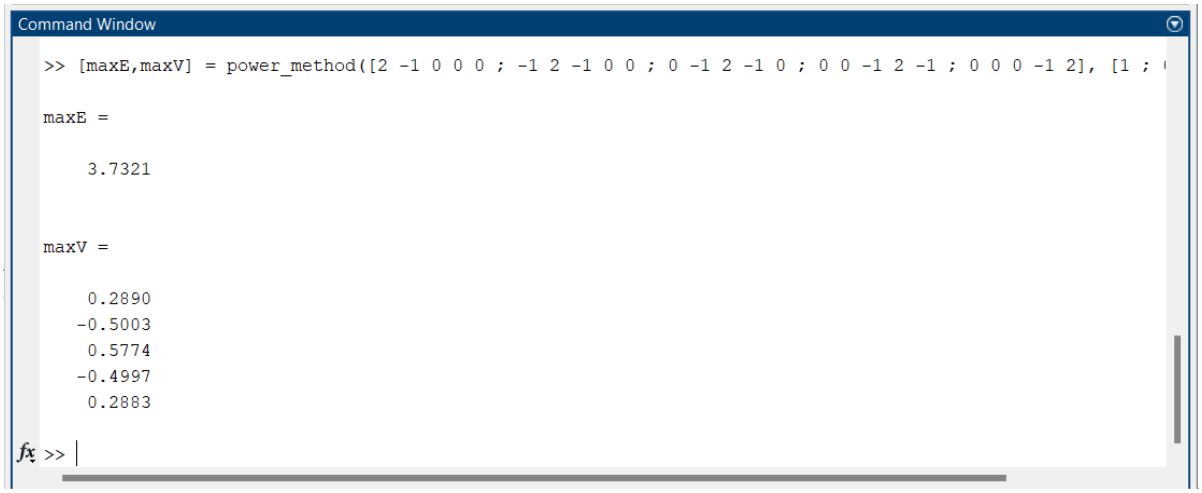

Figure 2: The result of the power method on the given matrix in 1c

Also, since the matrix is symmetric and also we know that the largest eigenvalue of the given matrix's inverse is the smallest eigenvalue of the original matrix, the smallest eigenvalue is equal to

$\lambda_{smallest} = \dfrac{1}{\lambda_{largest}} = \dfrac{1}{3.7321} = 0.2679$

(d) After the first multiplication of the initial starting vector with given matrix, we get a zero vector as follows

$$\begin{bmatrix} 0.2 & 0.3 & -0.5 \\ 0.6 & -0.8 & 0.2 \\ -1.0 & 0.1 & 0.9 \end{bmatrix} \times \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

which means that the initial starting vector has no component in the direction of v1 which is the eigenvector corresponding to the largest eigenvalue.That is, if we try to calculate the largest eigenvalue and the corresponding eigenvector with pen and paper, we can have such a case.But, on the computer due to the errors in the computations, this is not possible.Moreover, I have tried to compute the largest eigenvalue and the corresponding eigenvector on my computer by using the power method that I implemented, and it gives this result,

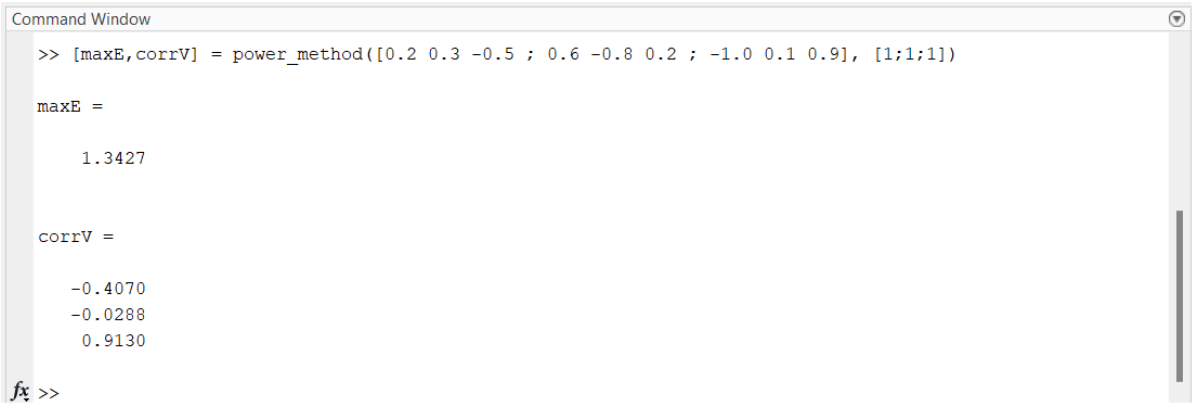

Figure 3: The result of the power method on the given matrix in 1d

2. (a) The given idea is actually something like making the largest root of the characteristic polynomial of a matrix zero out one by one by subtracting the given matrix $\lambda_i \frac{v_i v_i^T}{v_i^T v_i}$ after finding each eigenvector of a matrix.Let's consider

$$A - \lambda_i v_i v_i^T \text{ where } v_i^T v_i = 1$$

because we always normalize the eigenvectors in power method.Then, if we apply the power method on this new matrix after finding the first eigenvector that is corresponding to the largest eigenvalue, we have

$$(A - \lambda_i v_i v_i^T)v_j = Av_j - \lambda_i v_i v_i^T v_j = Av_j - \lambda_i v_i (v_i^T v_j)$$

As we said before, $v_i^T v_i = 1$, so

$$\text{when j} = 1,\ Av_1 - \lambda_1 v_1 v_1^T v_1 = \lambda_1 v_1 - \lambda_1 v_1 = 0$$
$$\text{when } j \neq 1, Av_j - \lambda_1 v_1 v_1^T v_j = \lambda_j v_j - \lambda_1 v_1 (0) = \lambda_j v_j.$$

Therefore, we zero out the largest eigenvalue and the corresponding eigenvector, and now the modified matrix has the same eigenpair with the original matrix except the largest one, so we now have the second largest eigenvalue of the original matrix, and we can easily find out it by using Rayleigh's coefficient like in the power method implementation.

As as side note, I think this idea works for the symmetric matrices only because to be able to make the cancellation (zero out), we need to have that each of the eigenvectors of the matrix has to be orthogonal to each other.I mean that if we construct a matrix which consists of all the eigenvectors, it must be an orthogonal matrix such that $v_i^T v_j = 0$ where $i \neq j$.Since we know that the eigenvectors of a real symmetric matrix are orthogonal to each other, we ony apply this method on these kind of matrices.

(b) I implemented the given idea in *power_k.m* file.I returns the largest k eigenvalue in a row vector, and the corresponding eigenvectors in a $k \times k$ matrix whose columns are the eigenvectors of the corresponding eigenvalues.

(c) I implemented the subspace iterations in *subspace_iteration.m* file.I returns the largest k eigenvalue in a column vector, and the corresponding eigenvectors in a $k \times k$ matrix.

(d) I run these two algorithms with different k values, and put the performance measurements of these two algorithms based on different k values.

Firstly, I run the two algorithms with the given matrix can_{229} to find the first 10 largest eigenvalues.I got the following result,

Profile Summary (Total time: 2.942 s)

Flame Graph

Generated 25-Ara-2022 17:09:05 using performance time.

Function Name	Calls	Total Time (s)	Self Time* (s)	Total Time Plot (dark band = self time)
hw3	1	2.942	0.001	
subspace_iteration	1	2.902	2.902	
power_k	1	0.039	0.039	

***Self time** is the time spent in a function excluding any time spent in child functions. The time includes any overhead time resulting from the profiling process.

Figure 4: Performance comparision of the two methods when k = 10.

Secondly, I run the two algorithms with the given matrix can_{229} to find the first 70 largest eigenvalues.I got the following result,

Profile Summary (Total time: 27.734 s)

Flame Graph

Generated 25-Ara-2022 17:13:08 using performance time.

Function Name	Calls	Total Time (s)	Self Time* (s)	Total Time Plot (dark band = self time)
hw3	1	27.734	0.001	
subspace_iteration	1	26.609	26.609	
power_k	1	1.124	1.124	

Figure 5: Performance comparision of the two methods when k = 70.

Lastly, I run the two algorithms with the given matrix can_{229} to find the first 229 largest eigenvalues.I got the following result,

Profile Summary (Total time: 459.310 s)

Flame Graph

Generated 25-Ara-2022 17:38:09 using performance time.

Function Name	Calls	Total Time (s)	Self Time* (s)	Total Time Plot (dark band = self time)
hw3	1	459.310	0.007	
subspace_iteration	1	451.858	451.858	
power_k	1	7.445	7.445	

***Self time** is the time spent in a function excluding any time spent in child functions. The time includes any overhead time resulting from the profiling process.

Figure 6: Performance comparision of the two methods when k = 70.

References

[1] This is the link that I get some help for QR factorization in MatLab : QR factorization in Matlab. *Mathworks*,

[2] This is the link that I get some help for norm of a vector in MatLab Norm of a Vector in Matlab. *Mathworks*,