SI231 - Matrix Computations, 2021 Fall

Solution of Homework Set #4

Prof. Yue Qiu

Acknowledgements:

1) Deadline: **2021-12-01 23:59:59**
2) **Late Policy details** can be found on piazza.
3) Submit your pdf homework in **Homework 4** on **Gradscope**. Make sure that you have correctly selected pages for each problem. If not, you probably will get 0 point.
4) No handwritten homework is accepted. You need to write LaTeX.
5) Use the given template and give your solution in English. Solution in Chinese is not allowed.
6) Your homework should be uploaded in the PDF format, and the naming format of the file is not specified.
7) For the calculation problems, you are highly required to write down your solution procedures in detail. And **all values must be represented by integers, fractions or square root**, floating points are not accepted.
8) When handing in your homework in gradescope, package all your codes into **your_student_id+hw4_code.zip**(must be zip) and upload it in **Homework 4 programming part**. In the package, you also need to include a file named README.txt/md to clearly identify the function of each file. Make sure that your codes can run and are consistent with your homework.

I. Eigenvalue and eigenspace

Problem 1. (15 points)

For a n $\times$ n matrix $\mathbf{A}$.

1) If $\mathbf{A}$ is a orthogonal projector, prove that $\mathbf{A}$'s eigenvalue is 0 or 1.
2) If $\mathbf{A}$ is a orthogonal projector, prove the eigenspace $\mathcal{S}_1$ associate with eigenvalue 1 and eigenspace $\mathcal{S}_0$ associate with eigenvalue 0 is orthogonal.
3) Prove that the similar transform can not bridge the defective matrix and non-defective.

Solution:

II. Similarity Transformation and Eigenvalue Decomposition

Problem 1. (5 × 5 points)

1) For a non-singular $n \times n$ matrix $\mathbf{A}$, if λ is the eigenvalue of $\mathbf{A}$, prove that λ^{-1} is the eigenvalue of $\mathbf{A}^{-1}$.
2) For similar matrices $\mathbf{A}$ and $\mathbf{B}$, prove that they have the same characteristic polynomial.
3) For similar matrices $\mathbf{A}$ and $\mathbf{B}$, there is an invertible matrix $\mathbf{P}$ such that $\mathbf{A} = \mathbf{P^{-1}BP}$. If $\mathbf{v}$ is the eigenvector of $\mathbf{A}$, please give the eigenvector of $\mathbf{B}$.
4) If matrix $\mathbf{A}$ is Hermitian, prove that the eigenvectors of $\mathbf{A}$ corresponding to different eigenvalues are orthogonal. (Hint: Eigenvalues of Hermitian matrix are real.)
5) Consider a 3×3 matrix $\mathbf{A} = \begin{bmatrix} 1 & -1 & 0 \\ -1 & 1 & 0 \\ 3 & 3 & 2 \end{bmatrix}$, which is similar with the diagonal matrix Λ. Please use the eigenvalue decomposition to compute the matrix Λ and the matrix $\mathbf{P}$ such that $\mathbf{A} = \mathbf{P}\Lambda\mathbf{P}^{-1}$. **You are required to convert the eigenvectors into unit vectors.**

Solution:

III. Eigenvalue Computations

Problem 1. (5 +16 +9 points)

Consider matrices

$$\mathbf{A} = \begin{bmatrix} 0 & 100 \\ 1 & 0 \end{bmatrix} \quad \text{and} \quad \mathbf{B} = \begin{bmatrix} 10 & 1 & 2 & 3 & 4 \\ 1 & 9 & -1 & 2 & -3 \\ 2 & -1 & 7 & 3 & -5 \\ 3 & 2 & 3 & 12 & -1 \\ 4 & -3 & -5 & -1 & 15 \end{bmatrix}$$

with $\alpha, \beta > 0$

1) Find the eigenvalues and eigenvectors of $\mathbf{A}$ by hand. (5 points)
2) Program **the power iteration** (See Algorithm 1) and **the inverse iteration** (See Algroithm 2) respectively and report the output of two algorithms for $\mathbf{A}$, do the two algorithms converge or not? Report what you have found (you can use plots to support your analysis). (13 points: programming takes 8 points and the analysis takes 5 points) After a few iterations, the sequence given by the power iteration fails to converge, explain why. (3 points)
3) Program **Modified Power iteration** (see Algorithm 3) and report the output of the two power iteration algorithms (algorithm 1 and 3) for $\mathbf{B}$ ($K = 100, tol = 10^{-7}$). Which algorithm is more accurate for the same tolerance (you can run 10 times and compare the average output).(9 points: programming takes 5 points and the analysis takes 4 points)

Remarks:

- Programming languages are not restricted. You are free to use MATLAB's **eig** or Python's **numpy.linalg.eig** to generate the eigenvalues and eigenvectors of $\mathbf{A}$ as a reference to study the convergence.
- The function accepts input: $n \times n$ matrix $\mathbf{A}$, a $n \times 1$ vector $\mathbf{q}$, tolerance, max iterations. And the function should return largest (smallest) eigenvalue and iterations k.
- You need to write the demo script which can show your outputs, i.e., eigenvalues, iterations, errors, figures.

Algorithm 1: Power iteration

Input : $\mathbf{A} \in \mathbb{C}^{n\times n}$,initial vector $\mathbf{q}^{(0)}$, tolerance tol, K

Initilization: $error = 1$, $k = 0$.

while $error > tol$ and $k < K$ **do**

$k = k + 1$

$\mathbf{z}^{(k)} = \mathbf{A}\mathbf{q}^{(k-1)}$

$\mathbf{q}^{(k)} = \mathbf{z}^{(k)}/\|\mathbf{z}^{(k)}\|_2$

$\lambda^{(k)} = (\mathbf{q}^{(k)})^H \mathbf{A}\mathbf{q}^{(k)}$

$error = \|\mathbf{q}^{(k)} - \mathbf{q}^{(k-1)}\|_2$

end

Output: $\lambda^{(k)}$, k

Algorithm 2: Inverse iteration

Input : $\mathbf{A} \in \mathbb{C}^{n\times n}$, μ,initial vector $\mathbf{q}^{(0)}$, tolerance tol,K

Initilization: $error = 1$, $k = 0$.

while $error > tol$ and $k < K$ **do**

$k = k + 1$

$\mathbf{z}^{(k)} = (\mathbf{A} - \mu\mathbf{I})^{-1}\mathbf{q}^{(k-1)}$

$\mathbf{q}^{(k)} = \mathbf{z}^{(k)}/\|\mathbf{z}^{(k)}\|_2$

$\lambda^{(k)} = (\mathbf{q}^{(k)})^H \mathbf{A}\mathbf{q}^{(k)}$

$error = \|\mathbf{q}^{(k)} - \mathbf{q}^{(k-1)}\|_2$

end

Output: $\lambda^{(k)}$,k

Algorithm 3: Modified Power iteration

Input : $\mathbf{A} \in \mathbb{C}^{n\times n}$,initial vector $\mathbf{q}^{(0)}$, tolerance tol,K

Initilization: $\mathbf{V}^{(0)} = \mathbf{A}$, $error = 1$, $k = 0$.

while $error > tol$ and $k < K$ **do**

$k = k + 1$

$\mathbf{U}^{(k)} = \mathbf{V}^{(k-1)} * \mathbf{V}^{(k-1)}$

$\mathbf{V}^{(k)} = \mathbf{U}^{(k)}/\text{trace}(\mathbf{U}^{(k)})$

$q = \mathbf{V}^k q^{(0)}$

$\lambda^{(k)} = \frac{(\mathbf{q})^H \mathbf{V}^{(0)}\mathbf{q}}{(\mathbf{q})^H \mathbf{q}}$

$error = \|\mathbf{V}^{(k)} - \mathbf{V}^{(k-1)}\|_2$

end

Output: $\lambda^{(k)}$, k

Solution:

IV. QR Iteration and Hessenberg QR Iteration

Problem 1. (18 + 4 + 7 points)

For $\mathbf{A} \in \mathbb{C}^{n\times n}$, consider the QR iteration (see Algorithm 4) for finding all the eigenvalues and eigenvectors of $\mathbf{A}$. In each iteration $\mathbf{A}^{(k)}$ is similar to $\mathbf{A}$. However, $\mathcal{O}(n^3)$ flops that required in each iteration make this algorithm computationally expensive. Then, we try to find a matrix structure that is preserved by the QR algorithm and that lowers the cost of a single iteration step. One simple matrix structure that is close to upper triangular form and is preserved by the QR algorithm is the Hessenberg form. Therefore, by Givens rotations each QR Iteration only takes $\mathcal{O}(n^2)$ flops (see Algorithm 5). Besides, QR iteration may not converge as shown in class, **shift** is required (see Algorithm 6).

Given matrices

$$\mathbf{D} = \begin{bmatrix} 10 & 0 & \dots & 0 \\ 0 & 9 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 0 & \dots & 0 & 1 \\ 0 & \dots & 1 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 1 & \dots & 0 & 0 \end{bmatrix}$$

1) Generate a dense symmetric matrix $\mathbf{A} \in \mathbb{R}^{10\times 10}$ whose eigenvalues are the diagonal entry of $\mathbf{D}$ and whose eigenvectors are an orthogonalized set of normalized Gaussian random vectors of length 10 (It can be computed via the Q factor from a QR factorization of a 10×10 Gaussain random matrix). Program **QR Iteration** and **Hessenberg QR Iteration** respectively. Use them on $\mathbf{A}$ and report error $\|diag(\mathbf{H}^{(k)}) - diag(\mathbf{D})\|_2$ in each iteration($K = 30$). Be specific, you show attach an *error vs iteration time* figure in the pdf.
2) Prove that in each iteration of QR Iteration with shifts $\mathbf{A}^{(k)}$ is similar to $\mathbf{A}$.
3) Program **QR Iteration with shifts** and use it on $\mathbf{B}$ with a fixed shift value $\mu = 2$. Report error $\|diag(\mathbf{H}^{(k)}) - diag(\mathbf{H}^{(k-1)})\|_2$ in each iteration($K = 100$). Be specific, you show attach an *error vs iteration* figure in the pdf.

Remark:

1) You can use build-in QR function when generating matrix, but do not use build-in QR when implementing QR Iteration algorithms.
2) Attach your figures in pdf which means no figures in pdf no point.

Algorithm 4: QR iteration

Input : $\mathbf{A} \in \mathbb{C}^{n\times n}$,$K$

Initilization: $\mathbf{A}^{(0)} = \mathbf{A}$

for $k = 1, 2, ..., K$ **do**

  $\mathbf{Q}^{(k)}\mathbf{R}^{(k)} = \mathbf{A}^{(k-1)}$ QR factorization of $\mathbf{A}^{(k-1)}$

  $\mathbf{A}^{(k)} = \mathbf{R}^{(k)}\mathbf{Q}^{(k)}$

end

Output: $\mathbf{A}^{(k)}$

Algorithm 5: Hessenberg QR iteration

Input : $\mathbf{A} \in \mathbb{C}^{n\times n}$,$K$

Initilization: $\mathbf{H}^{(0)} = \mathbf{Q}^H\mathbf{A}\mathbf{Q}$ Hessenberg reduction for $\mathbf{A}$

for $k = 1, 2, ..., K$ **do**

  $\mathbf{Q}^{(k)}\mathbf{R}^{(k)} = \mathbf{H}^{(k-1)}$ QR factorization of $\mathbf{H}^{(k-1)}$ using Givens QR

  $\mathbf{H}^{(k)} = \mathbf{R}^{(k)}\mathbf{Q}^{(k)}$

end

Output: $\mathbf{H}^{(k)}$

Algorithm 6: QR iteration with shifts

Input : $\mathbf{A} \in \mathbb{C}^{n\times n}$,$K$

Initilization: $\mathbf{A}^{(0)} = \mathbf{A}$

for $k = 1, 2, ..., K$ **do**

  Pick a shift $\mu^{(k)}$

  $\mathbf{Q}^{(k)}\mathbf{R}^{(k)} = \mathbf{A}^{(k-1)} - \mu^{(k)}\mathbf{I}$ QR factorization of $\mathbf{A}^{(k-1)} - \mu^{(k)}\mathbf{I}$

  $\mathbf{A}^{(k)} = \mathbf{R}^{(k)}\mathbf{Q}^{(k)} + \mu^{(k)}\mathbf{I}$

end

Output: $\mathbf{A}^{(k)}$

Solution: