Homework 3: PageRank

General instructions

- You must submit your results and solutions in a single PDF file. Please feel free to use Jupyter Notebook and export for quick PDF report.
- This assignment is due on April 7th 2024 by 23:00:00. Submissions will NOT be accepted after this deadline.
- If you have any questions, you can contact TA Zhang Wenrang via email at 223040237@link.cuhk.edu.cn, or during their office hour (Wednesday 14:00-15:00, ZhiXin 4F, seat 72).

Exercises (20*4 = 80 pts)

1. Construct a python function to implement Power Method:

   - Inputs :
     - $A \in \mathcal{M}_n(\mathbb{K})$.
     - $x \in \mathbb{C}^n$: initial guess.
     - optional : max_iter, tolerance for loop exit, $\cdots$
   - Returns :
     - a : principal eigenvalue of A.
     - v : one eigenvector corresponding to the principal eigenvalue of A, $\|v\|_2 = 1$.

   Test your function on one random symmetric 4×4 real matrix that you create. Compare your result to that of *numpy.linalg.eig*. Show that your function works correctly: (i) your v is indeed an eigenvector; (ii) your v is indeed associated to your a; (iii) divide your v by that of *numpy.linalg.eig*.

2. Construct a python function to implement QR decomposition:

   - Inputs :
     - $A \in \mathcal{M}_n(\mathbb{K})$ (NO need to consider singular inputs).
   - Returns :
     - $Q \in \mathcal{M}_n(\mathbb{K})$: $QQ^* = I$.
     - $R \in \mathcal{M}_n(\mathbb{K})$: upper triangular.

   Test your function on one random 4×4 real matrix that you create (make sure this matrix is non-singular). Show for your function: (i) $A = QR$; (ii) $QQ^* = I$; (iii) R is upper triangular.

3. Construct a mapping "**permutation**" that provides a permutation matrix from a complex vector following the order of the modulus of the entries.

- Inputs :
  - $v \in \mathbb{C}^n$.
- Returns :
  - $P \in \mathcal{M}_n(\mathbb{R})$ orthogonal such that $\forall i \in [n]$, $P_{i,j} = 1$ if v_j is the entry of v of the i^{th} biggest module.

Note that if $P = \textbf{permutation}(v)$, Pv should be a vector with entries having a decreasing module. Use any example input v, show that your code works correctly.

4. Derive a python function that uses QR decomposition to compute eigenvalues following the algorithm below. The permutation matrix P is here to ensure that the biggest eigenvalue of A will be at the left top corner of the output matrix A_k.

- Inputs :
  - $A \in \mathcal{M}_n(\mathbb{K})$
  - $tol \in \mathbb{R}$: tolerance for loop exit
  - optional : $x \in \mathbb{C}^n$ initial guess
- Returns :
  - $A_k \in \mathcal{M}_n(\mathbb{R})$: upper triangular matrix (diagonal if A is Hermitian) with diagonal values sorted by their norm.
  - $V \in \mathcal{M}_n(\mathbb{R})$: invertible matrix such that $A = VA_kV^*$.

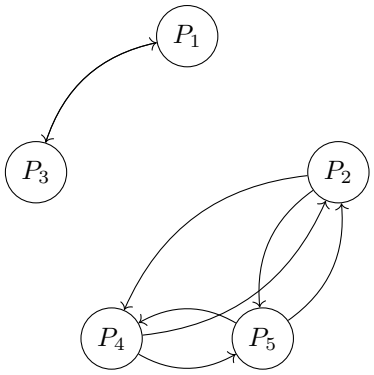
Algorithm 1 QR method.

If not given as input, consider an initial guess x
error = tol + 1
$A_k := A$
$V := I_n$
while error $>$ tol **do**
  $A_{\text{tmp}} := A_k$.
  Draw $\varepsilon \sim \mathcal{N}(0, 1)$.
  $Q_k, R_k :=$ QR decomposition of $A_k + \varepsilon I_n$.
  $A_k := R_kQ_k - \varepsilon I_n$.
  $P := \textbf{permutation}(\text{Diag}(A_k))$.
  $A_k := P^{-1}A_kP$.
  $V = VQ_kP$
  error $= \|\text{Diag}(A_k) - \text{Diag}(A_{\text{tmp}})\|$
Output A_k, V.

Test your function on a random 4×4 matrix A that you create and show that your function works correctly: (i) D is upper triangular (or diagonal if A is Hermitian); (ii) $VA_kV^* = A$; (iii) the eigenvector associated to the biggest eigenvalue of A is the first column of the output matrix V. [Hint]: To avoid potential instability, construct a Hermitian A.

Problem: Page Rank (4*5=20 pts)

1. Consider a Page Rank problem where



We want to find the scores $\{v_i\}_i$ of the 5 web pages, which means we want to solve

$$Av = v, \quad v \in \mathbb{R}^n_+,$$

where A is the connection matrix that was presented at the beginning of Lecture 5.

Construct the matrix A and use the power method you wrote in the exercises above to find v. Try 10 different initial random guesses. What results do you observe? Do you really obtain eigenvectors?

2. Use the below vectors as initial guess, what results do you observe? Do you obtain eigenvectors?

$$x_1 = (0, 0.57, 0, 0.57, 0.57)^T$$

$$x_2 = (0.7, 0, 0.7, 0, 0)^T$$

3. How do you explain your observations in above questions? Why does the algorithm not work well?

4. Try to solve the same problem with perturbed $A' = (1-\alpha)A + \frac{\alpha}{5}\mathbb{1}\mathbb{1}^T$ instead of A and provide the result. According to the lecture, what difference does this perturbation bring?

5. Use **Algorithm 1** to diagonalize A. Show your result as well as that of *numpy.linalg.eig*. [Hint]: Use a small *tol*. If your code does not converge fast, you can relaunch with different initial guesses for a better luck.