Assignment 4
MECHTRON 3X03: Scientific Computation

Due at 11:59 PM on Wednesday, December 6

Fall 2023

Submission Guidelines

There are 3 problems worth a total of 50 marks. Please read all of the files included in the Assignment 4 handout as they contain useful information. Submit the following files on Avenue:

1. A completed version of `assignment4_handout.jl` containing your solution to Problems 1-2. Do **not** change the name of this file. Other than `LinearAlgebra`, do **not** include (via `import` or `using` or any other means) any libraries as part of your submission (they will be detected and removed by the grading script and you will receive a mark of zero). Do **not** change the name or function signatures of any of the functions you are asked to implement in Problems 1-2. You may write helper functions to simplify your solution (include these in your completed version of `assignment4_handout.jl`).

2. A PDF called `assignment4.pdf` containing your plot and analysis for Problem 3 (no need to include Julia code here). This PDF can be any combination of typed/scanned/handwritten, so long as it is legible (you will receive a grade of zero on any section that cannot be understood).

The handout also contains a file called `assignment4_test.jl`. This contains some (but not all) of the code that we will be using to grade your submission. Do **not** hand this file in. You are advised to use this script to test your code and modify it while debugging.

The handout contains one last file called `assignment4_plot.jl`. You will use this code (along with your completed functions in `assignment4_handout.jl`) to produce the plot you need for Problem 3. Do **not** hand this file in. You may find it useful to modify this code while working on your solution to Problem 3, but the plot you submit should be the result of running this script without making any modifications.

Use of Generative AI Policy

If you use it, treat generative AI as you would a search engine: you may use it to answer general queries about scientific computing, but any specific component of a solution or lines of code must be cited (see the syllabus for citation guidelines).

This is an individual assignment. All submitted work must be your own, or appropriately cited from scholarly references. Submitting all or part of someone else's solution is an academic offence.

Problems

Problem 1 (10 points)**:** Implement the power method from class for real symmetric matrices in the function template `power_method_symmetric` in `assignment4_handout.jl`. Use the Bauer-Fike theorem for symmetric matrices with the input parameter `tol` as your termination criterion. This function will be tested by `assignment4_test.jl` with slightly different inputs. Do not change the function signature.

Problem 2 (20 points)**:** *If you are unfamiliar with the terminology of graph theory, please see the Appendix.* Google search was originally based on the PageRank [1] algorithm for ranking the "importance" of pages on the web based on the directed graph structure created by hyperlinks. In this model, each webpage is a vertex in the graph, and each link is a directed edge from a source page to a destination page. Imagine a probabilistic "web surfer" that moves from webpage to webpage by following a uniform distribution over the current webpage's outgoing links. If there are n webpages, the behaviour of this agent is described by a *transition matrix* $P \in \mathbb{R}^{n\times n}$ with elements

$$P_{i,j} = \text{probability of arriving on page } i \text{ from page } j. \tag{1}$$

That is, the jth column describes the probability distribution for the destination of a web surfer starting at webpage i. Since the sum of each column of P is 1 and the entries are all nonnegative, we call P a *column-stochastic* matrix. Also note that since the web surfer chooses each link with equal probability, the nonzero values in column j are all equal to $1/d_j^+$, where the *outdegree* d_j^+ is the number of edges originating from page j. For the simple example in Fig. 1, the transition matrix is

$$P = \begin{bmatrix} 0 & 0.5 & 0 \\ 1 & 0 & 1 \\ 0 & 0.5 & 0 \end{bmatrix}. \tag{2}$$

Use the following facts to implement the PageRank algorithm in the function template `page_rank` in `assignment4_handout.jl`:

1. A column-stochastic matrix P's largest eigenvalue (by magnitude) is $\lambda_1 = 1$.
2. The eigenvector corresponding to λ_1 describes the "steady-state" solution of the web surfer as the number of links followed approaches infinity. In other words, the eigenvector v satisfying $Pv = v$ contains only nonnegative entries (or nonnegative if multiplied by -1) and when normalized (by the 1-norm) represents a probability distribution over the state of the web surfer after an infinite amount of time.
3. The PageRank "score" of webpage i is the ith entry of the eigenvector corresponding to λ_1.

Finally, if webpage i has no outgoing links (i.e., it is a "dead-end" and $d_i^+ = 0$), assume that the surfer randomly lands on any webpage (including webpage i) with equal probability. Do not use any built-in eigensolvers from `LinearAlgebra` to solve this problem. Instead, implement the power method to find λ_1's eigenvector. A directed graph passed in to your function will be specified as an integer matrix whose rows correspond to edges with the source vertex in the first column and the destination vertex in the second column. For example, the edges of the graph in Fig. 1 will be encoded as

$$\text{edges} = \begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 2 & 3 \\ 3 & 2 \end{bmatrix}. \tag{3}$$

Note that P is not symmetric, but we know the value of the maximum eigenvalue and therefore do not need the Bauer-Fike theorem to decide when to terminate. Your function will be tested by `assignment4_test.jl` with slightly different inputs. Do not change the function signature.

Problem 3 (20 points)**:** Run the script `assignment4_plot.jl` with your solution to Problem 2. This script applies your PageRank implementation on a graph representation of Wikipedia users' votes in administrator elections (this data is found in `wiki-Vote.txt` in the assignment files and is provided by the

Stanford Large Network Dataset Collection [2]). Each vertex represents a Wikipedia user, and each edge represents a vote from the edge source to the edge destination in an online administrator election. Thus, the output of PageRank can be interpreted as a weighted popularity score of sorts. Note that this dataset contains “dead end" users who did not cast any votes and must be handled according to the instructions in Problem 3. Include the output plot in your PDF submission. This plot compares the PageRank of each user against their incoming degree (i.e., number of votes received). Provide the following information in your PDF submission:

a) (10 points) the top 5 and bottom 5 vertices in the graph according to PageRank; and

b) (10 points) an explanation for why the incoming degree and the PageRank of each user are not perfectly correlated.

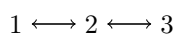

Figure 1: A very small directed graph for PageRank. In this scenario, website 1 has a link to website 2, website 3 has a link to website 3, and website 2 has links to both website 1 and 3.

References

[1] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank Citation Ranking: Bring Order to the Web," tech. rep., Stanford University, 1998.

[2] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection." http://snap.stanford.edu/data, June 2014.

Appendix - Graph Theory Primer

A directed graph $\mathcal{G}$ is a mathematical object consisting of a set of vertices (also called nodes) $\mathcal{V}$ and directed edges $\mathcal{E}$. In Figure 2, the vertices are

$$\mathcal{V} = \{1, 2, 3, 4\}, \tag{4}$$

and the edges can be stored in a 5×2 matrix as follows, where the first column contains the source vertex of each edge, and the second column contains the destination:

$$\mathcal{E} = \begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 2 & 3 \\ 3 & 2 \\ 2 & 4 \end{bmatrix}. \tag{5}$$

Each edge $e = (i, j) \in \mathcal{E}$ has a source vertex $i \in \mathcal{V}$ and a destination vertex $j \in \mathcal{V}$. Each edge is drawn as an arrow starting in the source vertex and pointing at the destination vertex (see Figure 2). The *indegree* d_i^- of a vertex $i \in \mathcal{V}$ is the number of edges in $\mathcal{E}$ whose destination vertex is i. Likewise, the *outdegree* d_i^+ is the number of edges in $\mathcal{E}$ whose source vertex is i. For example, for the graph in Figure 2, $d_4^+ = 0$ and $d_4^- = 1$.

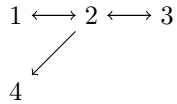

Figure 2: A small directed graph.