Homework 3 — Numerical Analysis Spring 2023

Instructions:

- Due 03/02 at 11:59pm on Gradescope.
- Write the names of anyone you work with on the top of your assignment. If you worked alone, write that you worked alone.
- Show your work.
- Include all code you use as copyable `monospaced` text in the PDF (i.e. not as a screenshot).
- Do not put the solutions to multiple problems on the same page.
- Tag your responses on gradescope. Each page should have a *single* problem tag. Improperly tagged responses will not receive credit.

Problem 1. In the last homework, we considered the following problem/task: You are given function $h : [-1, 1] \to \mathbb{R}$ and must return $\int_{-1}^{1} h(s)\mathrm{d}s$; i.e.

$$f(h) = \int_{-1}^{1} h(s)\mathrm{d}s.$$

Consider the following algorithm for this task:

$$\tilde{f}(h) = \sum_{i=0}^{100} \frac{1}{50} h(x_i), \qquad x_i = -1 + i/50.$$

(a) For each of the following inputs, compute the algorithm's output and compare it to the true solution $f(x)$.

input x	solution $f(x)$
$h(s) = 1$	2
$h(s) = s^2$	2/3
$h(s) = \sin(s)$	0

(b) Find an input for which the algorithm's output is very far from the true output. Explain why this is the case.

(c) Is this algorithms backwards stable? Justify your response.

Problem 2. For each problem, write down a matrix which performs the stated operations to a 3×4 matrix. Make sure to specify whether you should be applied on the left or right.

For reference, we will also show what the operation does to the following matrices:

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} 1 & 3 & -2 & 0 \\ 2 & -3 & 1 & 1 \\ 2 & -2 & 5 & 4 \end{bmatrix}$$

(a) Extract the second column.

$$\mathbf{A} \to \begin{bmatrix} 2 \\ 6 \\ 10 \end{bmatrix}, \qquad \mathbf{B} \to \begin{bmatrix} 3 \\ -3 \\ -2 \end{bmatrix}$$

(b) Extract the second column and place it in the third column of a 3×3 matrix of zeros.

$$\mathbf{A} \to \begin{bmatrix} 0 & 0 & 2 \\ 0 & 0 & 6 \\ 0 & 0 & 10 \end{bmatrix}, \qquad \mathbf{B} \to \begin{bmatrix} 0 & 0 & 3 \\ 0 & 0 & -3 \\ 0 & 0 & -2 \end{bmatrix}$$

(c) Swap the second and third columns.

$$\mathbf{A} \to \begin{bmatrix} 1 & 3 & 2 & 4 \\ 5 & 7 & 6 & 8 \\ 9 & 11 & 10 & 12 \end{bmatrix}, \qquad \mathbf{B} \to \begin{bmatrix} 1 & -2 & 3 & 0 \\ 2 & 1 & -3 & 1 \\ 2 & 5 & -2 & 4 \end{bmatrix}$$

(d) Sum up each column.

$$\mathbf{A} \to \begin{bmatrix} 15 & 18 & 21 & 24 \end{bmatrix}, \qquad \mathbf{B} \to \begin{bmatrix} 5 & -2 & 4 & 5 \end{bmatrix}$$

(e) Swap the second and third columns, then sum up each column.

$$\mathbf{A} \to \begin{bmatrix} 15 & 21 & 18 & 24 \end{bmatrix}, \qquad \mathbf{B} \to \begin{bmatrix} 5 & 4 & -2 & 5 \end{bmatrix}$$

(f) Sum all the entries (this one requires using two matrices).

$$\mathbf{A} \to \begin{bmatrix} 78 \end{bmatrix}, \qquad \mathbf{B} \to \begin{bmatrix} 12 \end{bmatrix}$$

Problem 3. Suppose we have defined

$$\mathbf{L}_1 = \begin{bmatrix} 1 & & & \\ \ell_{2,1} & 1 & & \\ \ell_{3,1} & & 1 & \\ \ell_{4,1} & & & 1 \end{bmatrix}, \qquad \mathbf{L}_2 = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & \ell_{3,2} & 1 & \\ & \ell_{4,2} & & 1 \end{bmatrix}, \qquad \mathbf{L}_3 = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & \ell_{4,3} & 1 \end{bmatrix}$$

Verify that

$$(\mathbf{L}_3\mathbf{L}_2\mathbf{L}_1)^{-1} = \begin{bmatrix} 1 & & & \\ -\ell_{2,1} & 1 & & \\ -\ell_{3,1} & -\ell_{3,2} & 1 & \\ -\ell_{4,1} & -\ell_{4,2} & -\ell_{4,3} & 1 \end{bmatrix}.$$

Problem 4. Suppose $\mathbf{A}$ is a $n \times n$ matrix with orthogonal columns; that is $[\mathbf{A}]_{:,i}^{\mathsf{T}}[\mathbf{A}]_{:,j} = 0$ for all $i \neq j$.

(a) What is $\mathbf{A}^{\mathsf{T}}\mathbf{A}$?

(b) Use (a) to determine $\mathbf{A}^{-1}$.

(c) Suppose $\mathbf{b}$ is a length n vector. Describe how you can solve the system $\mathbf{A}\mathbf{x} = \mathbf{b}$ using just $O(n^2)$ floating point operations.

Problem 5. Suppose

$$\mathbf{A} = \begin{bmatrix} 0 & 3 & -2 & 4 \\ 4 & -3 & 2 & 1 \\ 1 & 2 & 3 & -1 \\ 2 & 1 & 1 & 3 \end{bmatrix}$$

(a) Perform PLU factorization, using the row with the largest leading entry as the pivot row, to obtain a factorization $\mathbf{L}^{(3)}\mathbf{P}^{(3)}\mathbf{L}^{(2)}\mathbf{P}^{(2)}\mathbf{L}^{(1)}\mathbf{P}^{(1)}\mathbf{A} = \mathbf{U}$. Write each of the $\mathbf{L}^{(i)}$, $\mathbf{P}^{(i)}$, and $\mathbf{U}$ as you go, along with the current state of the matrix after applying each factor.

You should compute the factors exactly (i.e. using fractions), but you do not need to show you work when evaluating matrix-matrix products. It is recommended you use wolfram alpha, Mathematica, sympy, or some other symbolic math tool to assist you.

(b) Use (a) to find a factorization $\mathbf{PA} = \mathbf{LU}$.

(c) Perform regular LU factorization (without pivoting) on $\mathbf{PA}$. Show the row operation matrices you use along the way.

Problem 6. Access the file: `https://courses.chen.pw/na_s2023/hw3p6.py`

(a) Implement the functions `solve_LU(L,U,b)` and `solve_QR(Q,R,b)`. Provide the code for your implementations.

You can use the sample problems to check your algorithm. The errors should be small; say less than 10^{-12}.

Your implementations should run in time $O(n^2)$ and cannot use `np.linalg.solve` or a similar general purpose solver.

(b) For each algorithm, find the value of c so that the number of flops used by your algorithm is $cn^2 + O(n)$.