Homework 4 Numerical Analysis Fall 2024

Instructions:

- Due 10/21 at 6:00pm on Gradescope.
- You must follow the submission policy in the syllabus

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:

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

(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) Argue that the algorithm is not backwards stable. Justify your response.

(d) How could we restrict the input space to this algorithm to make it backwards stable?

Problem 2. For each $j = 1, 2, \dots, n-1$, define $\mathbf{L}_j \in \mathbb{R}^{n\times n}$ by

$$[\mathbf{L}]_{i,k} = \begin{cases} 1 & i = k \\ \ell_{i,j} & k = j \\ 0 & \text{o.w.} \end{cases}$$

For example:

$$\mathbf{L}_1 = \begin{bmatrix} 1 & & & & \\ \ell_{2,1} & 1 & & & \\ \ell_{3,1} & & 1 & & \\ \vdots & & & \ddots & \\ \ell_{n,1} & & & & 1 \end{bmatrix}, \quad \mathbf{L}_2 = \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & \ell_{3,2} & 1 & & \\ & \vdots & & \ddots & \\ & \ell_{n,2} & & & 1 \end{bmatrix}, \dots \quad \mathbf{L}_{n-1} = \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & \ell_{n,n-1} & 1 \end{bmatrix}$$

Verify that

$$(\mathbf{L}_{n-1}\cdots\mathbf{L}_2\mathbf{L}_1)^{-1} = \begin{bmatrix} 1 & & & & \\ -\ell_{2,1} & 1 & & & \\ -\ell_{3,1} & -\ell_{3,2} & 1 & & \\ \vdots & \vdots & & \ddots & \\ -\ell_{n,1} & -\ell_{n,2} & \cdots & -\ell_{n,n-1} & 1 \end{bmatrix}.$$

Hint: It suffices to check a certain product of matrices is the identity. Write down this product, and then compute it.

Problem 3. 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 $\mathbf{PA} = \mathbf{LU}$. You must show the work for how you obtain the factors (particularly the $\mathbf{L}$ factor).

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

(d) How do the steps you take in (a) and (c) compare?

Problem 4. In worksheet 4, we saw that applying Gaussian elimination to the matrix

$$\mathbf{A} = \begin{bmatrix} 10^{-20} & 1 \\ 1 & 1 \end{bmatrix}.$$

without pivoting results in an LU factorization that might have a large error.

Let's see that that is actually the case on a computer.

(a) Find a matrix $\mathbf{L}_1$ which introduces a zero in the bottom left corner by subtracting some multiple of the first row from the second row.

(b) Find the corresponding LU factorization for $\mathbf{A}$.

(c) Form the matrixes $\mathbf{A}$ and $\mathbf{L}_1$ in numpy, and multiply them to get $\mathbf{U}$. Form $\mathbf{L}$ by using the formula for $\mathbf{L} = \mathbf{L}_1^{-1}$.

Print out $\mathbf{A}$, $\mathbf{LU}$, and $\|\mathbf{A} - \mathbf{LU}\|_F$.

(d) Repeat this with partial pivoting. I.e. first find a permutation matrix $\mathbf{P}_1$ which swaps the rows of $\mathbf{A}$ so that the largest entry in the first column is in the first row. Multiply this matrix with $\mathbf{A}$. Then find $\mathbf{L}_1$ which introduces a zero in the bottom left corner by subtracting some multiple of the first row from the second row. Obtain $\mathbf{U} = \mathbf{L}_1(\mathbf{P}_1\mathbf{A})$ and $\mathbf{L}$.

Output the factorization and error.

Problem 5. Recall the growth factor for Gaussian elimination is the maximum ratio of the largest entry in $\mathbf{U}$ to the largest entry in $\mathbf{A}$. A bigger growth factor is correlated to a bigger loss of precision when the factorization is used to solve systems.

Apply PLU factorization (no pivoting if the rows have the same magnitude) to

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ -1 & 1 & 0 & 0 & 1 \\ -1 & -1 & 1 & 0 & 1 \\ -1 & -1 & -1 & 1 & 1 \\ -1 & -1 & -1 & -1 & 1 \end{bmatrix}.$$

Wilkinson showed that the growth factor for Gaussian elimination is 2^{n-1} in the worst case. This example shows that the growth factor of partial pivoting on a $n \times n$ matrix can be as large as 2^{n-1}. Other (more expensive) pivoting schemes can guarantee smaller growth factors. [1]

Problem 6. Let

$$\mathbf{Q} = \begin{bmatrix} | & | & & | \\ \mathbf{q}_1 & \mathbf{q}_2 & \cdots & \mathbf{q}_k \\ | & | & & | \end{bmatrix}.$$

(a) Suppose $\{\mathbf{q}_1, \ldots, \mathbf{q}_k\}$ are orthonormal. What is $\mathbf{Q}^\mathsf{T}\mathbf{Q}$?

(b) Suppose $\mathbf{x}$ is in the span of $\{\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_k\}$. Show that $\mathbf{x} = \mathbf{Q}\mathbf{c}$ for some vector $\mathbf{c} \in \mathbb{R}^k$.

(c) Suppose we know $\mathbf{x} = c_1\mathbf{q}_1 + c_2\mathbf{q}_2 + \cdots + c_k\mathbf{q}_k$, but we do not know the coefficients $c_1, c_2 \ldots, c_k$. Explain how you can obtain these coefficients from $\mathbf{Q}$ and $\mathbf{x}$.

[1] https://nhigham.com/2020/07/14/what-is-the-growth-factor-for-gaussian-elimination/