Homework3

Zric

January 8, 2026

1 Problem1: LU algorithm

1.1 problem description

Prove that the time complexity of the LU decomposition algorithm algorithm is $O(n^3)$,here L means lower triangular matrix and U upper triangular matrix.

1.2 solution

Proof:

For a nonsingular matrix $A \in \mathbb{R}^{n\times n}$ (with partial pivoting if needed), there exist a permutation matrix P, a unit lower triangular matrix L, and an upper triangular matrix U such that

$$PA = LU \tag{1}$$

The upper and lower triangular matrix satisfies:

$$a_{ij} = \sum_{s=1}^{n} l_{is}u_{sj}, \qquad l_{ii} = 1,\ l_{ij} = 0\ (j > i),\ u_{ij} = 0\ (i > j) \tag{2}$$

(1)Time complexity of LU decomposition: the decomposition can be devided into 3 steps:

step1: get the first column of L and the first row of U

$$\begin{cases} l_{i1} = a_{i1} & i = 1, 2..n \\ u_{1j} = \frac{a_{1j}}{l_{11}} & j = 1, 2..n \end{cases} \tag{3}$$

step2: get the second and following elements of L

$$l_{ij} = a_{ij} - \sum_{k=1}^{j-1} l_{ik}u_{kj} \qquad j = 2, 3, ...n \quad i = j, j+1, ...n \tag{4}$$

step3: use results from step2 to get elements of U

$$u_{ji} = \frac{1}{l_{jj}}(a_{ji} - \sum_{k=1}^{j-1} l_{jk}u_{ki}) \qquad j = 2, 3, ...n-1 \quad i = j+1, j+2, ...n \tag{5}$$

In the whole loop ,step2 and step3 is coupled with each other, each will use the output elements of the other one.The first step has complexity of $O(n)$. And in step2, the total number of loop is

$$\sum_{j=2}^{n}(n-j+1)(j-1) = \frac{1}{2}n^2(n-1) - \frac{1}{6}n(n-1)(2n-1) = \frac{1}{6}n^3 - \frac{1}{6}n \tag{6}$$

The same applies to step3, total number of loop in step3:

$$\sum_{j=2}^{n-1}(n-j)(j-1) = \frac{1}{6}n^3 - \frac{1}{2}n^2 + \frac{1}{3}n \tag{7}$$

(2)Time complexity of forward and backward substitution:

In solving $LUx = b$, forward substitution means solving $Ly = b$, backward substitution means solving $Ux = y$, the two process has the same time complexity $O(n^2)$, for a lower or upper triangular matrix, the total number of loop is:

$$\sum_{j=1}^{n}(j-1) = \frac{n(n-1)}{2} \tag{8}$$

Sum over equation(6)(7)(8), the leading term being $\frac{1}{3}n^3$, hence the LU decomposition algorithm has time complexity $O(n^3)$.

2 Problem2:

2.1 problem description

Solve systems of equations using the Gaussian elimination algorithm and partial-pivoting scheme (Note: Write a general program applicable to solving the following equations; select the pivot element with the largest coefficient among all columns for each elimination step; Optional: Compute the inverse matrix of the coefficient matrix).

$$\begin{cases} 2x_1 + 3x_2 + 5x_3 &= 5, \\ 3x_1 + 4x_2 + 8x_3 &= 6, \\ x_1 + 3x_2 + 3x_3 &= 5. \end{cases} \tag{9}$$

2.2 algorithm description

We solve $Ax = B$ by Gaussian elimination with partial pivoting on the augmented matrix $[A \mid B]$, here B can be generalized as $B \in \mathbb{R}^{n\times m}$.

(1) Forward elimination (with partial pivoting):

First for $k = 1, \ldots, n$, Select the pivot row $p = \max_{i\geq k} |a_{ik}|$ in column k and swap rows $k \leftrightarrow p$ in the whole augmented matrix.

Second for each $i = k+1, \ldots, n$, form the multiplier $l_{ik} = a_{ik}/a_{kk}$, then apply the row operation to each row below k

$$\text{row}_i \leftarrow \text{row}_i - l_{ik}\,\text{row}_k, \tag{10}$$

updating both the left block (turning A into an upper triangular U) and the right block B simultaneously. After this phase, the left block is U, and the right block is $B' = L^{-1}B$ (forward substitution has been implicitly applied by the same row operations).

(2) Back substitution (simultaneous for all right-hand sides): Solve $Ux = B'$ from bottom to top. For $i = n, \ldots, 1$ and for each RHS column r,

$$x_{ir} = \frac{1}{u_{ii}}\Big(b'_{ir} - \sum_{j=i+1}^{n} u_{ij}x_{jr}\Big). \tag{11}$$

(3) Matrix inverse (optional):

To get inversion matrix of A ,just set $B = I_n$ as identity matrix, and run the same procedure. The result x equals A^{-1}.

(4)Time complexity Let $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$.

Pivot search: $\sum_{k=1}^{n}(n-k+1) = O(n^2)$ comparisons.

Updating the left block (turning A into U): $2\sum_{k=1}^{n-1}(n-k)^2 = \frac{2}{3}n^3 + O(n^2)$ flops.

Updating the right block during elimination (equivalent to forward substitution for $n \times m$ RHS): about $mn^2 + O(mn)$ flops.

Back substitution for $n \times m$ RHS: about $mn^2 + O(mn)$ flops.

Total: flops $= \frac{2}{3}n^3 + 2mn^2 + O(n^2 + mn) = O(n^3)$.

2.3 output

```
Solution x:
['2.00000000', '2.00000000', '-1.00000000']

A^{-1}:
 6.00000000 -3.00000000 -2.00000000
 0.50000000 -0.50000000  0.50000000
-2.50000000  1.50000000  0.50000000

Check A x:
['5.00000000', '6.00000000', '5.00000000']

Check A * A^{-1}:
 1.00000000  0.00000000 -0.00000000
 0.00000000  1.00000000  0.00000000
 0.00000000  0.00000000  1.00000000
```

Figure 1: output of problem2.py

3 Problem3:

3.1 problem description

Solve the 1D Schrodinger equation with the potential (i) $V(x) = x^2$; (ii) $V(x) = x^4 - x^2$ with the variational approach using a Gaussian basis (either fixed widths or fixed centers). Consider the three lowest energy eigenstates.

The Gaussian basis functions are defined as: $\phi_i(x) = (\frac{v_i}{\pi})^{1/2}\exp(-v_i(x-s_i)^2)$. This function has two variational parameters: v_i the width of the Gaussian, and s_i the center

of the Gaussian. For simplicity, we only vary one of these parameters at a time and do calculations with either fixed widths or fixed centers.

3.2 algorithm description

To solve the one-dimensional Schrödinger equation in Hatree units

$$\hat{H}\psi(x) = E\psi(x), \qquad \hat{H} = -\tfrac{1}{2}\frac{d^2}{dx^2} + V(x), \tag{12}$$

with the potentials $V(x) = x^2$ and $V(x) = x^4 - x^2$

The trial wave function is expanded as

$$\psi(x) = \sum_i c_i\,\phi_i(x), \qquad \phi_i(x) = \sqrt{\tfrac{v_i}{\pi}}\,e^{-v_i(x-s_i)^2}, \tag{13}$$

where v_i is the Gaussian width and s_i its center.

For each pair of basis functions, analytical integrals are used to build:

$$S_{ij} = \langle\phi_i|\phi_j\rangle, \qquad T_{ij} = \tfrac{1}{2}\int \phi_i'(x)\phi_j'(x)\,dx, \qquad V_{ij} = \langle\phi_i|V(x)|\phi_j\rangle. \tag{14}$$

The Hamiltonian matrix is $H = T + V$. Because the basis is non-orthogonal ($S \neq I$),the eigenvalue equation can be write in matrix form.$c = [c_1, c_2, ..., c_n]^T$

$$Hc = \varepsilon Sc \tag{15}$$

S can be decomposed using Cholesky method as $S = R^T R$, R being the upper triangular matrix.Multiply R^{-T} on the left side:

$$R^{-T}HR^{-1}Rc = \varepsilon R^{-T}R^T Rc \to Ay = \varepsilon y, \quad A = R^{-T}HR^{-1}, \quad y = Rc \tag{16}$$

So we just need to find eigenvalues of A . In the code I use fixed-center basis, I take 9 Gaussian functions and set their centers in [-3,3] with the same interval, and they share the same width v for simplicity.

Then a grid of width $v \in [0.2, 3.0]$ is scanned. The value minimizing the ground-state energy $E_0(v)$ is chosen as the optimal width v_{best} to get E_0. Then the second and third excitation energies are computed in the same way.

3.3 output

```
[Fixed centers] potential = x2   basis size = 9
best width = 0.7220  energy 1  =0.70710683
best width = 1.1492  energy 2  =2.12228693
best width = 0.8644  energy 3  =3.53553854
[Fixed centers] potential = x4-x2   basis size = 9
best width = 1.6237  energy 1  =0.33801422
best width = 1.4339  energy 2  =1.62150652
best width = 0.9593  energy 3  =3.68652237
```

Figure 2: output of problem3.py

the output includes the best width for each energy.