CSCC37 A2

Leo (Si) Wang

Oct 31, 2022

Question 1

a)

$$\text{Let } m_k = \begin{bmatrix} 0 \\ \vdots \\ \frac{a_{(k+1)k}}{a_{kk}} \\ \vdots \\ \frac{a_{nk}}{a_{kk}} \end{bmatrix}$$

For a_{ij} are elements in the kth stage of eliminating A, i.e. $L_{k-1} \cdots L_1 A$

$$I - m_k e_k^T = I - \begin{bmatrix} 0 \\ \vdots \\ \frac{a_{(k+1)k}}{a_{kk}} \\ \vdots \\ \frac{a_{nk}}{a_{kk}} \end{bmatrix} \begin{bmatrix} 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \end{bmatrix}$$

$$= I - \begin{bmatrix} 0 & & \cdots & & 0 \\ \vdots & \ddots & & \iddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 \\ 0 & \cdots & \frac{a_{(k+1)k}}{a_{kk}} & \cdots & 0 \\ \vdots & & \vdots & & \vdots \\ 0 & \cdots & \frac{a_{nk}}{a_{kk}} & \cdots & 0 \end{bmatrix}$$

Note: the non-zero elements are on the kth column and start on row k+1.

$$= \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ \vdots & \ddots & & \iddots & \vdots \\ 0 & \cdots & 1 & \cdots & 0 \\ 0 & \cdots & -\frac{a_{(k+1)k}}{a_{kk}} & \cdots & 0 \\ \vdots & & \vdots & & \vdots \\ 0 & \cdots & -\frac{a_{nk}}{a_{kk}} & \cdots & 1 \end{bmatrix}$$

$$= L_k$$

b)

Let $L_k^{-1} = I + m_k e_k^T$, prove $L_k L_k^{-1} = I$

We know that $(m_k e_k^T)m_k$ is a matrix of 0's because the only non-zero elements in $m_k e_k^T$ are on the kth column and the non-zero elements in m_k start at row $k+1$, thus $(m_k e_k^T)^2$ is the zero matrix too.

$$\begin{aligned} L_k L_k^{-1} &= (I - m_k e_k^T)(I + m_k e_k^T) \\ &= I - I(m_k e_k^T) + I(m_k e_k^T) - (m_k e_k^T)^2 \\ &= I \end{aligned}$$

Thus $L_k^{-1} = I + m_k e_k^T$.

c)

For $j < k$ we know $(m_j e_j^T)(m_k e_k^T) =$ the zero matrix as the non-zero elements in $(m_j e_j^T)$ only occur on the jth column, while the non-zero elements in $(m_k e_k^T)$ only occur after the kth row. With $j < k$ there can never be a non-zero multiplication during the matrix multiplcation.

$$\begin{aligned} (L_k L_j)^{-1} &= L_j^{-1} L_k^{-1} \\ &= (I + m_j e_j^T)(I + m_k e_k^T) \\ &= I + m_j e_j^T + m_k e_k^T + (m_j e_j^T)(m_k e_k^T) \\ &= I + m_j e_j^T + m_k e_k^T \\ &= I + (I + m_j e_j^T) - I + (I + m_k e_k^T) - I \\ &= (I + m_j e_j^T) + (I + m_k e_k^T) - I \\ &= L_j^{-1} + L_k^{-1} - I \end{aligned}$$

d)

$$\begin{aligned} \tilde{L_k} &= P_i L_k P_i \\ &= P_i (I - m_k e_k^T) P_i \\ &= P_i I P_i - P_i (m_k e_k^T) P_i \\ &= I - \tilde{m_k} e_k^T, \text{ where } \tilde{m_k} = m_k \text{ with rows i, j swapped} \end{aligned}$$

We know post-multiplying P_i will swap columns i, j, but since we have $k < i < j$ this will have no effect on L_k nor $m_k e_k^T$ since the only non-zero elements are on column k. Additionally because of this property we know the row swap has changed $m_k e_k^T$ because the multipliers start at row $k+1$.

We can show this does not work for $i \leq k$, this will imply $i < j \leq k$. This is because the multipliers start on row $k+1$, so we would doing nothing by swapping rows of 0 if $i < j \leq k$.

Question 2

We can observe that if $A \in M^{nxn}$ is triangular, $det(A) = \prod_{i=1}^{n} a_{ii}$, meaning the determinant is the product of it's diagonal. The fact is shown in the process of finding a determinant where you choose the row or column to multiply by the determinant of the submatrix. Choosing the row/column with all zeros except the diagonal continuously through the submatrices, finally getting to the formula of only multiplying the diagonal entries.

From this fact we can get the following:

$$det(L_i) = 1, \text{ for any Gauss Transform } L_i$$
$$det(U_i) = \sum_{i=1}^{n} u_{ii}$$
$$det(P_i) = -1, \text{ for any permutation matrix } P_i$$
$$det(P) = \sum_{i=1}^{p} det(P_i), \text{ for p number of } P_i's$$

$$det(PA) = det(LU) \iff det(P)det(A) = det(L)det(U) \iff det(A) = \frac{n \sum_{i=1}^{n} u_{ii}}{(-1)^p}$$

Question 3

a)

$$L_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -\frac{4}{3} & 1 & 0 & 0 \\ -\frac{1}{3} & 0 & 1 & 0 \\ -\frac{2}{3} & 0 & 0 & 1 \end{bmatrix}, \; L_1A = \begin{bmatrix} 3 & 3 & 9 & 6 \\ 0 & 0 & -8 & -4 \\ 0 & 0 & 2 & 3 \\ 0 & 0 & -2 & 2 \end{bmatrix}$$

$$L_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}, \; L_2L_1A = \begin{bmatrix} 3 & 3 & 9 & 6 \\ 0 & 0 & -8 & -4 \\ 0 & 0 & 2 & 3 \\ 0 & 0 & 0 & 5 \end{bmatrix} = U$$

$$L_1^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{4}{3} & 1 & 0 & 0 \\ \frac{1}{3} & 0 & 1 & 0 \\ \frac{2}{3} & 0 & 0 & 1 \end{bmatrix}, \; L_2^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix}, \text{ Let } L = L_2^{-1}L_1^{-1}$$

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{4}{3} & 1 & 0 & 0 \\ \frac{1}{3} & 0 & 1 & 0 \\ \frac{2}{3} & 0 & -1 & 1 \end{bmatrix}$$

$$\implies L_1L_2A = U \implies A = L_2^{-1}L_1^{-1}U = LU$$

b)

$$\begin{aligned} det(A) &= \frac{det(L)det(U)}{det(P)} \\ &= \frac{n \times (3)(0)(2)(5)}{1} \\ &= 0 \end{aligned}$$

A det of 0 means the system will have infinitely many or no solutions.

c)

First solve $Ld = b$ for $d = Ux$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{4}{3} & 1 & 0 & 0 \\ \frac{1}{3} & 0 & 1 & 0 \\ \frac{2}{3} & 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \end{bmatrix} = \begin{bmatrix} 21 \\ 24 \\ 10 \\ 16 \end{bmatrix}$$

$$d_1 = 21$$
$$4/3d_1 + d_2 = 24 \implies d_2 = -4$$
$$1/3d_1 + d_3 = 10 \implies d_3 = 3$$
$$2/3d_1 - d_3 + d_4 = 16 \implies d_4 = 5$$
$$d = \begin{bmatrix} 21 \\ -4 \\ 3 \\ 5 \end{bmatrix}$$

Solve $Ux = d$ for x

$$\begin{bmatrix} 3 & 3 & 9 & 6 \\ 0 & 0 & -8 & -4 \\ 0 & 0 & 2 & 3 \\ 0 & 0 & 0 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 21 \\ -4 \\ 3 \\ 5 \end{bmatrix}$$
$$5x_4 = 5 \implies x_4 = 1$$
$$2x_3 + 3x_4 = 3 \implies x_3 = 0$$
$$0x_2 - 8x_3 - 4x_4 = -4 \implies x_2 = t \in \mathbb{R}$$
$$3x_1 + 3x_2 + 9x_3 + 6x_4 = 21 \implies x_1 = 5 - t$$
$$x = \begin{bmatrix} 5-t \\ t \\ 0 \\ 1 \end{bmatrix}, \text{ for } t \in \mathbb{R}$$

$\therefore$ There are infinite solutions.

d)

Because solving $Ax = b$ takes $O(n^2)$ time, but computing the LU factorization takes $O(n^3)$.
The LU factorization will be the same for any b so if you want to find the solutions to multiple b's, it is more efficient to not repeat the factorization.

Question 4

Here is the output table for running badsys on n from 4 to 15.

n	rcnd	re0	rr0	ref
--	----------	----------	----------	----------
4	0.00024768	3.7105e-15	4.0678e-17	1.0821e-15
5	1.0329e-05	2.7927e-13	6.0419e-17	1.9591e-13
6	3.6627e-07	2.9317e-12	1.9915e-17	5.0685e-14
7	1.1165e-08	3.3431e-11	1.1596e-16	4.397e-13
8	2.9656e-10	4.2625e-10	2.9577e-17	4.2488e-10
9	6.9489e-12	8.2195e-08	2.2721e-16	8.2177e-08
10	1.375e-13	2.0016e-06	8.7464e-17	4.9792e-10
11	2.4349e-15	1.9898e-06	2.2446e-16	1.9796e-06
12	3.9563e-17	0.0019801	3.9039e-17	0.0019801
13	5.9323e-19	0.036173	5.8799e-17	0.036158
14	8.2353e-21	4.4143	8.6243e-17	0.0038868
15	1.2419e-22	288.84	5.501e-17	288.84

This info is based on the algorithm not continuing if it finds the solution is not improving or subtractive cancellation is found in the iteration.

As the matrix get larger, elements approaching the bottom right corner become exponentially larger, causing a possible numerical singularity, especially with the bottom right element being the largest.
We can see that though the relative residual remains small, the relative error keeps increasing with the order of A. This can be seen best because of the reciprocal condition decreasing as n increases, meaning the condition number of A is getting exponentially big, combine this with the fact that relative round off is bound by the condition number times relative residual, and we can see why the relative residual is increasing.