CSC336 A3 Report

Tianshu Zhu
1002111225
Utorid zhutians

Q1

a):

Code:

```
errora = zeros(10, 2);
gammas = zeros(10, 1);
for k = 1:10
    gamma = 10^((-2)*k);
    gammas(k) = gamma;
    b = [1; 1];
    x = [1; -1];
    L1 = [1 0; 2/gamma 1];
    U1 = [gamma gamma-1; 0 2/gamma-1];
    y = L1\b;
    x_hat = U1\y;
    error = x_hat - x;
    errora(k, :) = error';
end
for k = 1:10
    fprintf('gamma: 10^(-2*%d); error1: %d; error2: %d\n', k, errora(k,1), errora(k,2));
end
```

Results:

gamma: 10^(-2*1); error1: 8.881784e-16; error2: 0
gamma: 10^(-2*2); error1: -1.101341e-13; error2: 0
gamma: 10^(-2*3); error1: 2.875566e-11; error2: 0
gamma: 10^(-2*4); error1: 5.024759e-09; error2: 0
gamma: 10^(-2*5); error1: 8.274037e-08; error2: 0
gamma: 10^(-2*6); error1: -2.212172e-05; error2: 0
gamma: 10^(-2*7); error1: -7.992778e-04; error2: 0
gamma: 10^(-2*8); error1: 1.102230e-01; error2: 0
gamma: 10^(-2*9); error1: -1; error2: 0

gamma: 10^(-2*10); error1: -1; error2: 0

The accuracy get worse as gamma decreases.

b):

Code:

```
errorb = zeros(10, 2);
for k = 1:10
    gamma = 10^((-2)*k);
    b_tilt = [1; 1];
    x = [1; -1];
    L2 = [1 0; gamma/2 1];
    U2 = [2 1; 0 gamma/2-1];
    y = L2\b;
    x_hat = U2\y;
    error = x_hat - x;
    errorb(k, :) = error';
end
for k = 1:10
    fprintf('gamma: 10^(-2*%d); error1: %d; error2: %d\n', k, errorb(k,1), errorb(k,2));
end
```

Results:

gamma: 10^(-2*1); error1: 0; error2: 0
gamma: 10^(-2*2); error1: 0; error2: 0
gamma: 10^(-2*3); error1: 0; error2: 0
gamma: 10^(-2*4); error1: 0; error2: 0
gamma: 10^(-2*5); error1: 0; error2: 0
gamma: 10^(-2*6); error1: 0; error2: 0
gamma: 10^(-2*7); error1: 0; error2: 0
gamma: 10^(-2*8); error1: 0; error2: 0
gamma: 10^(-2*9); error1: 0; error2: 0
gamma: 10^(-2*10); error1: 0; error2: 0

The accuracy did not change a lot as gamma decreases.

c):

Code:

```
errorc = zeros(10, 2);
for k = 1:10
    gamma = 10^((-2)*k);
    A = [gamma gamma-1; 2 1];
    b = [1; 1];
    x = [1; -1];
    L1 = [1 0; 2/gamma 1];
    U1 = [gamma gamma-1; 0 2/gamma-1];
    y = L1\b;
    x_hat = U1\y;
    r = b - A*x_hat;
    z = L1\r;
    e = U1\z;
    x_tilt = x_hat+e;
    error = x_tilt - x;
    errorc(k, :) = error';
end
for k = 1:10
    fprintf('gamma: 10^(-2*%d); error1: %d; error2: %d\n', k, errorc(k,1), errorc(k,2));
end
```

Results:

gamma: 10^(-2*1); error1: 0; error2: 0
gamma: 10^(-2*2); error1: 0; error2: 0
gamma: 10^(-2*3); error1: 0; error2: 0
gamma: 10^(-2*4); error1: 0; error2: 0
gamma: 10^(-2*5); error1: 0; error2: 0
gamma: 10^(-2*6); error1: 0; error2: 0
gamma: 10^(-2*7); error1: 0; error2: 0
gamma: 10^(-2*8); error1: 0; error2: 0
gamma: 10^(-2*9); error1: 0; error2: 0
gamma: 10^(-2*10); error1: 0; error2: 0

The accuracy did not change a lot as gamma decreases

Q2

a):

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

$$M_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad M_1 A = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & -1 & 1 & 0 & 2 \\ 0 & -1 & -1 & 1 & 2 \\ 0 & -1 & -1 & -1 & 2 \end{bmatrix}$$

$$M_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix} \qquad M_2 M_1 A = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 1 & 0 & 4 \\ 0 & 0 & -1 & 1 & 4 \\ 0 & 0 & -1 & -1 & 4 \end{bmatrix}$$

$$M_3 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{bmatrix} \qquad M_3 M_2 M_1 A = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 1 & 8 \\ 0 & 0 & 0 & -1 & 8 \end{bmatrix}$$

$$M_4 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \qquad M_4 M_3 M_2 M_1 A = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 1 & 8 \\ 0 & 0 & 0 & 0 & 16 \end{bmatrix}$$

$P = I$

$$U = M_4 M_3 M_2 M_1 A = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 1 & 8 \\ 0 & 0 & 0 & 0 & 16 \end{bmatrix}$$

$$\begin{aligned} L &= M_1^{-1} M_2^{-1} M_3^{-1} M_4^{-1} \\ &= (I + m_1 e_1^T)(I + m_2 e_2^T)(I + m_3 e_3^T)(I + m_4 e_4^T) \\ &= I + m_1 e_1^T + m_2 e_2^T + m_3 e_3^T + m_4 e_4^T \\ &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ -1 & -1 & 1 & 0 & 0 \\ -1 & -1 & -1 & 1 & 0 \\ -1 & -1 & -1 & -1 & 1 \end{bmatrix} \end{aligned}$$

b): $A = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ -1 & 1 & 0 & 0 & 1 \\ -1 & -1 & 1 & 0 & 1 \\ -1 & -1 & -1 & 1 & 1 \\ -1 & -1 & -1 & -1 & 1 \end{bmatrix}$

$Q_1 = I - (e_5 - e_1)(e_5 - e_1)^T$

$m_1 = (0 \ \ 1 \ \ 1 \ \ 1 \ \ 1)^T$

$M_1 = I - m_1 e_1^T$

$M_1 A Q_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & -2 \\ 0 & -1 & 1 & 0 & -2 \\ 0 & -1 & -1 & 1 & -2 \\ 0 & -1 & -1 & -1 & -2 \end{bmatrix}$

$Q_2 = I - (e_5 - e_2)(e_5 - e_2)^T$

$m_2 = (0 \ \ 0 \ \ 1 \ \ 1 \ \ 1)^T$

$M_2 = I - m_2 e_2^T$

$M_2 M_1 A Q_1 Q_2 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & -2 \\ 0 & 0 & -1 & 1 & -2 \\ 0 & 0 & -1 & -1 & -2 \end{bmatrix}$

$Q_3 = I - (e_5 - e_3)(e_5 - e_3)^T$

$m_3 = (0 \ \ 0 \ \ 0 \ \ 1 \ \ 1)^T$

$M_3 = I - m_3 e_3^T$

$M_3 M_2 M_1 A Q_1 Q_2 Q_3 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & -2 & 1 & 0 & 0 \\ 0 & 0 & -2 & 0 & 1 \\ 0 & 0 & 0 & 1 & -2 \\ 0 & 0 & 0 & -1 & -2 \end{bmatrix}$

$Q_4 = I - (e_5 - e_4)(e_5 - e_4)^T$

$m_4 = (0 \ \ 0 \ \ 0 \ \ 0 \ \ 1)^T$

$M_4 = I - m_4 e_4^T$

$M_4 M_3 M_2 M_1 A Q_1 Q_2 Q_3 Q_4 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & -2 & 1 & 0 & 0 \\ 0 & 0 & -2 & 1 & 0 \\ 0 & 0 & 0 & -2 & 1 \\ 0 & 0 & 0 & 0 & -2 \end{bmatrix}$

$P = I$

$Q = Q_1 Q_2 Q_3 Q_4 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$

$U = M_4 M_3 M_2 M_1 A Q_1 Q_2 Q_3 Q_4 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & -2 & 1 & 0 & 0 \\ 0 & 0 & -2 & 1 & 0 \\ 0 & 0 & 0 & -2 & 1 \\ 0 & 0 & 0 & 0 & -2 \end{bmatrix}$

$$\begin{aligned} L &= M_1^{-1} M_2^{-1} M_3^{-1} M_4^{-1} \\ &= (I + m_1 e_1^T)(I + m_2 e_2^T)(I + m_3 e_3^T)(I + m_4 e_4^T) \\ &= I + m_1 e_1^T + m_2 e_2^T + m_3 e_3^T + m_4 e_4^T \\ &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \end{aligned}$$

c):

Code:

```
n = 60;
A = ones(n, n);
A = A - triu(A);
A = eye(n)-A;
A = A+[ones(n-1,1);0]*[zeros(1,n-1),1];
Q = diag(ones(n-1,1),1);
Q(n,1) = 1;
[L1,U1,P1] = lu(A); % partial pivoting
[L2,U2] = lu(A*Q); % complete pivoting
x = ones(n,1);
b = A*x;
% poor approximation
y1 = L1\b;
x1 = U1\y1;
norm1 = norm(x-x1,inf);
% good approximation
y2 = L2\b;
z = U2\y2;
x2 = Q*z;
norm2 = norm(x-x2,inf);
% print result
fprintf('norm(x-x1,inf): %f; norm(x2-x,inf): %f\n',norm1,norm2);
```

Results:

norm(x-x1,inf): 1.000000; norm(x2-x,inf): 0.000000

Row partial pivoting in this case has large error.
Complete pivoting in this case has small error.

Q3

a): $A = \begin{bmatrix} 1 & 4 & 4 \\ 4 & 8 & 4 \\ 2 & 10 & 5 \end{bmatrix}$

$P_1 = I - (e_2 - e_1)(e_2 - e_1)^T = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ $\quad m_1 = \begin{bmatrix} 0 \\ 1/4 \\ 1/2 \end{bmatrix}$ $\quad M_1 = I - m_1 e_1^T = \begin{bmatrix} 1 & 0 & 0 \\ -1/4 & 1 & 0 \\ -1/2 & 0 & 1 \end{bmatrix}$ $\quad M_1 P_1 A = \begin{bmatrix} 4 & 8 & 4 \\ 0 & 2 & 3 \\ 0 & 6 & 3 \end{bmatrix}$

$P_2 = I - (e_3 - e_2)(e_3 - e_2)^T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$ $\quad m_2 = \begin{bmatrix} 0 \\ 0 \\ 1/3 \end{bmatrix}$ $\quad M_2 = I - m_1 m_2^T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1/3 & 1 \end{bmatrix}$ $\quad M_2 P_2 M_1 P_1 A = \begin{bmatrix} 4 & 8 & 4 \\ 0 & 6 & 3 \\ 0 & 0 & 2 \end{bmatrix}$

$\hat{m}_1 = P_2 m_1 = \begin{bmatrix} 0 \\ 1/2 \\ 1/4 \end{bmatrix}$ $\quad \hat{M}_1 = I - \hat{m}_1 e_1^T = \begin{bmatrix} 1 & 0 & 0 \\ -1/2 & 1 & 0 \\ -1/4 & 0 & 1 \end{bmatrix}$

$P = P_2 P_1 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$ $\quad L = \hat{M}_1^{-1} M_2^{-1} = I + \hat{m}_1 e_1^T + m_2 e_2^T = \begin{bmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ 1/4 & 1/3 & 1 \end{bmatrix}$ $\quad U = M_2 P_2 M_1 P_1 A = \begin{bmatrix} 4 & 8 & 4 \\ 0 & 6 & 3 \\ 0 & 0 & 2 \end{bmatrix}$

b): $Ax = b \iff LUx = Pb$

$b' = Pb = \begin{bmatrix} 0 \\ 3 \\ -1 \end{bmatrix}$ $\quad Ly = b' \Rightarrow y = \begin{bmatrix} 0 \\ 3 \\ -2 \end{bmatrix}$ $\quad Ux = y \Rightarrow x = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix}$

c): $A = \begin{bmatrix} 1 & 4 & 4 \\ 4 & 8 & 4 \\ 2 & 10 & 5 \end{bmatrix}$ $\quad \hat{A} = \begin{bmatrix} 1 & 4 & 4 \\ 4 & 8 & 4 \\ 2 & 5 & 5 \end{bmatrix}$

$\hat{A} = A - uv^T \iff uv^T = A - \hat{A} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 5 & 0 \end{bmatrix}$

Let $u = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$ $\quad v^T = [\,0 \;\; 5 \;\; 0\,]$

then $\hat{A} = A - uv^T$

d): $Ax = b$

$x = A^{-1}b = (A - uv^T)^{-1}b = A^{-1}b + \frac{A^{-1}uv^TA^{-1}}{1 - v^TA^{-1}u} b$

Let $x_1 = A^{-1}b$,

$\Leftrightarrow \quad Ax_1 = b$

$\Leftrightarrow \quad x_1 = [-1, 1, -1]^T$

Then $\quad x = x_1 + \frac{A^{-1}uv^Tx_1}{1 - v^TA^{-1}u}$

Let $\quad x_2 = A^{-1}u$

$\Leftrightarrow \quad Ax_2 = u \quad \Leftrightarrow \quad LUx_2 = Pu$

$\Leftrightarrow \quad x_2 = [-1/3, 1/4, -1/6]^T$

Then $\quad x = x_1 + \frac{A^{-1}uv^Tx_1}{1 - v^Tx_2}$

Let $\quad x_3 = A^{-1}uv^Tx_1$; $\quad uv^Tx_1 = [0 \ 0 \ 5]^T$

$\Leftrightarrow \quad Ax_3 = [0 \ 0 \ 5]^T$

$\Leftrightarrow \quad x_3 = [-5/3, 5/4, -5/6]$

Then $\quad x = x_1 + \frac{x_3}{1 - v^Tx_2}$

$= x_1 + \frac{x_3}{-0.25} \quad$ (Since $\quad v^Tx_2 = 1.25$)

$= \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix} - 4 \begin{bmatrix} -5/3 \\ 5/4 \\ -5/6 \end{bmatrix} = \begin{bmatrix} 17/3 \\ -4 \\ 7/3 \end{bmatrix}$

Q4

a):

Code:

```
function y = perm_a(p,x)
y = x;
num_row = size(p, 1);
for i = 1:num_row
    j = p(i);
    temp = y(i,:);
    y(i,:) = y(j,:);
    y(j,:) = temp;
end
```

b):

Code:

```
function q = perm_b(p)
np = size(p,1);
q = (1:np+1)';
q = perm_a(p, q);
```

c):

Code:

```
function y = perm_c(q,x)
y = x;
np = size(q, 1);
for i = 1:np
    j = q(i);
    y(i,:) = x(j,:);
end
```

Test:

Code:

```
p = [3;5;9;4;10;8;7;9;10];
x = [1:10]';
y1 = perm_a(p,x);
q = perm_b(p);
y2 = perm_c(q,x);
```

Result:

```
>> y1'
ans =
    3    5    9    4   10    8    7    1    2    6
>> y2'
ans =
    3    5    9    4   10    8    7    1    2    6
>> q'
ans =
    3    5    9    4   10    8    7    1    2    6
```