Linghao Shi RIN: 661954958

1. (a)
$$A=\begin{bmatrix} 0.6 & 2.2 & 4.4 \\ 1.5 & 1.1 & 7.2 \\ 3 & 1 & 2 \end{bmatrix}$$

Step 1:

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 \\ -\frac{1}{2} & 1 & 0 \\ -\frac{1}{5} & 0 & 1 \end{bmatrix}}_{L_1} \underbrace{\begin{bmatrix} 3 & 1 & 2 \\ 1.5 & 1.1 & 7.2 \\ 0.6 & 2.2 & 4.4 \end{bmatrix}}_{P_1 A} = \underbrace{\begin{bmatrix} 3 & 1 & 2 \\ 0 & 0.6 & 6.2 \\ 0 & 2 & 4 \end{bmatrix}}_{L_1 P_1 A}$$

Step 2:

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -\frac{3}{10} & 1 \end{bmatrix}}_{L_2} \begin{bmatrix} 3 & 1 & 2 \\ \frac{1}{5} & 2 & 4 \\ \frac{1}{2} & 0.6 & 6.2 \end{bmatrix} = \begin{bmatrix} 3 & 1 & 2 \\ \frac{1}{5} & 2 & 4 \\ \frac{1}{2} & \frac{3}{10} & 5 \end{bmatrix}$$

$$L=\begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{5} & 1 & 0 \\ \frac{1}{2} & \frac{3}{10} & 1 \end{bmatrix} \qquad U=\begin{bmatrix} 3 & 1 & 2 \\ 0 & 2 & 4 \\ 0 & 0 & 5 \end{bmatrix}$$

$$P=\begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \qquad A=\begin{bmatrix} 0.6 & 2.2 & 4.4 \\ 1.5 & 1.1 & 7.2 \\ 3 & 1 & 2 \end{bmatrix}$$

$PA = LU$

(b) $Ax=b$: $\begin{bmatrix} 1 & 0 & 1 \\ -2 & 1 & 1 \\ 1 & 2 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -2 \\ 17 \\ 3 \end{bmatrix}$

$p = \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix}$ $\quad P = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$

$$\begin{bmatrix} -2 & 1 & 1 \\ 1 & 2 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

$L = \begin{bmatrix} 1 & 0 & 0 \\ -\frac{1}{2} & 1 & 0 \\ -\frac{1}{2} & 0 & 1 \end{bmatrix}$ $\quad U = \begin{bmatrix} -2 & 1 & 1 \\ 0 & \frac{5}{2} & \frac{1}{2} \\ 0 & \frac{1}{2} & \frac{3}{2} \end{bmatrix}$

$L = \begin{bmatrix} 1 & 0 & 0 \\ -\frac{1}{2} & 1 & 0 \\ -\frac{1}{2} & \frac{1}{5} & 1 \end{bmatrix}$ $\quad U = \begin{bmatrix} -2 & 1 & 1 \\ 0 & \frac{5}{2} & \frac{1}{2} \\ 0 & 0 & \frac{7}{5} \end{bmatrix}$ $\quad L \cdot U = \begin{bmatrix} -2 & 1 & 1 \\ 1 & 2 & 0 \\ 1 & 0 & 1 \end{bmatrix}$

$P = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$ $\quad A = \begin{bmatrix} 1 & 0 & 1 \\ -2 & 1 & 1 \\ 1 & 2 & 0 \end{bmatrix}$ $\quad P \cdot A = \begin{bmatrix} -2 & 1 & 1 \\ 1 & 2 & 0 \\ 1 & 0 & 1 \end{bmatrix}$

verified.

$$\begin{bmatrix} 1 & 0 & 0 \\ -\frac{1}{2} & 1 & 0 \\ -\frac{1}{2} & \frac{1}{5} & 1 \end{bmatrix}\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 17 \\ 3 \\ -2 \end{bmatrix}$$

$$\begin{cases} y_1 = 17 \\ -\frac{1}{2}y_1 + y_2 = 3 \\ -\frac{1}{2}y_1 + \frac{1}{5}y_2 + y_3 = -2 \end{cases} \Rightarrow \begin{cases} y_1 = 17 \\ y_2 = 11.5 \\ y_3 = 4.2 \end{cases}$$

$$\begin{bmatrix} -2 & 1 & 1 \\ 0 & \frac{5}{2} & \frac{1}{2} \\ 0 & 0 & \frac{7}{5} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 17 \\ 11.5 \\ 4.2 \end{bmatrix}$$

$$\begin{cases} -2x_1 + x_2 + x_3 = 17 \\ \frac{5}{2}x_2 + \frac{1}{2}x_3 = 11.5 \\ \frac{7}{5}x_3 = 4.2 \end{cases} \Rightarrow \begin{cases} x_1 = -5 \\ x_2 = 4 \\ x_3 = 3 \end{cases}$$

check Ax:

$$\begin{cases} -5+3=-2 \\ 10+4+7=17 \\ -5+8=3 \end{cases}$$

$Ax=b$

(c) $P = \begin{bmatrix} 2 \\ 4 \\ 3 \\ 1 \end{bmatrix}$ $PA = \begin{bmatrix} -r_2- \\ -r_4- \\ -r_3- \\ -r_1- \end{bmatrix}$

$$AP = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} = \begin{array}{c c} \begin{array}{cccc} C_4 & C_1 & C_3 & C_2 \end{array} & \\ \begin{bmatrix} a_{14} & a_{11} & a_{13} & a_{12} \\ a_{24} & a_{21} & a_{23} & a_{22} \\ a_{34} & a_{31} & a_{33} & a_{32} \\ a_{44} & a_{41} & a_{43} & a_{42} \end{bmatrix} & \begin{array}{c} R_1 \\ R_2 \\ R_3 \\ R_4 \end{array} \end{array}$$

② (a) (i) $B = \begin{bmatrix} \frac{1}{4} & \frac{2}{3} \\ \frac{1}{2} & -\frac{1}{5} \end{bmatrix}$

$$\left.\begin{aligned} \|B\|_\infty &= \max\left\{\tfrac{11}{12}, \tfrac{7}{10}\right\} = \tfrac{11}{12} \\ \|B\|_1 &= \max\left\{\tfrac{3}{4}, \tfrac{13}{15}\right\} = \tfrac{13}{15} \end{aligned}\right\} \text{both } < 1, \text{ converge}$$

(ii) $B = \begin{bmatrix} \frac{6}{5} & \frac{8}{15} \\ -\frac{2}{5} & \frac{4}{15} \end{bmatrix}$

$$\left.\begin{aligned} \|B\|_\infty &= \max\left\{\tfrac{26}{15}, \tfrac{10}{15}\right\} = \tfrac{26}{15} \\ \|B\|_1 &= \max\left\{\tfrac{8}{5}, \tfrac{12}{15}\right\} = \tfrac{8}{5} \end{aligned}\right\} \text{both } > 1$$

$Bx = \lambda x$

$\det(B - \lambda x) = 0$

$(\frac{6}{5} - \lambda)(\frac{4}{15} - \lambda) + \frac{8}{15} \cdot \frac{2}{5} = 0$

$\frac{24}{75} + \lambda^2 - \frac{22}{15}\lambda + \frac{16}{75} = 0$

$15\lambda^2 - 22\lambda + 8 = 0$

$\lambda = \frac{22 \pm \sqrt{484 - 480}}{30} = \frac{22 \pm 2}{30}$

$\lambda_1 = \frac{4}{5}, \ \lambda_2 = \frac{2}{3}$

$\rho(B) = \max|\lambda| = \frac{4}{5} < 1$ converge

(b)

(i) $A = \begin{bmatrix} -5 & -2 & 2 \\ 7 & 10 & -1 \\ -3 & 2 & -7 \end{bmatrix}$

$\begin{cases} 5 > 2+2 \\ 10 > 7+1 \\ 7 > 3+2 \end{cases}$ $\Rightarrow$ this matrix is SDD

(ii) $A = \begin{bmatrix} 8 & 3 & -2 \\ 2 & 4 & -3 \\ 1 & 4 & 6 \end{bmatrix}$

$\begin{cases} 8 > 3+3 \\ 4 < 2+3 \quad \otimes \\ 6 > 1+4 \end{cases}$ Not SDD. Can't be made SDD

(iii) $A = \begin{bmatrix} 4 & -2 & 8 \\ -3 & 6 & 2 \\ 3 & -1 & 1 \end{bmatrix}$

$\begin{cases} 4 < 2+8 \quad \otimes \\ 6 > 3+2 \\ 1 < 3+1 \quad \otimes \end{cases}$ Not SDD, consider the matrix $\begin{bmatrix} 3 & -1 & 1 \\ -3 & 6 & 2 \\ 4 & -2 & 8 \end{bmatrix}$

can be made SDD

(c) $A = \begin{bmatrix} a & -2 & 2 \\ -1 & 7 & a \\ 2 & -3 & 2a \end{bmatrix}$ $\begin{cases} |a| > 4 \\ |a| < 6 \\ |a| > 2.5 \end{cases} \Rightarrow \begin{cases} -6 < a < 4 \\ 4 < a < 6 \end{cases}$

4.

This is the code to generate the matrices.

```
n = 10;
A = sparse(n,n);
xe = zeros(10, 1);
for i = 1:n
    A(i,i) = 2;
    if i >= 2
        A(i,i-1) = -1;
    end
    if i <= 9
        A(i,i+1) = -1;
    end
    xe(i) = 11 - i;
end
```

$Set\ b = A * xe, x0 = zeros(n, 1), tol\ to\ 10^{-4}, maxIterations\ to\ 100\ for\ below\ input$

(a) Jacobi method

```
function [x] = Jacobi(A,b,x0,tol,maxIterations,xe)
    k = 1;
    r = A - diag(diag(A));
    DiagInv = inv(diag(diag(A)));
    x = x0;
    norm_b = norm(b, "inf");
    delta = norm(b - A * x, "inf")/norm_b;
    while k < maxIterations
        x = DiagInv * (b- r * x);
        prevdelta = delta;
        delta = norm(b - A * x, "inf")/norm_b;
        if( mod(k,10) == 1 )
            fprintf('%s: k=%5d, delta=%8.2e, CR=%8.2e\n',"Jacobi", k,...
                delta, delta/prevdelta);
        end
        k = k + 1;
        if tol > delta
            break
        end
    end
    rfe = norm(xe - x,"inf")/norm(xe,"inf");
    fprintf('%s: RFE = %9.3e, numIterations=%d\n\n',"Jacobi", rfe, k);
end
```

```
>> x = Jacobi(A,b,x0,tol,maxIterations,xe)
Jacobi: k=    1, delta=5.00e-01, CR=5.00e-01
Jacobi: k=   11, delta=8.06e-02, CR=9.17e-01
Jacobi: k=   21, delta=4.30e-02, CR=9.35e-01
Jacobi: k=   31, delta=2.79e-02, CR=9.41e-01
Jacobi: k=   41, delta=1.86e-02, CR=9.54e-01
Jacobi: k=   51, delta=1.23e-02, CR=9.58e-01
Jacobi: k=   61, delta=8.14e-03, CR=9.59e-01
Jacobi: k=   71, delta=5.38e-03, CR=9.59e-01
Jacobi: k=   81, delta=3.56e-03, CR=9.59e-01
Jacobi: k=   91, delta=2.35e-03, CR=9.59e-01
Jacobi: RFE = 1.124e-02, numIterations=100


x =

    9.9680
    8.9411
    7.9141
    6.9009
    5.8876
    4.8921
    3.8967
    2.9176
    1.9386
    0.9693
```

(b) GaussSeidel method

```
function [x] = GaussSeidel(A,b,x0,tol,maxIterations,xe)
    LDinverse = inv(tril(A,-1) + diag(diag(A)));
    B = -LDinverse * triu(A,1);
    x = x0;
    norm_b = norm(b, "inf");
    delta = norm(b - A * x, "inf") / norm_b;
    k = 1;
    while k < maxIterations
        x = B * x + LDinverse * b;
        prevdelta = delta;
        delta = norm(b - A * x, "inf") / norm_b;
        if( mod(k,10) == 1 )
            fprintf("%s: k=%5d, delta=%8.2e, CR=%8.2e\n","Gauss-Seidel",...
                k, delta, delta/prevdelta);
        end
        if tol > delta
            break
        end
        k = k + 1;
    end
 rfe = norm(xe - x, "inf") / norm(xe, "inf");
 fprintf("%s: RFE = %9.3e, numIterations=%d\n\n","Gauss-Seidel",...
     rfe, k);
 end
```

```
>> x = GaussSeidel(A,b,x0,tol,maxIterations,xe)
Gauss-Seidel: k=    1, delta=2.50e-01, CR=2.50e-01
Gauss-Seidel: k=   11, delta=1.94e-02, CR=9.01e-01
Gauss-Seidel: k=   21, delta=8.23e-03, CR=9.20e-01
Gauss-Seidel: k=   31, delta=3.60e-03, CR=9.21e-01
Gauss-Seidel: k=   41, delta=1.57e-03, CR=9.21e-01
Gauss-Seidel: k=   51, delta=6.88e-04, CR=9.21e-01
Gauss-Seidel: k=   61, delta=3.01e-04, CR=9.21e-01
Gauss-Seidel: k=   71, delta=1.32e-04, CR=9.21e-01
Gauss-Seidel: RFE = 1.206e-03, numIterations=75


x =

    9.9960
    8.9925
    7.9900
    6.9885
    5.9879
    4.9884
    3.9898
    2.9919
    1.9944
    0.9972
```

(c) SOR method

```
function [x] = SOR(A,b,x0,tol,maxIterations,xe)
    x = x0;
    inverse = inv(diag(diag(A)) + 1.5 * tril(A,-1));
    B = inverse * ((1 - 1.5) * diag(diag(A)) - 1.5 * triu(A,1));
    wLDi = 1.5 * inverse;
    norm_b = norm(b, "inf");
    delta = norm(b - A * x, "inf")/norm_b;
    k = 1;
    while k < maxIterations
        x = B * x + wLDi * b;
        prevdelta = delta;
        delta = norm(b - A * x, "inf")/norm_b;
        if mod(k,10) == 1
            fprintf('%s: k=%5d, delta=%8.2e, CR=%8.2e\n',...
                "SOR", k, delta, delta/prevdelta);
        end
        if tol > delta
            break
        end
        k = k + 1;
    end
    rfe = norm(xe-x, "inf") / norm(xe, "inf");
    fprintf('%s: RFE = %9.3e, numIterations=%d\n\n', "SOR", rfe, k);
end
```

```
>> x = SOR(A,b,x0,tol,maxIterations,xe)
SOR: k=    1, delta=6.25e-02, CR=6.25e-02
SOR: k=   11, delta=2.88e-03, CR=7.99e-01
SOR: k=   21, delta=1.17e-04, CR=7.22e-01
SOR: RFE = 5.195e-04, numIterations=22


x =

    9.9974
    8.9958
    7.9949
    6.9948
    5.9952
    4.9959
    3.9968
    2.9977
    1.9986
    0.9994
```

(d)

Use SOR solver function to solve for different omega values using loop, we obtain the following results:

```
SOR: RFE = 1.261e-03, omega= 0.900, numIterations= 92
SOR: RFE = 1.206e-03, omega= 1.000, numIterations= 75
SOR: RFE = 1.138e-03, omega= 1.100, numIterations= 61
SOR: RFE = 1.083e-03, omega= 1.200, numIterations= 49
SOR: RFE = 9.525e-04, omega= 1.300, numIterations= 39
SOR: RFE = 8.192e-04, omega= 1.400, numIterations= 30
SOR: RFE = 5.195e-04, omega= 1.500, numIterations= 22
SOR: RFE = 2.220e-04, omega= 1.600, numIterations= 16
SOR: RFE = 9.023e-05, omega= 1.700, numIterations= 23
SOR: RFE = 8.734e-05, omega= 1.800, numIterations= 42
SOR: RFE = 8.469e-05, omega= 1.900, numIterations= 88
SOR: RFE = 1.000e+00, omega= 2.000, numIterations=100
```

As we can see, when omega = 1.6, it has the fastest convergence.