University of Waterloo

ECE204 Lab Report

Simulation Assignment #2

Section: 202

Kehan Xing (k5xing)

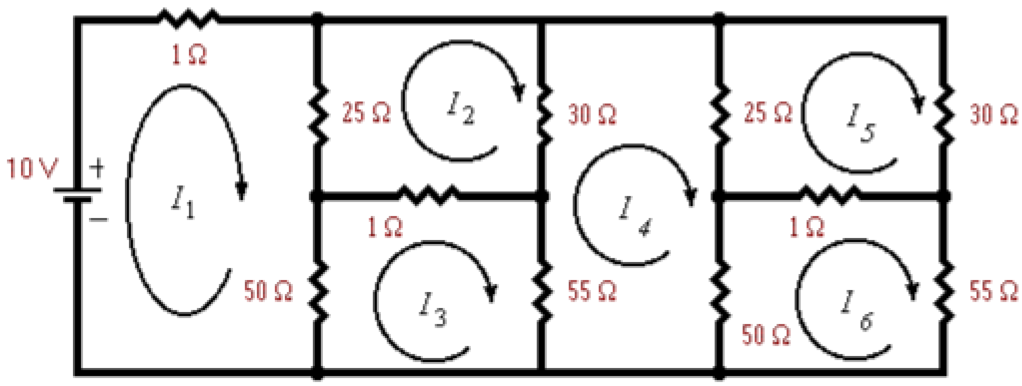
Jing Liu (j762liu)

Submission Date: 2019-10-03

Question:

Using MATLAB, calculate the value of the loop currents of the following circuit using the following algorithms:

a) Simple matrix inversion
b) Gaussian elimination
c) LU Factorization
d) Gauss-Seidel iteration



Matrix A (A.txt):

	1	2	3	4	5	6
1	76	-25	-50	0	0	0
2	-25	56	-1	-30	0	0
3	-50	-1	106	-55	0	0
4	0	-30	-55	160	-25	-50
5	0	0	0	-25	56	-1
6	0	0	0	-50	-1	106

Vector B (B.txt):

	1
1	10
2	0
3	0
4	0
5	0
6	0

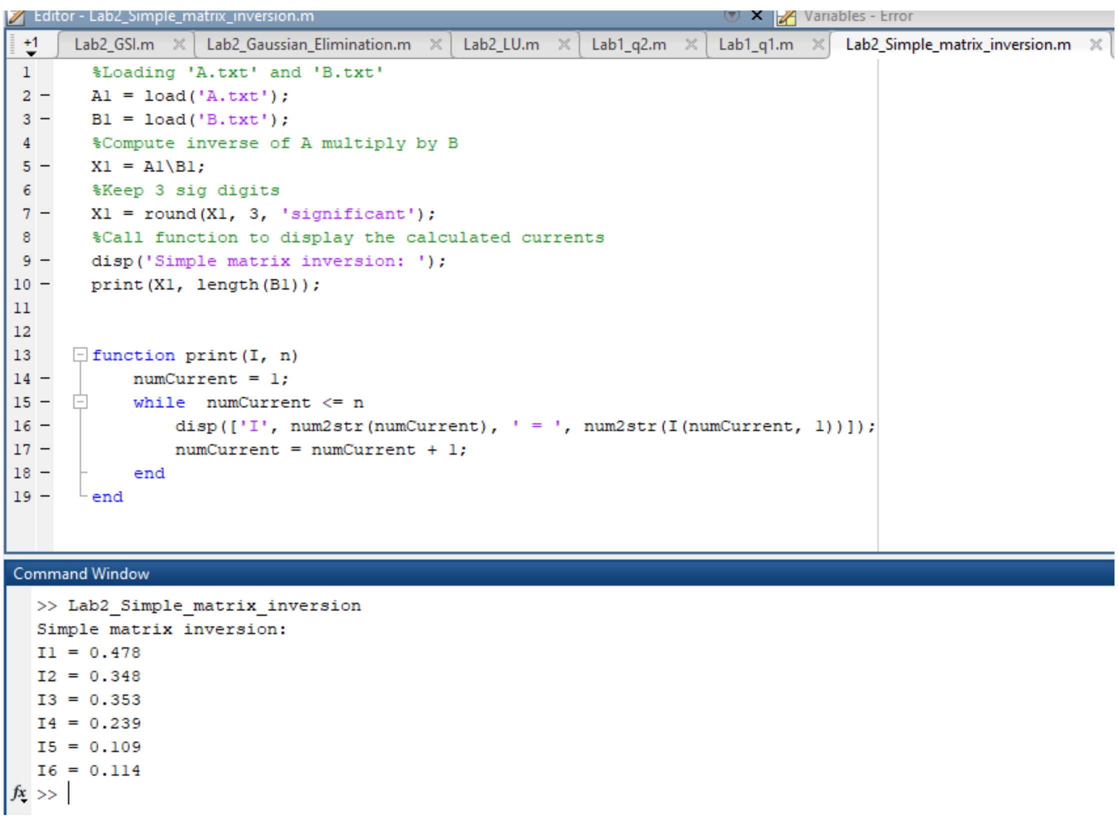
Algorithm 1.

Simple matrix inversion

Code:

```
%Loading 'A.txt' and 'B.txt'
A1 = load('A.txt');
B1 = load('B.txt');
%Compute inverse of A multiply by B
X1 = A1\B1;
%Keep 3 sig digits
X1 = round(X1, 3, 'significant');
%Call function to display the calculated currents
disp('Simple matrix inversion: ');
print(X1, length(B1));

function print(I, n)
    numCurrent = 1;
    while  numCurrent <= n
        disp(['I', num2str(numCurrent), ' = ', num2str(I(numCurrent, 1))]);
        numCurrent = numCurrent + 1;
    end
end
```

Display:



Algorithm2.

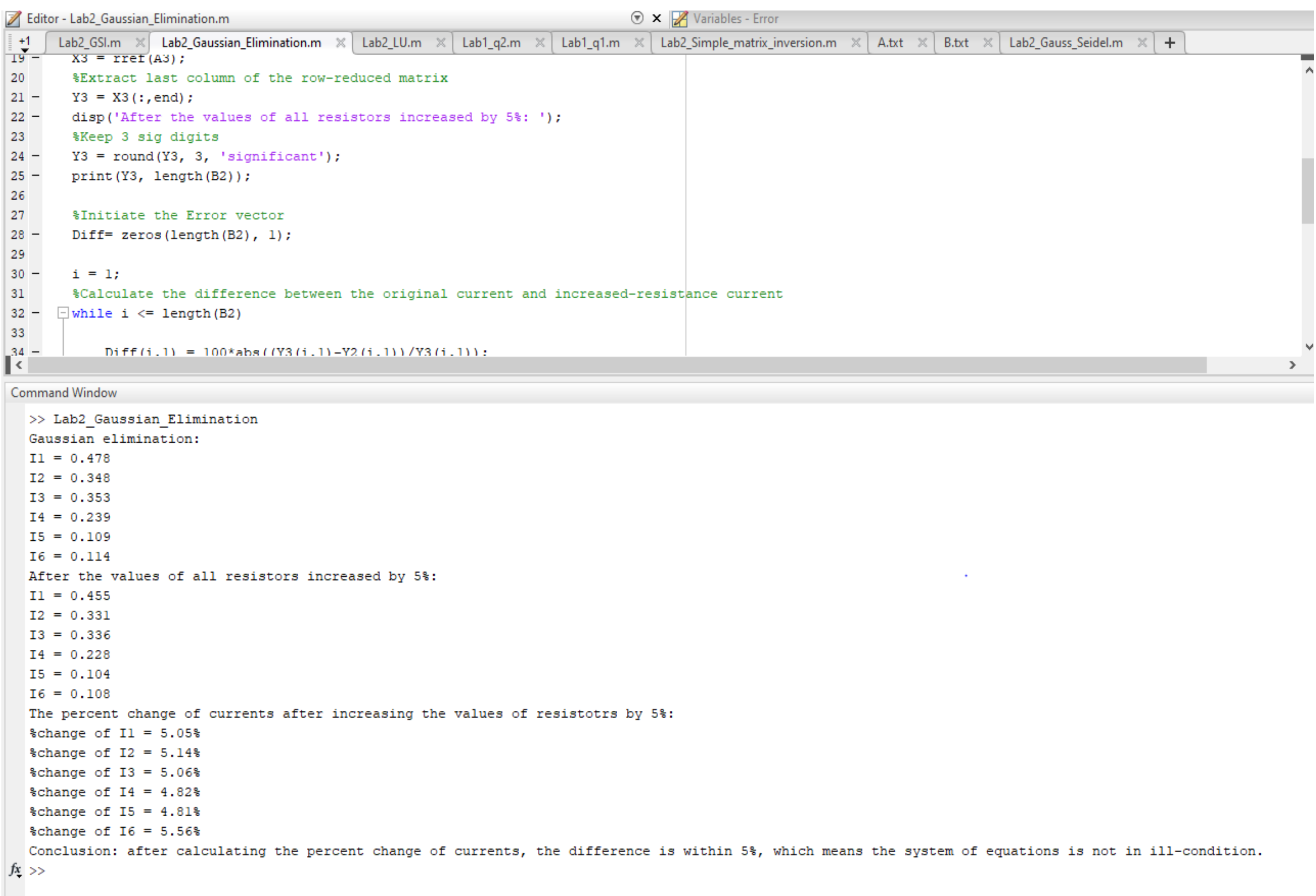
Gaussian Elimination

Code:

```
%Loading 'A.txt' and 'B.txt'
A2 = load('A.txt');
B2 = load('B.txt');

%Append vector B2 to the last column of A2
A2 = [A2 B2];
%Perform row reduction on A2
X2 = rref(A2);
%Extract last column of the row-reduced matrix
Y2 = X2(:,end);
%Keep 3 sig digits
Y2 = round(Y2, 3, 'significant');
disp('Gaussian elimination: ');
print(Y2, length(B2));

%Append vector B2 to the last column of A3 with increased resistance
A3 = [1.05 * load('A.txt') B2];
%Perform row reduction on A3
X3 = rref(A3);
%Extract last column of the row-reduced matrix
Y3 = X3(:,end);
disp('After the values of all resistors increased by 5%: ');
%Keep 3 sig digits
Y3 = round(Y3, 3, 'significant');
print(Y3, length(B2));

%Initiate the Error vector
Diff= zeros(length(B2), 1);

i = 1;
%Calculate the difference between the original current and increased-
resistance current
while i <= length(B2)

    Diff(i,1) = 100*abs((Y3(i,1)-Y2(i,1))/Y3(i,1));
    i = i + 1;

end

disp('The percent change of currents after increasing the values of
resistotrs by 5%: ')

%Keep 3 sig digits
Diff = round(Diff, 3, 'significant');

%display %change of current
numCurrent = 1;
```

```
    while  numCurrent <= n
        disp(['%change of I', num2str(numCurrent), ' = ',
num2str(Diff(numCurrent, 1)), '%']);
        numCurrent = numCurrent + 1;
    end

disp('Conclusion: after calculating the percent change of currents, the
difference is within 5%, which means the system of equations is not in ill-
condition.')

%define print function
function print(I, n)
    numCurrent = 1;
    while  numCurrent <= n
        disp(['I', num2str(numCurrent), ' = ', num2str(I(numCurrent, 1))]);
        numCurrent = numCurrent + 1;
    end
end
```

Display:



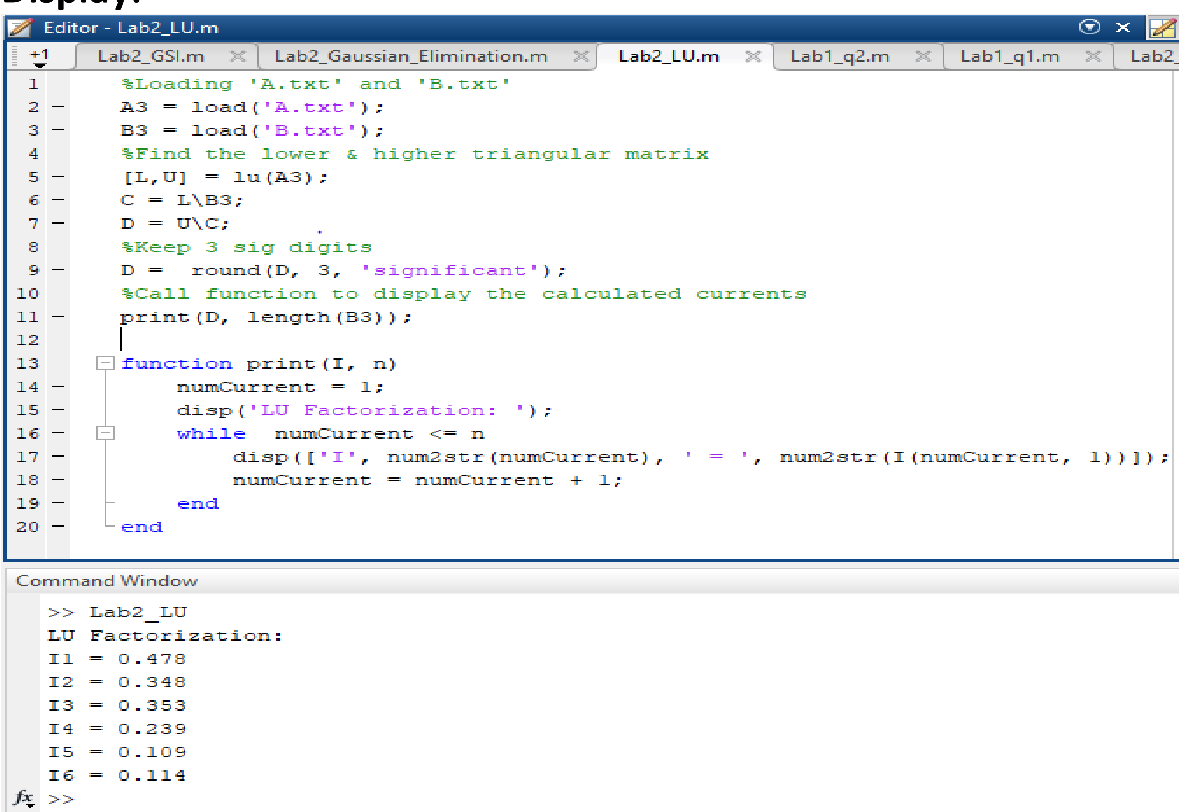
Algorithm3.

LU Factorization

Code:

```
%Loading 'A.txt' and 'B.txt'
A3 = load('A.txt');
B3 = load('B.txt');
%Find the lower & higher triangular matrix
[L,U] = lu(A3);
C = L\B3;
D = U\C;
%Keep 3 sig digits
D =  round(D, 3, 'significant');
%Call function to display the calculated currents
print(D, length(B3));

function print(I, n)
    numCurrent = 1;
    disp('LU Factorization: ');
    while  numCurrent <= n
        disp(['I', num2str(numCurrent), ' = ', num2str(I(numCurrent, 1))]);
        numCurrent = numCurrent + 1;
    end
end
```

Display:



Editor - Lab2_LU.m

Lab2_GSI.m | Lab2_Gaussian_Elimination.m | Lab2_LU.m | Lab1_q2.m | Lab1_q1.m | Lab2_

```
%Loading 'A.txt' and 'B.txt'
A3 = load('A.txt');
B3 = load('B.txt');
%Find the lower & higher triangular matrix
[L,U] = lu(A3);
C = L\B3;
D = U\C;
%Keep 3 sig digits
D =  round(D, 3, 'significant');
%Call function to display the calculated currents
print(D, length(B3));

function print(I, n)
    numCurrent = 1;
    disp('LU Factorization: ');
    while  numCurrent <= n
        disp(['I', num2str(numCurrent), ' = ', num2str(I(numCurrent, 1))]);
        numCurrent = numCurrent + 1;
    end
end
```

Command Window

```
>> Lab2_LU
LU Factorization:
I1 = 0.478
I2 = 0.348
I3 = 0.353
I4 = 0.239
I5 = 0.109
I6 = 0.114
>>
```

Algorithm4.

Gauss-Seidel Iteration

Code:

```
%Loading 'A.txt' and 'B.txt'
A4 = load('A.txt');
B4 = load('B.txt');

n = length(B4);

%initial guess values
X_curr = ones(n,1);
X_prev = ones(n,1);
Error = ones(n,1);

%set output conditions
flag = 0;
i = 0;

while max(Error) >= 0.0001

    row = 1;
    %solve from X1 to Xn
    while row <= n
        col = 1;
        X_curr(row, 1) = B4(row, 1);

        %solve specific Xi
        while col <= n
          %skip diagonal element
          if col ~= row
             X_curr(row, 1)=  X_curr(row, 1)- A4(row, col) * X_curr(col, 1);
          end

          col=col+1;
        end
        X_curr(row,1) = X_curr(row, 1)/A4(row, row);

        %update absolute approximate relative error for each Xi
        Error(row,1) = abs((X_curr(row,1)- X_prev(row,1))/X_curr(row,1));
        %update previous Xi values, in order to calculate errors
        X_prev(row, 1) = X_curr(row, 1);

        row = row + 1;
    end
    i = i + 1;
```

```
    %change flag to avoid displaying the output of the same condition
multiple times
    if max(Error)<0.01 && flag == 0
        display(['Required iterations for error less than 1%: ',
num2str(i)]);
        I = round(X_curr, 3, 'significant');
        %Call function to display the calculated currents
        print(I,n);
        flag = 1;
    elseif max(Error)<0.005 && flag == 1
        display(['Required iterations for error less than 0.5%: ',
num2str(i)]);
        I = round(X_curr, 3, 'significant');
        %Call function to display the calculated currents
        print(I,n);
        flag = 2;
    elseif max(Error)<0.001 && flag == 2
        display(['Required iterations for error less than 0.1%: ',
num2str(i)]);
        I = round(X_curr, 3, 'significant');
        %Call function to display the calculated currents
        print(I,n);
        flag = 3;
    elseif max(Error) < 0.0001 && flag == 3
        display(['Required iterations for error less than 0.01%: ',
num2str(i)]);
        I = round(X_curr, 3, 'significant');
        %Call function to display the calculated currents
        print(I,n);
        flag = 4;
    end

end

function print(I, n)
    numCurrent = 1;
        while  numCurrent <= n
            disp(['I', num2str(numCurrent), ' = ', num2str(I(numCurrent,
1))]);
            numCurrent = numCurrent + 1;
        end
end
```

Display:

Editor - Lab2_Gauss_Seidel.m

```matlab

            row = row + 1;
        end
        i = i + 1;
        %change flag to avoid displaying the output of the same condition multiple times
        if max(Error)<0.01 && flag == 0
            display(['Required iterations for error less than 1%: ', num2str(i)]);
            I = round(X_curr, 3, 'significant');
            %Call function to display the calculated currents
            print(I,n);
            flag = 1;
        elseif max(Error)<0.005 && flag == 1
```

Command Window

```
>> Lab2_Gauss_Seidel
Required iterations for error less than 1%: 24
I1 = 0.5
I2 = 0.367
I3 = 0.372
I4 = 0.253
I5 = 0.115
I6 = 0.12
Required iterations for error less than 0.5%: 29
I1 = 0.488
I2 = 0.356
I3 = 0.361
I4 = 0.245
I5 = 0.111
I6 = 0.117
Required iterations for error less than 0.1%: 38
I1 = 0.48
I2 = 0.35
I3 = 0.355
I4 = 0.24
I5 = 0.109
I6 = 0.114
Required iterations for error less than 0.01%: 52
I1 = 0.478
I2 = 0.348
I3 = 0.353
I4 = 0.239
I5 = 0.109
I6 = 0.114
>>
```