MATH 446/OR 481
Project 4

October 13, 2022

Project Conclusion

A MATLAB function is created which solves a matrix equation $Ax = b$ using simple Gaussian Elimination without row exchanges. The function is used to study the error characteristics for matrices $A1, A2, A3$ for various sizes - n.

Comparing the Relative Forward Error (RFE) for the matrices with two values $n = 8, 16$, we find that the errors are least for matrix $A2$ with a size of 16×16. Matrix $A3$ gives poorest results with a completely inaccurate solution when $n = 16$.

Matrix $A2$ is the easiest to solve for an accurate solution, followed by matrix $A1$, and finally matrix $A3$ is the hardest to solve for an accurate solution.

Matrix $A2$ gives an accurate solution (at least 4 correct digits) up to a threshold of $n = 355$. Matrix $A1$ can produce a similar accurate solution up to $n = 22$, and matrix $A3$ can only produce a solution with at least 4 correct digits up to $n = 9$.

The MATLAB code and results follow. Sections of the Command Window output are shown below.

Summary Table of Errors - n = 8

matrix	RFE	RBE	EMF	condition_number
"A1"	4.5430e-13	1.6193e-16	2.8055e+03	2.4129e+04
"A2"	2.1316e-13	1.5491e-14	1.3761e+01	4.3999e+01
"A3"	9.6302e-07	1.3596e-16	7.0833e+09	1.0694e+11

Summary Table of Errors - n = 16

matrix	RFE	RBE	EMF	condition_number
"A1"	7.9969e-09	8.5648e-16	9.3369e+06	2.1694e+08
"A2"	1.3456e-13	2.1875e-14	6.1513e+00	7.0610e+01
"A3"	3.9221e+01	4.7606e-16	8.2388e+16	2.7783e+18

```
LARGEST SIZE FOR ACCURATE SOLUTIONS (AT LEAST 4 CORRECT DIGITS)
Largest size for matrix A1 = 22
Largest size for matrix A2 = 355
Largest size for matrix A3 = 9
```

MATLAB Code - GaussElim.m

```
function x = GaussElim(A, b)
    % this function solves the matrix equation Ax = b
    % using simple Gaussian Elimination

    % get number of rows and cols of A
    n = size(A, 1);

    % elimination
    % i -> row, j -> col
    for j = 1 : n-1
        % check for zero pivot
        if abs(A(j, j)) < eps
            error('zero pivot encountered');
        end
        for i = j+1 : n
            mult = A(i, j)/A(j, j);
            for k = j+1 : n
                A(i, k) = A(i, k) - mult*A(j, k);
            end
        b(i) = b(i) - mult*b(j);
        end
    end

    % back-substitution
    for i = n : -1 : 1
        for j = i+1 : n
            b(i) = b(i) - A(i, j)*x(j, 1);
        end
        x(i, 1) = b(i)/A(i, i);
    end
end
```

MATLAB Code - Main Script

```
clc; clear; close all

% n = 8
n = 8;
fprintf('SOLVING FOR n = 8\n')

A1 = zeros(n);
A2 = A1; A3 = A1;

% define matrices
for i = 1:n
    for j = 1:n
        A1(i, j) = exp(2 + sin(i + j));
        A2(i, j) = sin(2 + exp(i + j));
        A3(i, j) = exp(2/(i+j));
    end
end

c = ones(n, 1);

b1 = A1*c;
b2 = A2*c;
b3 = A3*c;

% compute c (Ac = b) using Gaussian Elimination
format longE
c1 = GaussElim(A1, b1)
c2 = GaussElim(A2, b2)
c3 = GaussElim(A3, b3)

% compute RFE, RBE, EMF and condition number
RFE1 = norm(c - c1, 'Inf')/norm(c, 'Inf');
RBE1 = norm(b1 - A1*c1, 'Inf')/norm(b1, 'Inf');
EMF1 = RFE1/RBE1;
CN1 = cond(A1, 'Inf');

RFE2 = norm(c - c2, 'Inf')/norm(c, 'Inf');
RBE2 = norm(b2 - A2*c2, 'Inf')/norm(b2, 'Inf');
EMF2 = RFE2/RBE2;
```

```
CN2 = cond(A2, 'Inf');

RFE3 = norm(c - c3, 'Inf')/norm(c, 'Inf');
RBE3 = norm(b3 - A3*c3, 'Inf')/norm(b3, 'Inf');
EMF3 = RFE3/RBE3;
CN3 = cond(A3, 'Inf');

% print table of results
format shortE
fprintf('Summary Table of Errors - n = 8\n\n');
tbl = table;
tbl.matrix = ["A1", "A2", "A3"]';
tbl.RFE = [RFE1, RFE2, RFE3]';
tbl.RBE = [RBE1, RBE2, RBE3]';
tbl.EMF = [EMF1, EMF2, EMF3]';
tbl.condition_number = [CN1, CN2, CN3]';
disp(tbl)

% n =16
n = 16;
fprintf('SOLVING FOR n = 16\n')

A1 = zeros(n);
A2 = A1; A3 = A1;

% define matrices
for i = 1:n
    for j = 1:n
        A1(i, j) = exp(2 + sin(i + j));
        A2(i, j) = sin(2 + exp(i + j));
        A3(i, j) = exp(2/(i+j));
    end
end

c = ones(n, 1);

b1 = A1*c;
b2 = A2*c;
b3 = A3*c;

% compute c (Ac = b) using Gaussian Elimination
```

```
format longE
c1 = GaussElim(A1, b1)
c2 = GaussElim(A2, b2)
c3 = GaussElim(A3, b3)

% compute RFE, RBE, EMF and condition number
RFE1 = norm(c - c1, 'Inf')/norm(c, 'Inf');
RBE1 = norm(b1 - A1*c1, 'Inf')/norm(b1, 'Inf');
EMF1 = RFE1/RBE1;
CN1 = cond(A1, 'Inf');

RFE2 = norm(c - c2, 'Inf')/norm(c, 'Inf');
RBE2 = norm(b2 - A2*c2, 'Inf')/norm(b2, 'Inf');
EMF2 = RFE2/RBE2;
CN2 = cond(A2, 'Inf');

RFE3 = norm(c - c3, 'Inf')/norm(c, 'Inf');
RBE3 = norm(b3 - A3*c3, 'Inf')/norm(b3, 'Inf');
EMF3 = RFE3/RBE3;
CN3 = cond(A3, 'Inf');

% print table of results
format shortE
fprintf('Summary Table of Errors - n = 16\n\n');
tbl = table;
tbl.matrix = ["A1", "A2", "A3"]';
tbl.RFE = [RFE1, RFE2, RFE3]';
tbl.RBE = [RBE1, RBE2, RBE3]';
tbl.EMF = [EMF1, EMF2, EMF3]';
tbl.condition_number = [CN1, CN2, CN3]';
disp(tbl)

% find out what value of n leads to RFE > 0.5E-04 for each matrix
clear;
fprintf('LARGEST SIZE FOR ACCURATE SOLUTIONS (AT LEAST 4 CORRECT DIGITS)\n')
% A1
n = 17;
tol = 0.00005;

while (true)
    for i = 1:n
```

```
        for j = 1:n
            A1(i, j) = exp(2 + sin(i + j));
        end
    end

    c = ones(n, 1);
    b1 = A1*c;
    c1 = GaussElim(A1, b1);

    RFE1 = norm(c - c1, 'Inf')/norm(c, 'Inf');
    if (RFE1 < tol)
        n = n + 1;
    else
        break
    end
end

fprintf(['Largest size for matrix A1 = %d\n'], n)

% A2
n = 32;
tol = 0.00005;

while (true)
    for i = 1:n
        for j = 1:n
            A2(i, j) = sin(2 + exp(i + j));
        end
    end

    c = ones(n, 1);
    b2 = A2*c;
    c2 = GaussElim(A2, b2);

    RFE2 = norm(c - c2, 'Inf')/norm(c, 'Inf');
    if (RFE2 < tol)
        n = n + 1;
    else
        break
    end
end
```

```
fprintf(['Largest size for matrix A2 = %d\n'], n)

% A3
n = 2;
tol = 0.00005;

while (true)
    for i = 1:n
        for j = 1:n
            A3(i, j) = exp(2/(i+j));
        end
    end

    c = ones(n, 1);
    b3 = A3*c;
    c3 = GaussElim(A3, b3);

    RFE3 = norm(c - c3, 'Inf')/norm(c, 'Inf');
    if (RFE3 < tol)
        n = n + 1;
    else
        break
    end
end

fprintf(['Largest size for matrix A3 = %d\n'], n)
```

Command Window Output

```
SOLVING FOR n = 8

c1 =

    1.000000000000347e+00
    1.000000000000364e+00
    9.999999999999305e-01
    1.000000000000036e+00
    9.999999999999735e-01
    1.000000000000035e+00
    9.999999999997691e-01
    9.999999999995457e-01


c2 =

    1.000000000000031e+00
    1.000000000000213e+00
    9.999999999998493e-01
    1.000000000000108e+00
    9.999999999999413e-01
    1.000000000000175e+00
    9.999999999998289e-01
    9.999999999998845e-01


c3 =

    9.999999999159541e-01
    1.000000003886779e+00
    9.999999513424251e-01
    1.000000258461814e+00
    9.999993124049639e-01
    1.000000963015131e+00
    9.999993216945963e-01
    1.000000189280456e+00

Summary Table of Errors - n = 8
```

```
    matrix       RFE          RBE          EMF        condition_number
    ------    ----------   ----------   ----------   ----------------

    "A1"     4.5430e-13   1.6193e-16   2.8055e+03      2.4129e+04
    "A2"     2.1316e-13   1.5491e-14   1.3761e+01      4.3999e+01
    "A3"     9.6302e-07   1.3596e-16   7.0833e+09      1.0694e+11

SOLVING FOR n = 16

c1 =

    1.000000003154927e+00
    9.999999930706268e-01
    1.000000004349520e+00
    9.999999993472164e-01
    9.999999998231216e-01
    9.999999997455417e-01
    9.999999976311875e-01
    1.000000007996922e+00
    9.999999929443933e-01
    1.000000001763904e+00
    1.000000000208171e+00
    1.000000000199409e+00
    1.000000000882352e+00
    9.999999947446699e-01
    1.000000006556349e+00
    9.999999975816950e-01


c2 =

    1.000000000000135e+00
    9.999999999998765e-01
    1.000000000000055e+00
    1.000000000000020e+00
    9.999999999999708e-01
    9.999999999999591e-01
    1.000000000000088e+00
    9.999999999999971e-01
    1.000000000000031e+00
    9.999999999998996e-01
```

```
    1.000000000000049e+00
    9.999999999999681e-01
    9.999999999999981e-01
    1.000000000000010e+00
    1.000000000000018e+00
    1.000000000000051e+00


c3 =

    1.000000557072159e+00
    9.999610211525439e-01
    1.000741286870706e+00
    9.942048511783724e-01
    1.025342479258314e+00
    8.575166652054999e-01
    2.114343739878505e+00
    -4.912990263255757e+00
    1.976424400721395e+01
    -3.489525347794957e+01
    4.022136716805760e+01
    -1.575655317161230e+01
    -1.155013609327875e+01
    2.215140538128752e+01
    -1.022992075422704e+01
    3.215726607779362e+00

Summary Table of Errors - n = 16

    matrix       RFE           RBE           EMF        condition_number
    ------    ----------    ----------    ----------    ----------------

    "A1"     7.9969e-09    8.5648e-16    9.3369e+06       2.1694e+08
    "A2"     1.3456e-13    2.1875e-14    6.1513e+00       7.0610e+01
    "A3"     3.9221e+01    4.7606e-16    8.2388e+16       2.7783e+18

LARGEST SIZE FOR ACCURATE SOLUTIONS (AT LEAST 4 CORRECT DIGITS)
Largest size for matrix A1 = 22
Largest size for matrix A2 = 355
Largest size for matrix A3 = 9
```