MATH 446
Project 3

March 12, 2023

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 two matrices for various sizes - $n = 6, 10, 14$.

The matrices are defined as

$$A(i,j) = \sqrt{(1 + \sin(i+j)/2)} \tag{1}$$

and

$$A(i,j) = \sqrt{(1 + 2(i+j))} \tag{2}$$

The summary of errors is presented in the table below.

Command Window

```
Summary Table
        n             RFE           RBE           EMF        condition_number
    __________    __________    __________    __________    ________________

    6.0000e+00    1.2879e-14    2.2204e-16    5.8000e+01       6.5317e+02
    1.0000e+01    3.1599e-12    2.2204e-16    1.4231e+04       9.9821e+04
    1.4000e+01    1.5368e-10    3.7797e-16    4.0660e+05       6.8050e+06

Summary Table
        n             RFE           RBE           EMF        condition_number
    __________    __________    __________    __________    ________________

    6.0000e+00    2.2375e-08    2.2204e-16    1.0077e+08       3.1890e+09
    1.0000e+01    4.3888e-01    2.5225e-16    1.7399e+15       8.7833e+15
    1.4000e+01    8.2928e+01    1.6905e-15    4.9054e+16       6.0884e+17

fx >>
```

We see that for both kinds of matrices, those with larger sizes and condition numbers report higher RFE and EMF values as well. However, in the second case, the condition numbers are very large and the RFE is not negligible at all, leading to completely incorrect solution. Only the case for $n = 6$ has a respectable RFE - meaning a solution with at least some correct digits. Matrices for the second case with $n = 10, 14$ are complete failures.

The code for the Gaussian Elimination function and the main MATLAB script follow.

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

nlist = [6, 10, 14];
RFE = zeros(1, length(nlist));
RBE = zeros(1, length(nlist));
EMF = zeros(1, length(nlist));
CN = zeros(1, length(nlist));

for k = 1:length(nlist)
n = nlist(k);
A = zeros(n);

% define matrices
for i = 1:n
for j = 1:n
A(i, j) = sqrt(1 + sin(i + j)/2);
end
end

c = ones(n, 1); b = A*c;

% compute c (Ac = b) using Gaussian Elimination
format longE
c1 = GaussElim(A, b);

% compute RFE, RBE, EMF and condition number
RFE(k) = norm(c - c1, 'Inf')/norm(c, 'Inf');
RBE(k) = max(eps, norm(b - A*c1, 'Inf')/norm(b, 'Inf'));
EMF(k) = RFE(k)/RBE(k);
CN(k) = cond(A, 'Inf');
end

% print table of results
format shortE
fprintf('Summary Table\n');
tbl = table;
tbl.n = nlist';
tbl.RFE = RFE';
tbl.RBE = RBE';
```

```
tbl.EMF = EMF';
tbl.condition_number = CN';
disp(tbl)

% repeat for different matrix A
for k = 1:length(nlist)
n = nlist(k);
A = zeros(n);

% define matrices
for i = 1:n
for j = 1:n
A(i, j) = sqrt(1 + 2*(i + j));
end
end

c = ones(n, 1); b = A*c;

% compute c (Ac = b) using Gaussian Elimination
format longE
c1 = GaussElim(A, b);

% compute RFE, RBE, EAMF and condition number
RFE(k) = norm(c - c1, 'Inf')/norm(c, 'Inf');
RBE(k) = max(eps, norm(b - A*c1, 'Inf')/norm(b, 'Inf'));
EMF(k) = RFE(k)/RBE(k);
CN(k) = cond(A, 'Inf');
end

% print table of results
format shortE
fprintf('Summary Table\n');
tbl = table;
tbl.n = nlist';
tbl.RFE = RFE';
tbl.RBE = RBE';
tbl.EMF = EMF';
tbl.condition_number = CN';
disp(tbl)
```