Maddox Gonzalez

Contents

Due 2/16/2025

Project 3 - Solving Systems of Equations using Gauss Elimination

```
% Clear command window, close all graphs, clear workspace
clc; close all; clear;

% Read data from files
A = dlmread('A.txt');
b = dlmread('b.txt');

fprintf('\nOriginal A and b:\n');
disp(A);
disp(b);

% diagonal dominance
if diag_dom(A)
    fprintf("Matrix A is diagonally dominant.\n");
else
    fprintf("Matrix A is NOT diagonally dominant.\n");
end

% Gaussian elimination
[x, A_tri, b_tri] = gauss_simple(A, b);

fprintf('\nUpper Triangulated A and b:\n');
disp(A_tri);
disp(b_tri);

fprintf('\nSolution vector x:\n');
disp(x);

% Display problem 1 answers
disp(['x(39) = ', num2str(x(39), '%.3e')]);
disp(['x(51) = ', num2str(x(51), '%.3e')]);
disp(['x(102) = ', num2str(x(102), '%.3e')]);
disp(['x(113) = ', num2str(x(113), '%.3e')]);

% Display problem 1 part 2 answers
disp(['A(28,57) = ', num2str(A_tri(28,57), '%.3e')]);
disp(['b(11) = ', num2str(b_tri(11), '%.3e')]);


% Function to check if a matrix is diagonally dominant
function is_dd = diag_dom(A)
    n = size(A, 1);
    is_dd = true;
    for i = 1:n
        if abs(A(i,i)) < sum(abs(A(i,:))) - abs(A(i,i))
```

```
            is_dd = false;
            return;
        end
    end
end

% Function to perform Gaussian elimination (without pivoting)
function [x, A, b] = gauss_simple(A, b)
    fprintf("Converting A to upper triangular form...\n");
    n = length(b);
    x = zeros(n, 1);

    % Forward elimination
    for j = 1:n-1
        for i = j+1:n
            s = -A(i,j) / A(j,j);
            A(i,:) = A(i,:) + s * A(j,:); % Apply ERO to A
            b(i) = b(i) + s * b(j); % Apply ERO to b
        end
    end

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

```
Original A and b:
  Columns 1 through 7

    8.0000   -0.4998    0.4997   -0.4996    0.4994   -0.4992    0.4989
   -0.4998   16.0000   -0.4996    0.4994   -0.4992    0.4989   -0.4986
    0.4997   -0.4996   24.0000   -0.4992    0.4989   -0.4986    0.4983
   -0.4996    0.4994   -0.4992   32.0000   -0.4986    0.4983   -0.4979
    0.4994   -0.4992    0.4989   -0.4986   40.0000   -0.4979    0.4975
   -0.4992    0.4989   -0.4986    0.4983   -0.4979   48.0000   -0.4971
    0.4989   -0.4986    0.4983   -0.4979    0.4975   -0.4971   56.0000
   -0.4986    0.4983   -0.4979    0.4975   -0.4971    0.4967   -0.4962
    0.4983   -0.4979    0.4975   -0.4971    0.4967   -0.4962    0.4956
   -0.4979    0.4975   -0.4971    0.4967   -0.4962    0.4956   -0.4951
    0.4975   -0.4971    0.4967   -0.4962    0.4956   -0.4951    0.4945
   -0.4971    0.4967   -0.4962    0.4956   -0.4951    0.4945   -0.4938
    0.4967   -0.4962    0.4956   -0.4951    0.4945   -0.4938    0.4932
   -0.4962    0.4956   -0.4951    0.4945   -0.4938    0.4932   -0.4925
    0.4956   -0.4951    0.4945   -0.4938    0.4932   -0.4925    0.4918
   -0.4951    0.4945   -0.4938    0.4932   -0.4925    0.4918   -0.4910
    0.4945   -0.4938    0.4932   -0.4925    0.4918   -0.4910    0.4902
   -0.4938    0.4932   -0.4925    0.4918   -0.4910    0.4902   -0.4894
    0.4932   -0.4925    0.4918   -0.4910    0.4902   -0.4894    0.4885
   -0.4925    0.4918   -0.4910    0.4902   -0.4894    0.4885   -0.4876
    0.4918   -0.4910    0.4902   -0.4894    0.4885   -0.4876    0.4867
   -0.4910    0.4902   -0.4894    0.4885   -0.4876    0.4867   -0.4857
    0.4902   -0.4894    0.4885   -0.4876    0.4867   -0.4857    0.4847
   -0.4894    0.4885   -0.4876    0.4867   -0.4857    0.4847   -0.4837
    0.4885   -0.4876    0.4867   -0.4857    0.4847   -0.4837    0.4826
   -0.4876    0.4867   -0.4857    0.4847   -0.4837    0.4826   -0.4815
```

```
    0.0022
    0.0039
    0.0023
   -0.0016
   -0.0038
   -0.0026
    0.0011
    0.0035
    0.0029
   -0.0006
   -0.0033
   -0.0031

x(39) = -3.299e-03
x(51) = -6.458e-03
x(102) = -7.736e-04
x(113) = -3.751e-03
A(28,57) = -2.989e-01
b(11) = 3.094e-01
```

Published with MATLAB® R2024b