

Homework 3

AE370 - Spring 2018

Emilio R. Gordon

Problem: For the systems of equations

$$5x_1 - 15x_2 - x_3 + 2x_4 = 11$$

$$3x_1 + 20x_2 + 4x_3 = -8$$

$$x_1 + 6x_2 + 11x_3 - 3x_4 = 7$$

$$4x_1 + 11x_2 + 3x_3 + 12x_4 = 4$$

Solve the equations using the Jacobi method and the Gauss-seidel method. Stop when the system residual is of the same order of magnitude as the corresponding "exact solution" from Matlab's operator

Solution:

The Jacobi Method converges within a tolerance of 1e-05 after the 30 iteration with a final solution $u=[0.2159 \ -0.6594 \ 1.1352 \ 0.5820]$

The Gauss-Seidel Method converges within a tolerance of 1e-05 after the 14 iteration with a final solution $u=[0.2159 \ -0.6594 \ 1.1352 \ 0.5820]$

The Matlab Function $A \setminus B = u$ has the final solution of $u=[0.2160 \ -0.6594 \ 1.1352 \ 0.5820]$

As we can see, both methods converged to the Matlab "exact" value. Note that the Gauss-Seidel method converges in almost half the number of iterations needed for the Jacobi method.

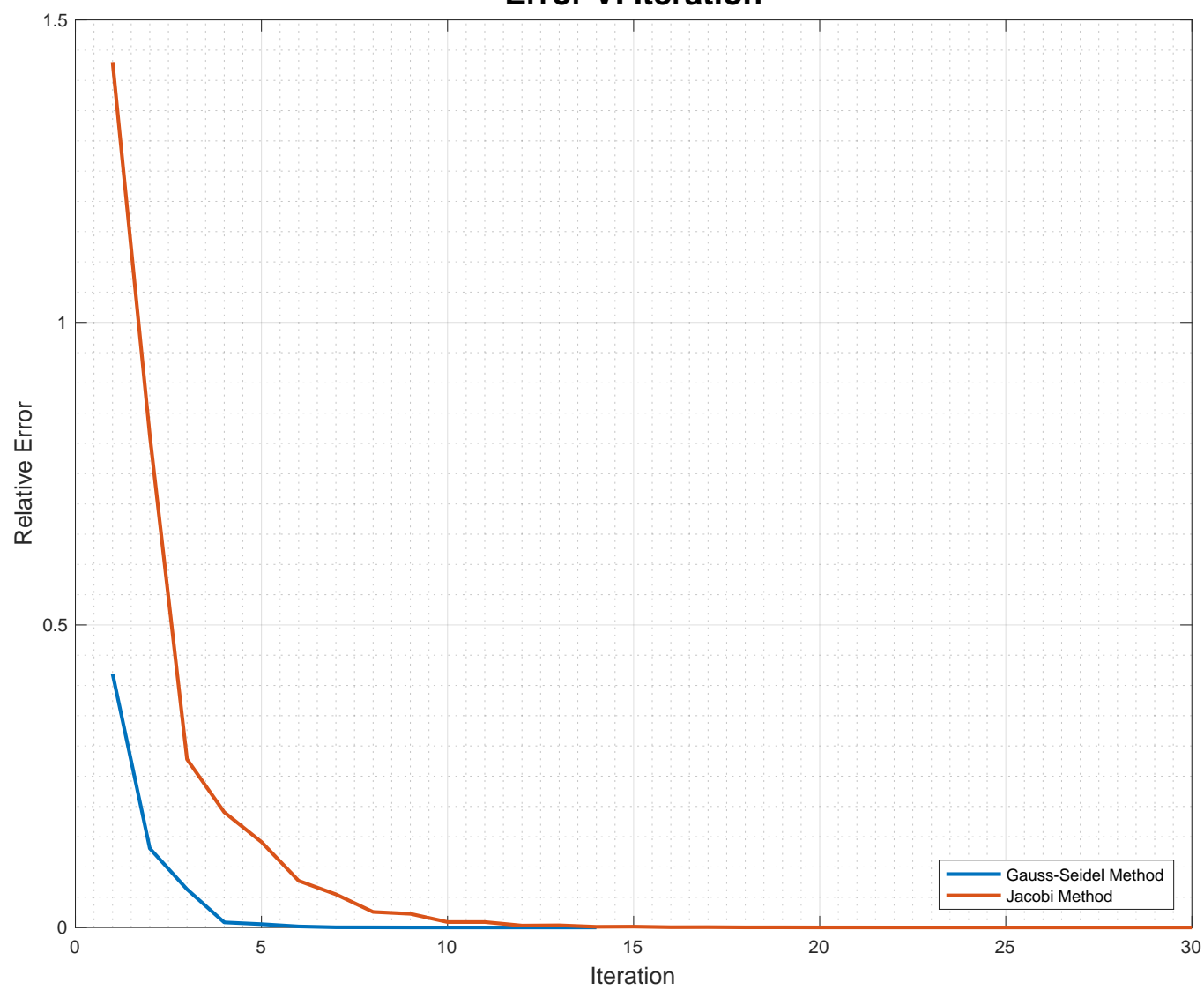
The following code was utilized to arrive at these solutions. Following the code are two plots. The first is the relative error of the magnitude of u compared for each method to the exact solution. The second plot is the convergence of each method looking at their residue.

```

1 function hw3
2     tol = 0.00001; % Tolerance
3     A = [5 -15 -1 2; 3 20 4 0; 1 6 11 -3; 4 11 3 12]; % A Matrix defined by problem statement
4     B = [11; -8; 7; 4]; % B Matrix defined by problem statement
5     exact = A\B; % Exact Matlab Solution
6     [uJ iterationsJ] = jacobi(A,B,tol); % Run Jacobi Function
7     [uGS iterationsGS] = gaussSeidel(A,B,tol); % Run Gauss-Seidel Function
8     plotter(A,B,tol) % Run Plotter Function
9 end
10 function [uJ iterationsJ] = jacobi(A,B,tolerance)
11     u = [0;0;0;0];
12     M = zeros(size(A));
13     N = -A; % Negative A
14     for i=1:size(A,1)
15         M(i,i) = A(i,i); % Diagonal A
16         N(i,i) = 0; % Zero Diagonal Components
17     end
18     i = 0;
19     while norm(B-A*u)>tolerance
20         i=i+1;
21         u = inv(M)*(B+N*u); %  $u_{k+1} = M^{-1} [B+N*u^k]$ 
22         iterations(i) = i;
23         uJ(1:4,i) = u;
24     end
25 end
26 function [uGS iterationsGS] = gaussSeidel(A,B,tolerance)
27     u = [0;0;0;0];
28     M = triu(A); % M is the Triangular Upper Bound of A
29     N = M-A; % N is the Triangular Lower Bound of A
30     i = 0;
31     while norm(B-A*u)>tolerance
32         i=i+1;
33         u = inv(M)*(B+N*u); %  $u_{k+1} = M^{-1} [B+N*u^k]$ 
34         iterations(i) = i;
35         uGS(1:4,i) = u;
36     end
37 end
38 function plotter(A,B,tol)
39     [uJ iterationsJ] = jacobi(A,B,tol);
40     [uGS iterationsGS] = gaussSeidel(A,B,tol);
41     for j=1:length(uGS)
42         GSRE(j) = norm(uGS(:,j)-(A\B))/norm(A\B); % Compute Relative Error
43         GSresidue(j) = norm(B-A*uGS(:,j)); % Compute Residue
44     end
45     for k=1:length(uJ)
46         JRE(k) = norm(uJ(:,k)-(A\B))/norm(A\B); % Compute Relative Error
47         Jresidue(k) = norm(B-A*uJ(:,k)); % Compute Residue
48     end
49     plot(iterationsGS, GSRE,'LineWidth',2)
50     hold
51     plot(iterationsJ, JRE,'LineWidth',2)
52     semilogy(iterationsGS,GSresidue,iterationsJ,Jresidue,'linewidth',2)
53     %Significant Plotting Code removed for space
54 end

```

Error V. Iteration



Jacobi V. Gauss-Seidel - Convergence

