Gauss-Seidel Solver (with Relaxation)

Owen Moogk (21006256)

Equations to be Solved

Circuit 1

Circuit 1

$V_1 = 200$

$\frac{V_1 - V_2}{10} + \frac{V_3 - V_2}{20} + \frac{V_5 - V_2}{5} = 0$

$\frac{V_2 - V_3}{20} + \frac{V_4 - V_3}{2} = 0$

$\frac{V_3 - V_4}{2} + \frac{V_5 - V_4}{5} = 0$

$\frac{V_2 - V_5}{5} + \frac{V_4 - V_5}{5} + \frac{V_6 - V_5}{25} = 0$

$V_6 = 0$

$$\left[\begin{array}{cccccc|c} 1 & 0 & 0 & 0 & 0 & 0 & 200 \\ 1/10 & -0.35 & 1/20 & 0 & 1/5 & 0 & 0 \\ 0 & 1/20 & -0.55 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & -0.7 & 1/5 & 0 & 0 \\ 0 & 1/5 & 0 & 1/5 & -0.44 & 1/25 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{array}\right]$$

Circuit 2

$V_1 = 0$

$V_2 = V_1 + 80$

$V_5 = V_1 + 50$

$$\frac{V_2 - V_3}{5} + \frac{V_1 - V_3}{15} + \frac{V_4 - V_3}{10} = 0$$

$$\frac{V_5 - V_4}{20} + \frac{V_4 - V_4}{25} + \frac{V_3 - V_4}{10} = 0$$

$$\left[\begin{array}{ccccc|c} 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 80 \\ -1 & 0 & 0 & 0 & 1 & 50 \\ 1/15 & 1/5 & -11/30 & 1/10 & 0 & 0 \\ 1/25 & 0 & 1/10 & -0.19 & 1/20 & 0 \end{array}\right]$$

Gauss Seidel Solution

Solution Demonstration

Circuit 1

```
Solution Vector (Gauss-Seidel):
  200.0000
  148.9375
  132.9574
  131.3708
  127.4129
         0

Solution vector (real):
  200.0000
  149.0040
  133.0677
  131.4741
  127.4900
         0
```

Circuit 3

```
Solution Vector (Gauss-Seidel):
         0
   80.0000
   55.1393
   42.1786
   50.0000

Solution vector (real):
         0
   80.0000
   55.1397
   42.1788
   50.0000
```

Gauss Seidel Discussion

1. Partial Pivoting: Partial pivoting was required to solve these systems of equations, as originally they were not in a diagonally dominant form. However, it was simpler to do this by hand than to do it in Matlab, so this was done manually before inputting them into the Matlab worksheets.
2. The initial values I selected were the corresponding zero vectors for each system size. Because each system is diagonally dominant, the initial value doesn't really matter, as convergence is guaranteed.
3. Note: The following findings were found using Circuit 1. A relaxation parameter of 2 cause divergence, however a relaxation parameter of 1.99 converged. However, this took very long, specifically 988 iterations (seen below). This is actually much larger than no relaxation (parameter = 1), which took 69 iterations[1]. The parameter that seemed optimal was 1.6, as it only took 22 iterations to converge with the same error. And of course, with a lower relaxation parameter, it took longer to converge. Using 0.8, it took 98 iterations to converge.

   The proof of all of this is pictured below, with the found solutions, compared to the real solution, and number of iterations.

Note that these matrices actually solve for voltages at all of the nodes. This is much simpler than solving for current, and required less circuit manipulation beforehand. From the solutions, we can then solve for currents through resistors.

The solution is as follows:

[1] Nice.

Circuit 1

Circuit 1

$I_{12} = \frac{200 - 148.9375}{10}$

$= 5.106\ A$

$I_{23} = \frac{148.9375 - 132.9574}{20}$

$= 0.799\ A$

$I_{34} = \frac{137.0677 - 132.4741}{2}$

$= 0.7968\ A$

$I_{45} = \frac{131.4741 - 127.4129}{5}$

$= 0.7968\ A$

$I_{56} = \frac{127.49 - 0}{25}$

$= 5.09\ A$

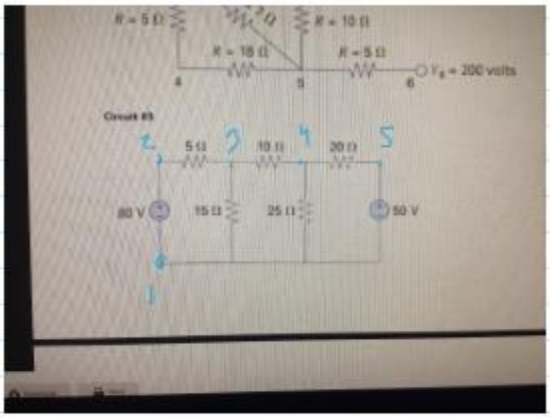
$I_{25} = \frac{148.9375 - 127.4129}{5}$

$= 4.304\ A$

Circuit 3

Note the node numbering chosen in the image.

Circuit 3



$I_{23} = \frac{80 - 55.139}{5}$

$= 4.97 \ A$

$I_{34} = \frac{55.139 - 42.179}{10}$

$= 1.296 \ A$

$I_{21} = \frac{55.139}{15}$

$= 3.676 \ A$

$I_{41} = \frac{42.178}{25}$

$\approx 1.687 \ A$

$I_{45} = \frac{42.178 - 50}{20}$

$= -0.39 \ A \quad \therefore I_{54} = 0.39 \ A$

```
>> Circuit1
Iteration:
   988

Solution Vector (Gauss-Seidel):
  199.9903
  148.9933
  133.0737
  131.4758
  127.4905
         0

Solution vector (real):
  200.0000
  149.0040
  133.0677
  131.4741
  127.4900
         0
```

Figure 1: Relaxation = 1.99

```
>> Circuit1
Iteration:
    22

Solution Vector (Gauss-Seidel):
  199.9974
  149.0085
  133.0655
  131.4732
  127.4908
         0

Solution vector (real):
  200.0000
  149.0040
  133.0677
  131.4741
  127.4900
         0
```

Figure 2: Relaxation = 1.6

```
>> Circuit1
Iteration:
    69

Solution Vector (Gauss-Seidel):
  200.0000
  148.9375
  132.9574
  131.3708
  127.4129
         0

Solution vector (real):
  200.0000
  149.0040
  133.0677
  131.4741
  127.4900
         0
```

Figure 3: Relaxation = 1

```
>> Circuit1
Iteration:
    98

Solution Vector (Gauss-Seidel):
  200.0000
  148.8966
  132.8903
  131.3052
  127.3622
         0

Solution vector (real):
  200.0000
  149.0040
  133.0677
  131.4741
  127.4900
         0
```

Figure 4: Relaxation = 0.8

Matlab Equation Solver Discussion

The solution from the Matlab equation solver was very similar to the solutions found with Gauss Seidel, and were only usually off by less than 0.1. The Matlab `linsolve` command uses direct methods to solve the system (Source: https://www.mathworks.com/help/matlab/math/iterative-methods-for-linear-systems.html), meaning that it is more accurate than the solution found using Gauss Seidel. This is because it is exact (approximately, if you ignore floating point errors), so it will always be better than an iterative method. The downside of this is that it usually takes longer, so for very large matrices with a lot of zeroes, also known as “Large and sparse”, an iterative method will have far superior performance.

Appendix

02/06/24 10:34 AM C:\Users\owenm\OneDrive - ...\Circuit1.m 1 of 1

```
% Define the coefficient matrix (A) and constant vector (b)
A = [
    1 0 0 0 0 0;
    1/10 -0.35 1/20 0 1/5 0;
    0 1/20 -0.55 1/2 0 0;
    0 0 1/2 -0.7 1/5 0;
    0 1/5 0 1/5 -0.44 1/25;
    0 0 0 0 0 1
];

% Define the constants vector (form Ax=b, this is b)
B = [200; 0; 0; 0; 0; 0];

% define parameters
initialGuess = [0; 0; 0; 0; 0; 0;];
maxIterations = 1000;
errorLevel = 0.0001;
relaxation = 0.8;

solution = GaussSeidel(A, B, initialGuess, maxIterations, errorLevel, relaxation);

% display the solution
disp("Solution Vector (Gauss-Seidel):");
disp(solution);


% Solve the system of equations
realSolution = linsolve(A, B);
% Display real solution
disp('Solution vector (real):');
disp(realSolution);
```

Figure 5: Circuit 1 Code

```
% Define the coefficient matrix (A) and constant vector (b)
A = [
    1 0 0 0 0;
    -1 1 0 0 0;
    1/15 1/5 -11/30 1/10 0;
    1/25 0 1/10 0.19 1/20;
    -1 0 0 0 1;
];

% Define the constants vector (form Ax=b, this is b)
B = [0; 80; 0; 0; 50;];

% define parameters
initialGuess = [0; 0; 0; 0; 0;];
maxIterations = 1000;
errorLevel = 0.0001;
relaxation = 1;

solution = GaussSeidel(A, B, initialGuess, maxIterations, errorLevel, relaxation);

% display the solution
disp("Solution Vector (Gauss-Seidel):");
disp(solution);


% Solve the system of equations
realSolution = linsolve(A, B);
% Display real solution
disp('Solution vector (real):');
disp(realSolution);
```

Figure 6: Circuit 3 Code

```
function [solutionVector] = GaussSeidel( ...
        coefMatrix, ...
        constantVector, ...
        initialGuess, ...
        maxIterations, ...
        errorLevel, ...
        relaxation...
    )

    % GAUSSSEIDEL This function takes in the coef matrix and constants vector,
    % and solves it using the gauss-seidel method

    % renaming variables for shorter equations
    A = coefMatrix;
    B = constantVector;
    P = initialGuess;

    % N is the number of variables that we are working in
    % This works out to be the length of b
    N = length(B);

    solutionVector = zeros(N,1);
    errorVector = zeros(N,1);

    % initial guess vector
    for j=1:maxIterations
        for i=1:N
            solutionVector(i) = ...
                ((B(i)-(A(i,[1:i-1, i+1:N])*P([1:i-1,i+1:N])))/A(i,i)) * relaxation ...
                + (1 - relaxation) * solutionVector(i) ...
            ;

            % this accounts for the case when P(i) = 0
            if solutionVector(i) == P(i)
                errorVector(i) = 0;
            else
                errorVector(i) = abs(solutionVector(i) - P(i)) / solutionVector(i);
            end
            P(i) = solutionVector(i);
        end
        if max(errorVector) < errorLevel
            disp("Iteration: ")
            disp(j)
            return
        end
    end
    warning("Warning: GaussSeidel was not able to find a solution within the number of↙
iterations specified.")
end
```

Figure 7: Gauss Seidel Code