Table of Contents

Dummy 1
Part 2 1
Part 3 2
Part 4 3
Part 5 4
Part 6 4
Local Functions 12

Dummy

Ignore this. It is used to make the outputs allign for the main code

```
A = [10, -1, 2, 0; -1, 11, -1, 3; 2, -1, 10, -1; 0, 3, -1, 8];
b = [6; 25; -11; 15];
N = 100;
tolerance = 10^-3;
x0 = zeros(size(b, 1), 1);

[jacobiSolution, jacobiIterations] = Jacobi(A, b, N, tolerance, x0);
[gaussSeidelSolution, gaussSeidelIterations] = GaussSeidel(A, b, N,
 tolerance, x0);
```

Part 2

Use the Jacobi method and the Gauss-Seidel method to solve the linear system

```
fprintf("Part 2: Use the Jacobi method and the Gauss-Seidel method to
 solve the linear system\n");

A = [10, -1, 2, 0; -1, 11, -1, 3; 2, -1, 10, -1; 0, 3, -1, 8];
b = [6; 25; -11; 15];
N = 100;
tolerance = 10^-3;
x0 = zeros(size(b, 1), 1);

fprintf("Part A: Jacobi method\n");
[jacobiSolution, jacobiIterations] = Jacobi(A, b, N, tolerance, x0);
fprintf('Jacobi Method Solution\n');
fprintf('%.4f\n', jacobiSolution);
fprintf('Jacobi Iterations: %.0f\n', jacobiIterations);
fprintf("Part B: Gauss-Seidel method\n");
[gaussSeidelSolution, gaussSeidelIterations] = GaussSeidel(A, b, N,
 tolerance, x0);
fprintf('Gauss-Seidel Method Solution\n');
fprintf('%.4f\n', gaussSeidelSolution);
fprintf('Gauss-Seidel Iterations: %.0f\n', gaussSeidelIterations);
```

```
Part 2: Use the Jacobi method and the Gauss-Seidel method to solve the
 linear system
```

```
Part A: Jacobi method
Jacobi Method Solution
0.9997
2.0004
-1.0004
1.0006
Jacobi Iterations: 9
Part B: Gauss-Seidel method
Gauss-Seidel Method Solution
1.0001
2.0000
-1.0000
1.0000
Gauss-Seidel Iterations: 5
```

Part 3

Use the Jacobi method and the Gauss-Seidel method to solve the linear system

```
fprintf("Part 3: Use the Jacobi method and the Gauss-Seidel method to
 solve the linear system\n");

A = [4, -1, 0, 0, 0, 0; -1, 4, -1, 0, 0, 0; 0, -1, 4, 0, 0, 0; 0, 0,
 0, 4, -1, 0; 0, 0, 0, -1, 4, -1; 0, 0, 0, 0, -1, 4];
b = [0; 5; 0; 6; -2; 6];
N = 100;
tolerance = 10^-4;
x0 = zeros(size(b, 1), 1);

fprintf("Part A: Jacobi method\n");
[jacobiSolution, jacobiIterations] = Jacobi(A, b, N, tolerance, x0);
fprintf('Jacobi Method Solution\n');
fprintf('%.5f\n', jacobiSolution);
fprintf('Jacobi Iterations: %.0f\n', jacobiIterations);
fprintf("Part B: Gauss-Seidel method\n");
[gaussSeidelSolution, gaussSeidelIterations] = GaussSeidel(A, b, N,
 tolerance, x0);
fprintf('Gauss-Seidel Method Solution\n');
fprintf('%.5f\n', gaussSeidelSolution);
fprintf('Gauss-Seidel Iterations: %.0f\n', gaussSeidelIterations);
```

```
Part 3: Use the Jacobi method and the Gauss-Seidel method to solve the
 linear system
Part A: Jacobi method
Jacobi Method Solution
0.35713
1.42857
0.35713
1.57143
0.28569
1.57143
Jacobi Iterations: 11
Part B: Gauss-Seidel method
```

```
Gauss-Seidel Method Solution
0.35714
1.42857
0.35714
1.57143
0.28571
1.57143
Gauss-Seidel Iterations: 7
```

Part 4

Use the best method to solve the linear system

```
fprintf("Part 4: Use the best method to solve the linear system\n");

A = [2, -1, 1; 2, 2, 2; -1, -1, 2];
b = [-1; 4; -5];
N = 100;
tolerance = 10^-3;
x0 = zeros(size(b, 1), 1);

fprintf("Part A: Calculate Spectral Radius\n");
D = diag(diag(A));
L = -tril(A,-1);
U = -triu(A,1);
Tj = inv(D)*(L+U);
Tg = inv(D-L)*U;
pTj = max(abs(eig(Tj)));
fprintf('p(Tj) is %.4f\n', pTj);
pTg = max(abs(eig(Tg)));
fprintf('p(Tg) is %.4f\n', pTg);

fprintf("Since p(Tg) is less than 1 and p(Tj) is not, it is best to
 use the Gauss-Seidel method.\n");

fprintf("Part B: Solve\n");
[gaussSeidelSolution, gaussSeidelIterations] = GaussSeidel(A, b, N,
 tolerance, x0);
fprintf('Gauss-Seidel Method Solution\n');
fprintf('%.4f\n', gaussSeidelSolution);
fprintf('Gauss-Seidel Iterations: %.0f\n', gaussSeidelIterations);

Part 4: Use the best method to solve the linear system
Part A: Calculate Spectral Radius
p(Tj) is 1.1180
p(Tg) is 0.5000
Since p(Tg) is less than 1 and p(Tj) is not, it is best to use the
 Gauss-Seidel method.
Part B: Solve
Gauss-Seidel Method Solution
1.0003
1.9996
-1.0000
```

```
Gauss-Seidel Iterations: 15
```

Part 5

Use the best method to solve the linear system

```
fprintf("Part 5: Use the best method to solve the linear system\n");

A = [1, 2, -2; 1, 1, 1; 2, 2, 1];
b = [7; 2; 5];
N = 100;
tolerance = 10^-3;
x0 = zeros(size(b, 1), 1);

fprintf("Part A: Calculate Spectral Radius\n");
D = diag(diag(A));
L = -tril(A,-1);
U = -triu(A,1);
Tj = inv(D)*(L+U);
Tg = inv(D-L)*U;
pTj = max(abs(eig(Tj)));
fprintf('p(Tj) is %.4f\n', pTj);
pTg = max(abs(eig(Tg)));
fprintf('p(Tg) is %.4f\n', pTg);

fprintf("Since p(Tj) is less than 1 and p(Tg) is not, it is best to
 use the Jacobi method.\n");

fprintf("Part B: Solve\n");
fprintf("Part A: Jacobi method\n");
[jacobiSolution, jacobiIterations] = Jacobi(A, b, N, tolerance, x0);
fprintf('Jacobi Method Solution\n');
fprintf('%.4f\n', jacobiSolution);
fprintf('Jacobi Iterations: %.0f\n', jacobiIterations);

Part 5: Use the best method to solve the linear system
Part A: Calculate Spectral Radius
p(Tj) is 0.0000
p(Tg) is 2.0000
Since p(Tj) is less than 1 and p(Tg) is not, it is best to use the
 Jacobi method.
Part B: Solve
Part A: Jacobi method
Jacobi Method Solution
1.0000
2.0000
-1.0000
Jacobi Iterations: 4
```

Part 6

Calculate the probability of reaching the left endpoint before the right

```
fprintf("Part 6: Calculate the probability of reaching the left
 endpoint before the right\n");

fprintf("Part A: alpha = 1/2\n");

N = 1000;
tolerance = 10^-10;

n = 10;
A = eye(n) - diag(.5*ones(n-1,1), -1) - diag(.5*ones(n-1,1), 1);
b = zeros(n, 1);
b(1) = 1/2;
x0 = zeros(size(b, 1), 1);

[gaussSeidelSolution, gaussSeidelIterations] = GaussSeidel(A, b, N,
 tolerance, x0);
fprintf('Gauss-Seidel Method Solution for n = %.0d\n', n);
fprintf('%.12f\n', gaussSeidelSolution);
fprintf('Gauss-Seidel Iterations: %.0f\n', gaussSeidelIterations);

n = 50;
A = eye(n) - diag(.5*ones(n-1,1), -1) - diag(.5*ones(n-1,1), 1);
b = zeros(n, 1);
b(1) = 1/2;
x0 = zeros(size(b, 1), 1);

[gaussSeidelSolution, gaussSeidelIterations] = GaussSeidel(A, b, N,
 tolerance, x0);
fprintf('Gauss-Seidel Method Solution for n = %.0d\n', n);
fprintf('%.12f\n', gaussSeidelSolution);
fprintf('Gauss-Seidel Iterations: %.0f\n', gaussSeidelIterations);

n = 100;
A = eye(n) - diag(.5*ones(n-1,1), -1) - diag(.5*ones(n-1,1), 1);
b = zeros(n, 1);
b(1) = 1/2;
x0 = zeros(size(b, 1), 1);

[gaussSeidelSolution, gaussSeidelIterations] = GaussSeidel(A, b, N,
 tolerance, x0);
fprintf('Gauss-Seidel Method Solution for n = %.0d\n', n);
fprintf('%.12f\n', gaussSeidelSolution);
fprintf('Gauss-Seidel Iterations: %.0f\n', gaussSeidelIterations);

fprintf("Part B alpha = 1/3\n");

alpha = 1/3;

n = 10;
A = eye(n) - diag(alpha*ones(n-1,1), -1) - diag((1-alpha)*ones(n-1,1),
 1);
b = zeros(n, 1);
b(1) = alpha;
x0 = zeros(size(b, 1), 1);
```

```
[gaussSeidelSolution, gaussSeidelIterations] = GaussSeidel(A, b, N,
 tolerance, x0);
fprintf('Gauss-Seidel Method Solution for n = %.0d\n', n);
fprintf('%.12f\n', gaussSeidelSolution);
fprintf('Gauss-Seidel Iterations: %.0f\n', gaussSeidelIterations);

n = 50;
A = eye(n) - diag(alpha*ones(n-1,1), -1) - diag((1-alpha)*ones(n-1,1),
 1);
b = zeros(n, 1);
b(1) = alpha;
x0 = zeros(size(b, 1), 1);

[gaussSeidelSolution, gaussSeidelIterations] = GaussSeidel(A, b, N,
 tolerance, x0);
fprintf('Gauss-Seidel Method Solution for n = %.0d\n', n);
fprintf('%.12f\n', gaussSeidelSolution);
fprintf('Gauss-Seidel Iterations: %.0f\n', gaussSeidelIterations);

n = 100;
A = eye(n) - diag(alpha*ones(n-1,1), -1) - diag((1-alpha)*ones(n-1,1),
 1);
b = zeros(n, 1);
b(1) = alpha;
x0 = zeros(size(b, 1), 1);

[gaussSeidelSolution, gaussSeidelIterations] = GaussSeidel(A, b, N,
 tolerance, x0);
fprintf('Gauss-Seidel Method Solution for n = %.0d\n', n);
fprintf('%.12f\n', gaussSeidelSolution);
fprintf('Gauss-Seidel Iterations: %.0f\n', gaussSeidelIterations);

Part 6: Calculate the probability of reaching the left endpoint before
 the right
Part A: alpha = 1/2
Gauss-Seidel Method Solution for n = 10
0.909090908751
0.818181817555
0.727272726432
0.636363635393
0.545454544441
0.454545453573
0.363636362779
0.272727272044
0.181818181349
0.090909090674
Gauss-Seidel Iterations: 243
Gauss-Seidel Method Solution for n = 50
0.979512177339
0.959031024692
0.938559826506
0.918101817047
0.897660168336
```

```
0.877237978359
0.856838259610
0.836463927994
0.816117792142
0.795802543177
0.775520744953
0.755274824830
0.735067064984
0.714899594314
0.694774380952
0.674693225416
0.654657754422
0.634669415379
0.614729471583
0.594838998130
0.574998878556
0.555209802213
0.535472262399
0.515786555229
0.496152779261
0.476570835877
0.457040430401
0.437561073960
0.418132086079
0.398752597982
0.379421556600
0.360137729262
0.340899709050
0.321705920788
0.302554627653
0.283443938366
0.264371814951
0.245336081007
0.226334430490
0.207364436932
0.188423563103
0.169509171039
0.150618532422
0.131748839263
0.112897214844
0.094060724876
0.075236388841
0.056421191461
0.037612094250
0.018806047125
Gauss-Seidel Iterations: 1000
Gauss-Seidel Method Solution for n = 100
0.982159680198
0.964337155987
0.946541271412
0.928780817028
0.911064516854
0.893401015522
0.875798865638
```

0.858266515409
0.840812296539
0.823444412437
0.806170926752
0.788999752276
0.771938640213
0.754995169861
0.738176738714
0.721490552998
0.704943618682
0.688542732941
0.672294476131
0.656205204244
0.640281041891
0.624527875792
0.608951348804
0.593556854475
0.578349532139
0.563334262548
0.548515664044
0.533898089271
0.519485622417
0.505282076994
0.491290994137
0.477515641424
0.463959012203
0.450623825416
0.437512525914
0.424627285239
0.411970002871
0.399542307919
0.387345561236
0.375380857950
0.363649030388
0.352150651366
0.340886037851
0.329855254936
0.319058120150
0.308494208047
0.298162855072
0.288063164677
0.278194012668
0.268554052752
0.259141722273
0.249955248115
0.240992652733
0.232251760319
0.223730203057
0.215425427460
0.207334700762
0.199455117350
0.191783605214
0.184316932401
0.177051713441

```
0.169984415756
0.163111365999
0.156428756341
0.149932650672
0.143618990708
0.137483601995
0.131522199787
0.125730394803
0.120103698840
0.114637530247
0.109327219231
0.104168013014
0.099155080820
0.094283518687
0.089548354113
0.084944550519
0.080467011535
0.076110585111
0.071870067455
0.067740206782
0.063715706904
0.059791230649
0.055961403106
0.052220814722
0.048564024238
0.044985561484
0.041479930033
0.038041609719
0.034665059044
0.031344717464
0.028075007573
0.024850337199
0.021665101412
0.018513684471
0.015390461699
0.012289801323
0.009206066269
0.006133615938
0.003066807969
Gauss-Seidel Iterations: 1000
Part B alpha = 1/3
Gauss-Seidel Method Solution for n = 10
0.499755739952
0.249633609971
0.124572545018
0.062042012570
0.030776746367
0.015144113278
0.007327796741
0.003419638477
0.001465559347
0.000488519782
Gauss-Seidel Iterations: 98
Gauss-Seidel Method Solution for n = 50
```

```
0.499999999735
0.249999999650
0.124999999655
0.062499999699
0.031249999754
0.015624999808
0.007812499855
0.003906249893
0.001953124923
0.000976562445
0.000488281211
0.000244140598
0.000122070294
0.000061035144
0.000030517570
0.000015258783
0.000007629391
0.000003814695
0.000001907347
0.000000953673
0.000000476836
0.000000238418
0.000000119209
0.000000059604
0.000000029802
0.000000014901
0.000000007451
0.000000003725
0.000000001863
0.000000000931
0.000000000466
0.000000000233
0.000000000116
0.000000000058
0.000000000029
0.000000000015
0.000000000007
0.000000000004
0.000000000002
0.000000000001
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
Gauss-Seidel Iterations: 129
Gauss-Seidel Method Solution for n = 100
0.499999999735
0.249999999650
```

0.124999999655
0.062499999699
0.031249999754
0.015624999808
0.007812499855
0.003906249893
0.001953124923
0.000976562445
0.000488281211
0.000244140598
0.000122070294
0.000061035144
0.000030517570
0.000015258783
0.000007629391
0.000003814695
0.000001907347
0.000000953673
0.000000476836
0.000000238418
0.000000119209
0.000000059604
0.000000029802
0.000000014901
0.000000007451
0.000000003725
0.000000001863
0.000000000931
0.000000000466
0.000000000233
0.000000000116
0.000000000058
0.000000000029
0.000000000015
0.000000000007
0.000000000004
0.000000000002
0.000000000001
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000

```
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
0.000000000000
Gauss-Seidel Iterations: 129
```

Local Functions

```
function [solution, iterations] = Jacobi(A, b, N, tolerance, x0)
answerFound = false;
n = size(b,1);
x = zeros(n, 1);
k = 1;
```

```
while k <= N && ~answerFound
    for i = 1:n
        x(i) = (1/A(i,i))*(-A(i,:)*x0(:) + A(i,i)*x0(i) + b(i));
    end
    if norm(x-x0, inf)/norm(x, inf) < tolerance
        iterations = k;
        answerFound = true;
    end
    x0 = x;
    k = k + 1;
end
if answerFound == false
    iterations = N;
end
solution = x;
end

function [solution, iterations] = GaussSeidel(A, b, N, tolerance, x0)
answerFound = false;
n = size(b,1);
x = zeros(n, 1);
k = 1;
while k <= N && ~answerFound
    for i = 1:n
        x(i) = (1/A(i,i))*(-A(i,1:i-1)*x(1:i-1) - A(i,i+1:end)*x0(i
+1:end) + b(i));
    end
    if norm(x-x0, inf)/norm(x, inf) < tolerance
        iterations = k;
        answerFound = true;
    end
    x0 = x;
    k = k + 1;
end
if answerFound == false
    iterations = N;
end
solution = x;
end
```

Published with MATLAB® R2020a