

Michael Dang – 16257750

MATH434

HW3

#3

a.

```
jacobi_with_iter.m
1 A = [2 -1 1; 2 2 2; -1 -1 2];
2 b = [-1 4 5]';
3 x0 = [0 0 0]';
4 %Part a
5 D = diag(diag(A));
6 R = A - D;
7 T = - inv(D) * R;
8 spec_radius = max(abs(eig(T)));
9 disp('Eigenvalue for matrix A')
10 spec_radius
11 %Part b
12 iter = 25;
13 N = length(b);
14 x = zeros(N,1);
15
16 for j=1:iter
17     for i=1:N
18         x(i) = (b(i)/A(i,i)) - (A(i,[1:i-1,i+1:N])*x0([1:i-1,i+1:N]))/A(i,i);
19     end
20     fprintf('Iteration # %d\n', j)
21     x
22     x0 = x;
23 end
24
25
26
```

```
>> jacobi_with_iter
Eigenvalue for matrix A

spec_radius =

    1.1180

Iteration # 1

x =

   -0.5000
    2.0000
    2.5000

Iteration # 2

x =

   -0.7500
    0
    3.2500

Iteration # 3

x =

   -2.1250
   -0.5000
    2.1250
```

b.

```
jacobi_with_iter.m
1 A = [2 -1 1; 2 2 2; -1 -1 2];
2 b = [-1 4 5]';
3 x0 = [0 0 0]';
4 %Part a
5 D = diag(diag(A));
6 R = A - D;
7 T = - inv(D) * R;
8 spec_radius = max(abs(eig(T)));
9 disp('Eigenvalue for matrix A')
10 spec_radius
11 %Part b
12 iter = 25;
13 N = length(b);
14 x = zeros(N,1);
15
16 for j=1:iter
17     for i=1:N
18         x(i) = (b(i)/A(i,i)) - (A(i,[1:i-1,i+1:N])*x0([1:i-1,i+1:N]))/A(i,i);
19     end
20     fprintf('Iteration # %d\n', j)
21     x
22     x0 = x;
23 end
24
25
26
```

```
x =

    3.1757
   -7.3895
   10.8705

Iteration # 23

x =

   -9.6300
  -12.0461
    0.3931

Iteration # 24

x =

   -6.7196
   11.2369
   -8.3381

Iteration # 25

x =

    9.2875
   17.0577
    4.7587

fx >>
```

After 25 iterations, Jacobi method fail. The approximate solution is far away from the exact solution.

c.

The image shows a MATLAB script in the Editor window and its output in the Command Window. The script defines a matrix A, a vector b, and an initial guess x0. It calculates the spectral radius and then enters a loop for iterations. The first iteration is shown in the Command Window output.

```

1 A = [2 -1 1; 2 2 2; -1 -1 2];
2 b = [-1 4 5]';
3 x0 = [0 0 0]';
4 %Part c
5 D = diag(diag(A));
6 U = triu(A,1);
7 L = tril(A,-1);
8 T = inv(L+D)*U;
9 spec_radius = max(abs(eig(T)));
10 disp('Eigenvalue for matrix A')
11 spec_radius
12 %Part d
13 iter = 25;
14 N = length(b);
15 x = zeros(N,1);
16 y = zeros(N,1);
17 tol=1e-5; %10^-5
18
19 for j=1:iter
20     for i=1:N
21         x(i) = (b(i)/A(i,i)) - (A(i,[1:i-1,i+1:N])*x0([1:i-1,i+1:N]))/A(i,i);
22         x0(i) = x(i);
23     end
24     fprintf('Iteration # %d\n', j)
25     %X
26     if abs(y-x)< tol
27         break
28     end
29     y=x
30 end

```

Command Window Output:

```

Eigenvalue for matrix A
spec_radius =
    0.5000
Iteration # 1
x =
   -0.5000
    2.5000
    3.5000
y =
   -0.5000
    2.5000
    3.5000
Iteration # 2
x =
   -1.0000
   -0.5000
    1.7500

```

d.

The image shows the same MATLAB script as above, but the Command Window output displays the results for the 23rd and 24th iterations. The values have converged to a stable state.

```

1 A = [2 -1 1; 2 2 2; -1 -1 2];
2 b = [-1 4 5]';
3 x0 = [0 0 0]';
4 %Part c
5 D = diag(diag(A));
6 U = triu(A,1);
7 L = tril(A,-1);
8 T = inv(L+D)*U;
9 spec_radius = max(abs(eig(T)));
10 disp('Eigenvalue for matrix A')
11 spec_radius
12 %Part d
13 iter = 25;
14 N = length(b);
15 x = zeros(N,1);
16 y = zeros(N,1);
17 tol=1e-5; %10^-5
18
19 for j=1:iter
20     for i=1:N
21         x(i) = (b(i)/A(i,i)) - (A(i,[1:i-1,i+1:N])*x0([1:i-1,i+1:N]))/A(i,i);
22         x0(i) = x(i);
23     end
24     fprintf('Iteration # %d\n', j)
25     %X
26     if abs(y-x)< tol
27         break
28     end
29     y=x
30 end
31

```

Command Window Output:

```

y =
   -1.2222
    0.8889
    2.3333
Iteration # 23
x =
   -1.2222
    0.8889
    2.3333
y =
   -1.2222
    0.8889
    2.3333
Iteration # 24
x =
   -1.2222
    0.8889
    2.3333
fx >>

```

I'm not sure what you mean by within 10^{-5} in l infinity norm.