《机械工程中的数值分析技术》

作业

学　　生：易弘睿

学　　号：20186103

专业班级：机械一班

作业编号：2021061305

重庆大学-辛辛那提大学联合学院

二〇二一年六月

Catalog

Lec11 Eigen-Val of matrix

1.1 Question 12.2

12.2 **(a)** Use the Gauss-Seidel method to solve the following system until the percent relative error falls below $\varepsilon_s = 5\%$:

$$\begin{bmatrix} 0.8 & -0.4 & \\ -0.4 & 0.8 & -0.4 \\ & -0.4 & 0.8 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \begin{Bmatrix} 41 \\ 25 \\ 105 \end{Bmatrix}$$

(b) Repeat **(a)** but use overrelaxation with $\lambda = 1.2$.

The Matlab code is below:

```
clear;clc;close all;
A = [0.8 -0.4 0;-0.4 0.8 -0.4;0 -0.4 0.8];
b = [41;25;105];
ea = 5;
% 12.2(a)
x1 = GaussSeidel(A,b,ea);
disp('x1 = ')
disp(x1)
% 12.2(b)
r = 1.2;
x2 = SOR(A,b,r,ea);
disp('x2 = ')
disp(x2)
function x = GaussSeidel(A,b,es,maxit)
% GaussSeidel: Gauss Seidel method
% x = GaussSeidel(A,b): Gauss Seidel without relaxation
% input:
% A = coefficient matrix
% b = right hand side vector
% es = stop criterion (default = 0.00001%)
% maxit = max iterations (default = 50)
% output:
% x = solution vector
    if nargin < 2
        error('at least 2 input arguments required')
    end
```

```
    if nargin < 4 | isempty(maxit)
        maxit = 50;
    end
    if nargin < 3 | isempty(es)
        es = 0.00001;
    end
    [m,n] = size(A);
    if m ~= n
        error('Matrix A must be square');
    end
        C = A;
    for i = 1:n
        C(i,i) = 0;
        x(i) = 0;
    end
        x = x';
    for i = 1:n
        C(i,1:n) = C(i,1:n)/A(i,i);
    end
    for i = 1:n
        d(i) = b(i)/A(i,i);
    end
        iter = 0;
    while (1)
        xold = x;
        for i = 1:n
            x(i) = d(i)-C(i,:)*x;
            if x(i) ~= 0
                ea(i) = abs((x(i) - xold(i))/x(i)) * 100;
            end
        end
        iter = iter+1;
        if max(ea)<=es | iter >= maxit
            break
        end
    end
    disp('The number of iteration by using Gauss-Seidel method to
solve the system is: ')
    disp(iter)
end
function x = SOR(A,b,r,es,maxit)
% x = SOR(A,b,La,es,maxit):  use overrelaxation with λ = 1.2

% input:
```

```
% A = coefficient matrix
% b = right hand side vector
% r = weighting factor
% es = stop criterion (default = 0.00001%)
% maxit = max iterations (default = 50)
% output:
% x = solution vector
    if nargin < 3
        error('at least 2 input arguments required'),
    end
    if nargin < 5 | isempty(maxit)
        maxit = 50;
    end
    if nargin < 4 | isempty(es)
        es = 0.00001;
    end
    [m,n] = size(A);
    if m ~= n
        error('Matrix A must be square');
    end
    C = A;
    for i = 1:n
        C(i,i) = 0;
        x(i) = 0;
    end
    x = x';
    for i = 1:n
        C(i,1:n) = C(i,1:n)/A(i,i);
    end
    for i = 1:n
        d(i) = b(i)/A(i,i);
    end
    iter = 0;
    while (1)
        xold = x;
        for i = 1:n
            x(i) = d(i) - C(i,:) * x;
            x(i) = x(i) * r + (1 - r) * xold(i);
            if x(i) ~= 0
                ea(i) = abs((x(i) - xold(i))/x(i)) * 100;
            end
        end
        iter = iter+1;
        if max(ea) <= es | iter >= maxit
```

```
            break
        end
    end
    disp('The number of iteration by using overrelaxation with λ = 
1.2 is: ')
    disp(iter)
end
```

The output is below:

```
The number of iteration by using Gauss-Seidel method to solve the system is:
     6

x1 =
  167.8711
  239.1211
  250.8105

The number of iteration by using overrelaxation with λ = 1.2 is:
     4

x2 =
  171.4230
  244.3887
  253.6222
```

1.2 Question 12.6

12.6 Use the Gauss-Seidel method **(a)** without relaxation and **(b)** with relaxation ($\lambda = 1.2$) to solve the following system to a tolerance of $\varepsilon_s = 5\%$. If necessary, rearrange the equations to achieve convergence.

$$\begin{aligned} 2x_1 - 6x_2 - x_3 &= -38 \\ -3x_1 - x_2 + 7x_3 &= -34 \\ -8x_1 + x_2 - 2x_3 &= -20 \end{aligned}$$

The Matlab code is below:

```
%% 12.6
```

```
clear;clc;close all;
A = [2 -6 -1;-3 -1 7;-8 1 -2];
b = [-38;-34;-20];
ea = 5;
r = 1.2;
% 12.6(a)
x1 = GaussSeidel(A,b,ea);
disp('x1 = ')
disp(x1)
% 12.6(b)
x2 = SOR(A,b,r,ea);
disp('x2 = ')
disp(x2)
% rearrange the equations to achieve convergence
A_R = [-8 1 -2;2 -6 -1;-3 -1 7];
b_R = [-20;-38;-34];
% 12.6(a_R)
x1_R = GaussSeidel(A_R,b_R,ea);
disp('x1_R = ')
disp(x1_R)
% 12.6(b_R)
x2_R = SOR(A_R,b_R,r,ea);
disp('x2_R = ')
disp(x2_R)
function x = GaussSeidel(A,b,es,maxit)
% GaussSeidel: Gauss Seidel method
% x = GaussSeidel(A,b): Gauss Seidel without relaxation
% input:
% A = coefficient matrix
% b = right hand side vector
% es = stop criterion (default = 0.00001%)
% maxit = max iterations (default = 50)
% output:
% x = solution vector
    if nargin < 2
        error('at least 2 input arguments required')
    end
    if nargin < 4 | isempty(maxit)
        maxit = 50;
    end
    if nargin < 3 | isempty(es)
        es = 0.00001;
    end
    [m,n] = size(A);
```

```
    if m ~= n
        error('Matrix A must be square');
    end
    C = A;
    for i = 1:n
        C(i,i) = 0;
        x(i) = 0;
    end
    x = x';
    for i = 1:n
        C(i,1:n) = C(i,1:n)/A(i,i);
    end
    for i = 1:n
        d(i) = b(i)/A(i,i);
    end
    iter = 0;
    while (1)
        xold = x;
        for i = 1:n
            x(i) = d(i)-C(i,:)*x;
            if x(i) ~= 0
                ea(i) = abs((x(i) - xold(i))/x(i)) * 100;
            end
        end
        iter = iter+1;
        if max(ea)<=es | iter >= maxit, break, end
    end
    if iter == 50
        disp('The number of iteration by using Gauss-Seidel method to solve the system is: ')
        disp(iter)
        disp('It shows that divergent')
    else
        disp('The number of iteration by using Gauss-Seidel method to solve the system afer rearranging is: ')
        disp(iter)
    end
end
function x = SOR(A,b,r,es,maxit)
% x = SOR(A,b,La,es,maxit):  use overrelaxation with λ = 1.2
% input:
% A = coefficient matrix
% b = right hand side vector
```

```
% r = weighting factor
% es = stop criterion (default = 0.00001%)
% maxit = max iterations (default = 50)
% output:
% x = solution vector
    if nargin < 3
        error('at least 2 input arguments required')
    end
    if nargin < 5 | isempty(maxit)
        maxit = 50;
    end
    if nargin < 4 | isempty(es)
        es = 0.00001;
    end
    [m,n] = size(A);
    if m ~= n
        error('Matrix A must be square');
    end
    C = A;
    for i = 1:n
        C(i,i) = 0;
        x(i) = 0;
    end
    x = x';
    for i = 1:n
        C(i,1:n) = C(i,1:n)/A(i,i);
    end
    for i = 1:n
        d(i) = b(i)/A(i,i);
    end
    iter = 0;
    while (1)
        xold = x;
        for i = 1:n
            x(i) = d(i) - C(i,:) * x;
            x(i) = x(i) * r + (1 - r) * xold(i);
            if x(i) ~= 0
                ea(i) = abs((x(i) - xold(i))/x(i)) * 100;
            end
        end
        iter = iter+1;
        if max(ea) <= es | iter >= maxit
            break
        end
```

```
    end
    if iter == 50
        disp('The number of iteration by using overrelaxation with λ
= 1.2 is: ')
        disp(iter)
        disp('It shows that divergent')
    else
        disp('The number of iteration by using overrelaxation with λ
= 1.2 afert rearranging is: ')
        disp(iter)
    end
end
```

The output is below:

```
The number of iteration by using Gauss-Seidel method to solve the
system is:
    50

It shows that divergent
x1 =
   1.0e+49 *

   -0.4871
    0.5399
    2.2183

The number of iteration by using overrelaxation with λ = 1.2 is:
    50

It shows that divergent
x2 =
   1.0e+55 *

    0.2304
   -1.6419
   -2.0718

The number of iteration by using Gauss-Seidel method to solve the
system afer rearranging is:
     3
```

```
x1_R =
    4.0047
    7.9917
   -1.9992

The number of iteration by using overrelaxation with λ = 1.2 afert rearranging is:
     6

x2_R =
    4.0122
    8.0273
   -1.9790
```

1.3 Question 12.9

12.9 Determine the solution of the simultaneous nonlinear equations:

$$x^2 = 5 - y^2$$
$$y + 1 = x^2$$

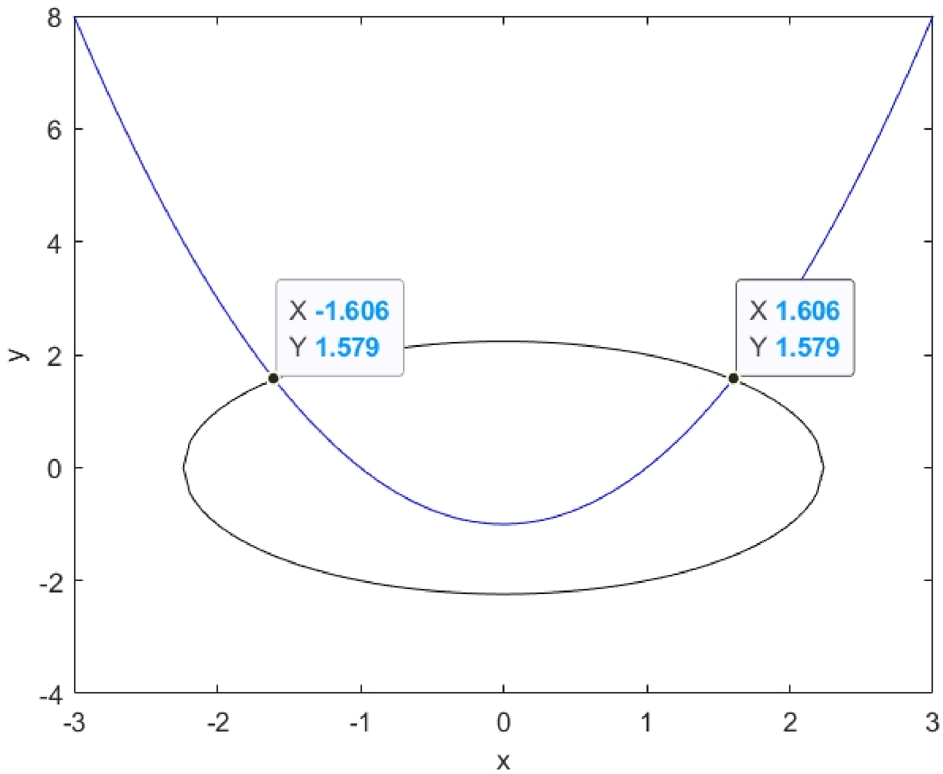
(a) Graphically.
(b) Successive substitution using initial guesses of $x = y = 1.5$.
(c) Newton-Raphson using initial guesses of $x = y = 1.5$.

a) The Matlab code is below:

```
clear; clc; close all;
% 12.9(a)
r = sqrt(5);
x1 = linspace(-r,r,100);
x2 = linspace(-3,3,100);
yu = zeros(1,length(x1));
yd = zeros(1,length(x1));
y = zeros(1,length(x1));
for i = 1:length(x1)
    yu(i) = sqrt(5 - x1(i)^2);
    yd(i) = -sqrt(5 - x1(i)^2);
end
for i = 1:length(x2)
    y(i) = x2(i)^2 - 1;
```

```
end
plot(x1, yu, 'k-')
hold on
plot(x1, yd, 'k-')
plot(x2, y, 'b-')
xlabel('x')
ylabel('y')
```

The output is below:



b) The Matlab code is below:

```
clear; clc; close all;
% (b)
x = 1.5;
y = 1.5;
maxit = 1;
iter = 0;
while (1)
    x = (y + 1) / x;
    y = (5 - x^2) / y;
    f1 = x^2 + y^2 - 5;
    f2 = y - x^2 + 1;
```

```
    iter = iter + 1;
    if (f1 == 0 && f2 == 0) | iter >= maxit
        break;
    end
end
disp('x is: ')
disp(x)
disp('y is: ')
disp(y)
```

The output is below:

```
x is:
   1.6667

y is:
1.4815
```

c) The Matlab code is below:

```
clear; clc; close all;
% (c)
es = 0.00001;
maxit = 50;
x0 = [1.5; 1.5];
x = x0;
iter = 0;
while (1)
    [J, f] = func(x);
    dx = J \ f;
    x = x - dx;
    iter = iter + 1;
    ea = 100 * max(abs(dx ./ x));
    if iter >= maxit | ea <= es
        break;
    end
end
disp('x = ')
disp(x)
disp('The number of iteration is: ')
disp(iter)

function [J, f] = func(x)
    f = zeros(1, 2);
```

```
    f(1) = x(1)^2 + x(2)^2 - 5;
    f(2) = -x(1)^2 + x(2) + 1;
    f = f';
    J = zeros(2, 2);
    J(1, 1) = 2 * x(1);
    J(1, 2) = 2 * x(2);
    J(2, 1) = -2 * x(1);
    J(2, 2) = 1;
end
```

The output is below:

```
x =
   1.6005
   1.5616

The number of iteration is:
    4
```

1.3 Question 12.11

12.11 The steady-state distribution of temperature on a heated plate can be modeled by the *Laplace equation:*

$$0 = \frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}$$

If the plate is represented by a series of nodes (Fig. P12.11), centered finite differences can be substituted for the second derivatives, which result in a system of linear algebraic equations. Use the Gauss-Seidel method to solve for the temperatures of the nodes in Fig. P12.11.

The Matlab code is below:

```
clear;clc;close all;
A = [4 -1 -1 0;-1 4 0 -1;-1 0 4 -1;0 -1 -1 4];
b = [175;125;75;25];
ea = 5;
x = GaussSeidel(A,b,ea);
disp('x = ')
disp(x)
```

```
function x = GaussSeidel(A,b,es,maxit)
% GaussSeidel: Gauss Seidel method
% x = GaussSeidel(A,b): Gauss Seidel without relaxation
% input:
% A = coefficient matrix
% b = right hand side vector
% es = stop criterion (default = 0.00001%)
% maxit = max iterations (default = 50)
% output:
% x = solution vector
    if nargin < 2,error('at least 2 input arguments required'),end
    if nargin < 4 | isempty(maxit),maxit = 50;end
    if nargin < 3 | isempty(es),es = 0.00001;end
    [m,n] = size(A);
    if m ~= n, error('Matrix A must be square'); end
    C = A;
    for i = 1:n
        C(i,i) = 0;
        x(i) = 0;
    end
    x = x';
    for i = 1:n
        C(i,1:n) = C(i,1:n)/A(i,i);
    end
    for i = 1:n
        d(i) = b(i)/A(i,i);
    end
    iter = 0;
    while (1)
        xold = x;
        for i = 1:n
            x(i) = d(i)-C(i,:)*x;
            if x(i) ~= 0
                ea(i) = abs((x(i) - xold(i))/x(i)) * 100;
            end
        end
        iter = iter+1;
        if max(ea)<=es | iter >= maxit, break, end
    end
    disp('The number of iteration by using Gauss-Seidel method to
solve the system is: ')
    disp(iter)
end
```

The output is below:

```
The number of iteration by using Gauss-Seidel method to solve the
system is:
     4

x =
   68.3105
   56.0303
   43.5303
   31.1401
```