同濟大學

计算方法实验报告（三）

学院　机械与能源工程学院

专业　机械设计制造及其自动化

学号　1852951

姓名　李腾

指导教师　李梦茹、陈茂林

完成日期　2020 年 11 月 27 日

目录

一、Jacobi 迭代法和 Gauss-Seidel 迭代法 3

1.1 实验代码 3

1.2 参数生成截图 4

二、非线性方程的解法（二分法） 5

2.1 实验代码 5

2.2 参数生成截图 6

三、非线性方程的解法（Newton 迭代法） 6

3.1 实验代码 7

3.2 参数生成截图 7

一、Jacobi 迭代法和 Gauss-Seidel 迭代法

对下列方程组，分别用 Jacobi 迭代法和 Gauss-Seidel 迭代法迭代求解，并观察是否收敛。

$$\begin{pmatrix} 1 & 2 & -2 \\ 1 & 1 & 1 \\ 2 & 2 & 1 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = randi\,(20,3,1)$$

1.1 实验代码

Jacobi 迭代函数

```
% The jacobi iterative function
% precision is the deviation between two successive iterations
% A is the coefficient matrix
% x0 is the initial value of x
% b is the constant matrix
function [y,iteration_count] = jacobi(A,b,x0,precision)
D=diag(diag(A));
U=-triu(A,1);
L=-tril(A,-1);
B=D\(L+U);
C=D\b;
y=B*x0+C;
iteration_count=1;
while norm(y-x0)>precision
    x0=y;
    y=B*x0+C;
    iteration_count=iteration_count+1;
end
end
```

Gauss-Seidel 迭代函数

```matlab
% The Gauss_seidel iterative function
% precision is the deviation between two successive iterations
% A is the coefficient matrix
% x0 is the initial value of x
% b is the constant matrix
function [y,iteration_count] = Gauss_seidel(A,b,x0,precision)
D=diag(diag(A));
U=-triu(A,1);
L=-tril(A,-1);
G=(D-L)\U;
C=(D-L)\b;
y=G*x0+C;
iteration_count=1;
while norm(y-x0)>precision
    x0=y;
    y=G*x0+C;
    iteration_count=iteration_count+1;
end
end
```

实验调用函数

```matlab
clear;
rand('seed',1852951);

A=[ 1,2,-2 ; 1,1,1 ; 2,2,1 ];
b=randi(20,3,1);
x0=[0;0;0];
precision=1e-6;
[y1,iteration_count1]=jacobi(A,b,x0,precision)
[y2,iteration_count2]=Gauss_seidel(A,b,x0,precision)
```

1.2 参数生成截图

名称	值
A	[1,2,-2;1,1,1;2,...
b	[3;3;7]
iteration_co...	4
iteration_co...	1015
precision	1.0000e-06
x0	[0;0;0]
y1	[7;-3;-1]
y2	[NaN;NaN;NaN]

```
y1 =

     7
    -3
    -1


iteration_count1 =

     4


y2 =

   NaN
   NaN
   NaN


iteration_count2 =

        1015
```

二、非线性方程的解法（二分法）

用二分法求方程 $x^2-x-a=0$ 的正根，其中$a=0.5+1.5rand(1)$，要求误差小于$b=0.03+0.03rand(1)$。

2.1 实验代码

二分法迭代函数

```
dichotomy_solve.m  ×  test2.m  ×  +
% The dichotomy iterative function
% x_upper is the upper limit of the interval
% x_down is the lower limit of the interval
% fun is the function corresponding to the equation
% error is required error of solution
function [iterations,y]=dichotomy_solve(x_upper,x_dowm,fun,error,iteration_count)
x=(x_upper+x_dowm)/2;
f3=fun(x);
f1=fun(x_upper);
if(f1*f3<0)
    m=x-x_upper;
    if(m>error)
        x_dowm=x;
        iteration_count=iteration_count+1;
        [iteration_count,y]=dichotomy_solve(x_upper,x_dowm,fun,error,iteration_count);
    else
        y=x;
    end
else
    m=x_dowm-x;
    if(m>error)
        x_upper=x;
        iteration_count=iteration_count+1;
        [iteration_count,y]=dichotomy_solve(x_upper,x_dowm,fun,error,iteration_count);
    else
        y=x;
    end
end
iterations=iteration_count;
end
```

实验调用函数

dichotomy_solve.m　test2.m

```
clear;
rand('seed',1852951);

syms x y;
a=0.5+1.5*rand(1);
fun(x)=x*x-x-a;
error=0.03+0.03*rand(1);
iteration_count=1;
[iterations,y]=dichotomy_solve(0,100,fun,error,iteration_count);
```

2.2 参数生成截图

名称	值
a	0.6879
error	0.0343
fun	*1x1 symfun*
iteration_co...	1
iterations	12
x	*1x1 sym*
y	1.4893

```
iterations =

    12


y =

    1.4893
```

三、非线性方程的解法（Newton 迭代法）

对方程$ax^b-e^x=0$（其中，$a=1+2rand(1)$，$b=0.8+1.5rand(1)$ ），用$Newton$迭代法计算。

3.1 实验代码

Newton 迭代函数

```
% The newton iterative function
% f is the function corresponding to the equation
% f_diff is the first derivative of f
% max_iteration is the maximum number of iterations
% error is required error of solution
function [iteration_count, y]=newton(x0, f, f_diff, max_iteration, error)
y=x0-f(x0)/f_diff(x0);
vpa(y, 5);
iteration_count=1;
while abs(y-x0)>=error && iteration_count < max_iteration
    iteration_count=iteration_count+1;
    x0=y;
    y=vpa(x0-f(x0)/f_diff(x0), 6);
end
end
```

实验调用函数

```
clear;
rand('seed', 1852951);

syms x;
a=1+2*rand(1);
b=0.8+1.5*rand(1);
f(x)=a*x^b-exp(x)+3;
f_diff(x)=a*b*x^(b-1)-exp(x);
error=0.001;
x0=1;
max_iteration=1000;
[iteration_count, y]=newton(x0, f, f_diff, max_iteration, error)
```

3.2 参数生成截图

名称	值
a	1.2505
b	1.0145
error	1.0000e-03
f	*1x1 symfun*
f_diff	*1x1 symfun*
iteration_co...	5
max_iteration	1000
x	*1x1 sym*
x0	1
y	*1x1 sym*

```
>> test3

iteration_count =

     5


y =

1.61653
```