

# 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 1851960

姓名 郑光泽

指导教师 李梦茹、陈茂林

完成日期 2020 年 11 月 27 日

# 1. Jacobi 迭代法和 Gauss-Seidel 迭代法求解

## 1.1. 题目

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。

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

## 1.2. 代码

### 1.2.1. 主函数

```
clear;
rand('seed', 1851960);

%% settings
Const = [ 1,2,-2 ; 1,1,1 ; 2,2,1 ];
B      = randi(20,3,1);
x0     = [0;0;0];
pre    = 1e-6;

%% results
[y1,iteration_count1] = jacobi(Const,B,x0,pre);
[y2,iteration_count2] = Gauss_seidel(Const,B,x0,pre);
```

### 1.2.2. Jacobi 迭代法

```
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
```

### 1.2.3. Gauss-Seidel 迭代法

```
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
```

### 1.3. 工作区变量

工作区	
名称 ▲	值
B	[17;17;2]
Const	[1,2,-2;1,1,1;2,2,1]
iteration_co...	4
iteration_co...	1010
pre	1.0000e-06
x0	[0;0;0]
y1	[-111;96;32]
y2	[NaN;NaN;NaN]

### 1.4. 结论

可以观察到通过 Jacobi 迭代法得到的结果  $y_1 = [-111; 96; 32]$  收敛，通过 Gauss-Seidel 迭代法的结果  $y_2 = [\text{NaN}; \text{NaN}; \text{NaN}]$  不收敛。

## 2. 非线性方程的解法（二分法）

### 2.1. 题目

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

### 2.2. 代码

#### 2.2.1. 主函数

```

clear;
rand('seed',1851960);

%% settings
syms x y;
a      = 0.5 + 1.5*rand(1);
fun(x) = x*x-x-a;
error  = 0.03 + 0.03*rand(1);
iter_c = 1;

%% main function
[iter,y] = two(0,100,fun,error,iter_c);

```

## 2.2.2. 二分法

```

function [iter,y]=two(x_upper,x_down,fun,error,iter_c)
x = (x_upper+x_down)/2;
f3 = fun(x);
f1 = fun(x_upper);
if(f1*f3<0)
    m = x-x_upper;
    if(m>error)
        x_down = x;
        iter_c = iter_c+1;
        [iter_c,y] = two(x_upper,x_down,fun,error,iter_c);
    else
        y = x;
    end
else
    m = x_down-x;
    if(m > error)
        x_upper = x;
        iter_c = iter_c+1;
        [iter_c,y] = two(x_upper,x_down,fun,error,iter_c);
    else
        y = x;
    end
end
iter = iter_c;
end

```

## 2.3. 工作区变量

工作区	
名称 ▲	值
a	1.7487
error	0.0541
fun	1x1 symfun
iter	11
iter_c	1
x	1x1 sym
y	1.9043

## 3. 非线性方程的解法（Newton迭代法）

### 3.1. 题目

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。

### 3.2. 代码

#### 3.2.1. 主函数

```
clear;
rand('seed',1851960);

%% settings
syms x;
a = 1+2*rand(1);
b = 0.8+1.5*rand(1);
f(x) = a*x^b-exp(x)+1;           % generate zero point
f_diff(x) = a*b*x^(b-1)-exp(x);

error = 0.001;
x0 = 1;
max_iter = 1000;

%% main function
[iter_c,y] = newton(x0,f,f_diff,max_iter,error);
```











#### 3.2.2. 二分法

```
function [iter_c,y] = newton(x0,f,f_diff,max_iter,error)
y = x0-f(x0)/f_diff(x0);
vpa(y,5);
iter_c = 1;

while abs(y-x0) >= error && iter_c < max_iter
    iter_c = iter_c+1;
    x0 = y;
    y = vpa(x0-f(x0)/f_diff(x0),6);
end
end
```

### 3.3. 工作区变量

---

工作区	
名称 ▲	值
 a	2.6650
 b	2.0054
 error	1.0000e-03
 f	<i>1x1 symfun</i>
 f_diff	<i>1x1 symfun</i>
 iter_c	5
 max_iter	1000
 x	<i>1x1 sym</i>
 x0	1
 y	<i>1x1 sym</i>