计算方法Final Project

> *PB19051035*周佳豪

1.

$\alpha = 0$ 时,记 $F = \sum_{i=1}^{n} \frac{1}{2}\left(\frac{u_i - u_{i-1}}{h}\right)^2 h - \sum_{i=1}^{n-1} f_i u_i h$

展开 F 可得: $F = \frac{(u_1 - u_0)^2 + \ldots + (u_n - u_{n-1})^2}{2h} - h(f_1 u_1 + \ldots + f_{n-1} u_{n-1})$

对 $u_1, \ldots u_{n-1}$ 求偏导得 $\frac{\partial F}{\partial u_i} = \frac{2u_i - u_{i-1} - u_{i+1}}{h} - f_i h$, 其中 $i = 1, \ldots, n-1$

令 $\frac{\partial F}{\partial u_i} = 0$ 可得 $f_i = \frac{2u_i - u_{i-1} - u_{i+1}}{h^2}$,其中 $i = 1, \ldots, n-1$

故得线性方程组 $A_h u_h = f_h$ 为:

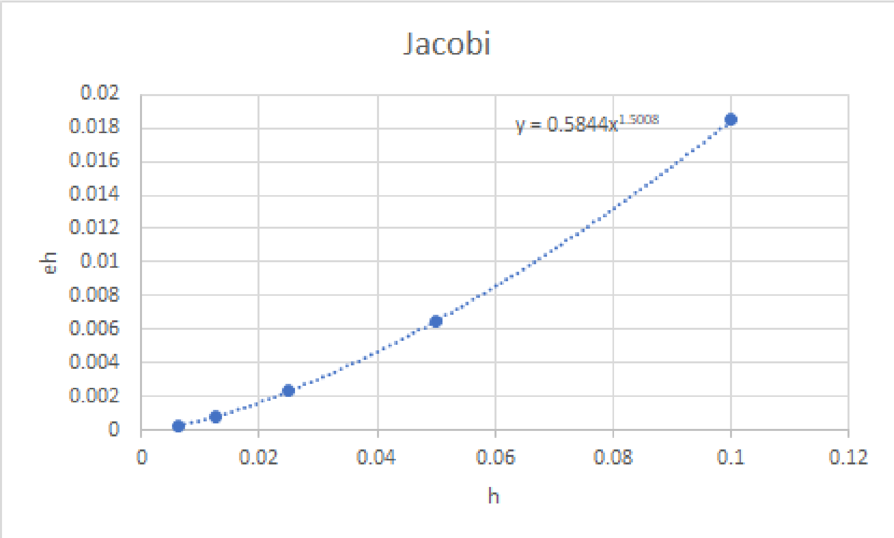
$$\begin{pmatrix} \frac{2}{h^2} & -\frac{1}{h^2} & & & & & \\ -\frac{1}{h^2} & \frac{2}{h^2} & -\frac{1}{h^2} & & & & \\ & -\frac{1}{h^2} & \frac{2}{h^2} & -\frac{1}{h^2} & & & \\ & & & \cdots & & & \\ & & & & \cdots & & \\ & & & & -\frac{1}{h^2} & \frac{2}{h^2} & -\frac{1}{h^2} \\ & & & & & -\frac{1}{h^2} & \frac{2}{h^2} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ \cdots \\ u_{n-1} \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \\ \cdots \\ f_{n-1} \end{pmatrix}$$

2.

N	JACOBI迭代误差	GAUSS-SEIDEL迭代误差
10	0.0185	0.0185
20	0.0065	0.0065
40	0.0023	0.0023
80	0.00081298	0.00081324
160	0.00028818	0.00028968

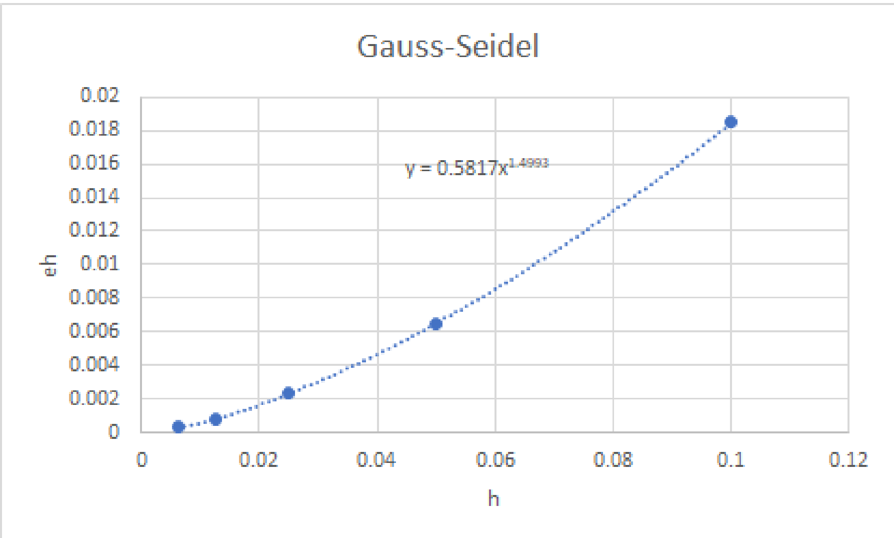
3.

Jacobi



可发现拟合结果为 $e_h = 0.5844h^{1.5008}$,故 $\beta = 1.5008$

Gauss-Seidel



可发现拟合结果为 $e_h = 0.5817h^{1.4993}$,故 $\beta = 1.4993$

4.

N	JACOBI迭代次数	GAUSS-SEIDEL迭代次数
10	408	195
20	1539	729
40	5727	2697

N	JACOBI迭代次数	GAUSS-SEIDEL迭代次数
80	21127	9891
160	77331	35970

可以看出Gauss-Seidel算法的收敛速度比Jacobi算法更快。同时随着n的增大，两种算法都需要更多次的迭代才能收敛。

5.

$\alpha = 1$ 时,记 $F = \sum_{i=1}^{n} \frac{1}{2}\left(\frac{u_i - u_{i-1}}{h}\right)^2 h + \sum_{i=1}^{n-1}\left(\frac{u_i^4}{4} - f_i u_i\right)h$

展开 F 可得:
$F = \frac{(u_1-u_0)^2+\ldots+(u_n-u_{n-1})^2}{2h} + h\left(\frac{u_1^4+\ldots+u_{n-1}^4}{4} - (f_1u_1+\ldots+f_{n-1}u_{n-1})\right)$

对 $u_1, \ldots u_{n-1}$ 求偏导得 $\frac{\partial F}{\partial u_i} = \frac{2u_i - u_{i-1} - u_{i+1}}{h} + h(u_i^3 - f_i)$，其中
$i = 1, \ldots, n-1$

令 $\frac{\partial F}{\partial u_i} = 0$ 可得 $f_i = \frac{2u_i - u_{i-1} - u_{i+1}}{h^2} + u_i^3$,其中 $i = 1, \ldots, n-1$

6.

根据题5可得方程组：

$$\begin{cases} 2u_1 - u_2 + h^2(u_1^3 - f_1) = 0 \\ 2u_i - u_{i-1} - u_{i+1} + h^2(u_i^3 - f_i) = 0, i = 2, \ldots, n-2 \\ 2u_{n-1} - u_{n-2} + h^2(u_{n-1}^3 - f_{n-1}) = 0 \end{cases}$$

进而得到方程组：

$$\begin{cases} (2+3h^2u_{1,k}^2)\delta u_1 - \delta u_2 = -(2u_{1,k} - u_{2,k} + h^2(u_{1,k}^3 - f_1)) \\ (2+3h^2u_{i,k}^2)\delta u_i - \delta u_{i-1} - \delta u_{i+1} = -(2u_{i,k} - u_{i-1,k} - u_{i+1,k} + h^2(u_{i,k}^3 - f_i)), i = 2, \ldots, n \\ (2+3h^2u_{n-1,k}^2)\delta u_{n-1} - \delta u_{n-2} = -(2u_{n-1,k} - u_{n-2,k} + h^2(u_{n-1,k}^3 - f_{n-1})) \end{cases}$$

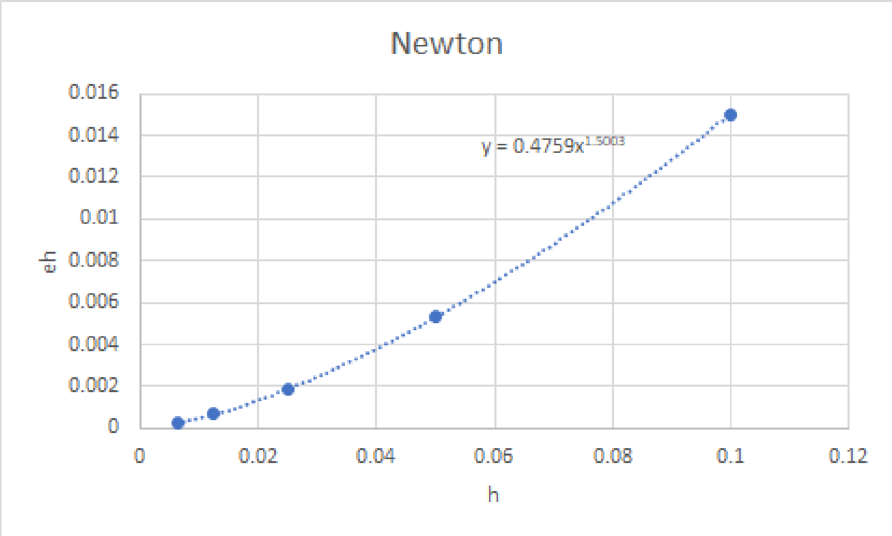
然后 $u_{k+1} = u_k + \delta u$ ，重复操作直至 $||u_{k+1} - u_k||_2 < \epsilon$

Newton迭代算法得到的结果与精确解的误差如下表：

N	NEWTON迭代误差
10	0.015
20	0.0053
40	0.0019

N	NEWTON迭代误差
80	0.00066227
160	0.00023414

拟合结果如下图：



可得公式 $e_h = 0.4759h^{1.5003}$,故得Newton算法的收敛阶为1.5003

附录

Jacobi

```
n=160;

errors = 1e-10;
h=1/n;
A = zeros(n-1,n-1);
D = zeros(n-1,n-1);
I = eye(n-1,n-1);
b = zeros(n-1,1);

ue = zeros(n-1,1);
%get b,ue
for i=1:n-1
    b(i,1)= f(i*h);
    ue(i,1)= u(i*h);
end
%get A,D
for i=1:n-1

    A(i,i)=2/(h*h);
```

```
    D(i,i)=2/(h*h);
    if(i>1)
        A(i,i-1)=-1/(h*h);
    end
    if(i<n-1)
        A(i,i+1)=-1/(h*h);
    end

end
InvD = inv(D);
R = I-InvD*A;
g = InvD*b;
x1 = zeros(n-1,1);
x2 = x1+1;
num = 0;

while norm(x1-x2,inf)>errors
    x1 = x2;
    x2 = R*x1+g;
    num = num + 1;
end

fprintf("n=\n");
n
fprintf("x2=\n");
x2
fprintf("ue=\n");
ue
fprintf("eh=\n");
eh=norm(x2-ue,2);
eh
fprintf("迭代次数为: \n");
num



function result = f(x)
    result = pi*pi*sin(pi*x);
end

function result = u(x)
    result = sin(pi*x);
end
```

Gauss-Seidel

```
n=160;
errors = 1e-10;
h=1/n;
A = zeros(n-1,n-1);
D = zeros(n-1,n-1);
L = zeros(n-1,n-1);
U = zeros(n-1,n-1);
I = eye(n-1,n-1);
b = zeros(n-1,1);
ue = zeros(n-1,1);
% get b and ue
for i=1:n-1
    b(i,1)= f(i*h);
    ue(i,1)= u(i*h);
end
% get A,D,L,U
for i=1:n-1

    A(i,i)=2/(h*h);
    D(i,i)=2/(h*h);
    if(i>1)
        A(i,i-1)=-1/(h*h);
        L(i,i-1)=-1/(h*h);
    end
    if(i<n-1)
        A(i,i+1)=-1/(h*h);
        U(i,i+1)=-1/(h*h);
    end

end

InvD_Add_L = inv(D+L);
S = -InvD_Add_L*U;
F = InvD_Add_L*b;
x1 = zeros(n-1,1);
x2 = x1+1;
num = 0;
while norm(x1-x2,inf)>errors
    x1 = x2;
    x2 = S*x1+F;
    num = num+1;
end
fprintf("n=\n");
n
fprintf("x2=\n");
```

```
x2
fprintf("ue=\n");
ue
fprintf("eh=\n");
eh=norm(x2-ue,2);
eh
fprintf("迭代次数为: \n");
num

function result = f(x)
    result = pi*pi*sin(pi*x);
end

function result = u(x)
    result = sin(pi*x);
end
```

Newton

```
n=160;
errors = 1e-8;
h=1/n;
b = zeros(n-1,1);
ue = zeros(n-1,1);
delta_u = zeros(n-1,1);

%get b,ue
for i=1:n-1
    b(i,1)= f(i*h);
    ue(i,1)= u(i*h);
end

u1 = zeros(n-1,1);
u2 = u1+1;
B = zeros(n-1,1);
A = zeros(n-1,n-1);
num=0;
while norm(u1-u2,inf)>errors
    u1 = u2;
    for i=1:n-1
        A(i,i) = (2+3*h^2*u2(i,1)^2);

        if(i==1)
            A(i,i+1) = -1;
            B(i,1) = -(2*u2(1,1)-u2(2,1)+h^2*(u2(1,1)^3-b(1,1)));
        end
```

```
        if( i>=2 && i<= n-2)
            A(i,i+1) = -1;
            A(i,i-1) = -1;
            B(i,1) = -(2*u2(i,1)-u2(i-1,1)-u2(i+1,1)+h^2*
(u2(i,1)^3-b(i,1)));
        end
        if(i==n-1)
            A(i,i-1) = -1;
            B(i,1) = -(2*u2(n-1,1)-u2(n-2,1)+h^2*(u2(n-1,1)^3-b(n-
1,1)));
        end
    end
    delta_u = A\B;
    u2 = u1+delta_u;
    num = num + 1;
end
fprintf("n=\n");
n
fprintf("u2=\n");
u2
fprintf("ue=\n");
ue
fprintf("eh=\n");
eh=norm(u2-ue,2);
eh
fprintf("迭代次数为：\n");
num




function result = f(x)
    result = pi*pi*sin(pi*x)+sin(pi*x).^3;
end

function result = u(x)
    result = sin(pi*x);
end
```