

计算方法第 5 次编程作业报告

PB16020184

付湘睿

目录

1	收敛阶及收敛值的推导	2
1.1	Newton 迭代法	2
1.1.1	情形 1	2
1.1.2	情形 2	2
1.2	弦截法	2
1.2.1	情形 1 ^[1]	2
1.2.2	情形 2	3
2	题目具体收敛阶分析及验证	4
2.1	结果	4
2.2	Newton 迭代法	5
2.2.1	第一组 $x_0 = 0.1$	5
2.2.2	第二组 $x_0 = 0.2$	5
2.2.3	第三组 $x_0 = 0.9$	5
2.2.4	第四组 $x_0 = 9.0$	5
2.3	弦截法	6
2.3.1	第一组 $x_0 = -0.1, x_1 = 0.1$	6
2.3.2	第二组 $x_0 = -0.2, x_1 = 0.2$	6
2.3.3	第三组 $x_0 = -2.0, x_1 = 0.9$	6
2.3.4	第四组 $x_0 = 0.9, x_1 = 9.0$	6
	Appendices	7
A	源代码	7

1 收敛阶及收敛值的推导

1.1 Newton 迭代法

递推公式:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \quad (1)$$

可以令 $\varphi(x) = x - \frac{f(x)}{f'(x)}$, 则 $\varphi'(x) = \frac{f(x)f''(x)}{[f'(x)]^2}$, $\varphi''(x) = \frac{[f'(x)]^3 f''(x) + f(x)[f'(x)]^2 f'''(x) - 2f(x)f'(x)[f''(x)]^2}{[f'(x)]^4}$

设 α 为 $f(x)$ 的精确根, 定义 $x_n = \alpha + e_n$, 由定义知 $\alpha = \varphi(\alpha)$, $\varphi'(\alpha) = 0$

1.1.1 情形 1

若 $\varphi''(\alpha) \neq 0$, 通过 Taylor 展开, 有

$$x_{n+1} - \alpha = \varphi(x_n) - \varphi(\alpha) = \varphi'(x_n)(x_n - \alpha) + \frac{(x_n - \alpha)^2}{2} \varphi''(\xi_n) = \frac{(x_n - \alpha)^2}{2} \varphi''(\xi_n)$$

因此有

$$\lim_{n \rightarrow \infty} \frac{|e_{n+1}|}{|e_n|^2} = \frac{|\varphi''(\alpha)|}{2}$$

1.1.2 情形 2

若 $\varphi''(\alpha) = 0$, $\varphi'''(\alpha) \neq 0$, 通过 Taylor 展开, 有

$$x_{n+1} - \alpha = \varphi(x_n) - \varphi(\alpha) = \varphi'(x_n)(x_n - \alpha) + \frac{(x_n - \alpha)^2}{2} \varphi''(\alpha) + \frac{(x_n - \alpha)^3}{6} \varphi'''(\xi_n)$$

因此有

$$\lim_{n \rightarrow \infty} \frac{|e_{n+1}|}{|e_n|^3} = \frac{|\varphi'''(\alpha)|}{6}$$

1.2 弦截法

递推公式:

$$x_{n+1} = x_n - f(x_n) \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}$$

记每次迭代的误差为 $e_n = x_n - \alpha$, 其中 α 为 $f(x)$ 的精确根, 则 $x_n = \alpha + e_n$, 因此

$$e_{n+1} = e_n - f(\alpha + e_n) \frac{e_n - e_{n-1}}{f(\alpha + e_n) - f(\alpha + e_{n-1})} \quad (2)$$

1.2.1 情形 1^[1]

设 $f'(\alpha) \neq 0$, $f''(\alpha) \neq 0$, 将 $f(x_n)$ 在 α 处 Taylor 展开, 有

$$f(\alpha + e_n) = f(\alpha) + e_n f'(\alpha) + \frac{f''(\xi_n)}{2} e_n^2 = e_n f'(\alpha) \left[1 + \frac{f''(\xi_n)}{2f'(\alpha)} e_n \right] \quad (3)$$

因此

$$f(\alpha + e_n) - f(\alpha + e_{n-1}) = f'(\alpha) \left[(e_n - e_{n-1}) + \frac{f''(\xi_n)e_n^2 - f''(\xi_{n-1})e_{n-1}^2}{2f'(\alpha)} \right] \quad (4)$$

将(3),(4)代入(2)中, 经过化简得

$$\frac{e_{n+1}}{e_n e_{n-1}} = \frac{e_n f''(\xi_n) - e_{n-1} f''(\xi_{n-1})}{2f'(\alpha)(e_n - e_{n-1}) + f''(\xi_n)e_n^2 - f''(\xi_{n-1})e_{n-1}^2}$$

因此

$$\lim_{n \rightarrow +\infty} \frac{e_{n+1}}{e_n e_{n-1}} = \frac{f''(\alpha)}{2f'(\alpha)} \quad (5)$$

设 p 为收敛阶, 则有

$$\lim_{n \rightarrow +\infty} \frac{|e_{n+1}|}{|e_n|^p} = C \quad (6)$$

联立(5),(6)可得

$$\lim_{n \rightarrow +\infty} \frac{|e_n|^{p-1}}{|e_{n-1}|} = \frac{\left| \frac{f''(\alpha)}{2f'(\alpha)} \right|}{C}$$

因此

$$\lim_{n \rightarrow \infty} \frac{|e_{n+1}|}{|e_n|^{\frac{1}{p-1}}} = \frac{\left| \frac{f''(\alpha)}{2f'(\alpha)} \right|^{\frac{1}{p-1}}}{C^{\frac{1}{p-1}}} \quad (7)$$

对比(6),(7), 得

$$\begin{cases} C = \frac{\left| \frac{f''(\alpha)}{2f'(\alpha)} \right|^{\frac{1}{p-1}}}{C^{\frac{1}{p-1}}} \\ p = \frac{1}{p-1} \end{cases} \quad (8)$$

由于 $p > 1$, 因此解得 $C = \left| \frac{f''(\alpha)}{2f'(\alpha)} \right|^{\frac{\sqrt{5}-1}{2}}$, $p = \frac{1+\sqrt{5}}{2} \approx 1.618$

1.2.2 情形 2

若 $f'(\alpha) \neq 0$, $f''(\alpha) = 0$, $f'''(\alpha) \neq 0$, 仿照情形 1 的方法, 将(2)和 $f(\alpha + e_n)$ 的带 *lagrange* 余项的三阶 Taylor 展开公式联立并取极限可得到递推关系:

$$\lim_{n \rightarrow \infty} \frac{e_{n+1}}{e_n e_{n-1} (e_n + e_{n-1})} = \frac{f'''(\alpha)}{6f'(\alpha)}$$

若弦截法收敛, 则 $p > 1$, 由收敛阶定义, 设

$$\lim_{n \rightarrow +\infty} \frac{|e_{n+1}|}{|e_n|^p} = C \quad (9)$$

因此

$$\lim_{n \rightarrow \infty} \frac{e_{n+1}}{e_n} = 0$$

因此该情形下的递推公式可以化为:

$$\lim_{n \rightarrow \infty} \frac{e_{n+1}}{e_n e_{n-1}^2} = \frac{f'''(\alpha)}{6f'(\alpha)} \quad (10)$$

将(9)与(10)联立, 可得:

$$\begin{cases} C = \left(\frac{|f'''(\alpha)|}{6f'(\alpha)} \right)^{\frac{2}{p-1}} \\ p = \frac{2}{p-1} \end{cases} \quad (11)$$

注意到 $p > 1$, 可解得 $C = \left| \frac{f'''(\alpha)}{6f'(\alpha)} \right|^{\frac{2}{3}}$, $p = 2$

2 题目具体收敛阶分析及验证

函数形式为 $f(x) = \frac{x^3}{3} - x$, 因此 $f'(x) = x^2 - 1$, $f''(x) = 2x$, $f'''(x) = 2$

2.1 结果

```
E:\vital\计算方法\作业5\PB16020184_付桐鑫_05.exe
Newton迭代法
第1组:
e(1)/e(0) 3=6.734006734006734E-001
e(2)/e(1) 3=6.666669689790854E-001
e(3)/e(2) 3=1.496000316767681E+000
初值=0.1, 根=-1.262177448353619E-029, 迭代步数=3
第2组:
e(1)/e(0) 3=6.944444444444444E-001
e(2)/e(1) 3=6.666872434334389E-001
e(3)/e(2) 3=6.66687924618858E-001
e(4)/e(3) 3=0.000000000000000E+000
初值=0.2, 根=0.000000000000000E+000, 迭代步数=4
第3组:
e(1)/e(0) 2=1.192091095282856E-001
e(2)/e(1) 2=4.118140528433625E-001
e(3)/e(2) 2=6.288996151655457E-001
e(4)/e(3) 2=8.118061720326165E-001
e(5)/e(4) 2=8.637006819364022E-001
e(6)/e(5) 2=8.660376797859792E-001
e(7)/e(6) 2=0.000000000000000E+000
初值=0.9, 根=-1.732050807568877E+000, 迭代步数=7
第4组:
e(1)/e(0) 2=8.221687836487031E-002
e(2)/e(1) 2=1.288753578078540E-001
e(3)/e(2) 2=2.053141342924571E-001
e(4)/e(3) 2=3.310932028720106E-001
e(5)/e(4) 2=5.240104039907028E-001
e(6)/e(5) 2=7.418945128981886E-001
e(7)/e(6) 2=8.528362334138895E-001
e(8)/e(7) 2=8.658942653614247E-001
e(9)/e(8) 2=1.853602114439544E+000
e(10)/e(9) 2=0.000000000000000E+000
初值=9.0, 根=1.732050807568877E+000, 迭代步数=10
```

(a) Newton 迭代法

```
E:\vital\计算方法\作业5\PB16020184_付桐鑫_05.exe
弦截法
第1组:
e(1)/e(0) 2.000=1.000000000000000E+001
e(2)/e(1) 2.000=6.776263578034402E-019
初值x1=-0.1, x2=0.1, 根=-6.776263578034403E-021, 迭代次数=2
第2组:
e(1)/e(0) 2.000=5.000000000000000E+000
e(2)/e(1) 2.000=0.000000000000000E+000
初值x1=-0.2, x2=0.2, 根=0.000000000000000E+000, 迭代次数=2
第3组:
e(1)/e(0) 1.618=9.879143065877728E-002
e(2)/e(1) 1.618=2.642710845322429E+002
e(3)/e(2) 1.618=1.622614998602094E-004
e(4)/e(3) 1.618=1.120314941258962E+000
e(5)/e(4) 1.618=5.780769971134903E+000
e(6)/e(5) 1.618=1.177614006180881E-002
e(7)/e(6) 1.618=7.11656998610632E+000
e(8)/e(7) 1.618=2.73371408054227E-001
e(9)/e(8) 1.618=2.33312904438166E+000
e(10)/e(9) 1.618=5.203552820280419E-001
e(11)/e(10) 1.618=1.290550957088552E+000
e(12)/e(11) 1.618=7.398135710654231E-001
初值x1=-2.0, x2=0.9, 根=1.732050807753635E+000, 迭代次数=12
第4组:
e(1)/e(0) 1.618=9.786156238509619E+000
e(2)/e(1) 1.618=3.268545727571223E-002
e(3)/e(2) 1.618=1.107610470250097E+000
e(4)/e(3) 1.618=8.835786678217929E+000
e(5)/e(4) 1.618=3.426525911077311E-002
e(6)/e(5) 1.618=9.784717936782917E-001
e(7)/e(6) 1.618=3.220422759493634E+000
e(8)/e(7) 1.618=3.068372790008541E-001
e(9)/e(8) 1.618=1.216150119306952E+000
e(10)/e(9) 1.618=1.042249057264986E+000
e(11)/e(10) 1.618=9.000287736029059E-001
e(12)/e(11) 1.618=9.025521904464798E-001
e(13)/e(12) 1.618=9.259411766746183E-001
e(14)/e(13) 1.618=9.083092606224866E-001
初值x1=0.9, x2=9.0, 根=1.732050807453568E+000, 迭代次数=14
```

(b) 弦截法

图 1: 程序运行输出结果

2.2 Newton 迭代法

$$\varphi(x) = x - \frac{f(x)}{f'(x)} = \frac{2x^3}{3(x^2-1)}, \varphi'(x) = \frac{2x^3(x^2-3)}{3(x^2-1)^2}, \varphi''(x) = \frac{4x(x^2+3)}{3(x^2-1)^3}, \varphi'''(x) = -\frac{4(x^4+6x^2+1)}{(x^2-1)^4}$$

2.2.1 第一组 $x_0 = 0.1$

通过计算结果判断, 最终 x_n 收敛到 0, 由于 $\varphi''(0) = 0$, $\varphi'''(0) = -4 \neq 0$, 因此属于情形 2, 理论收敛阶为 3, 且

$$\lim_{n \rightarrow \infty} \frac{|e_{n+1}|}{|e_n|^3} = \frac{|\varphi'''(0)|}{6} = \frac{2}{3} \approx 0.666667$$

由图1(a)可知随着迭代步数增加, $\frac{|e_{n+1}|}{|e_n|^3}$ 越来越接近 $\frac{2}{3}$, 当判断收敛时, $\frac{|e_{n+1}|}{|e_n|^3}$ 的值突然发生变化, 经过分析认为这是由于收敛时 e_n 过小, 导致浮点计算误差和浮点数储存的精度限制因素变得不可忽略。因此程序运行结果验证了此时的收敛阶为 3

2.2.2 第二组 $x_0 = 0.2$

与 $x_0 = 0.1$ 的情况类似, 最终 x_n 收敛到 0, 从图1(a)中的数据也能看出收敛阶确实为 3

2.2.3 第三组 $x_0 = 0.9$

通过计算结果判断, 最终 x_n 收敛到 $-\sqrt{3}$, 由于 $\varphi''(-\sqrt{3}) = -\sqrt{3} \neq 0$, 因此属于情形 1, 理论收敛阶为 2, 且

$$\lim_{n \rightarrow \infty} \frac{|e_{n+1}|}{|e_n|^2} = \frac{|\varphi''(-\sqrt{3})|}{2} \approx 0.866025$$

由图1(a)可知随着迭代步数增加, $\frac{|e_{n+1}|}{|e_n|^2}$ 越来越接近 0.866025, 当判断收敛时, $\frac{|e_{n+1}|}{|e_n|^2}$ 的值突然发生变化, 经过分析认为这是由于收敛时 e_n 过小, 导致浮点计算误差和浮点数储存的精度限制因素变得不可忽略。因此程序运行结果验证了此时的收敛阶确实为 2

2.2.4 第四组 $x_0 = 9.0$

通过计算结果判断, 最终 x_n 收敛到 $\sqrt{3}$, 由于 $\varphi''(\sqrt{3}) = \sqrt{3} \neq 0$, 因此属于情形 1, 理论收敛阶为 2, 且

$$\lim_{n \rightarrow \infty} \frac{|e_{n+1}|}{|e_n|^2} = \frac{|\varphi''(\sqrt{3})|}{2} \approx 0.866025$$

由图1(a)可知随着迭代步数增加, $\frac{|e_{n+1}|}{|e_n|^2}$ 越来越接近 0.866025, 当判断收敛时, $\frac{|e_{n+1}|}{|e_n|^2}$ 的值突然发生变化, 经过分析认为这是由于收敛时 e_n 过小, 导致浮点计算误差和浮点数储存的精度限制因素变得不可忽略。因此程序运行结果验证了此时的收敛阶确实为 2

2.3 弦截法

2.3.1 第一组 $x_0 = -0.1, x_1 = 0.1$

通过计算结果判断, 最终 x_n 收敛到 0, 由于 $f'(0) = -1 \neq 0$, $f''(0) = 0$, $f'''(0) = 2 \neq 0$, 因此属于情形 2, 理论收敛阶为 2, 且

$$\lim_{n \rightarrow \infty} \frac{|e_{n+1}|}{|e_n|^2} = \left| \frac{f'''(0)}{6f'(0)} \right|^{\frac{2}{3}} \approx 0.480750$$

然而从1(b)中没有观察到此结果, 分析为如下原因: 一、收敛速度过快, 仅两步即达到精度要求, 导致收敛时计算的比值受浮点数精度影响较大; 二、初始比值完全受初值影响, 不能反映收敛情况。

2.3.2 第二组 $x_0 = -0.2, x_1 = 0.2$

通过计算结果判断, 最终 x_n 收敛到 0, 该情况分析内容与 $x_0 = -0.1, x_1 = 0.1$ 的情况相同。

2.3.3 第三组 $x_0 = -2.0, x_1 = 0.9$

通过计算结果判断, 最终 x_n 收敛到 $\sqrt{3}$, 由于 $f'(\sqrt{3}) = 2 \neq 0$, $f''(\sqrt{3}) = 2\sqrt{3} \neq 0$, 因此属于情形 1, 理论收敛阶为 $\frac{1+\sqrt{5}}{2} \approx 1.618$, 且

$$\lim_{n \rightarrow +\infty} \frac{|e_{n+1}|}{|e_n|^{\frac{1+\sqrt{5}}{2}}} = \left| \frac{f''(\sqrt{3})}{2f'(\sqrt{3})} \right|^{\frac{\sqrt{5}-1}{2}} \approx 0.914938$$

由图1(b)可知随着迭代步数增加, $\frac{|e_{n+1}|}{|e_n|^{\frac{1+\sqrt{5}}{2}}}$ 越来越接近 0.914938。因此程序运行结果验证了此时的收敛阶确实为 $\frac{1+\sqrt{5}}{2}$

2.3.4 第四组 $x_0 = 0.9, x_1 = 9.0$

通过计算结果判断, 最终 x_n 收敛到 $\sqrt{3}$, 由于 $f'(\sqrt{3}) = 2 \neq 0$, $f''(\sqrt{3}) = 2\sqrt{3} \neq 0$, 因此属于情形 1, 理论收敛阶为 $\frac{1+\sqrt{5}}{2} \approx 1.618$, 且

$$\lim_{n \rightarrow +\infty} \frac{|e_{n+1}|}{|e_n|^{\frac{1+\sqrt{5}}{2}}} = \left| \frac{f''(\sqrt{3})}{2f'(\sqrt{3})} \right|^{\frac{\sqrt{5}-1}{2}} \approx 0.914938$$

由图1(b)可知随着迭代步数增加, $\frac{|e_{n+1}|}{|e_n|^{\frac{1+\sqrt{5}}{2}}}$ 越来越接近 0.914938。因此程序运行结果验证了此时的收敛阶确实为 $\frac{1+\sqrt{5}}{2}$

参考文献

- [1] http://www.math.drexel.edu/~tolya/300_secant.pdf

Appendices

A 源代码

```
1  #include<stdio.h>
2  #include<stdlib.h>
3  #include<math.h>
4  //定义最大迭代步数
5  #define MAXREPT 10000
6  //定义判断条件
7  #define epsilon 0.00000001
8  //Newton 迭代法求根
9  int NEWTON(double (*f)(double),double (*g)(double),double x,int flag){
10     double x0=x,x1,alpha;
11     int k,n;
12     /* 分情况设置精确根及理论收敛阶, 对于收敛的根为 0 的情况, 收敛阶为 3,
13     对于收敛的根为正负根 3 的情况, 收敛阶为 2*/
14     if(flag<=1){
15         alpha=0;
16         n=3;
17     }
18     else if(flag==2){
19         n=2;
20         alpha=-sqrt(3);
21     }
22     else{
23         n=2;
24         alpha=sqrt(3);
25     }
26     for(k=1;k<=MAXREPT;k++){
27         x1=g(x0);
28         //输出每一步的收敛阶验证情况
```

```

29     printf("|e(%d)|/|e(%d)|^d=%.15E\n",\
30 k,k-1,n,fabs(x1-alpha)/pow(fabs(x0-alpha),n));
31     if(fabs(x1-x0)<epsilon){
32         printf(" 初值 =%.11f, 根 =%.15E, 迭代步数 =%d\n\n",x,x1,k);
33         return 0;
34     }
35     x0=x1;
36 }
37 printf(" 在%.11f 附近 f(x) 无根\n",x);
38 return 1;
39 }
40 //弦截法求根
41 int XJ(double (*f)(double),double X1,double X2,int flag){
42     double f1,f2,x1=X1,x2=X2,x,alpha,n;
43     f1=f(x1);
44     int k;
45     //分情况设置精确根及收敛阶
46     if(flag<=1){
47         alpha=0;
48         n=2;
49     }
50     else{
51         alpha=sqrt(3);
52         n=(1+sqrt(5))/2;
53     }
54     printf("|e(%d)|/|e(%d)|^%.31f=%.15E\n",\
55 1,0,n,fabs(x2-alpha)/pow(fabs(x1-alpha),n));
56     for(k=2;k<=MAXREPT;k++){
57         f2=f(x2);
58         x=x2-f2*(x2-x1)/(f2-f1);
59         //输出每一步的收敛阶验证情况
60         printf("|e(%d)|/|e(%d)|^%.31f=%.15E\n",\
61 k,k-1,n,fabs(x-alpha)/pow(fabs(x2-alpha),n));
62         if(fabs(x-x2)<epsilon || fabs( f(x) )<epsilon ){
63             printf(" 初值 x1=%.11f,x2=%.11f, 根 =%.15E, 迭代次数 =%d\n\n",\
64 X1,X2,x,k);
65             return 0;
66         }

```



```

67     //为下一次迭代准备数值
68     f1=f2;
69     x1=x2;
70     x2=x;
71 }
72 printf(" 在初始值 x1=%.11f,x2=%.11f 附近 f(x) 无根\n",X1,X2);
73 return 1;
74 }
75 //返回 f(x) 的值
76 double f(double x){
77     return (x*x*x/3-x);
78 }
79 //返回 g(x)=x-f(x)/f'(x) 的值
80 double g(double x){
81     return (x-(x*x*x/3-x)/(x*x-1));
82 }
83 int main(void){
84     //Newton 迭代法初始值储存在数组里
85     double x0[4]={0.1,0.2,0.9,9.0};
86     //定义函数指针
87     double (*F)(double);
88     double (*G)(double);
89     F=f;
90     G=g;
91     int i;
92     //分别输出 Newton 迭代法结果
93     printf("Newton 迭代法\n\n");
94     for(i=0;i<4;i++){
95         printf(" 第%d 组:\n",i+1);
96         NEWTON(F,G,x0[i],i);
97     }
98     //弦截法初始值储存在二维数组里
99     double X[4][2]={{-0.1,0.1},{-0.2,0.2},{-2.0,0.9},{0.9,9.0}};
100    //分别输出弦截法结果
101    printf(" 弦截法\n\n");
102    for(i=0;i<4;i++){
103        printf(" 第%d 组:\n",i+1);
104        XJ(F,X[i][0],X[i][1],i);

```

```
105     }  
106     //暂停程序  
107     system("pause");  
108     return 0;  
109 }
```