计算方法 B 课程实验报告 3

姓名：章东泉　　**学号：**PB17121706

问题一　非线性方程求根

Part1 计算结果

取定误差限 $\varepsilon = 1.0\mathrm{E}-009$，当 $|f(x_k)| < \varepsilon$ 时停止迭代。

使用 Newton 法计算方程 $2x^4 + 24x^3 + 61x^2 - 16x + 1 = 0$ 的解，分别代入初值 $x_0 = 0.0$ 与 $x_0 = 3.0$，有如下结果：

表 1-1　Newton 法初值 0.0

迭代步数 k	x_k	$f(x_k)$
0	0.0000000000E+000	1.0000000000E+000
1	6.2500000000E-002	2.4417114258E-001
2	9.2675144823E-002	6.0357821710E-002
3	1.0750916023E-001	1.4994760152E-002
4	1.1485323376E-001	3.7248898748E-003
5	1.1848368152E-001	9.1626064336E-004
6	1.2024260677E-001	2.1577268802E-004
7	1.2102581790E-001	4.2847681852E-005
8	1.2128383271E-001	4.6530959359E-006
9	1.2131962667E-001	8.9569062833E-008
10	1.2132034327E-001	3.5900726836E-011

表 1-2　Newton 法初值 3.0

迭代步数 k	x_k	$f(x_k)$
0	3.0000000000E+000	1.3120000000E+003
1	1.9192751236E+000	3.9180729077E+002
2	1.1936133228E+000	1.1368262566E+002
3	7.3112114546E-001	3.1859876182E+001
4	4.5362080609E-001	8.6190504416E+000
5	2.9663693142E-001	2.2633471161E+000
6	2.1197535112E-001	5.8197426653E-001
7	1.6779403531E-001	1.4770709460E-001
8	1.4519438879E-001	3.7206334228E-002
9	1.3376760731E-001	9.3253679860E-003
10	1.2803649704E-001	2.3222638207E-003
11	1.2519603327E-001	5.6755417123E-004
12	1.2383872242E-001	1.2927049695E-004
13	1.2327102895E-001	2.2587102744E-005
14	1.2311856261E-001	1.6284754711E-006
15	1.2310571808E-001	1.1556329893E-008
16	1.2310562562E-001	5.9885429948E-013

使用弦截法计算方程$2x^4+24x^3+61x^2-16x+1=0$的解，分别代入初值对$x_0=0.0, x_1=0.5$与$x_0=0.1, x_1=1.5$，有如下结果：

表 1-3　弦截法初值 0.0,0.5

迭代步数 k	x_k	$f(x_k)$
0	0.0000000000E+000	1.0000000000E+000
1	5.0000000000E-001	1.1375000000E+001
2	-4.8192771084E-002	1.9100839450E+000
3	-1.5882177241E-001	4.9849583298E+000
4	2.0528956326E-002	6.9745241532E-001
5	4.9704099508E-002	3.5839401507E-001
6	8.0543024888E-002	1.1965360731E-001
7	9.5999095782E-001	4.7582804260E-002
8	1.0620354980E-001	1.7777847602E-002
9	1.1229022963E-001	6.8102183582E-003
10	1.1606968076E-001	2.5795784215E-003
11	1.1837415259E-001	9.7415265691E-004
12	1.1977247782E-001	3.6072356011E-004
13	1.2059475518E-001	1.2741741436E-004
14	1.2104383231E-001	3.9876767168E-005
15	1.2124841215E-001	9.3453570273E-006
16	1.2131102788E-001	1.1695181286E-006

17	1.2131998479E-001	4.4816937494E-008
18	1.2132034170E-001	2.3240165348E-010

表 1-4　弦截法初值 0.1,1.5

迭代步数 **k**	x_k	$f(x_k)$
0	1.0000000000E-001	3.4200000000E-002
1	1.5000000000E+000	2.0537500000E+002
2	9.9766826661E-002	3.4920022729E-002
3	9.9528703766E-002	3.5662994682E-002
4	1.1095871197E-001	8.7722830757E-003
5	1.1468740718E-001	3.8969392942E-003
6	1.1766781216E-001	1.3876451955E-003
7	1.1931598272E-001	5.3099453171E-004
8	1.2033760050E-001	1.9023231922E-004
9	1.2090792407E-001	6.3397280115E-005
10	1.2119299484E-001	1.7038550552E-005
11	1.2129776891E-001	2.8550135170E-006
12	1.2131885895E-001	1.8556909953E-007
13	1.2132032525E-001	2.3119210990E-009
14	1.2132034354E-001	1.9196866319E-012

Part2 算法分析

计算结果使用 C 语言得出，对函数值 $f(x)$，$f'(x)$ 的计算以及 Newton 法与弦截法的迭代分别用函数实现。

其中，Newton 法的迭代方程如下：

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)},$$

弦截法的迭代方程如下：

$$x_{k+1} = x_k - f(x_k)\frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})}.$$

迭代直到函数值的绝对值小于误差界限。

Part3 结果分析

四次迭代计算得到的结果整理如下：

表 3-1　迭代结果总汇

迭代公式/初值	x_k	$f(x_k)$
Newton/0.0	1.2132034327E-001	3.5900726836E-011
Newton/3.0	1.2310562562E-001	5.9885429948E-013
弦截/0.0,0.5	1.2132034170E-001	2.3240165348E-010
弦截/0.1,1.5	1.2132034354E-001	1.9196866319E-012

下面做分析。

绘图得到函数在(0,0)点附近的图像大致如下：

图 3-2　(0,0)附近函数图像

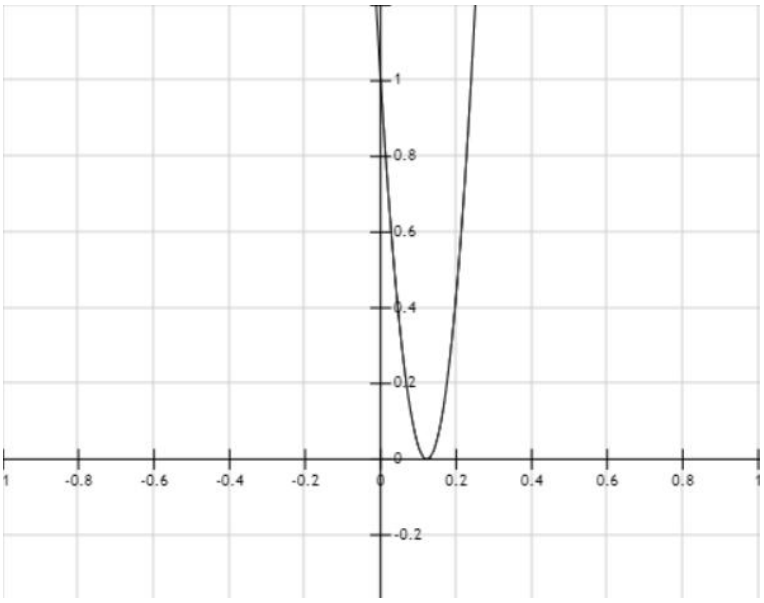
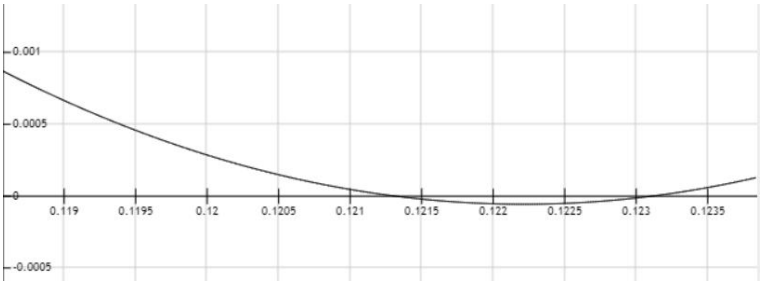


图 3-3　根附近精细图像



第一次 Newton 迭代是从 0.0 处向右逼近根；而使用弦截法时注意到题设函数 $f(x)=2x^4+24x^3+61x^2-16x+1$ 在 x>1 时增长快速，故取实验中初值，第二次迭代之后的点全部落在左侧根的左边，所以这三次计算全部在逼近函数左侧，0.1212 附近的根；而第二次 Newton 法从 3.0 处向左逼近，从而逼近了函数右侧在 0.1230 附近的根。

Part4 实验小结和代码展示

实验中看到了由于初值选择不同导致迭代次数的差异。总体上来说，Newton 法在单根附近的收敛速度快于弦截法，但是 Newton 可能

由于初值选择不当而导致迭代公式不收敛，弦截法牺牲了一些速度但是一定程度上弥补了 Newton 法此方面的不足，各有利弊。

图 4-1　函数值及其导函数值计算

```
double f_x(double x)
{
    double result;
    result = 2*pow(x, 4.0) + 24*pow(x, 3.0) + 61*pow(x, 2.0) - 16*x + 1.0;
    return result;
}

double f_prime_x(double x)
{
    double result;
    result = 8*pow(x, 3.0) + 72*pow(x, 2.0) + 122*x - 16;
    return result;
}
```

图 4-2　Newton 迭代计算

```
while(y >= epsilon)
{
    printf("%d: %.10e\t%.10e\n",k, x, f_x(x));
    x = x - f_x(x)/f_prime_x(x);
    y = f_x(x) >= 0 ? f_x(x) : -1*f_x(x);
    k++;
}
```

图 4-3　弦截法迭代计算

```
while(y >= epsilon)
{
    printf("%d: %.10e\t%.10e\n", k, x0, f_x(x0));
    double x2 = x1 - f_x(x1)*((x1 - x0)/(f_x(x1) - f_x(x0)));
    x0 = x1;
    x1 = x2;
    y = f_x(x1) >= 0 ? f_x(x1) : -1*f_x(x1);
    k++;
}
```

图 4-4　代码原始输出

```
SECANT: 1.0000000000e-001       3.4200000000e-002
0: 1.0000000000e-001    3.4200000000e-002
1: 1.5000000000e+000    2.0537500000e+002
2: 9.9766826661e-002    3.4920022729e-002
3: 9.9528703766e-002    3.5662994682e-002
4: 1.1095871197e-001    8.7722830757e-003
5: 1.1468740718e-001    3.8969392942e-003
6: 1.1766781216e-001    1.3876451955e-003
7: 1.1931598272e-001    5.3099453171e-004
8: 1.2033760050e-001    1.9023231922e-004
9: 1.2090792407e-001    6.3397280115e-005
10: 1.2119299484e-001   1.7038550552e-005
11: 1.2129776891e-001   2.8550135170e-006
12: 1.2131885895e-001   1.8556909953e-007
13: 1.2132032505e-001   2.3119210990e-009
14: 1.2132034354e-001   1.9196866319e-012
```