实验报告 4

姓名：徐煜森　　学号：　PB16110173

1. 算法分析

本次实验主体为两层循环，分别使用两组初值进行迭代法求根。为了程序的通用性，将两种迭代法主体部分写成函数，使用函数指针传入需要计算的函数和初值，在需要计算其他函数的根或不同初值时只需传入相应的函数指针或初值即可。

以下为本次实验主体部分：

```
for (int i = 0; i < N; i++) {
    cout << "Newton i = " << i << endl;
    cout << "x0 = " << NewtonX0[i] << endl;
    Newton(fx, fdx, NewtonX0[i]);
    cout << endl;
}

for (int i = 0; i < N; i++) {
    cout << "Secant i = " << i << endl;
    cout << "x0 = " << SecantX0X1[i][0] << endl;
    cout << "x1 = " << SecantX0X1[i][1] << endl;
    Secant(fx, SecantX0X1[i][0], SecantX0X1[i][1]);
    cout << endl;
}
```

以下为使用 Newton 迭代法求解根的函数：

```c
void Newton(double(*fx)(double), double(*fdx)(double), double x0) {
    printf("k\t\tx_k\t\tf(x_k)\n");
    int k = 0;
    double xk = x0;
    double fxk = (*fx)(xk);
    printf("%d\t\t%.10e\t\t%.10e\n", k, xk, fxk);
    while (abs(fxk) >= epsilon) {
        k++;
        xk = xk - (*fx)(xk) / (*fdx)(xk);
        fxk = (*fx)(xk);
        printf("%d\t\t%.10e\t\t%.10e\n", k, xk, fxk);
    }
    return;
}
```

以下为使用弦截法求根的函数：

```c
void Secant(double(*fx)(double), double x0, double x1) {
    printf("k\t\tx_k\t\tf(x_k)\n");
    int k = 0;
    double xk = x0;
    double fxk = (*fx)(xk);
    printf("%d\t\t%.10e\t\t%.10e\n", k, xk, fxk);
    k++;
    double xkplus1 = x1;
    fxk = (*fx)(xkplus1);
    printf("%d\t\t%.10e\t\t%.10e\n", k, xkplus1, fxk);
    while (abs(fxk) >= epsilon) {
        k++;
        double tmp = xkplus1;
        xkplus1 = xkplus1 - (*fx)(xkplus1)*((xkplus1 - xk) / ((*fx)(xkplus1) - (*fx)(xk)));
        fxk = (*fx)(xkplus1);
        printf("%d\t\t%.10e\t\t%.10e\n", k, xkplus1, fxk);
        xk = tmp;
    }
    return;
}
```

2. 计算结果

Newton 迭代法

表 1 Newton 迭代法 $X_0 = 0$

K	X_k	$F(X_k)$
0（初值）	0.0000000000e+00	1.0000000000e+00

1	6.2500000000e-02	2.4417114258e-01
2	9.2675144823e-02	6.0357821710e-02
3	1.0750916023e-01	1.4994760152e-02
4	1.1485323376e-01	3.7248898748e-03
5	1.1848368152e-01	9.1626064336e-04
6	1.2024260677e-01	2.1577268802e-04
7	1.2102581790e-01	4.2847681852e-05
8	1.2128383271e-01	4.6530959360e-06
9	1.2131962667e-01	8.9569062944e-08
10	1.2132034327e-01	3.5900615813e-11

从表 1 中看出，使用 Newton 迭代法求该方程的根，初值为 0 时需要 10 步就能收敛。

表 2 Newton 迭代法 $X_0 = 1$

K	X_k	$F(X_k)$
0（初值）	1.0000000000e+00	7.2000000000e+01
1	6.1290322581e-01	1.9916142676e+01
2	3.8571317721e-01	5.3253351976e+00
3	2.5960364887e-01	1.3863606793e+00
4	1.9251296856e-01	3.5450987947e-01
5	1.5779817659e-01	8.9686549723e-02

6	1.4012814469e-01	2.2547538196e-02
7	1.3122111065e-01	5.6408367721e-03
8	1.2676835045e-01	1.3986621658e-03
9	1.2457983507e-01	3.3654327128e-04
10	1.2356510397e-01	7.2212049152e-05
11	1.2318374768e-01	1.0190518405e-05
12	1.2310877106e-01	3.9378256034e-07
13	1.2310563114e-01	6.9058336827e-10
14	1.2310562562e-01	2.2204460493e-15

从表 2 中看出，使用 Newton 迭代法求该方程的根，初值为 1 时需要 14 步才能收敛。与表 1 对比可发现，不同的初始值对 Newton 迭代法的收敛速度有影响。另外，初始值的选取对最后收敛到的值也有些许影响。

弦截法

表 3 弦截法 $X_0 = 0, X_1 = 0.1$

K	X_k	$F(X_k)$
0（初值）	0.0000000000e+00	1.0000000000e+00
1（初值）	1.0000000000e-01	3.4200000000e-02
2	1.0354110582e-01	2.4179518330e-02

3	1.1208582811e-01	7.0954731540e-03
4	1.1563468594e-01	2.9655182228e-03
5	1.1818294682e-01	1.0792226046e-03
6	1.1964090533e-01	4.0681640637e-04
7	1.2052299333e-01	1.4401732385e-04
8	1.2100638906e-01	4.6100547817e-05
9	1.2123397833e-01	1.1307750277e-05
10	1.2130794544e-01	1.5591680876e-06
11	1.2131977559e-01	7.0957434595e-08
12	1.2132033965e-01	4.8875381520e-10
13	1.2132034356e-01	1.5520917884e-13

从表 3 中看出，使用弦截法求该方程的根，初值为 0 和 0.1 时需要 13 步才能收敛。

表 4 弦截法 $X_0 = 0.5, X_1 = 1.0$

K	X_k	$F(X_k)$
0（初值）	5.0000000000e-01	1.1375000000e+01
1（初值）	1.0000000000e+00	7.2000000000e+01
2	4.0618556701e-01	6.2280271014e+00
3	3.4995656522e-01	3.9299549639e+00
4	2.5379881582e-01	1.2691171507e+00

5	2.0793527302e-01	5.3000859083e-01
6	1.7504690856e-01	1.9898234938e-01
7	1.5527746672e-01	7.7353635526e-02
8	1.4270446360e-01	2.9543153285e-02
9	1.3493532719e-01	1.1322065118e-02
10	1.3010780715e-01	4.3180216959e-03
11	1.2713162110e-01	1.6401064889e-03
12	1.2530883671e-01	6.1561581910e-04
13	1.2421352668e-01	2.2446674520e-04
14	1.2358496664e-01	7.6000812034e-05
15	1.2326320210e-01	2.1431886963e-05
16	1.2313682942e-01	3.9677813742e-06
17	1.2310811800e-01	3.1191157945e-07
18	1.2310566840e-01	5.3467786865e-09
19	1.2310562568e-01	7.4583672571e-12

从表 4 中看出，使用弦截法求该方程的根，初值为 0.5 和 1 时需要 19 步才能收敛。这组初始值是本次实验中收敛速度最慢的。同样，初始值的选择对弦截法的收敛速度和收敛结果也有影响。

3. 结果分析与对比

综合比较表 1..4，可以发现弦截法的收敛速度普遍比 Newton 法

慢，这与预期相符合，因理论上对于单根 Newton 法的收敛阶为 2，而弦截法约为 1.618。另外也可看出，即使是同一种迭代求根方法，不同的初始值也会影响收敛速度和收敛结果。

本次实验的真实值约为 0.1213203436，可以看出两种方法的第一组初始值最终收敛结果几乎等于真实值，而第二组的收敛结果则有些偏差，不过最终也收敛至误差允许的范围内。可以看出，当初始值距离根足够近时，两种方法都能收敛至真实值附近。

4. 实验小结

通过本次实验，可以发现 Newton 法在实现上格式简单、收敛速度快，但在实际应用中，对函数求导过程较为复杂或根本不可行，此时可以使用弦截法替换。另外，迭代法求根的效果与初始值的选取有密切关系，如果初始值选取的不好有可能导致算法不收敛，实践中需要注意判断这种情况，避免算法不收敛时导致的死循环。