Programming Report of Chapter 1

张志心 混合 2106

日期：2023 年 9 月 30 日

1 设计思路

1.1 函数类

函数类作为一个抽象的模板类，其中 `operator()` 函数为纯虚函数，用于函数的求值运算，`Diff()` 函数为求导运算，（对于向量函数而言，即为求梯度运算）。

```
template <class Xtype, class Ytype = Xtype>
class Function{
public:
    virtual Ytype operator() (const Xtype& x) const = 0;
    virtual Ytype Diff (const Xtype& x) const {custom_assert(0, "Diff() has not be designed\
        n");}
};
```

为了简化求导的运算，还对本次实验中使用的常见的 `F: double` $\longrightarrow$ `double` 类型的函数的求导运算进行了特化，使用

$$f'(x) \simeq \frac{f(x+\epsilon) - f(x-\epsilon)}{2\epsilon}.$$

作为导数的近似，取 `10*std::numeric_limits<NUM>::epsilon()` 作为 ϵ 的值。当继承类并未定义 `Diff()` 函数时，默认采用如下函数作为求导函数：

```
template<>
NUM Function<NUM, NUM>::Diff (const NUM& x) const {
    // the default difference calculator
    return ((*this)(x+EPS_MIN)-(*this)(x-EPS_MIN))/(2*EPS_MIN);
}
```

在使用时，需要定义一个具体的函数作为 `class function` 的继承，并定义其求值运算（必要的）和求导运算（可选择的）。比如函数类 $f(x) = x - tan(x)$ 的定义如下：

```
class func : public Function<double, double>{
  public:
    double operator() (const double& x) const {
        return x-tan(x);
    }
    double Diff(const double& x) const {
        return 1-1/(sq(cos(x)));
    }
};
```

1.2 方程求解类

`class EquationSolver` 为方程求解基类，其中只含有一个 `solve()` 的纯虚函数。

代码实现见 `EquationSolver.hpp`，在使用时，请先引用该文件（`#include "EquationSolver.hpp"`）。

1.2.1 Bisection Method

`class Bisection_Method: public EquationSolver` 为二分法的求解类，用于求解方程 $f(x) = 0$ 在初始区间上的解。其成员变量包括：求解函数 $f(x)$ 、迭代初始区间 $[a, b]$ 、迭代终止条件 ϵ 和 δ 、以及最大的迭代次数 $Maxiter$。当以下任何一个条件不满足时，结束迭代：

$$\begin{cases} |f(\frac{a+b}{2})| \le \epsilon \\ |a-b| \le \delta \\ iter \le Maxiter. \end{cases}$$

二分法求解类的构造函数定义如下：

```
Bisection_Method(const Function<Type> &F, const Type &a, const Type &b,
                 const NUM& eps = 1e-7, const NUM& delta = 1e-8, const int& Maxiter = 50) :
    F(F), a(a), b(b), Maxiter(Maxiter), eps(eps), delta(delta) {}
```

可以使用 `Bisection_Method solver(f, a, b);` 构造一个二分法求解实例（其中参数采用默认值），然后调用 `solver.solve()` 来进行求解运算。

1.2.2 Newton Method

`class Newton_Method: public EquationSolver` 为牛顿法的求解类，用于求解方程 $f(x) = 0$ 在初始值附近的解。其成员变量包括：求解函数 $f(x)$ 、初始值 x_0 、迭代终止条件 ϵ 、以及最大的迭代次数 $Maxiter$。当以下任何一个条件不满足时，结束迭代：

$$\begin{cases} |f(x_n)| \le \epsilon \\ iter \le Maxiter. \end{cases}$$

牛顿法求解类的构造函数定义如下：

```
Newton_Method(const Function<Type> &F, const Type &x0,
                 const NUM& eps = 1e-7, const int& Maxiter = 50) :
    F(F), x0(x0), Maxiter(Maxiter), eps(eps){}
```

可以使用 `Newton_Method solver(f, x0);` 构造一个牛顿法求解实例（其中参数采用默认值），然后调用 `solver.solve()` 来进行求解运算。

1.2.3 Secant Method

`class Secant_Method: public EquationSolver` 为割线法的求解类，用于求解方程 $f(x) = 0$ 在初始值附近的解。其成员变量包括：求解函数 $f(x)$ 、初始值 x_0, x_1 、迭代终止条件 ϵ 和 δ 、以及最大的迭代次数 $Maxiter$。当以下任何一个条件不满足时，结束迭代：

$$\begin{cases} |f(x_n)| \le \epsilon \\ |x_n - x_{n-1}| \le \delta \\ iter \le Maxiter. \end{cases}$$

割线法求解类的构造函数定义如下：

```
Secant_Method(const Function<Type> &F, const Type& x0, const Type& x1,
              const NUM& eps = 1e-7, const NUM& delta = 1e-8, const int& Maxiter = 50) :
        F(F), x0(x0), x1(x1), Maxiter(Maxiter), eps(eps), delta(delta){}
```

可以使用 `Secant_Method solver(f, x0, x1);` 构造一个割线法求解实例（其中参数采用默认值），然后调用 `solver.solve()` 来进行求解运算。

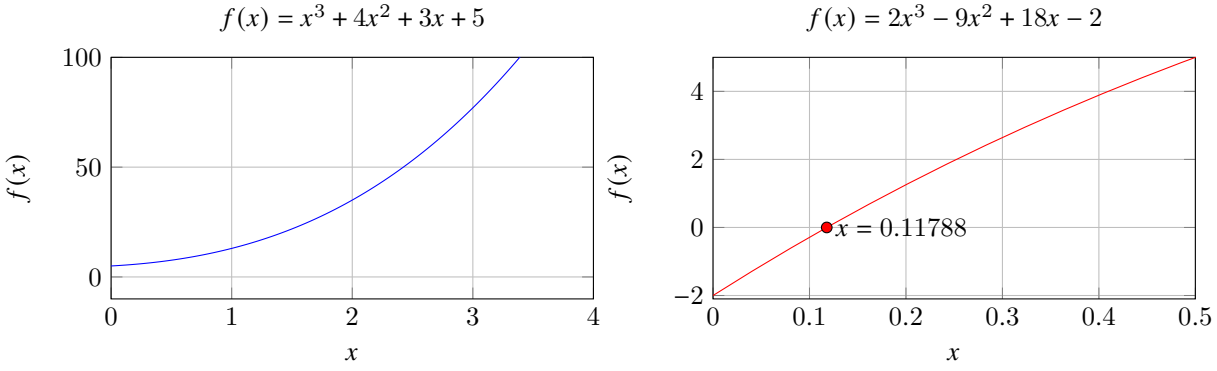
2 求解结果

2.1 问题 B

设置 $\epsilon = 10^{-7}, \delta = 10^{-8}$ 二分法求解结果如下：

$f(x)$	$[a,b]$	近似解 x^*	$\lvert f(x^*)\rvert$	迭代次数 $iter$	终止区间长度 b^*-a^*
$x^{-1}-\tan x$	$[0,\pi/2]$	0.8603336024	0.0000000495	24	0.0000000936
$x^{-1}-2^x$	$[0,1]$	0.6411857605	0.0000000562	17	0.0000076294
$2^{-x}+e^x+2\cos x-6$	$[1,3]$	1.8293836117	0.0000000400	22	0.0000004768
$\frac{x^3+4x^2+3x+5}{2x^3-9x^2+18x-2}$	$[0,4]$	0.1178765632	-94846963.165	29	0.0000000075

二分法没有办法正确求解第四个函数 $f(x)=\dfrac{x^3+4x^2+3x+5}{2x^3-9x^2+18x-2}$ 的根。这是由于该函数的分子在 $[0,4]$ 上恒大于 0，而分母在 $[0,4]$ 上有一个零点，导致二分法实际在求解的是分母的零点，而原函数该区间上没有零点，其在 $x\simeq 0.11788$ 处发散。



2.2 问题 C

即为求解 $f(x)=x-\tan x$ 的零点，$f'(x)=1-\dfrac{1}{\cos x^2}$。

设置 $\epsilon = 10^{-7}$，牛顿法求解结果如下：

$f(x)$	x_0	近似解 x^*	$\lvert f(x^*)\rvert$	迭代次数 $iter$
$f(x)=x-\tan x$	4.5	4.4934094579	0.0000000000	3
	7.7	7.7252518369	0.0000000000	4

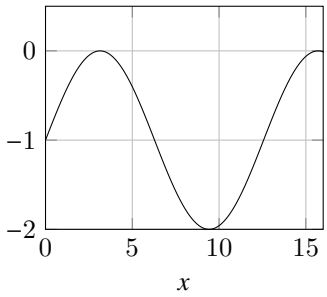
2.3 问题 D

使用割线法求解 $\sin(\frac{x}{2})-1, e^x-\tan x, x^3-12x^2+3x+1$，设置迭代终止条件 $\epsilon=10^{-7}, \delta=10^{-8}$。为每一个函数都设置了不同的初始值 x_0, x_1，得到结果如下表所示。

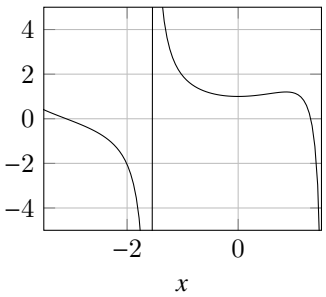
$f(x)$	x_0	x_1	近似解 x^*	$\lvert f(x^*)\rvert$	迭代次数 $iter$	$\lvert x_n-x_{n-1}\rvert$
$f(x)=\sin(\frac{x}{2})-1$	0	$\frac{\pi}{2}$	3.1409322315	.0000000545	16	0.0004081633
	14	15	15.7074076584	0.0000000386	15	0.0003433855
$f(x)=e^x-\tan x$	1	1.4	1.3063269404	0.0000000000	9	0.0000000151
	-2.8	-3.1	-3.0964122941	0.0000000103	2	0.0001338862
$f(x)=x^3-12x^2+3x+1$	0	-0.5	-0.1886854043	0.0000000063	6	0.0000019974
	8	9	11.7371421792	0.0000000004	10	0.0000001458

可以发现，不同的初始值会导致不同的求解结果。这是因为函数有不止一个零点，而割线法更倾向于找到离初始点近的零点。下为三个函数的函数图像：

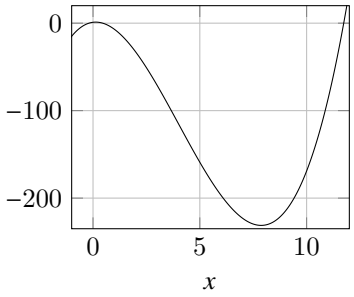
(a) $\sin(\frac{x}{2})-1$



(b) $e^x-\tan(x)$



(c) x^3-12x^2+3x+1



2.4 问题 E

求解方程：

$$f(h)=V-L\left[\frac{1}{2}\pi r^2-r^2\arctan\frac{h}{r}-h\sqrt{r^2-h^2}\right]=0.$$

对函数 $f(h)$ 进行求导得到：

$$f'(h)=L\left[\frac{r}{\sqrt{1-(h/r)^2}}+\frac{x^2}{\sqrt{r^2-h^2}}+\sqrt{r^2-h^2}\right].$$

问题所求水深即为 $r-h$。

代入 $V=12.4, L=10, r=1$，使用二分法、牛顿法、割线法求解结果如下：

Method	初始条件	近似解 x^*	$\lvert f(x^*)\rvert$	迭代次数 $iter$	ϵ	δ
Bisection	$[a,b]=[0,r]$	0.1640625000	0.0414932414	6	10^{-7}	0.02
Newton	$x_0=0$	0.1653981634	0.0151449150	1	0.01×20	
Secant	$x_0=0, x_1=r$	0.1662278952	0.0012200043	3	10^{-7}	$\frac{1}{2}[(\frac{0.01}{0.547})^{0.618}-0.01]$

最后，使用 Bisection 法，Newton 法，Secant 法得到的问题的答案分别为 0.8359375、0.8346018366、0.8337721048。

备注：关于 δ 和 ϵ 的选取问题，

1. 对于二分法，$\left|\frac{a_n+b_n}{2}-x^*\right| \le \frac{1}{2}\left|a_n-b_n\right|$，所以将 δ 设置为 2×0.01；
2. 对于牛顿法，考虑在 x^* 处 Taylor 展开：

$$f(x_n) \simeq f(x^*) + (x_n - x^*)f'(x^*) \Rightarrow \left|x_n - x^*\right| \le \left|\frac{f(x_n)}{f'(x^*)}\right|.$$

根据估计，函数在零点处导数约为 20，因此取 $\epsilon = 20\times 0.01$，事实上，还可以为牛顿法增加一个 δ 项，然而这样会增加其计算负担，牛顿法的效率通常很高，精确性也很好，所以一般不需要增加迭代终止条件；

3. 对于割线法的收敛性，有

$$|x_n - x^*| \le M|x_{n-1} - x^*|^{1.618} = M(-x_n + x_{n-1} + x_n - x^*)^{1.618}.$$

$$\delta = M(\delta + 0.01)^{1.618} \Rightarrow \delta = (\frac{0.01}{M})^{0.618} - 0.01.$$

取 $M = \left|\frac{f''(x^*)}{f'(x^*)}\right|$ 作为其近似值，并且为整个式子增加一个 $\frac{1}{2}$ 的因子，得到 δ 如上表所示。

2.5 问题 F

该问题需要把角度转化为弧度制，记 $k = \frac{\pi}{180}$。求解方程如下：

$$f(x) = A\sin kx\cos kx + B\sin^2 kx - C\cos kx - E\sin kx = 0.$$

$$f'(x) = k\left[A(\cos^2 kx - \sin^2 kx) + 2B\sin kx\cos kx + C\sin kx - E\cos kx\right].$$

(a) 和 (b) 使用牛顿法，采用了不同的参数值（$D = 55, 30$），求解结果如下：

x_0	近似解 x^*	$\lvert f(x^*)\rvert$	迭代次数 $iter$	ϵ
33°	32.9721748224°	0.0000000000	2	10^{-7}
33°	33.1689038218°	0.0000000013	2	10^{-7}

对于 (c) 题，在 $[0°, 160°]$ 中以 10° 为间距取了若干组 x_0, x_1 作初始值，可以发现不同的初始值将会收敛到不同的零点，具体结果如下：

x_0	x_1	近似解 x^*	$\lvert f(x^*)\rvert$	迭代次数 $iter$
0°	10°	11.4999999493°	0.0000000586	8
10°	20°	32.9721748162°	0.0000000057	6
20°	30°	32.9721748224°	0.0000000000	5
30°	40°	32.9721748157°	0.0000000061	4
40°	50°	32.9721748224°	0.0000000000	5
50°	60°	32.9721748224°	0.0000000000	5
60°	70°	32.9721748225°	0.0000000001	6
70°	80°	11.4999999999°	0.0000000001	7
80°	90°	216867.0278251776°	0.0000000000	13
90°	100°	1407.0278251776°	0.0000000001	8
100°	110°	147.0278252723°	0.0000000452	8
110°	120°	147.0278251774°	0.0000000001	6
120°	130°	147.0278251735°	0.0000000019	6
130°	140°	147.0278251776°	0.0000000000	6
140°	150°	147.0278251767°	0.0000000004	5
150°	160°	147.0278251356°	0.0000000200	7

可以发现，其中 147° 与 33° 两个恰好互补，前者恰好是车从左边上右边的坡的对称情景。

割线法的结果与初值点的选取有很大的关系，一般情况下，割线法会容易偏向于找到初值点附近零点，这是基于每次求割线后与坐标轴的交点一般不会离当前两个点太远，若在初值点附近存在零点，则容易收敛到附近的零点上。

然而也会有特殊的情况，比如当割线斜率的绝对值太小时，割线与坐标轴的交点离当前的两个点就会很远。

进一步的，考虑割线法的收敛性质：

$$|x_n - x^*| \leq M|x_{n-1} - x^*||x_{n-2} - x^*|$$

其中，$M = \dfrac{\max_{x\in\mathcal{B}} |f''(x)|}{2\min_{x\in\mathcal{B}} |f'(x)|}, \mathcal{B} = [x^* - \delta, x^* + \delta], \delta = \max(|x_1 - x^*|, |x_0 - x^*|)$，

当 M 充分小的时候，一定会收敛到 x^*，否则 $\{x_n\}$ 就可能会收敛到其他零点，甚至不收敛。

参考文献

[1] 张庆海. “Notes on Numerical Analysis and Numerical Methods for Differential Equations”. In: (2023).