**非线性方程求解**

# 一、历史背景

代数方程的求根问题是一个古老的数学问题。理论上，*n*次代数方程在复数域内一定有*n*个根(考虑重数)。早在16世纪就找到了三次、四次方程的求根公式，但直到19世纪才证明大于等于5次的一般代数方程式不能用代数公式求解，而对于超越方程就复杂的多，如果有解，其解可能是一个或几个，也可能是无穷多个。一般也不存在根的解析表达式。因此需要研究数值方法求得满足一定精度要求的根的近似解。

# 二、算法及原理

本文设计的有三种算法如下



## 1、二分法

原理：若*f*∈C[*a*, *b*]，且*f*(*a*)·*f*(*b*)<0，则*f*在(*a*, *b*)上至少有一实根。

基本思想：逐步将区间分半，通过判别区间端点函数值的符号，进一步搜索有根区间，将有根区间缩小到充分小，从而求出满足给定精度的根的近似值。

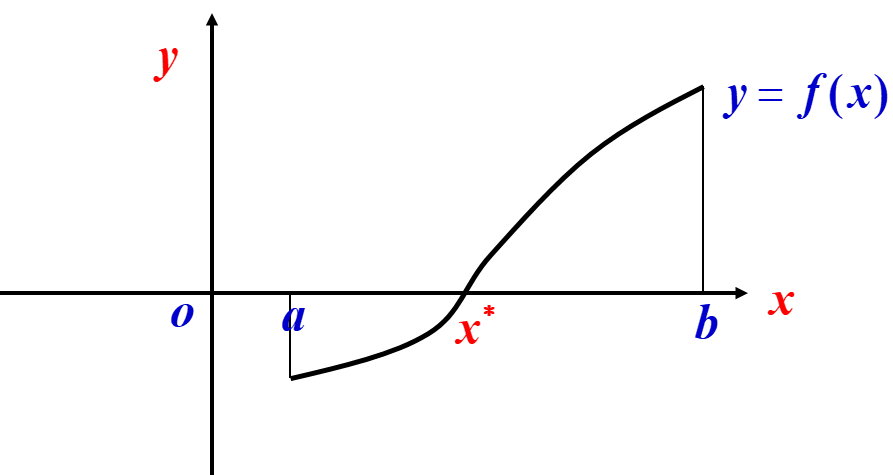


图1 二分法几何意义

二分法的算法描述：

给定区间[*a*, *b*] ，求*f*(*x*)=0 在该区间上的根*x*.

输入: *a*和*b*; 容许误差 *TOL*; 最大对分次数 *N max* .

输出: 近似根 *x*.

*Step 1* Set *k* = 1;

Step 2 Compute x=f((a + b)/2);

*Step 3* While ( *k* ≤ *N max*) do steps 4-6

Step 4 If |x| < TOL , STOP; Output the solution x.

Step 5 If x\*f(a)<0 , Set b=x;

*Else* Set *a=x*;

Step 6 Set k=k+1; Compute x=f((a + b)/2);Go To Step 3 ;

*Step 7* Output the solution of equation: *x*; STOP.

## 2、一般迭代法

原理：根据原函数，构造迭代形式，若迭代函数收敛，则通过若干次迭代可得到的近似解。

基本思想：迭代法是一种逐步逼近的方法，首先给出一个粗糙的初始值点，利用同一个迭代公式，反复计算，最终能得到一个较为准确的近似解。

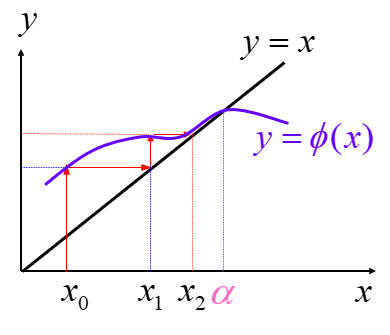


图2 一般迭代法几何意义

算法描述：

给定初始近似值 *x*0 ，求*x* = *g*(*x*) 的解.

输入: 初始近似值 *x*0; 容许误差 *TOL*;

最大迭代次数 *Nmax*.

输出: 近似解 *x* 或失败信息.

*Step 1* Set *i* = 1;

*Step 2* While ( *i* ≤ *Nmax*) do steps 3-6

*Step 3* Set *x* = *g*(*x*0); /\* 计算 *xi* \*/

*Step 4* If | *x* − *x*0 | < *TOL* then Output (*x*); /\*成功\*/

STOP;

*Step 5* Set *i* ++;

*Step 6* Set *x*0 = *x* ; /\* 更新 *x*0 \*/

*Step 7* Output (The method failed after *Nmax* iterations); /\*不成功 \*/

STOP.

## 3、牛顿迭代法

原理：若令一般迭代法中的，则为牛顿迭代法，通过若干次迭代可得到的近似解。

基本思想：基于一般迭代法，对迭代函数进行优化，使迭代过程加速。

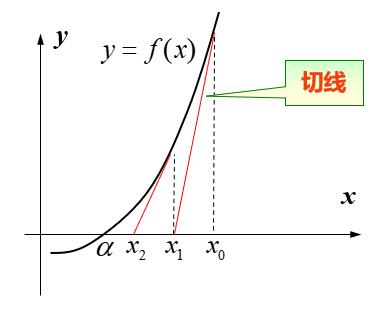
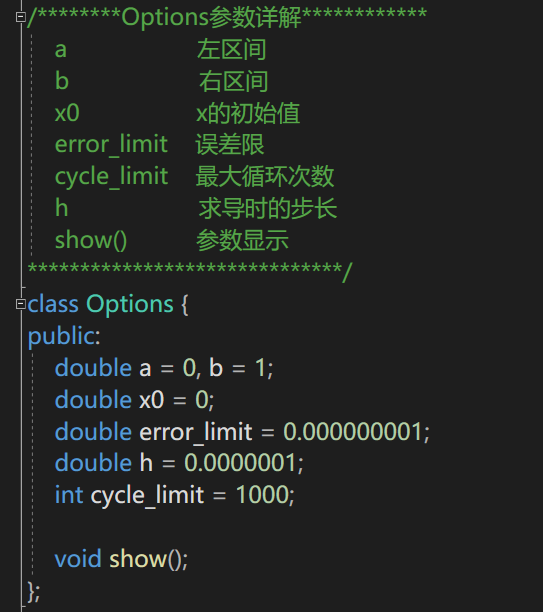


图3 牛顿迭代发几何意义

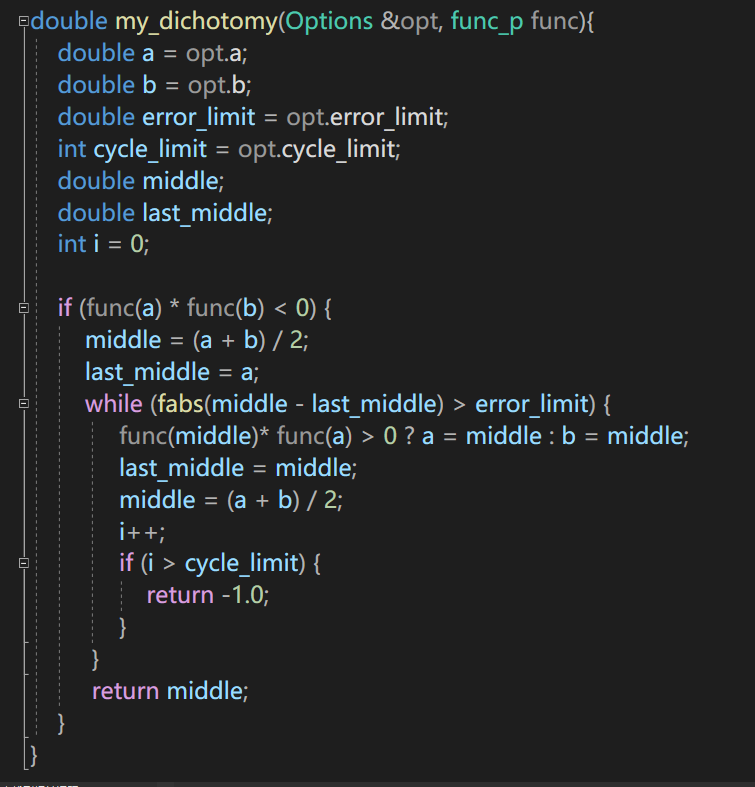
算法描述：同一般迭代法，这里不再赘述。

# 三、代码实现

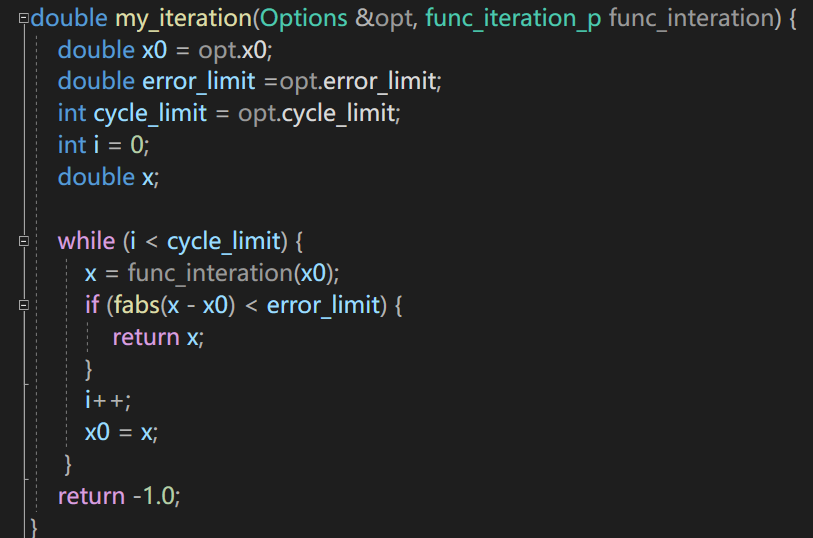
## 1、参数类Options



## 2、二分法实现



## 3、一般迭代法实现



## 4、牛顿迭代法实现

