现代数值计算方法

第七章 非线性方程迭代解法

第七章 非线性方程迭代解法

在工程计算和科学研究中, 如电路和电力系统计算、非线性力学、非线性积分和微分方程等多领域都要遇到非线性方程的求根问题. 本章主要讨论求解一元非线性方程

$$f(x) = 0 \tag{7.1}$$

的数值方法, 其中 $f(x)$ 是连续的非线性函数. 而方程按 $f(x)$ 是多项式或超越函数又分别称为代数方程和超越方程. 例如, 代数方程

$$x^4 - 8x^3 + 26x^2 - 43x + 17 = 0,$$

超越方程

$$\sin\frac{\pi x}{2} - \mathrm{e}^{-x} = 0.$$

Back
Close

已经证明, 对于 5 次及 5 次以上的一元多项式方程不存在精确的求根公式, 至于超越方程就更难求其精确解了. 鉴于此, 如何求得满足一定精度的方程的近似根, 已成为广大科研工作者迫切需要解决的问题.

本章的目的就是介绍求解非线性方程的数值方法. 主要介绍二分法, 迭代法及其加速方法, 牛顿型算法, 并讨论算法的收敛性、收敛速度和计算效率等问题.

§7.1 根的搜索与二分法

§7.1.1 隔根区间

在用近似方法求方程的根时, 需要知道方程的根所在的区间. 如果在区间 $[a,b]$ 内只有方程 $f(x)=0$ 的一个根, 则称区间 $[a,b]$ 为隔根区间. 通常可以用逐步扫描法来寻找方程 $f(x)=0$ 的隔根区间. 算法的基本思想是: 先确定方程 $f(x)=0$ 的所有实根所在区间 $[a,b]$, 再

按选定的步长 $h=(b-a)/n$ (n 为正整数), 逐点计算 $x_k=a+kh$ 处的函数值 $f(x_k)\,(k=0,1,\cdots,n)$, 当 $f(x_k)$ 与 $f(x_{k+1})$ 的值异号时, 则 $[x_k,x_{k+1}]$ 即为方程 $f(x)=0$ 的一个隔根区间.

算法 7.1 (逐步搜索法)

步 1 输入 a,b,h;

步 2 置 $a1:=a,\ b1:=a+h$;

步 3 while $(b1<b)$ (循环开始)

if $f(a1)\cdot f(b1)<0$ then

$x1(k):=a1,\ x2(k):=b1;$

else

$a1:=b1;\ b1:=b1+h;$

continue; (返回到循环的入口)

end

$a1 := b1;\ b1 := b1 + h;$

$k := k + 1;$

end (循环结束)

步 4 输出有根区间 $[x1(k), x2(k)]$.

根据算法 7.1 编制 MATLAB 程序如下：

- 算法 7.1 的 MATLAB 程序

```
%masearch.m
function masearch(fun,a,b,h)
%用途: 搜索非线性方程f(x)=0的有根区间
%格式: masearch(fun,a,b,h)  fun为函数表达式,
% a, b为区间左右端点, h为搜索步长
n=(b-a)/h; x1=zeros(1,n); x2=zeros(1,n);
```

```
a1=a; b1=a1+h; k=1;
while(b1<b)
  if feval(fun,a1)*feval(fun,b1)<0
    x1(k)=a1; x2(k)=b1;
  else
    a1=b1; b1=a1+h; continue;
  end
  a1=b1; b1=a1+h;  k=k+1;
end
for i=1:k
  if x1(i)-x2(i)~=0  [x1(i),x2(i)]  end
end
```

例 7.1 试用逐步搜索法确定方程

$$f(x) = x^3 + x^2 - 3x - 3 = 0$$

的有根区间.

容易看出, 当 $|x| > 3$ 时, $f(x)$ 保持符号不变, 故其根必定全部落在区间 $[-3, 3]$ 内, 即可初步确定 $a = -3$, $b = 3$. 取步长 $h = 0.6$, 利用算法 7.1 的通用程序 masearch.m, 在 MATLAB 命令窗口执行:

```
>> masearch(inline('x^3+x^2-3*x-3'),-3,3,0.6)
```

得计算结果:

```
ans =
   -1.8000   -1.2000
ans =
```

```
    -1.2000   -0.6000

ans =

     1.2000    1.8000
```

即方程有三个根, 分别在区间 $[-1.8.-1.2]$, $[-1.2,-0.6]$, $[1.2,1.8]$ 内.

§7.1.2 二分法及其程序实现

二分法的基本思想是通过计算隔根区间的中点, 逐步将隔根区间缩小,从而可得到方程的近似根数列. 具体地说, 设 $f(x)$ 为连续函数, 又设方程的隔根区间为 $[a,b]$, 即 $f(a)f(b)<0$. 记 $a_0:=a$, $b_0:=b$, 取其中点 $x_0=(a_0+b_0)/2$, 若 $f(a_0)f(x_0)<0$, 则去掉右半区间, 即令 $a_1:=a_0$, $b_1:=x_0$; 否则, 去掉左半区间, 即令 $a_1:=x_0$, $b_1=b_0$. 一般

地, 记当前有根区间为 $[a_k, b_k]$, 取

$$x_k = \frac{a_k + b_k}{2}, \tag{7.2}$$

若 $f(a_k)f(x_k) < 0$, 则令 $a_{k+1} := a_k$, $b_{k+1} := x_k$; 否则, 令 $a_{k+1} := x_k$, $b_{k+1} := b_k$; 再取 $x_{k+1} = (a_{k+1} + b_{k+1})/2$, 一直做下去, 直到满足精度为止.

算法 7.2 (二分法)

步 1 由算法 7.1 得到隔根区间 $[a, b]$, 设定精度要求 ε;

步 2 置 $x := (a + b)/2$;

步 3 若 $f(x) = 0$, 输出 x, 停算; 否则, 转步 4;

步 4 若 $f(a) \cdot f(x) < 0$, 则置 $b := x$; 否则, 置 $a := x$;

步 5 置 $x = (a + b)/2$, 若 $|b - a| < \varepsilon$, 输出 x, 停算; 否则, 转步 3.

根据算法 7.2 编制 MATLAB 通用程序如下：

- 二分法 MATLAB 程序

```
%mabisec.m
function x=mabisec(fun,a,b,ep)
%用途: 用二分法求方程f(x)=0有根区间[a,b]中的一个根
%格式: x=mabisec(fun,a,b,ep)  fun为函数表达式,
%       a,b为区间左右端点,ep为精度,x返回近似根
x=(a+b)/2.0;  k=0;
while abs(feval(fun,x))>ep|(b-a>ep)
   if feval(fun,x)*feval(fun,a)<0
      b=x;
   else
      a=x;
```

```
    end
    x=(a+b)/2.0;  k=k+1;
end
disp(['k=',num2str(k)])
```

例 7.2 用二分法程序 mabisec.m 求方程 $f(x) = xe^x - 1 = 0$ 在 $[0, 1]$ 内的一个实根. 取定精度 $\varepsilon = 10^{-5}$.

解 在 MATLAB 命令窗口执行:

```
>> x=mabisec(inline('x*exp(x)-1'),0,1,1e-5)
```

得计算结果:

```
k=16
x =
    0.56714630126953
```

即迭代 16 之后, 得到满足给定精度的近似根.

§7.1.3 二分法的收敛性分析

现在来估计由 (7.2) 式产生的点列 $\{x_k\}$ 与方程的根 x^* 之间的误差是多少. 根据二分法的基本思想, x_k 与 x^* 的误差不会超过区间 $[a_k, b_k]$ 长度的一半, 并注意到每一个小区间的长度是前一个区间长的一半, 因此有

$$|x_k - x^*| \leq \frac{b_k - a_k}{2} = \frac{b_{k-1} - a_{k-1}}{2^2} = \cdots = \frac{b_0 - a_0}{2^{k+1}},$$

即

$$|x_k - x^*| \leq \frac{1}{2^{k+1}}(b - a). \tag{7.3}$$

由上式可知, 当 $k \to \infty$ 时, 有 $|x_k - x^*| \to 0$, 即

$$\lim_{k\to\infty} x_k = x^*.$$

Back
Close

从以上推导过程可以看到, 序列 $\{x_k\}$ 的收敛性与初始区间 $[a,b]$ 无关. 对于任意给定的初始区间 $[a,b]$, 序列 $\{x_k\}$ 均是收敛的, 因此, 二分法是大范围收敛的.

例 7.3 用二分法求方程 $x^3-3x-1=0$ 在区间 $[1,2]$ 内的根, 使其精度达到两位有效数字. 问需要将区间二分多少次? 并求出满足精度的近似根.

解 根据 (7.3) 可以估计二分次数 k 的大小. 设

$$|x_k-x^*|\leq\frac{1}{2^{k+1}}(b-a)\leq\varepsilon,$$

其中 $a=1,\ b=2$, 精度 $\varepsilon=0.05$, 那么可求得 $k\geq(\ln 20/\ln 2)-1\approx 3.3219$, 取 $k=4$ 即可. 用公式 (7.2) 求解得 $x^*\approx x_4=1.9063$, 具体过程见下表:

k	a_k	b_k	x_k	$b_k - a_k$	$f(a_k)f(x_k)$
0	1	2	1.5	1	+
1	1.5	2	1.75	0.5	+
2	1.75	2	1.875	0.25	+
3	1.875	2	1.9375	0.125	−
4	1.875	1.9375	1.90625	0.0625	

根据上述讨论, 二分法具有计算简单, 方法可靠并且有大范围收敛性的优点. 缺点是收敛缓慢 (只有线性收敛速度), 并且不能求重根和复根. 通常用二分法为其它的求根方法 (例如牛顿法) 提供较好的初始近似值, 再用其它的求根方法精确化.

作业： P167: 7.1.