第二章 非线性方程的数值解法

科学工程中的许多现实问题常常涉及求解非线性方程

$$f(x) = 0, \quad x \in [a, b] \tag{1}$$

一般而言，非线性方程的精确求解是非常困难的，例如对于高于 4 次的代数方程，理论分析已证明其不存在精确求根公式。为此，我们必须借助于数值算法来求解各种非线性方程，本节将介绍一些常用的数值算法。为保证方程 (1) 在求解区间$[a, b]$ 内存在实根，以下我们恒设

$$f(x) \in \mathbb{C}([a, b]), \quad f(a)\, f(b) < 0,$$

且方程 (1) 在$[a, b]$ 内有唯一实根x^*，此时区间$[a, b]$ 称为方程的根的一个*隔离区间*。

§2.1 二分法

在本节，我们介绍非线性标量方程(1)的二分法,其基本计算思路如下：

Step 1. 把区间$[a,b]$ 二等分，记等分节点

$$a_0 = a, \quad x_0 = \frac{a+b}{2}, \quad b_0 = b.$$

计算x_0 处的函数值$f(x_0)$，若$f(x_0) = 0$, 则取$x^* = \frac{a+b}{2}$；否则，转至 Step 2;

Step 2. 若

$$f(a)f(\frac{a+b}{2}) < 0,$$

则记$a_1 = a, b_1 = \frac{a+b}{2}$; 若

$$f(\frac{a+b}{2})f(b) < 0,$$

则记$a_1 = \frac{a+b}{2}, b_1 = b$。取近似根

$$x_1 = \frac{a_1+b_1}{2} \in [a_1, b_1];$$

Step 3. 重复上述步骤，得根x^* 的下列隔离区间序列

$$[a_0, b_0] \supset [a_1, b_1] \supset [a_2, b_2] \supset \cdots \supset [a_k, b_k] \supset \cdots .$$

这里，二分k 次后隔离区间$[a_k, b_k]$ 的长度为

$$b_k - a_k = \frac{1}{2^k}(b - a),$$

近似根$x_k = \frac{a_k+b_k}{2} \in [a_k, b_k]$，其有误差估计

$$|x^* - x_k| \leq \frac{b_k - a_k}{2} = \frac{b - a}{2^{k+1}}.$$

因此，若要近似根x_k 达到预定精度ε：$|x^* - x_k| < \varepsilon$, 只需$\frac{b-a}{2^{k+1}} < \varepsilon$, 即当

$$k > \frac{ln(b-a) - ln(2\varepsilon)}{ln2}$$

时，可终止迭代，取$x_k = \frac{a_k+b_k}{2}$ 作为欲求近似根。

算法 2.1　二分法

```
function  x=half(a,b,tol)
c=(a+b)/2;  k=1;
m=1+round((log(b-a)-log(2*tol))/log(2));
while  k<=m
   if  f(c)==0
      c=c;
   return;
   elseif  f(a)*f(c)<0
      b=(a+b)/2;
   else
      a=(a+b)/2;
   end
 c=(a+b)/2;  k=k+1;
end
k
c
```

二分方法是方程求根问题的一种直接搜索方法，其优点是算法简单直观，数值解的精度易于判别。该算法的局限性是仅适用于标量方程，且事先要确定方程的根的一个隔离区间，当隔离区间较长时其计算速度较慢。

例 2.1 用二分法求方程$x = 4\sin x$ 在区间$[\frac{\pi}{2}, \pi]$ 内的根，要求绝对误差小于10^{-8}.

解 记$f(x) = x - 4\sin x$, 则$f(x) \in \mathbb{C}([\frac{\pi}{2}, \pi])$，且其满足

$$f(\frac{\pi}{2}) = \frac{\pi}{2} - 4 < 0, \quad f(\pi) = \pi > 0.$$

此外，$\forall x \in [\frac{\pi}{2}, \pi]$ 有

$$f'(x) = 1 - 4\cos x > 0,$$

即$f(x)$ 在区间$[\frac{\pi}{2}, \pi]$ 上单调递增。因此，原方程在区间$[\frac{\pi}{2}, \pi]$ 内有唯一根。
编制函数文件 **f.m**:

$$function \;\; y = f(x)$$
$$y = x - 4\sin x$$

运行 Matlab 命令$half(\pi/2, \pi, 10^{-8})$ 得满足精度要求的数值解

$$x = 2.47457678796451,$$

其需迭代 28 次。 ■

§2.2 Jacobi迭代法

§2.2.1 Jacobi迭代的基本原理

将非线性方程 (1) 等价地写成

$$x = \varphi(x), \tag{2}$$

其中$\varphi(x)$为连续函数。该方程的求根问题在几何上可视为求曲线$y = \varphi(x)$ 与直线$y = x$ 的交点P^* 的横坐标x^*。因此，我们可从这一几何观点出发来构造求解方程 (2) 的数值算法。

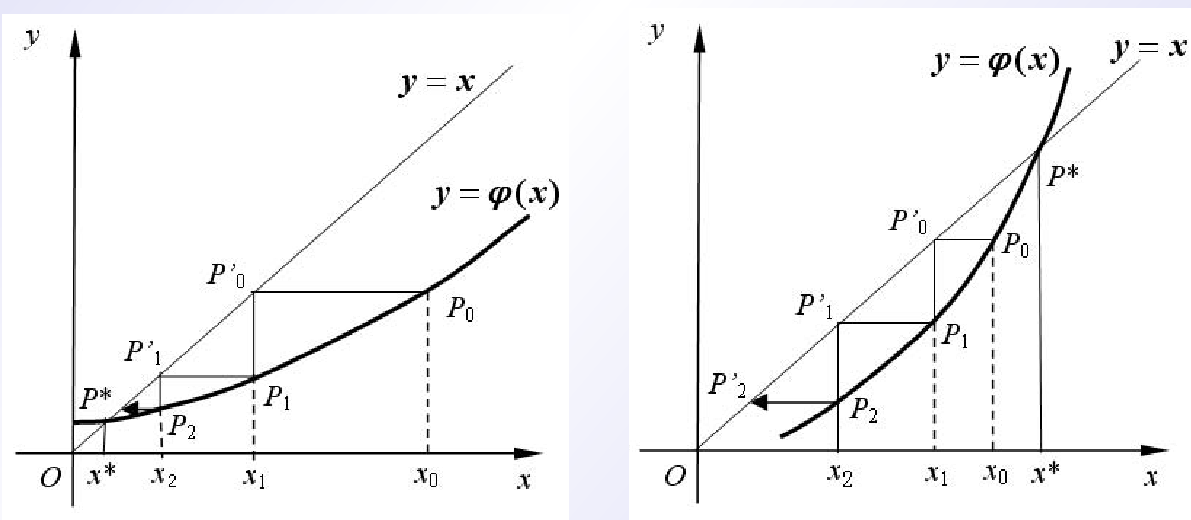

图 2.1. Jacobi迭代

访问主页
标题页
◀◀ ▶▶
◀ ▶
第 6 页 共 23 页
返回
全屏显示
关闭
退出

重复上述迭代过程，可得点列$P_1, P_2, \cdots, P_k, \cdots$（见图 2.1），其横坐标满足公式

$$x_{k+1} = \varphi(x_k), \quad k = 0, 1, 2, \cdots . \tag{3}$$

上迭代过程表明，若迭代函数φ 及初始逼近值x_0 选择恰当，则点列$\{P_k\}_{i=k}^{\infty}$逐步逼近P^*，即迭代序列x_k 收敛于x^*(见图 2.1 中左图)；否则，迭代过程发散（见图 2.1 中右图）。由公式 (2) 确定的方法称为Jacobi迭代法。

例 2.2 利用 Jacobi 迭代法求方程

$$x^3 - 2x^2 + x - 2 = 0$$

在 x=1.8 附近的近似根x_k，并使其满足$|x_k - x_{k-1}| < 10^{-8}$.

解 其方程可写成下列等价形式

$$x = \sqrt[3]{2x^2 - x + 2},$$

由此得 Jacobi 迭代公式

$$x_{k+1} = \sqrt[3]{2x_k^2 - x_k + 2}, \quad k = 0, 1, 2, \cdots, \ x_0 = 1.8.$$

根据该迭代公式，经 31 次迭代后，可得其方程在 x=1.8 附近且满足精度要求的近似根$x_{31} = 1.99999998890913$。

若取其方程的另一等价形式

$$x = -x^3 + 2x^2 + 2,$$

则有迭代格式

$$x_{k+1} = -x_k^3 + 2x_k^2 + 2, \quad k = 0, 1, 2, \cdots, \ x_0 = 1.8.$$

根据该迭代公式，经 10次迭代后，计算机发生溢出，无法获得满足题设精度要求的近似根。 ■

§2.2.2 迭代过程的收敛性

迭代过程只有在一定条件下才可能收敛。一个发散的过程是没有任何实际意义的。下面我们将给出两个迭代收敛的充分性定理。

定理2.1 设迭代函数$\varphi(x) \in \mathbb{C}([a,b])$，且满足下列条件
1) 对任意$x \in [a,b]$, 有

$$a \leq \varphi(x) \leq b. \tag{4}$$

2) 存在正数$L < 1$,使对任意$x \in [a,b]$, 有

$$|\varphi\prime(x)| \leq L < 1. \tag{5}$$

则迭代过程$x_{k+1} = \varphi(x_k)$对于任意初值$x_0 \in [a,b]$均收敛于方程$x = \varphi(x)$ 的根x^*，且有如下事后误差估计式

$$|x^* - x_k| \leq \frac{1}{1-L}|x_{k+1} - x_k|. \tag{6}$$

证 先证x^*的存在唯一性。作函数

$$g(x) = x - \varphi(x), \quad x \in [a,b].$$

显然，由已知条件$g(x) \in \mathbb{C}([a,b])$，且有

$$g(a) = a - \varphi(a) \le 0, \quad g(b) = b - \varphi(b) \ge 0,$$

则存在着$x^* \in [a,b]$使得$g(x^*) = 0$，即$x^* = \varphi(x^*)$。若另存在着$\tilde{x}^* \in [a,b]$使得$g(\tilde{x}^*) = 0$，则

$$|x^* - \tilde{x}^*| = |\varphi(x^*) - \varphi(\tilde{x}^*)| = |\varphi'(\xi)(x^* - \tilde{x}^*)| \le L|x^* - \tilde{x}^*|, \ \xi \in (a,b),$$

且因此有$L \ge 1$，矛盾！故方程$x = \varphi(x)$在$[a,b]$上必有唯一根。

下证收敛性。由微分中值定理

$$|x^* - x_k| = |\varphi(x^*) - \varphi(x_{k-1})| = |\varphi'(\xi)(x^* - x_{k-1})| \le L|x^* - x_{k-1}|,$$

式中ξ是x^*与x_{k-1}之间的一点。据此递推得

$$|x^* - x_k| \le L^k |x^* - x_0|.$$

因此，$\lim\limits_{k\to\infty} x_k = x^*$。

最后，我们证明误差估计式(6)。对任意正整数p有

$$
\begin{aligned}
&|x_{k+p}-x_k| \le \sum_{i=1}^{p}|x_{k+i}-x_{k+i-1}| = \sum_{i=1}^{p}|\varphi(x_{k+i-1})-\varphi(x_{k+i-2})| \\
= &\sum_{i=1}^{p}|\varphi'(\xi_i)(x_{k+i-1}-x_{k+i-2})| \ \ (\xi_i\in(a,b)) \\
\le &\sum_{i=1}^{p}L|x_{k+i-1}-x_{k+i-2}| \le \cdots\cdots \\
\le &\sum_{i=1}^{p}L^{i-1}|x_{k+1}-x_k| = \tfrac{1-L^p}{1-L}|x_{k+1}-x_k|
\end{aligned}
$$

在上式中固定k并令$p\to\infty$，则得

$$|x^*-x_k| \le \frac{1}{1-L}|x_{k+1}-x_k|. \quad \blacksquare$$

由此可见，在实际计算中，若用户给定的预定精度为ε：$|x^*-x_k|<\varepsilon$，则我们可用条件

$$|x_{k+1}-x_k|<\varepsilon$$

来控制迭代过程的终止。

一般说来，定理2.1中的条件在较大的有根区间上是很难保证的，为此我们通常在根x^*的附近考察其收敛性。

定义2.1 称迭代过程$x_{k+1} = \varphi(x_k)$在根x^*的附近具有局部收敛性，若存在x^*的邻域$\triangle : |x - x^*| \leq \delta$，使当$x_0 \in \triangle$时，迭代序列$\{x_k\}$收敛于$x^*$。

定理2.2 设$\varphi(x) \in \mathbb{C}^1([x^* - \delta, x^* + \delta])\ (\delta > 0)$，且$|\varphi'(x^*)| < 1$，则迭代过程$x_{k+1} = \varphi(x_k)$具有局部收敛性。

例2.3 求方程

$$x = e^{-x}$$

在$x = 0.5$附近的一个根，预定精度$\varepsilon = 10^{-5}$.

解 作函数$g(x) = x - e^{-x}$，其满足

$$g(0.5) \approx -0.1065 < 0, \quad g(0.6) \approx 0.0512 > 0.$$

因此，方程在区间$[0.5, 0.6]$内有根，且

$$|\varphi'(x)| \leq \max_{0.5 \leq x \leq 0.6} |(e^{-x})'| = \exp(-0.5) < 1, \ \ x \in [0.5, 0.6].$$

据定理2.1，迭代公式$x_{k+1} = e^{-x_k}$对于初值$x_0 = 0.5$是收敛的。经过18次迭代后，得$x_{18} = 0.56714$，它满足所规定的精度要求。该方程取5位有效数字的准确根为$x^* = 0.56714$.

§2.2.3 迭代法的收敛速度

衡量一个算法的有效性，除考察它是否收敛外，还需研究它的收敛速度，所谓收敛速度即指数值解收敛于精确解的速度。

定义2.2 若数值解x_k逼近于精确解x^*的误差$e_k := |x^* - x_k|$满足

$$\lim_{k\to\infty} \frac{|e_{k+1}|}{|e_k|^p} = C, \quad C > 0 \text{为常数},$$

则称相应数值方法是p阶收敛的。特别，当$p = 1$，$0 < C < 1$时，称其为线性收敛；$p = 2$时，称其为平方收敛。

定理2.3 设迭代函数$\varphi(x)$在方程$x = \varphi(x)$的根x^*附近p阶连续可微，且

$$\varphi'(x^*) = \varphi''(x^*) = \cdots = \varphi^{(p-1)}(x^*) = 0, \quad \varphi^{(p)}(x^*) \neq 0,$$

则迭代过程$x_{k+1} = \varphi(x_k)$是p阶收敛的。

证 由于$\varphi'(x^*) = 0$, 则根据定理2.2，迭代过程$x_{k+1} = \varphi(x_k)$ 具有局部收敛性。将$\varphi(x_k)$在x^*处作泰勒展开

$$
\begin{aligned}
\varphi(x_k) &= \varphi(x^*) + \sum_{i=1}^{p-1} \frac{\varphi^{(i)}(x^*)}{i\,!}(x_k - x^*)^i + \frac{\varphi^{(p)}(\xi)}{p\,!}(x_k - x^*)^p \\
&= \varphi(x^*) + \frac{\varphi^{(p)}(\xi)}{p\,!}(x_k - x^*)^p, \quad \xi\text{是}x_k\text{与}x^*\text{之间的某点},
\end{aligned}
$$

即

$$\varphi(x_k) - \varphi(x^*) = \frac{\varphi^{(p)}(\xi)}{p\,!}(x_k - x^*)^p.$$

从而

$$|x_{k+1} - x^*| = \left|\frac{\varphi^{(p)}(\xi)}{p\,!}\right| |x_k - x^*|^p.$$

因此

$$\lim_{k\to\infty} \frac{|e_{k+1}|}{|e_k|^p} = \lim_{k\to\infty} \left|\frac{\varphi^{(p)}(\xi)}{p\,!}\right| = \left|\frac{\varphi^{(p)}(x^*)}{p\,!}\right| \neq 0.$$

故迭代过程$x_{k+1} = \varphi(x_k)$是p阶收敛的。

§2.3 加速迭代方法

§2.3.1 加速方法的构造

Jacobi迭代方法简单易行，但是其收敛速度却在实际计算较为缓慢。为此，我们有必要改进和加速其迭代过程。设x_k是第k次迭代逼近值，利用迭代公式计算得到第$k+1$次迭代逼近值，这里记作$\bar{x}_{k+1}$，即$\bar{x}_{k+1} = \varphi(x_k)$。假设$\varphi'(x)$在根$x^*$的附近变化不大，且其值约为$L$，则由微分中值定理得

$$x^* - \bar{x}_{k+1} = \varphi'(\xi)(x^* - x_k) \approx L(x^* - x_k).$$

因此，我们有事后误差估计式

$$x^* - \bar{x}_{k+1} \approx \frac{L}{1-L}(\bar{x}_{k+1} - x_k), \tag{7}$$

并由此得迭代加速公式

$$x_{k+1} = \frac{1}{1-L}[\varphi(x_k) - Lx_k]. \tag{8}$$

例2.4 利用加速迭代公式求方程

$$x = e^{-x}$$

在$x = 0.5$附近的一个根，预定精度$\varepsilon = 10^{-5}$.

解 取

$$L = \frac{d[\exp(-x)]}{dx}\Big|_{x=0.5} = -\exp(-x)\Big|_{x=0.5} = -0.6065$$

得加速迭代公式

$$x_{k+1} = \frac{1}{1-L}\Big[\exp(-x_k) - Lx_k\Big], \quad k = 0, 1, 2, \cdots, \ x_0 = 0.5.$$

应用该迭代公式，经 4次迭代后，可得其方程在 x=0.5 附近且满足精度要求的近似根$x_4 = 0.5671$。与例2.3中所用的Jacobi迭代相比较，加速迭代公式大大提高了迭代过程的收敛速度。 ■

§2.3.2 Aitken加速方法

例 2.4表明，加速迭代格式大大加快了 Jacobi 迭代方法的收敛速度，但该方法的缺陷是需要确定L值，而这在实际问题的计算中往往是非常困难的。对于非线性方程 (2)，我们有一种克服该困难的方法，即所谓的Aitken加速迭代法。记

$$\tilde{x}_{k+1}=\varphi(x_k), \quad \hat{x}_{k+1}=\varphi(\tilde{x}_{k+1}),$$

则由 Taylor 展开定理近似地有

$$x^*-\tilde{x}_{k+1}\approx l(x^*-x_k), \quad x^*-\hat{x}_{k+1}\approx l(x^*-\tilde{x}_{k+1}),$$

由上二式消去未知常数l 得

$$x^*\approx \hat{x}_{k+1}-\frac{(\hat{x}_{k+1}-\tilde{x}_{k+1})^2}{\hat{x}_{k+1}-2\tilde{x}_{k+1}+x_k}.$$

故得Aitken 加速迭代公式

$$\begin{cases} \tilde{x}_{k+1}=\varphi(x_k), \\ \hat{x}_{k+1}=\varphi(\tilde{x}_{k+1}), \\ x_{k+1}=\hat{x}_{k+1}-\frac{(\hat{x}_{k+1}-\tilde{x}_{k+1})^2}{\hat{x}_{k+1}-2\tilde{x}_{k+1}+x_k}, \end{cases} \tag{9}$$

例2.5 利用 Aitken 加速迭代公式求方程

$$x^3 - 2x^2 + x - 2 = 0$$

在 x=1.8 附近的近似根x_k，并使其满足$|x_k - x_{k-1}| < 10^{-8}$.

解 取

$$x_0 = 1.8, \quad \varphi(x) = \sqrt[3]{2x^2 - x + 2},$$

应用 Aitken 加速迭代公式 (9)求解方程，经 3 次迭代后，可得其方程在 x=1.8 附近且满足精度要求的近似根$x_3 = 2.00000000000000$。

若另取

$$\varphi(x) = -x^3 + 2x^2 + 2,$$

应用 Aitken 加速迭代公式 (9) 求解方程，经 6 次迭代后，可得其方程在 x=1.8 附近且满足精度要求的近似根$x_6 = 2$。 ■

与例 2.2所用的Jacobi迭代相比较，Aitken迭代法不但大大加速了 Jacobi迭代公式的收敛速度，而且在某些情形下，Aitken 迭代方法可将一个发散的Jacobi迭代过程改造成一个收敛的迭代过程。

§2.4 Newton迭代法

为求解非线性方程 (1)，我们考虑如下迭代方法：在方程 (1) 的解的隔离区间$[a,b]$ 上选取适当迭代初值x_0，过曲线$y=f(x)$ 的点$(x_0,f(x_0))$ 引切线

$$l_1:\quad y=f(x_0)+f'(x_0)(x-x_0)$$

与x 轴相交于点

$$x_1=x_0-\frac{f(x_0)}{f'(x_0)};$$

进一步，过曲线$y=f(x)$ 的点$(x_1,f(x_1))$ 引切线

$$l_2:\quad y=f(x_1)+f'(x_1)(x-x_1)$$

与x 轴相交于点

$$x_2=x_1-\frac{f(x_1)}{f'(x_1)};$$

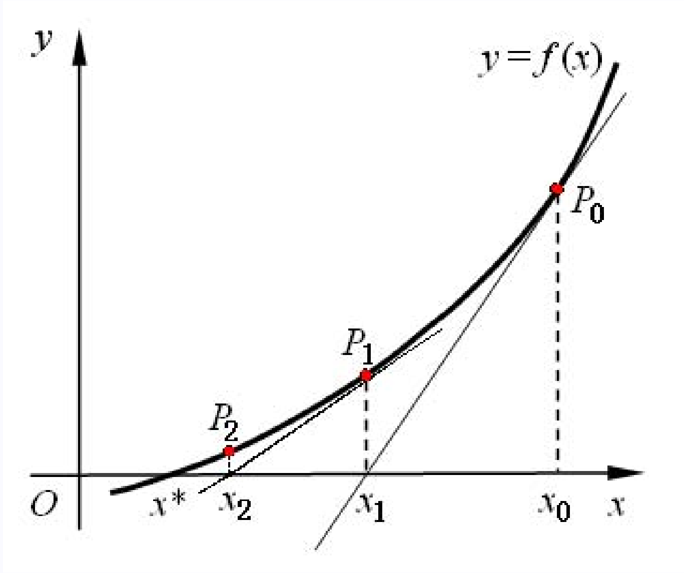

图 2.2. Newton迭代

如此循环往复，可得一列逼近x^* 的点$x_0, x_1, \cdots, x_k, \cdots$(见图 2.2)，其一般表达式为

$$x_k = x_{k-1} - \frac{f(x_{k-1})}{f'(x_{k-1})}, \quad k = 1, 2, \cdots \tag{10}$$

该公式即称为 Newton 迭代公式，其相应求解方法称为Newton 迭代法或切线法。

例 2.6 用 Newton 迭代法求方程$x = 4\sin x$ 在区间$[\frac{\pi}{2}, \pi]$ 内的根，并使其满足$|x_k - x_{k-1}| < 10^{-8}$。

解 记$f(x) = x - 4\sin x$,，并取$x_0 = \frac{\pi}{2}$。应用 Newton 迭代法 (10) 于方程$x = 4\sin x$，经 7 次迭代后，可得满足精度要求的数值解$x_7 = 2.47457678736983$。 ■

将例 2.1 与例 2.6 相比较，显见 Newton 迭代法的收敛速度要远远快于二分法的收敛速度。Newton迭代可以看成是关于方程

$$x = \varphi(x), \quad \varphi(x) := x - \frac{f(x)}{f'(x)}$$

的迭代公式。如果x^*为方程$f(x) = 0$的一个单根，则有$f(x^*) = 0$, $f'(x^*) \neq 0$，从而

$$\varphi'(x^*) = \frac{f(x^*) f''(x^*)}{[f'(x^*)]^2} = 0$$

由定理2.3可知，Newton迭代在根x^*的邻近至少是平方收敛的。

由前可知，Newton 迭代法具较快的收敛速度。但 Newton 迭代法有一个不足之处，即其要求选择一个恰当的迭代初值x_0，方可保证迭代过程快速收敛。为尽量避免因初值x_0 选择不当而导致迭代过程缓慢收敛或发散，我们在 Newton 迭代公式中引入一个下山因子λ：$0 < \lambda \leq 1$，从而产生下述Newton 下山法

$$x_{k+1} = x_k - \lambda [f'(x_k)]^{-1} f(x_k), \quad k = 0, 1, 2, \cdots . \tag{11}$$

在实际计算中，λ 可依次取为$1, \frac{1}{2}, \frac{1}{2^2}, \cdots, \frac{1}{2^k}, \cdots$，并且采用双精度控制：

$$\|x_{k+1} - x_k\|_2 < \varepsilon_1 \text{ 或 } \|f(x_k)\| < \varepsilon_2,$$

这里ε_1，ε_2 为预定精度。