第5章 方程求根
Zeros and Roots

苏 芮
srhello@zju.edu.cn
开物苑4-202

上节课知识回顾

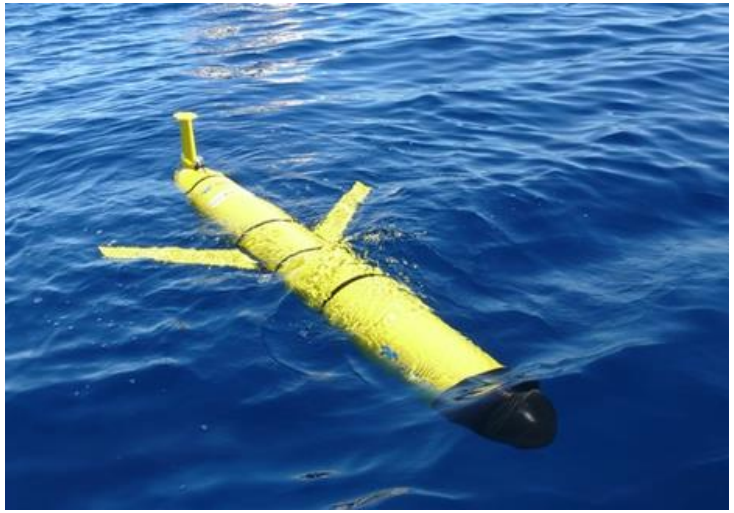
1. 最小二乘法

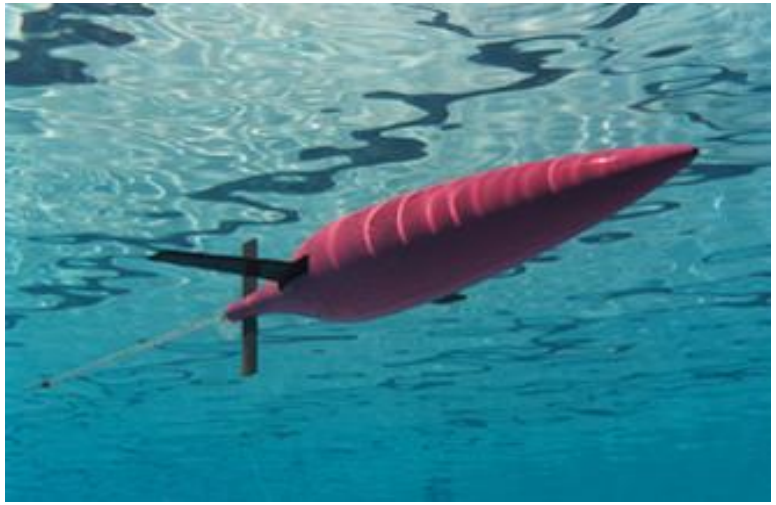
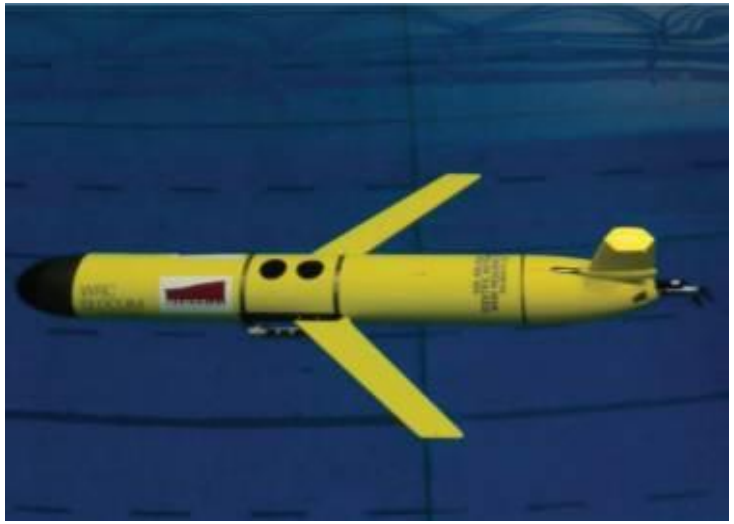
- 与插值法的区别？适用范围？
- 为什么用最小二乘（2范数）？
- 如何构建法方程组？（基本思路是什么？）
- 最小二乘直线拟合？最小二乘指数拟合？非线性最小二乘指数拟合？
- QR分解法

2. 大作业选题讲解

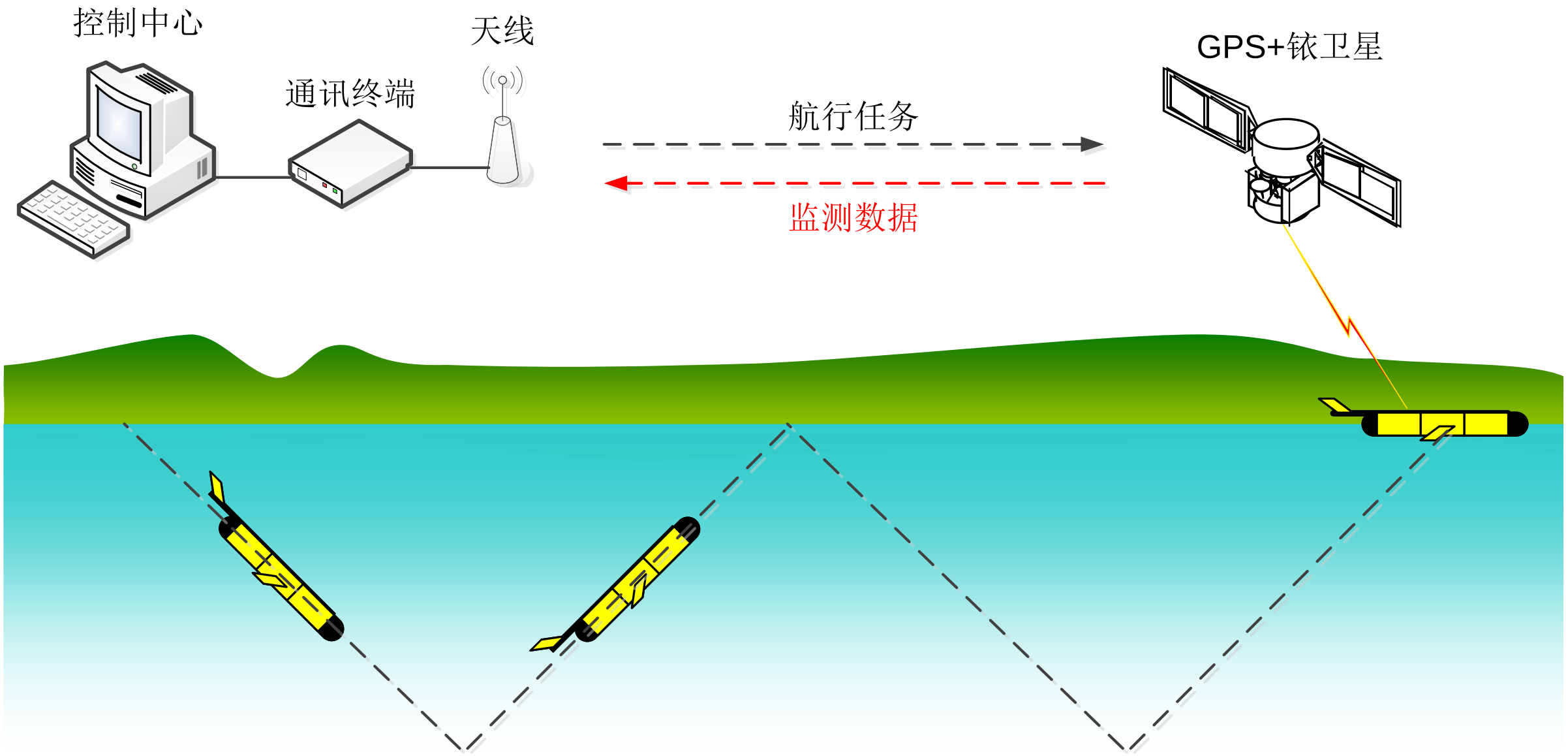
鼓励结合国防军工选题，要和课程有相关性，要初步符合科研论文格式（绪论、研究内容、研究方法、实验验证，结果分析等，参考机械工程学报或本科毕设模板）

水下滑翔机



水下滑翔机



水下滑翔机几何特征与水动力特性

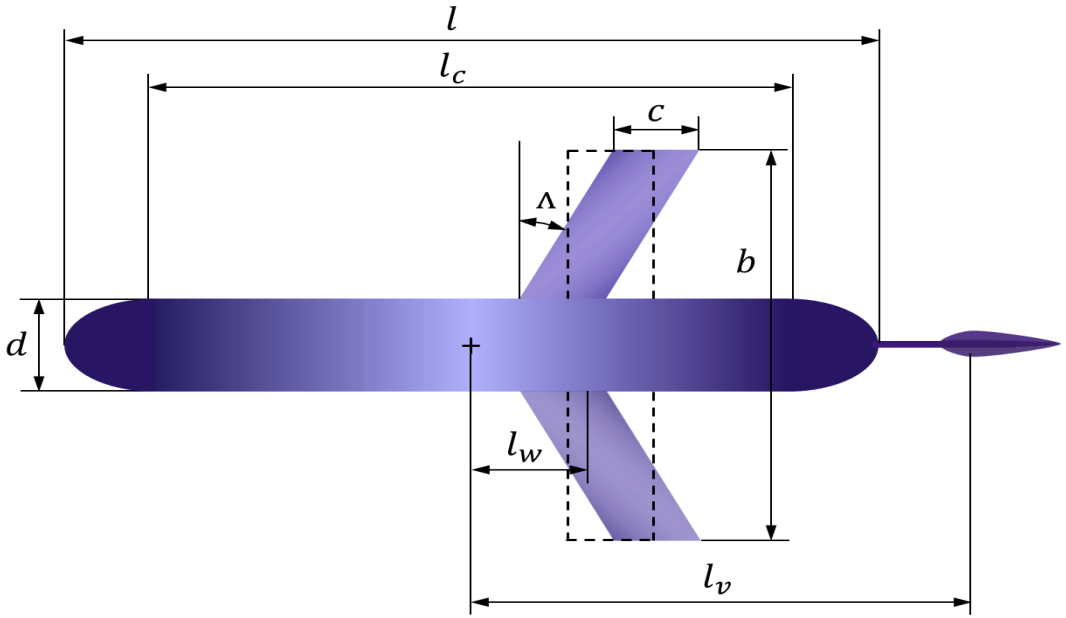
- **水下滑翔机关键几何特征的参数化表达**

机身细长比： $f = \dfrac{l_c}{d},$

机翼长宽比： $AR = \dfrac{b^2}{S},$

机翼翼展比： $\kappa = \dfrac{b}{d},$

机翼厚度比： $\tilde{t} = \dfrac{t}{\bar{c}}$



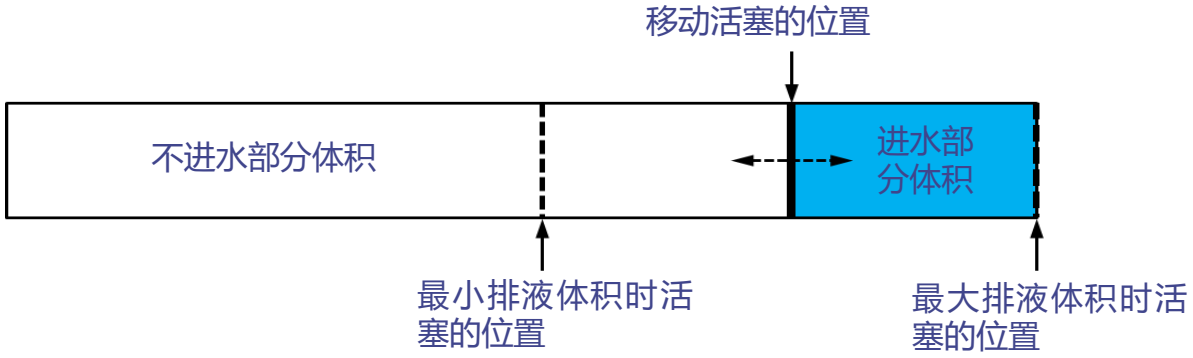
机翼相对位置： $\bar{l}_w = l_w / l_c$ **尾翼相对位置：** $\bar{l}_v = l_v / l_c$ **尾翼相对面积：** $S_v = S_v / S_f$

浮力肺能力： $\mathrm{V} = (1+\bar{\eta})\mathrm{V}_{\mathrm{NB}},\quad \bar{\eta} = \mathrm{V}/\mathrm{V}_{\mathrm{NB}} - 1$

$$\tilde{W} = W - B = \rho g \mathrm{V}_{\mathrm{NB}} - \rho g(1+\eta)\mathrm{V}_{\mathrm{NB}}$$

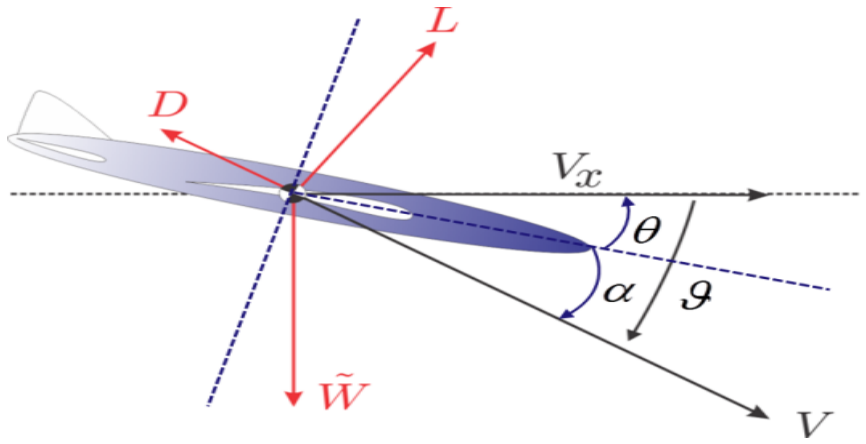
$$= -\eta \rho g \mathrm{V}_{\mathrm{NB}} = -\left(\frac{\eta}{1+\bar{\eta}}\right)\rho g \mathrm{V}$$

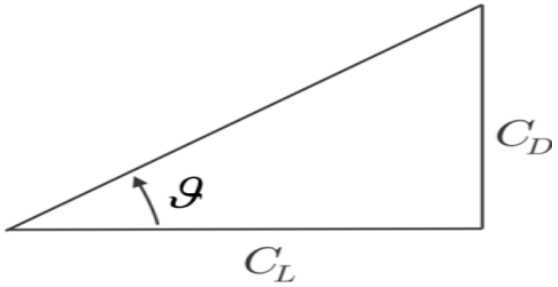
$$\eta \in [-\bar{\eta}, \bar{\eta}]$$



水下滑翔机运动性能分析

- 纵剖面锯齿形稳态运动
  - 纵剖面锯齿形稳态运动受力分析





速度方向受力平衡：

$$C_D\left(\frac{1}{2}\rho V^2 S\right)=\tilde{W}\sin\vartheta$$

$$V=\sqrt{\frac{2\tilde{W}\sin\vartheta}{\rho S C_D}}$$

$$V_x=V\cos\vartheta$$

$$V_z=V\sin\vartheta$$

$$\sin\vartheta=\frac{C_D}{\sqrt{C_D^2+C_L^2}}\qquad\cos\vartheta=\frac{C_L}{\sqrt{C_D^2+C_L^2}}$$

水下滑翔机运动性能分析

- **两种特殊的滑翔状态**

最大水平速度滑翔状态：

$$V_x = \sqrt{\frac{2\tilde{W}}{\rho S}} \frac{C_L}{[C_L^2 + C_D^2]^{3/4}}$$

$$\max\left(\frac{C_L}{[C_L^2 + C_D^2]^{3/4}}\right) \Rightarrow$$

$$\sigma = 2K_d C_{D_0}$$

$$C_{L_{\max V_x}} = \frac{\sqrt{2}}{4K_d}\sqrt{-(1+\sigma)+\sqrt{(1+\sigma)^2+8\sigma^2}}$$

$$C_{D_{\max V_x}} = C_{D_0} + \frac{1}{8K_d}\left[-(1+\sigma)+\sqrt{(1+\sigma)^2+8\sigma^2}\right]$$

$$\tan\vartheta_{\max V_x} = \frac{C_{D_{\max V_x}}}{C_{L_{\max V_x}}},\ \vartheta_{\mathrm{md}} \to \arctan(1/\sqrt{2}) \approx 35.5^\circ$$

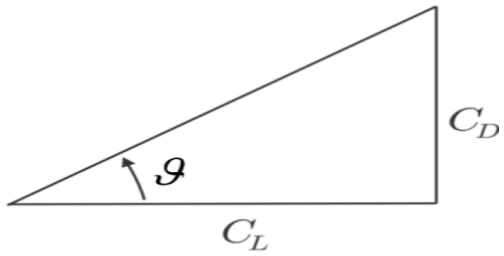
最小阻力滑翔状态：最小滑翔角状态，产生最大的水平航行范围

$$\tan\vartheta = \frac{C_D}{C_L} = \frac{C_{D_0} + K_d C_L^2}{C_L}$$

$$\max\left(\frac{C_{D_0} + K_d C_L^2}{C_L}\right) \Rightarrow$$

$$C_{D_{\mathrm{md}}} = 2C_{D_0},\ C_{L_{\mathrm{md}}} = \sqrt{\frac{C_{D_0}}{K_d}}$$

$$\tan\vartheta_{\mathrm{md}} = \frac{C_{D_{\mathrm{md}}}}{C_{L_{\mathrm{md}}}},\ \vartheta_{\mathrm{md}} \to \arctan\left(2\sqrt{C_{D_0}K_d}\right) = 7^\circ$$

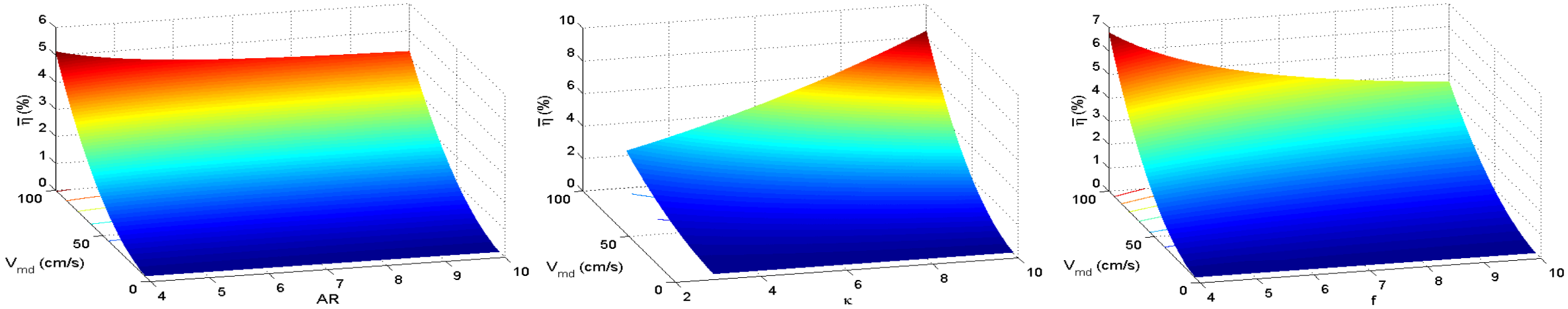


水下滑翔机运动性能分析

- **水下滑翔机几何特征对纵剖面内运动性能的影响**
  - 对浮力肺能力的影响：

$$\bar{\eta}_{\mathrm{md}} = \frac{1}{\Psi_{\mathrm{md}} - 1} \quad , \quad \Psi_{\mathrm{md}} = \frac{1}{\mathrm{Re}_{l_{\mathrm{md}}}^2} \frac{\pi}{2} \frac{g l_c^{\,3}}{\nu^2} \frac{AR}{\kappa^2} \frac{1}{\sqrt{C_{D_{\mathrm{md}}}^2 + C_{L_{\mathrm{md}}}^2}}$$

$$C_{D_{\mathrm{md}}} = 2C_{D_0} \ , \ C_{L_{\mathrm{md}}} = \sqrt{\frac{C_{D_0}}{K_d}} \qquad AR = 6.5,\ \kappa = 6,\ f = 7,\ \tilde{t} = 0.12$$



结论：　对于给定的最小阻力速度，要获得越小的浮力肺能力，增大机翼的长宽比，减小机翼的翼展比，并增大机身的细长比。

方程求根

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。**

方程求根

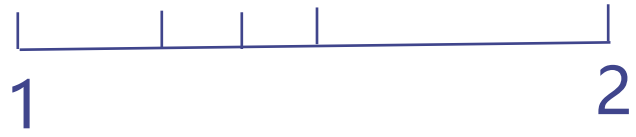
方程数值求解基本方法：

(1)找到有根区间(Find the area for the root)

(2)不断接近准确值(Approach)

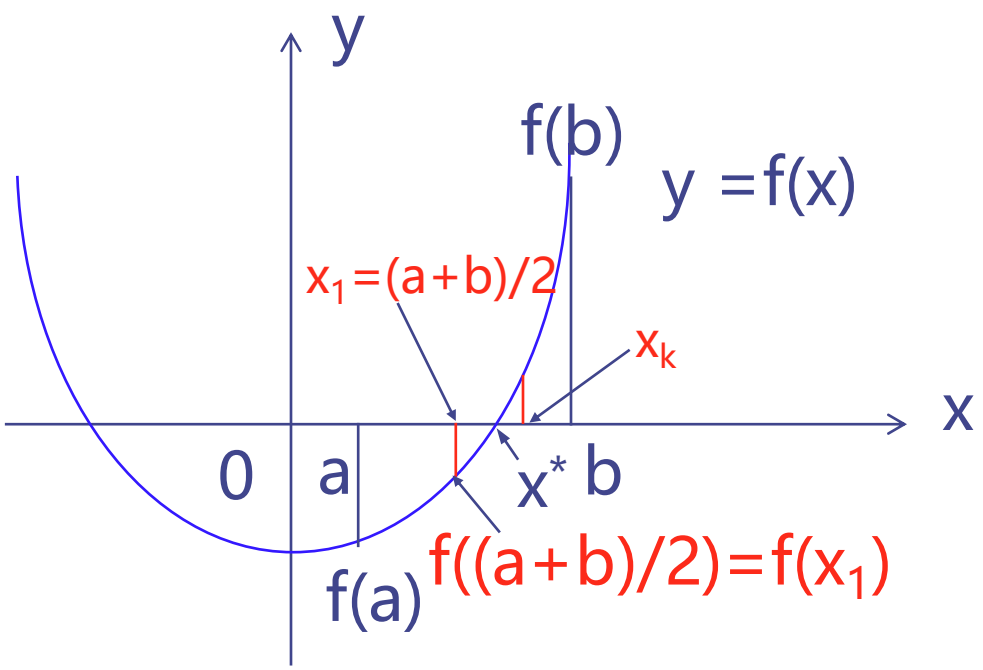
二分法

$$x^2 - 2 = 0 \qquad \sqrt{2}$$



$$1\frac{1}{2},\ 1\frac{1}{4},\ 1\frac{3}{8},\ 1\frac{5}{16},\ 1\frac{13}{32},\ 1\frac{27}{64},\ \ldots.$$

二分法



$$[a,b],[a_1,b_1],[a_2,b_2],\cdots,[a_k,b_k]$$

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

$$\frac{b-a}{2^{k+1}}\leqslant\varepsilon$$

若取$[a_k,b_k]$的中点 $x_k=\frac{a_k+b_k}{2}$作为所求根的近似值 $\{x_k\}:x_0,x_1,x_2,\cdots,x_k,\cdots$

所谓二分法，是使用对分区间的方法，保留有根区间，舍去无根区间，并且如此不断地对分下去，以逐步逼近方程根的方程求解方法

二分法

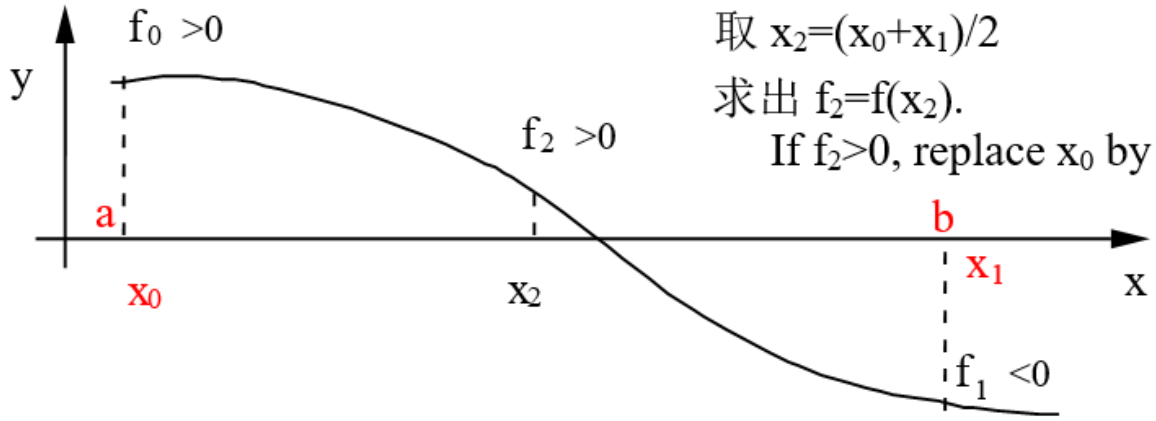
设定 $x_0=a$, $x_1=b$; 使得 $f_0*f_1<0$

假设: $f_0>0$, $f_1<0$.

取 $x_2=(x_0+x_1)/2$

求出 $f_2=f(x_2)$.

If $f_2>0$, replace x_0 by x_2



继续取 $x_2=(x_0+x_1)/2$.

判断: is $|f_2| <$ tolerance?

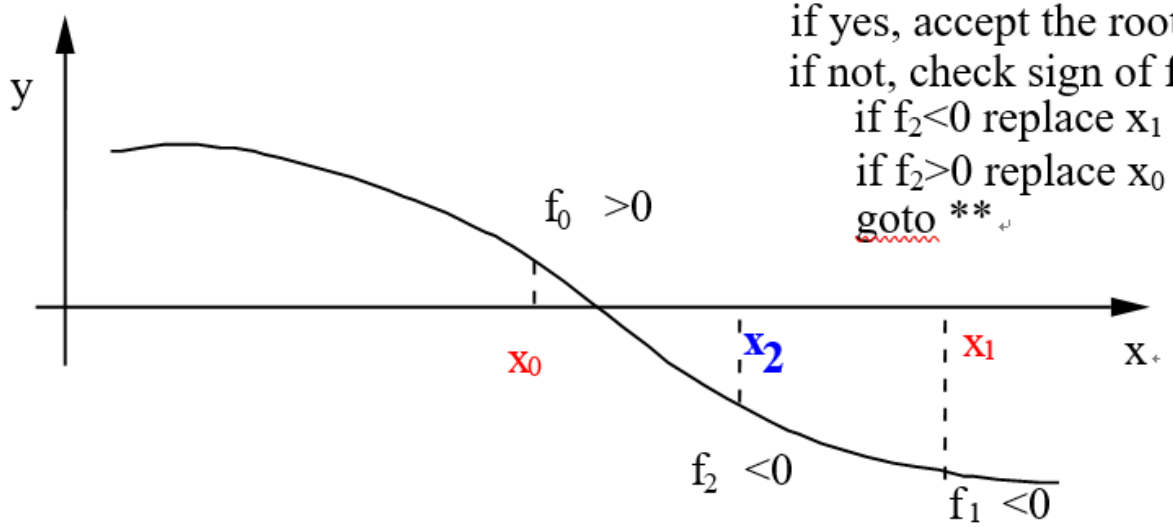
if yes, accept the root as $x=x_2$.

if not, check sign of f_2

if $f_2<0$ replace x_1 by x_2,

if $f_2>0$ replace x_0 by x_2

goto **

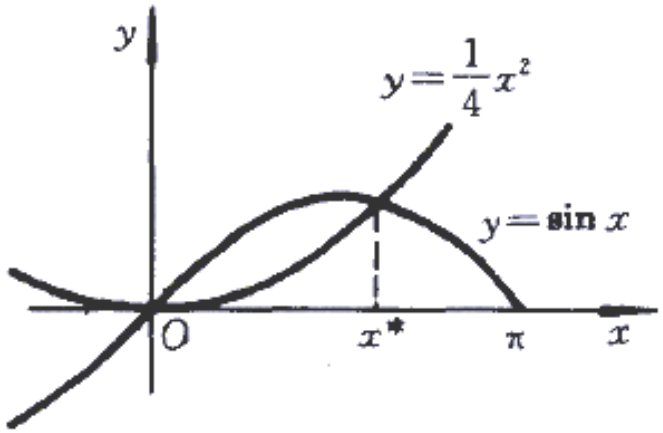


二分法

```
k = 0;
while abs(b-a) > eps*abs(b)
    x = (a + b)/2;
    if sign(f(x)) == sign(f(b))
        b = x;
    else
        a = x;
    end
    k = k + 1;
end
```

二分法

例 1 用二分法求方程 $f(x)=\sin x-\frac{x^2}{4}=0$ 的非零实根的近似值，使误差不超过 10^{-2}.



$x^* \in [1.5, 2]$.

k	a_k	b_k	x_k	$f(x_k)$
0	1.5	2	1.75	0.218361
1	1.75	2	1.875	0.0751795
2	1.875	2	1.9375	−0.00496228
3	1.875	1.9375	1.90265	0.0404208
4	1.90625	1.9375	1.921875	0.156014
5	1.921875	1.9375	1.9296875	0.00536340

二分法应用

荷兰的克里查罗(V.V.Kritchallo)的 分形作品《霜》(*Rime*)

分形理论

分形理论是当今世界十分风靡和活跃的新理论、新学科。分形的概念是美籍数学家曼德布罗特(B.B.Mandelbort)首先提出的。1967年他在美国权威的《科学》杂志上发表了题为《英国的海岸线有多长?》的著名论文。

自相似原则和迭代生成原则是分形理论的重要原则。它表征分形在通常的几何变换下具有不变性,即标度无关性。由自相似性是从不同尺度的对称出发,也就意味着递归。分形形体中的自相似性可以是完全相同,也可以是统计意义上的相似。标准的自相似分形是数学上的抽象,迭代生成无限精细的结构,如科契(Koch)雪花曲线、谢尔宾斯基(Sierpinski)地毯曲线等。这种有规分形只是少数,绝大部分分形是统计意义上的无规分形。

一种谢宾斯基三角构造方法

背景

谢宾斯基三角（Sierpiński triangle）是一种分形集合，具有自相似的特性。整体形状像一个正三角形，并且每条边被二等分，使得三角形被细分成更小的等边三角形。

构造方法之一：混沌游戏（Chaos game）

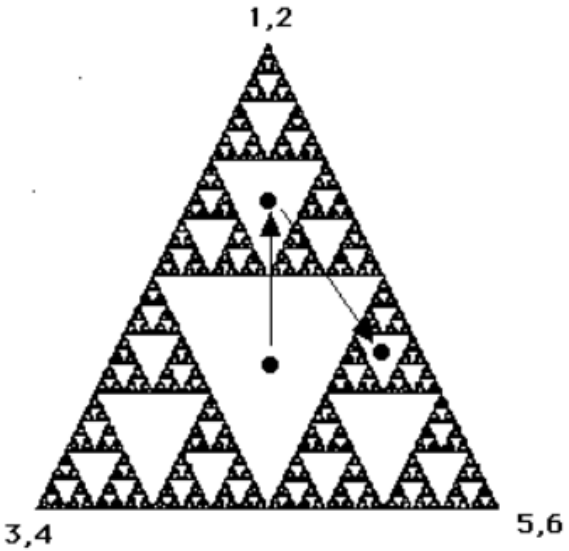
首先固定一个最外圈的大三角形，方便起见我们将三个顶点取为：$P_1(0,0)$，$P_2(1,0)$，$P_3\left(\frac{1}{2},\frac{\sqrt{3}}{2}\right)$初始点 v_0 取为三角形内（或边上）任意一点，比如（0,0）。

之后利用递推式

$v_{n+1} = 1/2 * (v_n + P_{r_n})$，$r_n$ 是从$[1,2,3]$等概率随机取的数。

绘制 从$v0$ 到v_∞ 的点。如果第一个点 v_0 是 三角上的一个点，那么所有的点 v_n都在谢宾斯基三角形上。如果位于最外圈三角形内的第一个点 v_0 不是谢尔宾斯基三角形上的一个点，那么任何一个点 v_n都不会位于谢尔宾斯基三角形上，但是它们会在三角形上收敛。

为什么Chaos Game可以构造出谢宾斯基三角？如图可以看出，比如起始点在中心，经过一步步的迭代，每次都会进入到更小的三角形的中心，而永远无法落到三角形的边上。经过足够多的迭代后，这些落点就将空白三角形的“轮廓”勾勒了出来。



代码实现

根据以上信息，容易推得：

$P_1(0,0)$

$P_2(1,0)$

$P_3\left(\frac{1}{2},\frac{\sqrt{3}}{2}\right)$

1. 选择P1 $x_{n+1} = \frac{1}{2} * x_n$ ， $y_{n+1} = \frac{1}{2} * y_n$

2. 选择P2 $x_{n+1} = \frac{1}{2} * (x_n + 1)$ ， $y_{n+1} = \frac{1}{2} * y_n$

3. 选择P3 $x_{n+1} = \frac{1}{2} * (x_n + \frac{1}{2})$ ， $y_{n+1} = \frac{1}{2} * (y_n + \frac{\sqrt{3}}{2})$

可以将以上三个式子写成：

$$v_{n+1} = Av_n + b,\ \ v_n = \begin{bmatrix} x_n \\ y_n \end{bmatrix}$$

$$A = \begin{bmatrix} 1/2 & 0 \\ 0 & 1/2 \end{bmatrix}$$

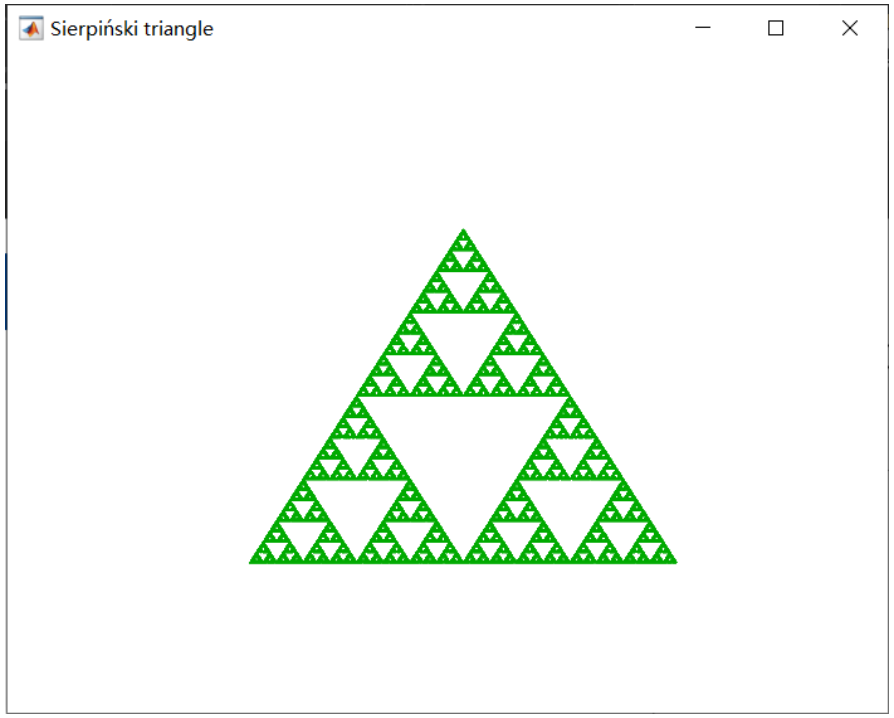
等概率选取$b1, b2, b3$作为b即可。

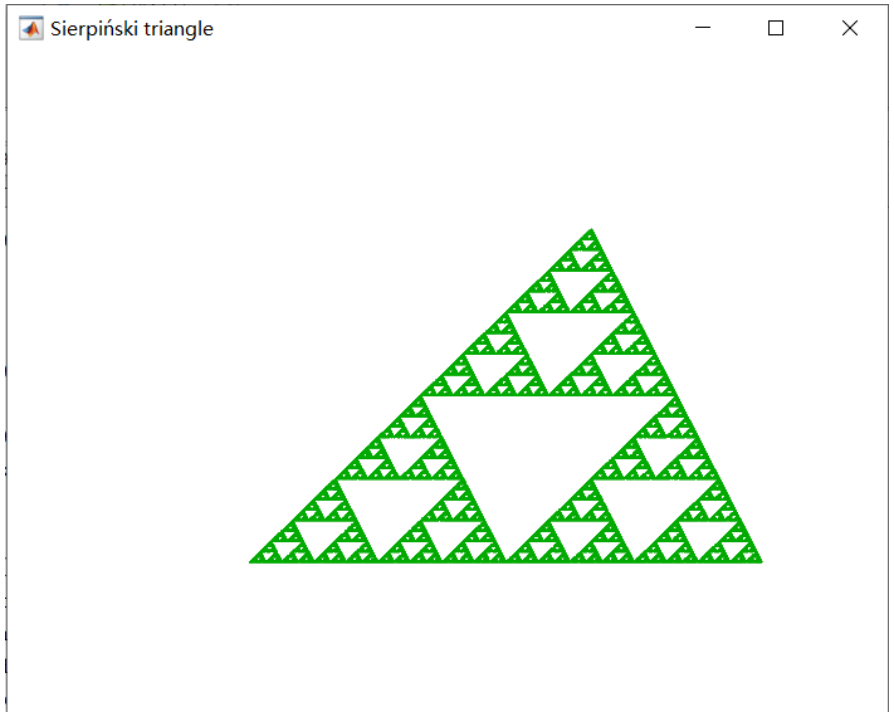
$$b1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix},\ \ b2 = \begin{bmatrix} 1/2 \\ 0 \end{bmatrix},\ b3 = \begin{bmatrix} 1/4 \\ \sqrt{3}/4 \end{bmatrix}$$

```
p  = [ 1/3 2/3 1.00];
A = [ .5  0; 0  .5];
b1 = [0; 0];
b2 = [.5; 0];
b3 = [.25; sqrt(3)/4];

x = [0; 0];
xs = zeros(2,n);
xs(:,1) = x;
for j = 2:n
   r = rand;
   if r < p(1)
      x = A*x + b1;
   elseif r < p(2)
      x = A*x + b2;
   else
      x = A*x + b3;
   end
```

最后，实际上最外圈的三角形并不一定是等边三角形，可以通过改变b的值来形成不同形状的三角形，也可以改变缩放的比例，即改变A。还可以拓展成四边形、五边形等。





分形网络制造

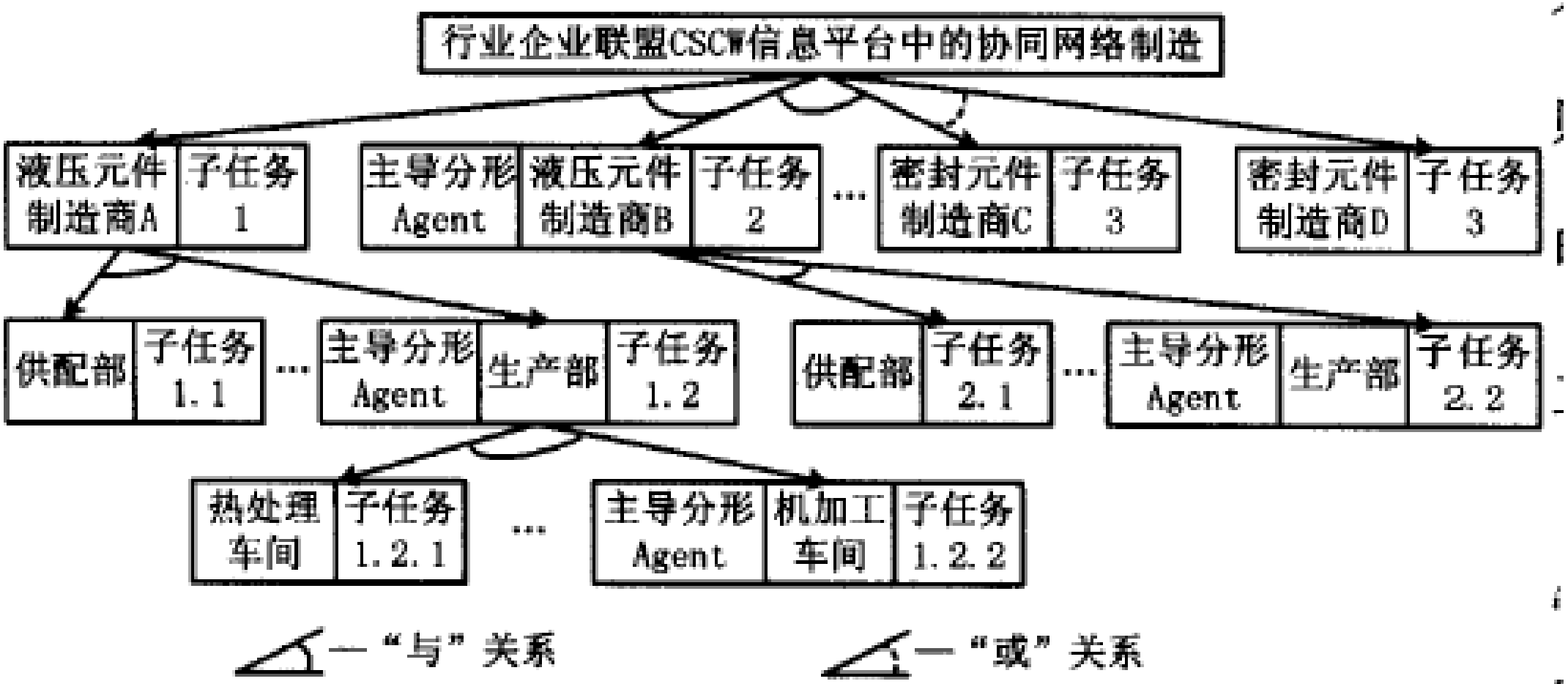


图1　分形网络协同制造的目标结构树

牛顿法

求解非线性方程 $f(x)=0$

是否可以将它转换成线性方程进行求解？如何转化？

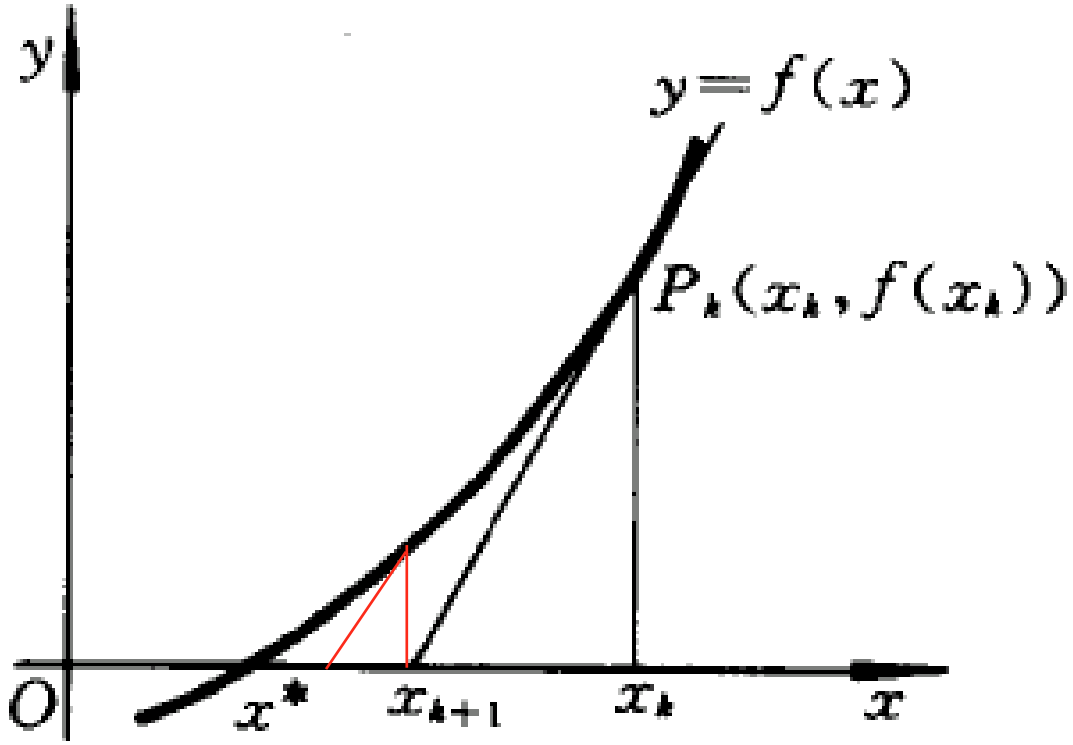
把函数 $f(x)$在 x_k 处泰勒展开

$$f(x_k)+f'(x_k)(x-x_k)=0$$

$$x_{k+1}=x_k-\frac{f(x_k)}{f'(x_k)} \qquad (k=0,1,2,\cdots)$$

牛顿法

几何特征



牛顿法

```
k = 0;
while abs(x - xprev) > eps*abs(x)
    xprev = x;
    x = x - f(x)/fprime(x)
    k = k + 1;
end
```

牛顿法

应用示例

例 4 用牛顿迭代法求方程 $x-\cos x=0$ 的实根，要求准确到 $|x_{k+1}-x_k|<10^{-5}$.

在 $[0,\frac{\pi}{2}]$ 上　当 $x_0=1$

$x_0\in[0,\frac{\pi}{2}]$　且 $f(x_0)f''(x_0)>0$

$$x_{k+1}=x_k-\frac{x_k-\cos x_k}{1+\sin x_k}\qquad(k=0,1,2,\cdots)$$

表 2-3

k	x_k
0	1
1	0.750364
2	0.739113
3	0.739086
4	0.739085

牛顿法

推论： 求平方根的牛顿迭代法 $\sqrt{M}$

$$f(x) = x^2 - M$$

$$x_{n+1} = x_n - \frac{x_n^{\,2} - M}{2x_n}$$

$$x_{n+1} = \frac{x_n + \frac{M}{x_n}}{2}$$

```
while abs(x - xprev) > eps*abs(x)
    xprev = x;
    x = 0.5*(x + M/x);
end
```

牛顿法

推论： 求平方根的牛顿迭代法 $\sqrt{M}$

$$x_{n+1} = \frac{x_n + \dfrac{M}{x_n}}{2}$$

```
while abs(x - xprev) > eps*abs(x)
    xprev = x;
    x = 0.5*(x + M/x);
end
```

$\sqrt{2}$ $x_0 = 1$

```
1.50000000000000
1.41666666666667
1.41421568627451
1.41421356237469
1.41421356237309
1.41421356237309
```

牛顿法

求：$\sqrt{5}$

$$p_k = \frac{p_{k-1} + \dfrac{A}{p_{k-1}}}{2} \qquad p_0 = 2$$

$$p_1 = \frac{2 + 5/2}{2} = 2.25$$

$$p_2 = \frac{2.25 + 5/2.25}{2} = 2.236111111$$

$$p_3 = \frac{2.236111111 + 5/2.236111111}{2} = 2.236067978$$

$$P_4 = \frac{2.36067978 + 5/2.236067978}{2} = 2.236067978$$

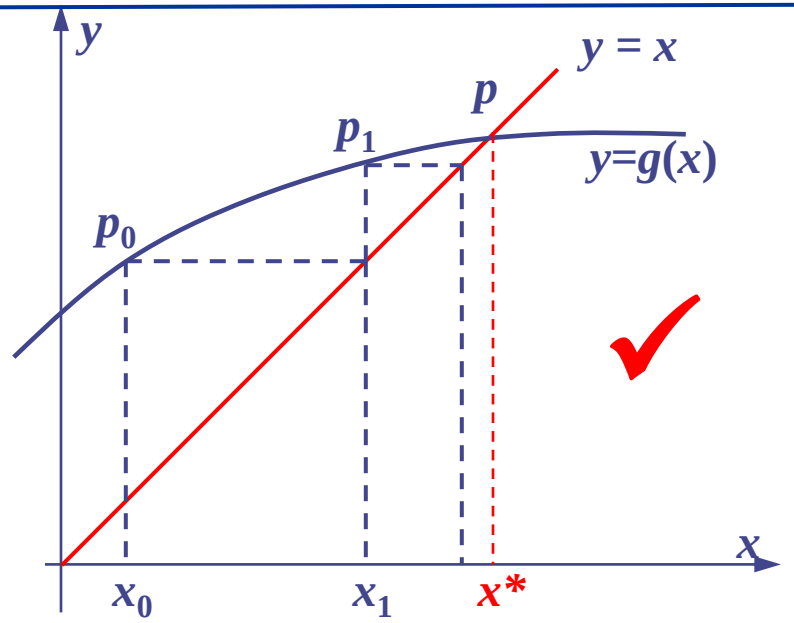
牛顿法

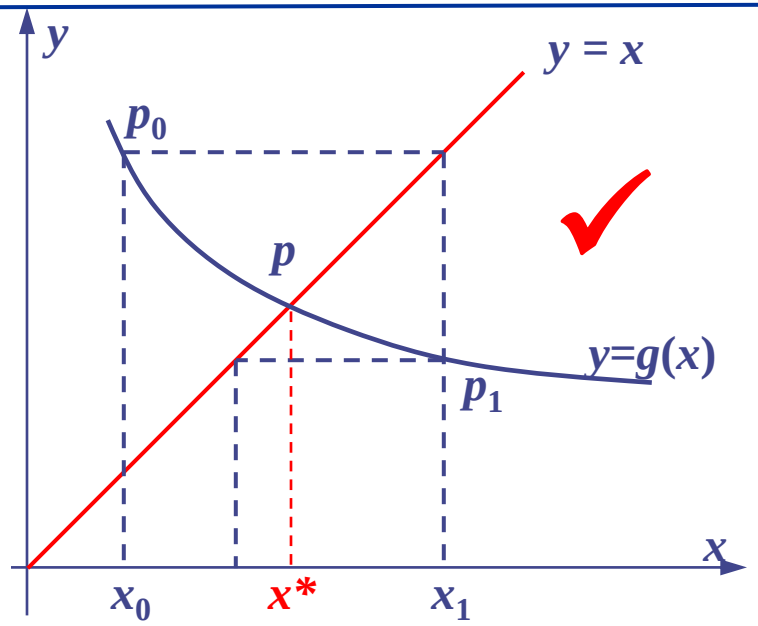
收敛性判断?

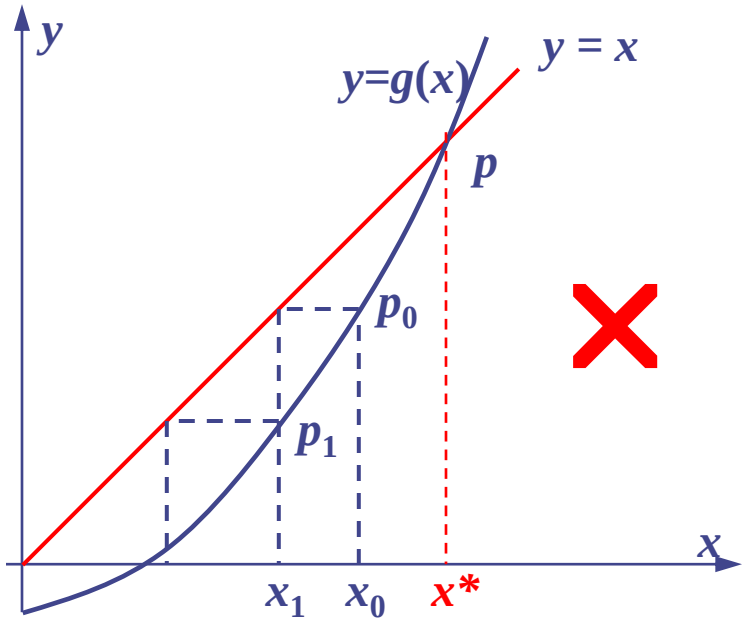
$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \qquad (k = 0, 1, 2, \cdots)$$

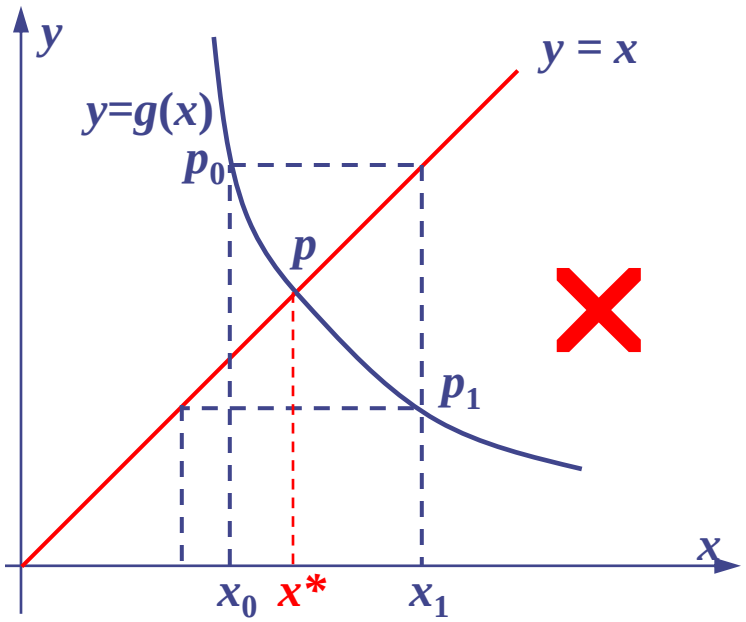
$$x_{k+1} = g(x_k)$$

牛顿法


y
y = x
p
p1
y=g(x)
p0
x
x0
x1
x*


y
y = x
p0
p
y=g(x)
p1
x
x0
x*
x1


y
y=g(x)
y = x
p
p0
p1
x
x1
x0
x*


y
y=g(x)
y = x
p0
p
p1
x
x0
x*
x1

牛顿法

牛顿迭代法的收敛性判断

$$\left|g'(x_k)\right|<1$$

定理 3 对于方程 $f(x)=0$，若存在区间 (a,b)，使

(1)在 (a,b) 内存在方程的单根 x^*；

(2)$f''(x)$ 在 (a,b) 内连续.

则牛顿迭代法在 x^* 附近具有局部收敛性.

牛顿法

定理3证明

$$g(x)=x-\frac{f(x)}{f'(x)} \qquad g'(x)=\frac{f(x)f''(x)}{[f'(x)]^2}$$

$f''(x)$在(a,b)内连续

$g'(x)$在(a,b)内连续

对于任意给定的正数ε，必存在δ，使得$g'(x)$在x^*邻域（$x^*-\delta$，$x^*+\delta$)内

$|g'(x)-g'(x^*)|<\varepsilon$，取$\varepsilon=1$，得$|g'(x)-g'(x^*)|<1$，即$|g'(x)|<1$

$$|g'(x)|<1$$

上节课知识回顾

1. 方程求根

- 与方程组求根的区别?
- 如何用数值法进行方程求根?
  （1. 确定有根区间；2. 不断逼近真实解）
- 二分法的基本思路?
- 牛顿迭代法的基本思路？为什么比二分法收敛的快？如何判断牛顿迭代法是否会收敛?

牛顿法

误差估计? $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$

$$0 = f(x^*) = f(x_n) + f'(x_n)(x^* - x_n) + \frac{1}{2} f''(\xi)\,(x^* - x_n)^2$$

$$x_n - x^* = \frac{f(x_n)}{f'(x_n)} + \frac{\frac{1}{2} f''(\xi)\,(x^* - x_n)^2}{f'(x_n)}$$

$$e_{n+1} = x_{n+1} - x^* = x_n - \frac{f(x_n)}{f'(x_n)} - x^* = x_n - x^* - \frac{f(x_n)}{f'(x_n)} = \frac{f(x_n)}{f'(x_n)} + \frac{\frac{1}{2} f''(\xi)\,(x^* - x_n)^2}{f'(x_n)} - \frac{f(x_n)}{f'(x_n)}$$

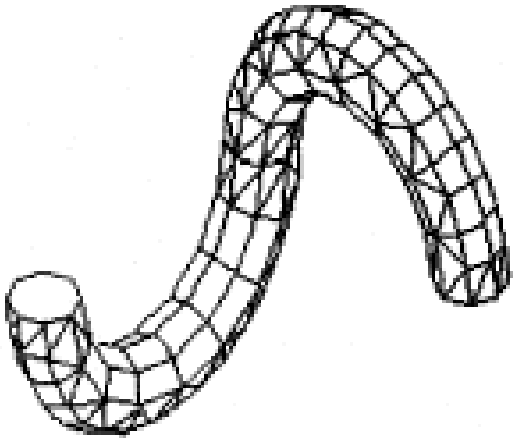
$$e_{n+1} = \frac{1}{2}\frac{f''(\xi)}{f'(x_n)} e_n^2$$

牛顿法

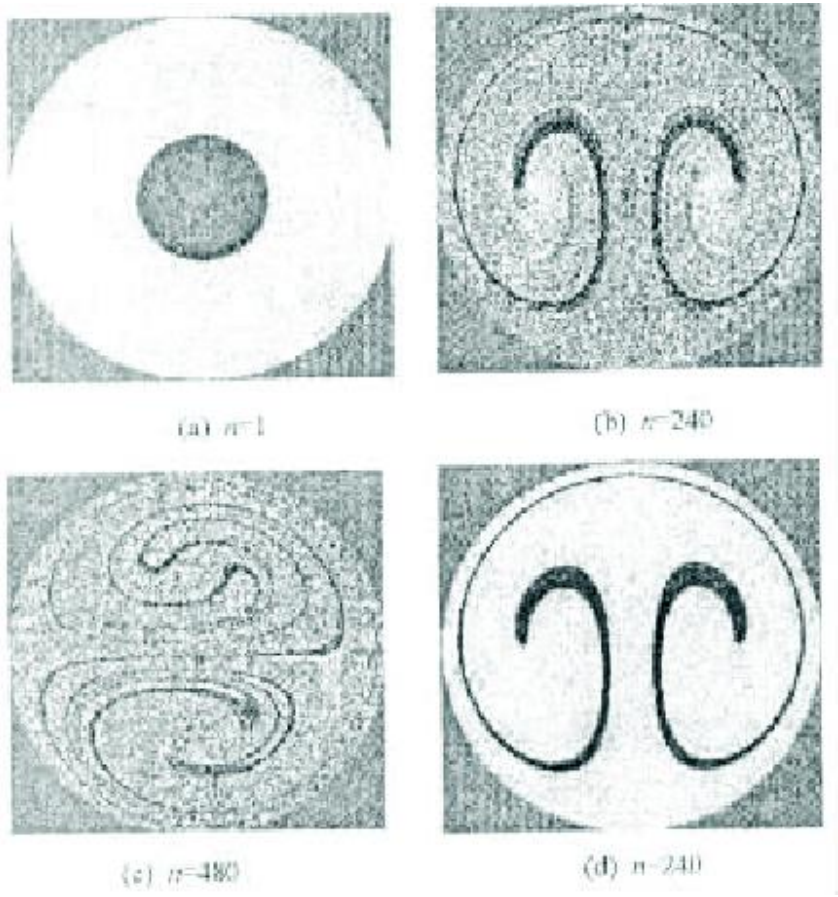
牛顿法解方程之混沌情况

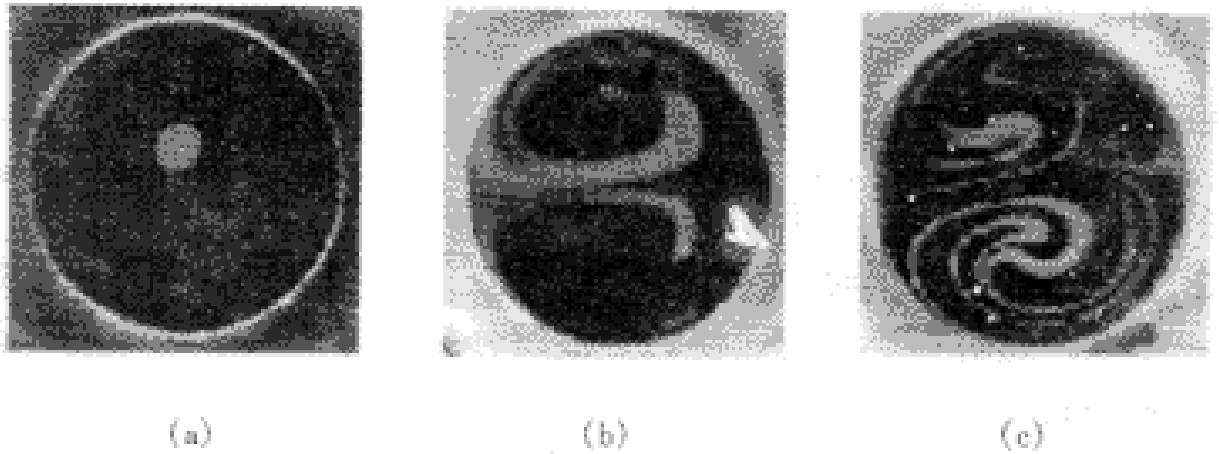
牛顿法解复方程专题：对复方程f(z) =0 , f(z) 为复多项式函数，设函数 g(z) = z - f(z) / f ‘(z) ,其中 f’ (z) 为函数f(z)的导函数。则函 g(Z) 就是复多项式方程求解的牛顿迭代公式。对于选定的起始点， g(z) 迭代大多都会收敛于多项式f(z) = 0 的某个根，但也可能存在许多点，使 g(z) 迭代根本就不收敛，甚至可能出现混沌的状态。

程序中的函数为 f(z) = z^n - 1 , n>= 2， 也就是求解方程 z^n = 1 的根。



混沌
混合





牛顿法

牛顿迭代法收敛性

定理 4 对方程 $f(x)=0$，若存在区间$[a,b]$，使

(1)$f''(x)$在$[a,b]$上连续；

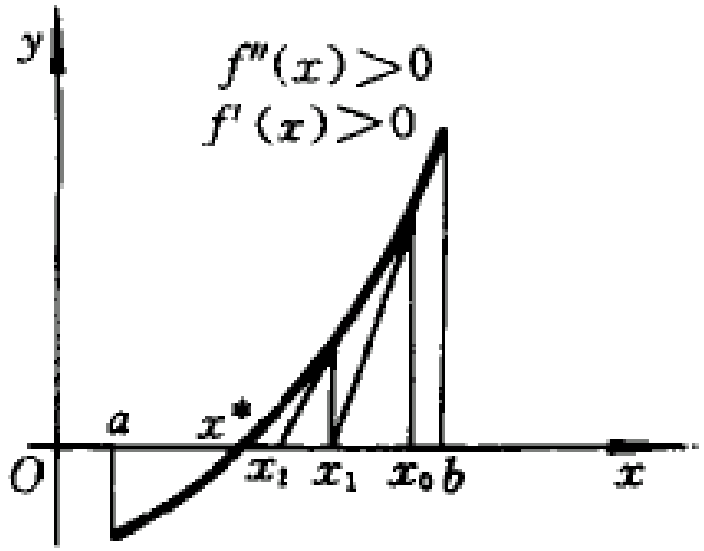
(2)$f(a)f(b)<0$；

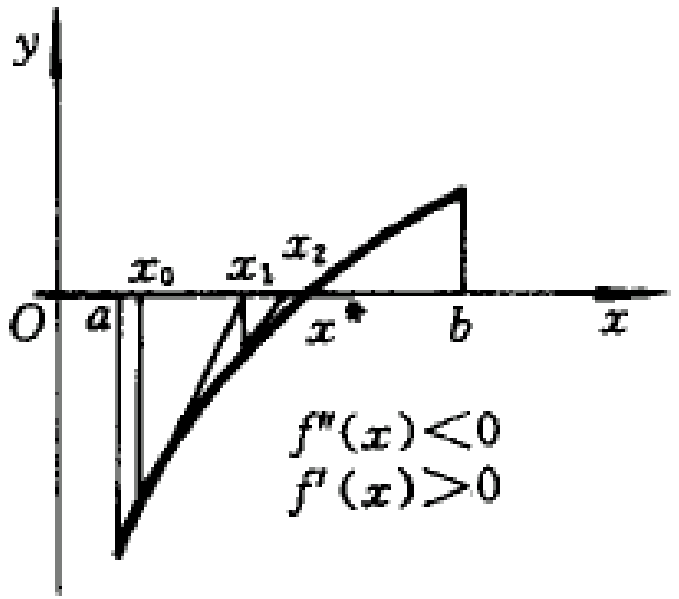
(3)对任意 $x\in[a,b]$都有 $f'(x)\neq 0$；

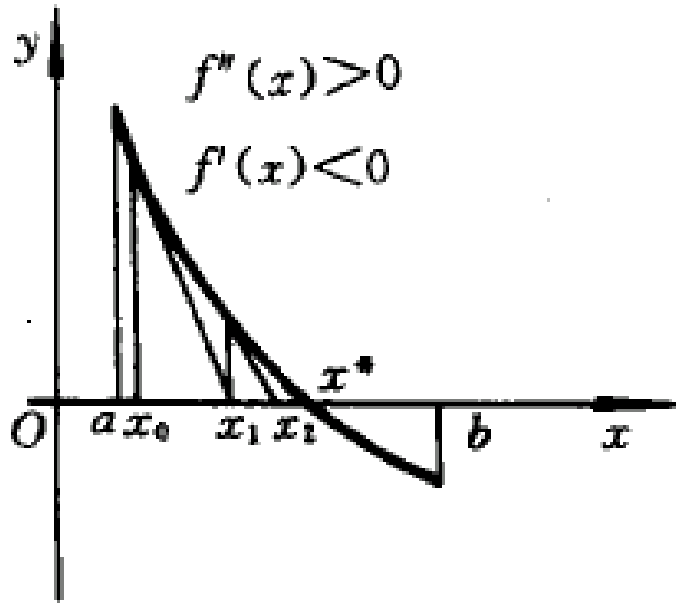
(4)$f''(x)$在$[a,b]$上保号.

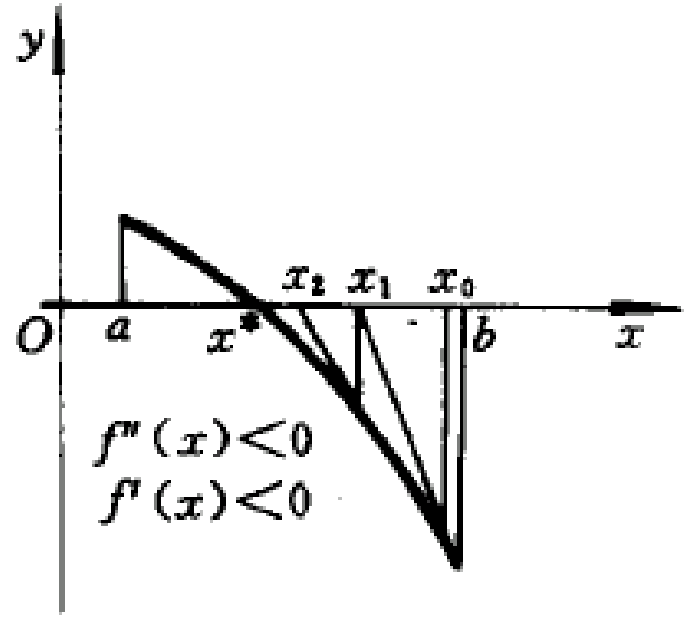
则当初值 $x_0\in[a,b]$且 $f(x_0)f''(x_0)>0$ 时，牛顿迭代过程(2.8)产生的迭代序列$\{x_k\}$收敛于方程 $f(x)=0$ 在$[a,b]$上的唯一实根 x^*.

$$x_0\in[a,b]\text{及 } f(x_0)f''(x_0)>0$$


y
f''(x)>0
f'(x)>0
a
x*
O
x₂
x₁
x₀
b
x


y
x₀
x₁
x₂
O
a
x*
b
x
f''(x)<0
f'(x)>0


y
f''(x)>0
f'(x)<0
x*
O
a
x₀
x₁
x₂
b
x

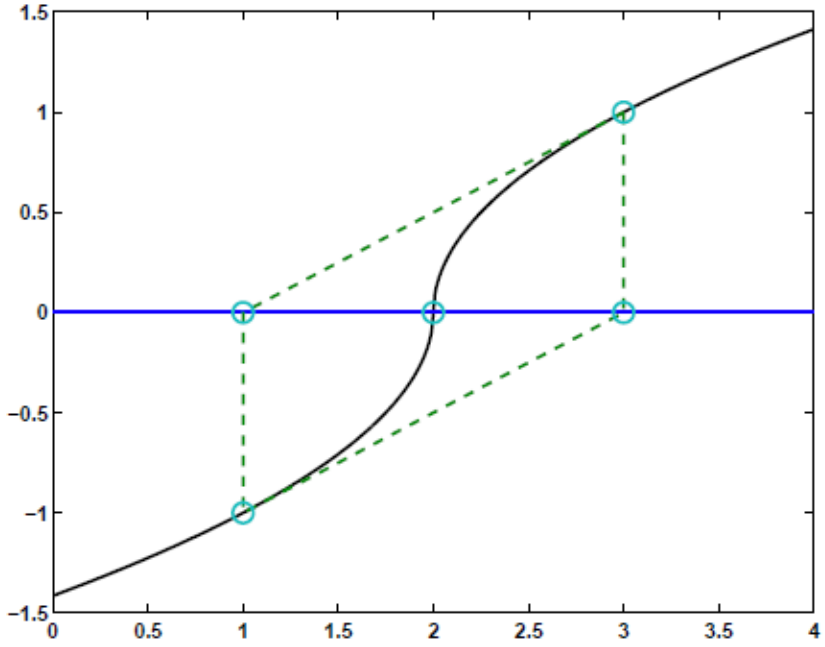

y
x₂
x₁
x₀
O
a
x*
b
x
f''(x)<0
f'(x)<0

牛顿法

一个特殊例子

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

$$x_{n+1} - a = -(x_n - a)$$

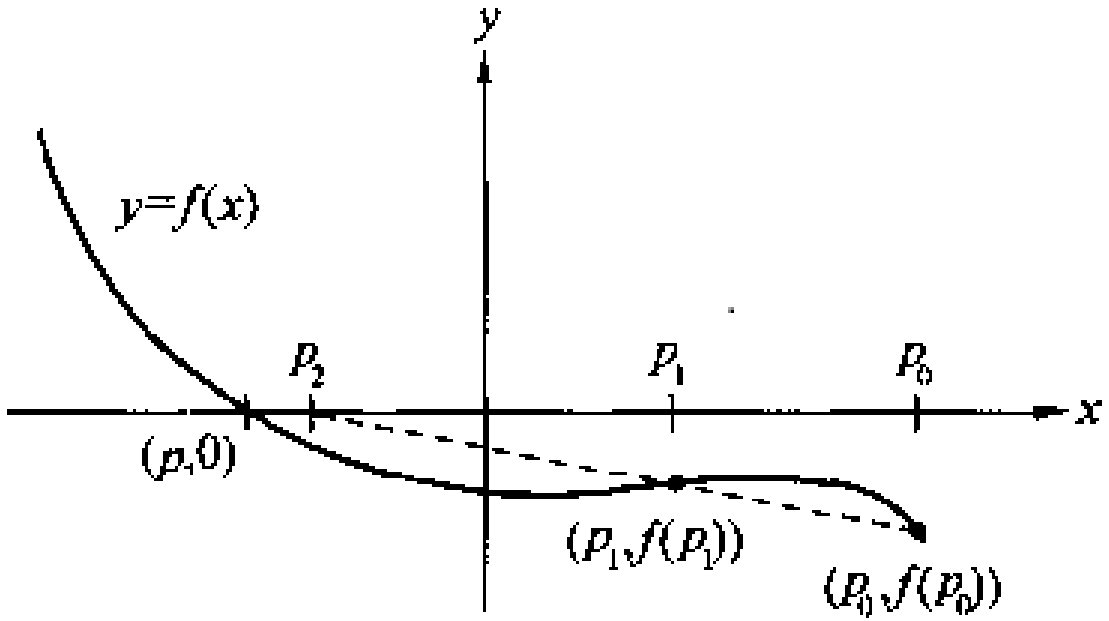


$$f(x) = sign(x-a)\sqrt{|x-a|}$$

割线法

$$s_n = \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}},$$

$$x_{n+1} = x_n - \frac{f(x_n)}{s_n}.$$



割线法

$\sqrt{2}$

```
while abs(b-a) > eps*abs(b)
    c = a;
    a = b;
    b = b + (b - c)/(f(c)/f(b)-1);
    k = k + 1;
end
```

```
1.33333333333333
1.40000000000000
1.41463414634146
1.41421143847487
1.41421356205732
1.41421356237310
1.41421356237310
```

割线法

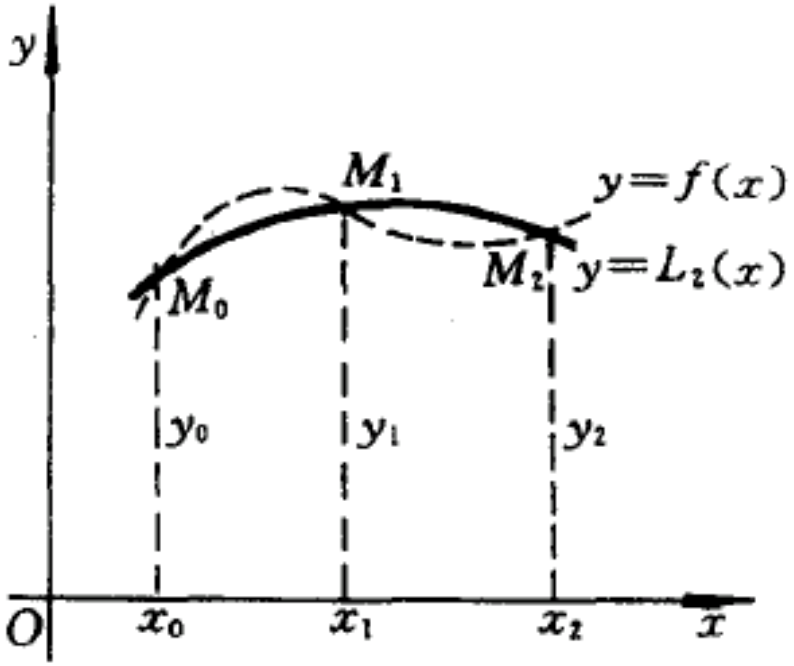
误差估计

$$e_{n+1} = \frac{1}{2}\frac{f''(\xi)f'(\xi_n)f'(\xi_{n-1})}{f'(\xi)^3}\ e_n e_{n-1}$$

$$e_{n+1} = O(e_n e_{n-1}).$$

逆二次插值

$$L_2(x) = y_0 \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} + y_1 \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} + y_2 \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}$$



抛物线插值

$$a = P(f(a)),\ b = P(f(b)),\ c = P(f(c))$$

逆二次插值

```
    k = 0;
    while abs(c-b) > eps*abs(c)

    x = polyinterp([f(a),f(b),f(c)],[a,b,c],0)
    a = b;
    b = c;
    c = x;
    k = k + 1;
end
```

这个"纯粹"的**IQI**算法的问题是，多项式插值数据点的横坐标，这里是***f(a)***、***f(b)***和***f(c)***，互不相同。但这里无法保证。例如，通过求解***f(x)=x^2-2=0*** 计算$\sqrt{2}$，并从***a*=-2** 、***b*=0** 、***c*=2** 开始，则一开始就出现***f(a)=f(c)***的情况，第一步就无法执行。如果开始时的参数接近这种奇异的情况，例如从***a*=-2.001** 、***b*=0** 、***c*=1.999**开始，则得到的下一步迭代解近似于***x*=500**。

Zeroin算法

核心：将二分法的可靠性和割线法及IQI算法的收敛速度结合起来

- Start with a and b so that $f(a)$ and $f(b)$ have opposite signs.
- Use a secant step to give c between a and b.
- Repeat the following steps until $|b-a| < \epsilon|b|$ or $f(b) = 0$.
- Arrange a, b, and c so that
  - $f(a)$ and $f(b)$ have opposite signs,
  - $|f(b)| \leq |f(a)|$,
  - c is the previous value of b.
- If $c \neq a$, consider an IQI step.
- If $c = a$, consider a secant step.
- If the IQI or secant step is in the interval $[a, b]$, take it.
- If the step is not in the interval, use bisection.

fzerotx , feval

```
bessj0 = inline('besselj(0,x)');
for n = 1:10
   z(n) = fzerotx(bessj0,[(n-1) n]*pi);
end
x = 0:pi/50:10*pi
y = besselj(0,x); plot(z,zeros(1,10),'o',x,y,'-')
line([0 10*pi],[0 0],'color','black')
axis([0 10*pi -0.5 1.0])
```

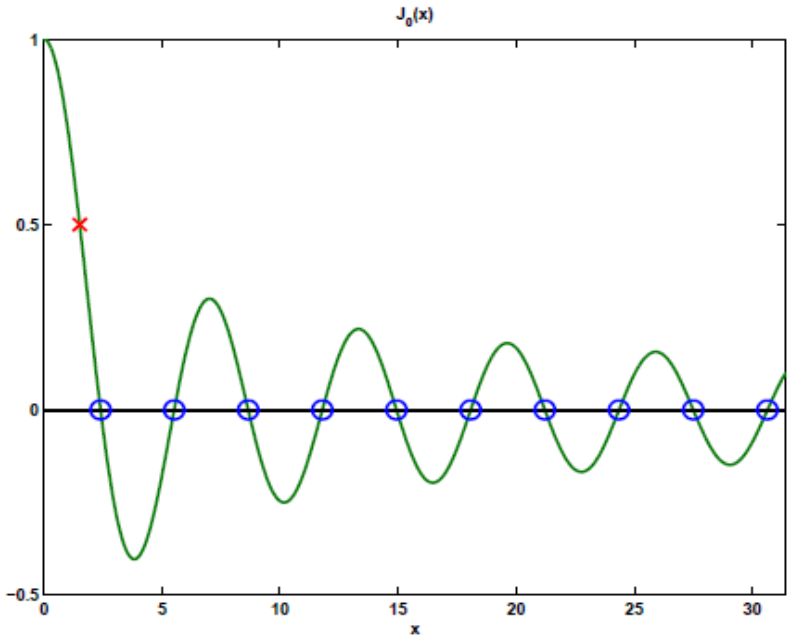


Figure 4.2. *Zeros of* $J_0(x)$.

fzerogui

fzerogui(@(x)besselj(0,x),[0,3.83])

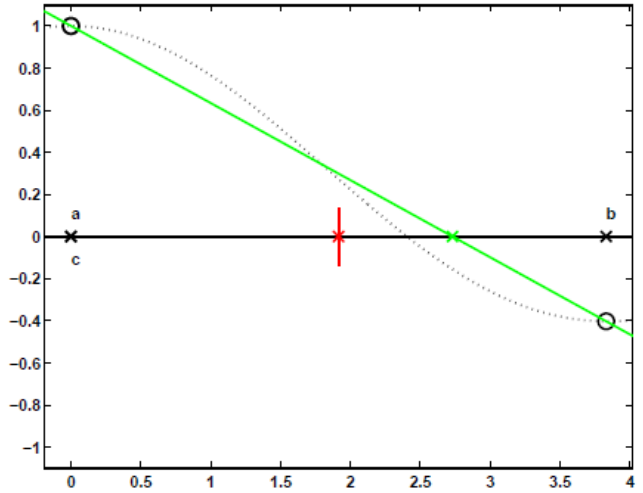


Figure 4.3. *Initially, choose secant or bisection.*

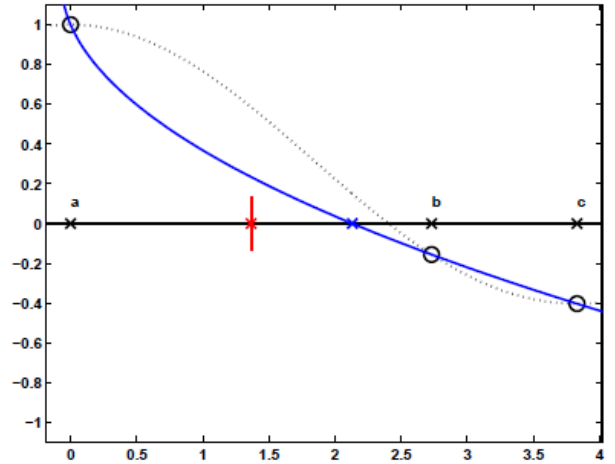


Figure 4.4. *Choose IQI or bisection.*

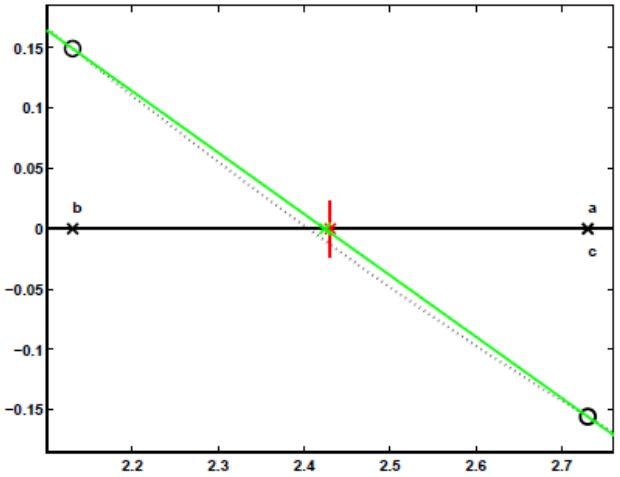


Figure 4.5. *Secant and bisection points nearly coincide.*

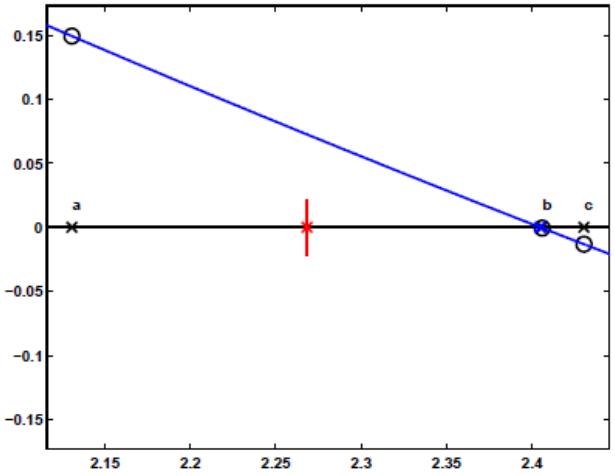


Figure 4.6. *Nearing convergence.*

寻找函数为某个值的解和反向插值

（1）给定一个函数 $F(x)$ 和值 η ，求 ξ 使得 $F(\xi)=\eta$ ；

（2）给定对未知函数 $F(x)$ 采样得到的一些数据点 (x_k, y_k)，以及一个值 η ，求 ξ 使得 $F(\xi)=\eta$ ；

（1）$f(x) = F(x) - \eta$ $\qquad$ $f(\xi) = 0$

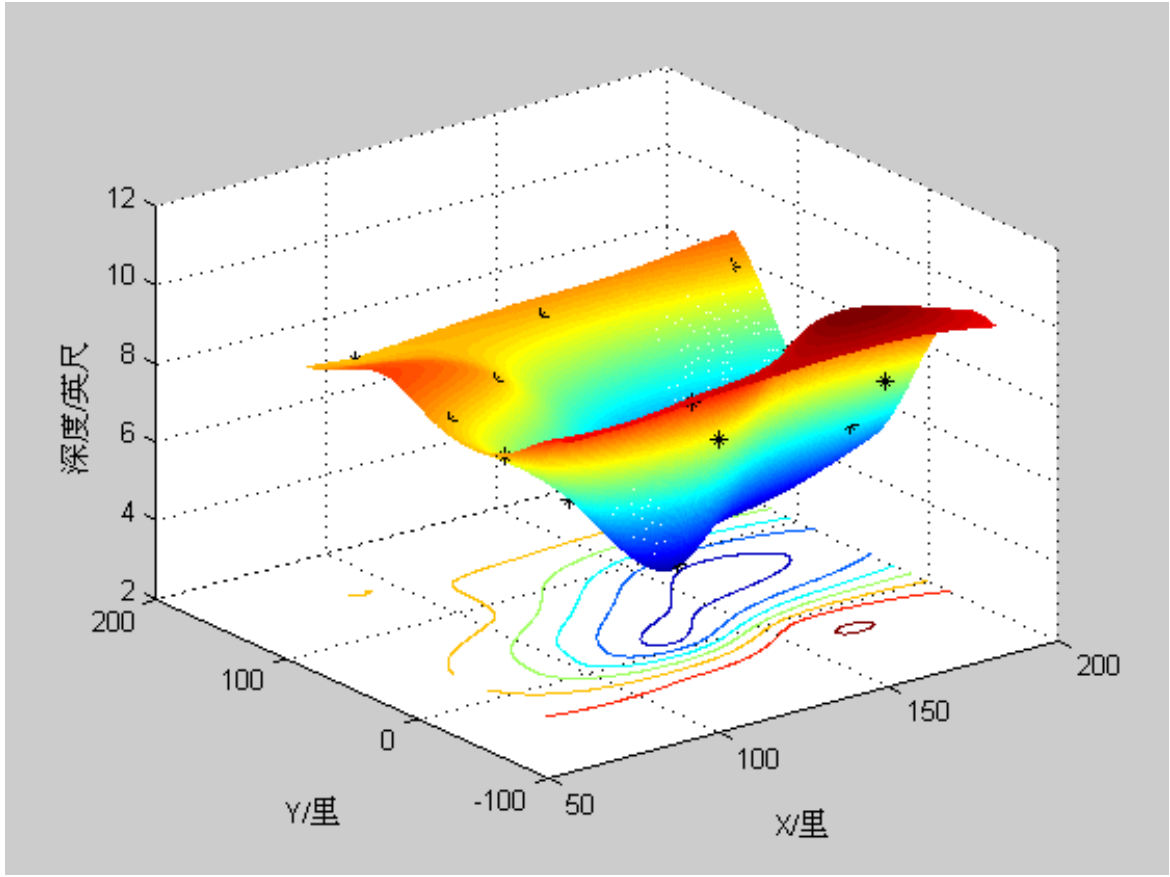
$F(\xi) = \eta$

（2）采用某种插值，$f(x) = P(x) - \eta$

对于有些情况，可以采用反向插值

最优化 和 fmintx

Min fun(x)
Sub. To [C.E.]
[B.E.]



最优化 和 fmintx

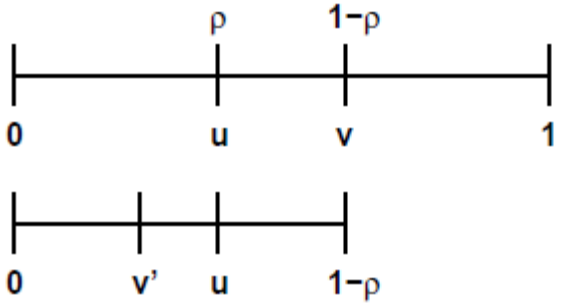
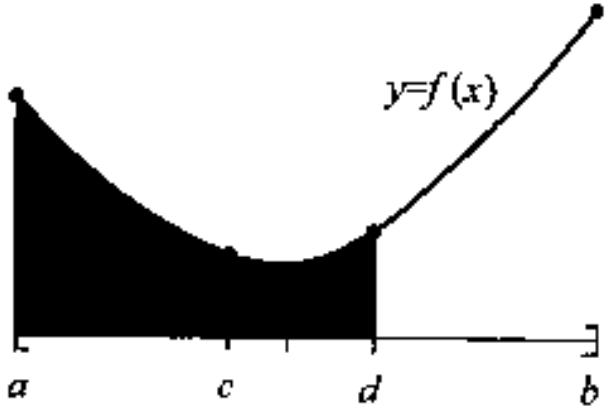


Figure 4.7. *Golden section search.*

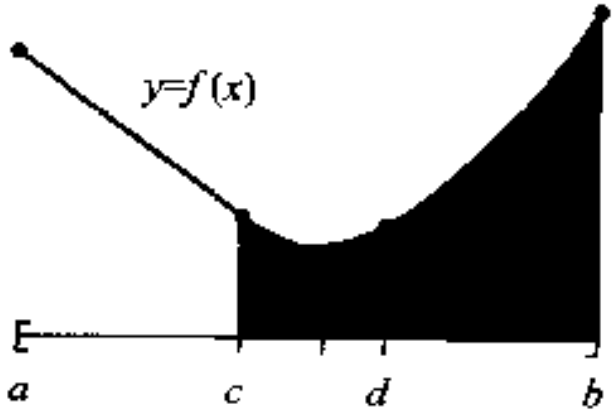
$$\frac{\rho}{1-\rho} = \frac{1-\rho}{1},$$

$$\rho^2 - 3\rho + 1 = 0.$$

$$\rho = 2 - \phi = (3 - \sqrt{5})/2 \approx 0.382.$$



如果 $f(c) \leqslant f(d)$，则从右边压缩并使用区间 $[a, d]$



如果 $f(d) < f(c)$，则从左边压缩并使用区间 $[c, b]$

最优化 和 fmintx

$$f'(x)=0$$

$$h(x)=\frac{1}{(x-0.3)^2+0.01}+\frac{1}{(x-0.9)^2+0.04}.$$

step	x	f(x)
init:	0.1458980337	-25.2748253202
gold:	0.8541019662	-20.9035150009
gold:	-0.2917960675	2.5391843579
para:	0.4492755129	-29.0885282699
para:	0.4333426114	-33.8762343193
para:	0.3033578448	-96.4127439649
gold:	0.2432135488	-71.7375588319
para:	0.3170404333	-93.8108500149
para:	0.2985083078	-96.4666018623
para:	0.3003583547	-96.5014055840
para:	0.3003763623	-96.5014085540
para:	0.3003756221	-96.5014085603

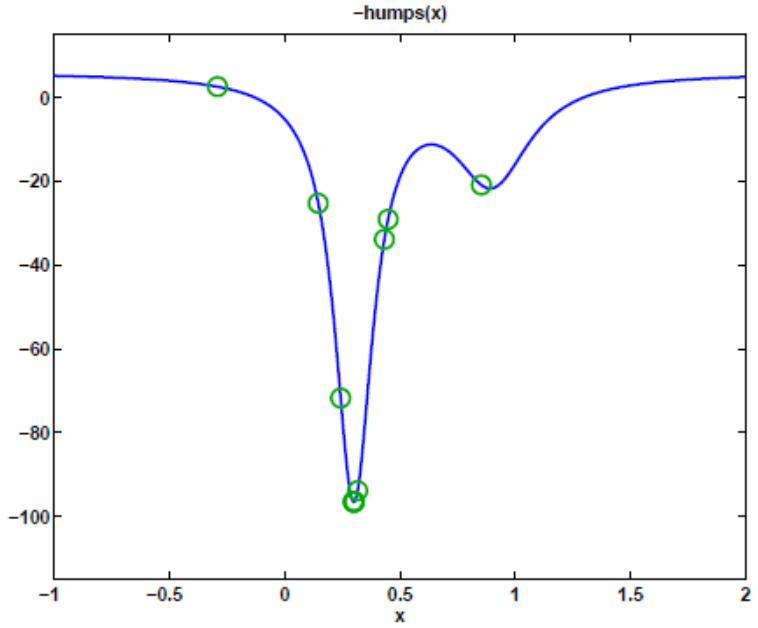
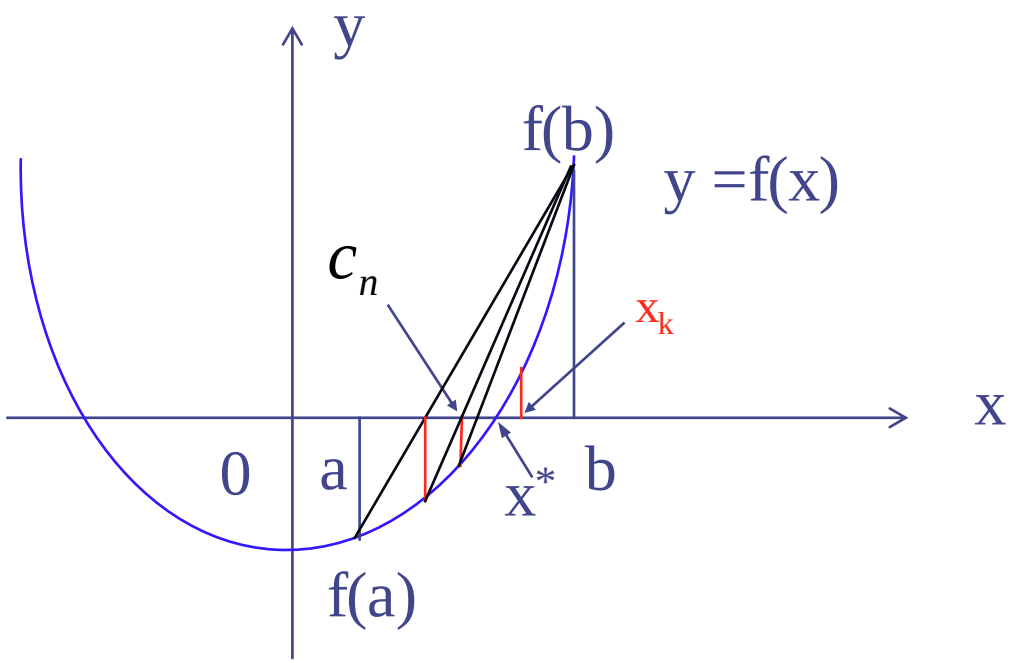


Figure 4.8. *Finding the minimum of* -humps(x).

试值法



$$c_n = b_n - \frac{f(b_n)(b_n - a_n)}{f(b_n) - f(a_n)}$$

横向收敛区：

纵坐标的封闭性通常通过$|f(p_n)<\varepsilon|$来检查。

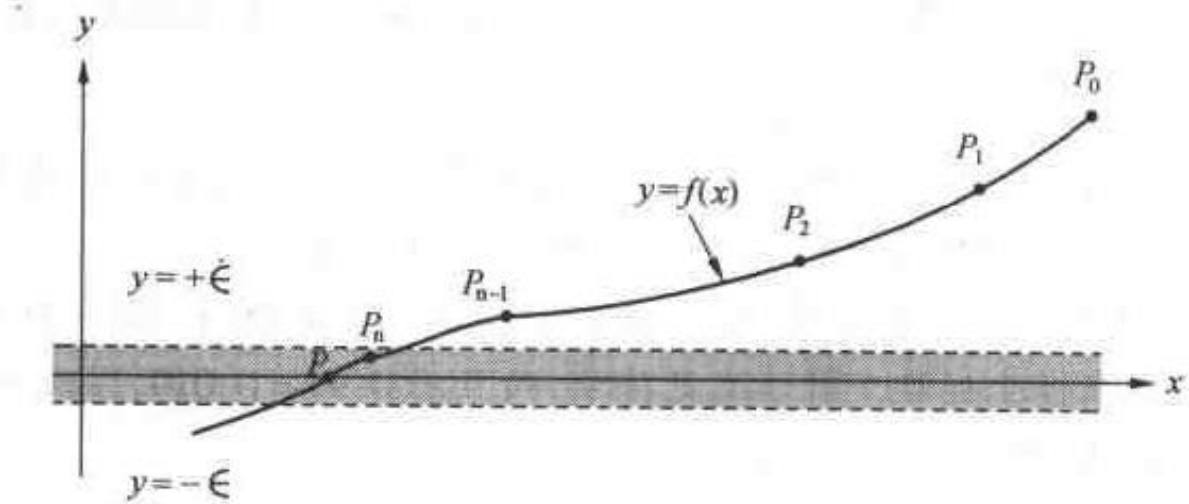


图 2.11(a)　定位函数 $f(x)=0$ 的解的横向收敛区

纵向收敛区：

$|p_n - p_{n-1}| < \delta$ （评价绝对误差）

$\dfrac{2|p_n - p_{n-1}|}{|p_n| - |p_{n-1}|}$ （评价相对误差）

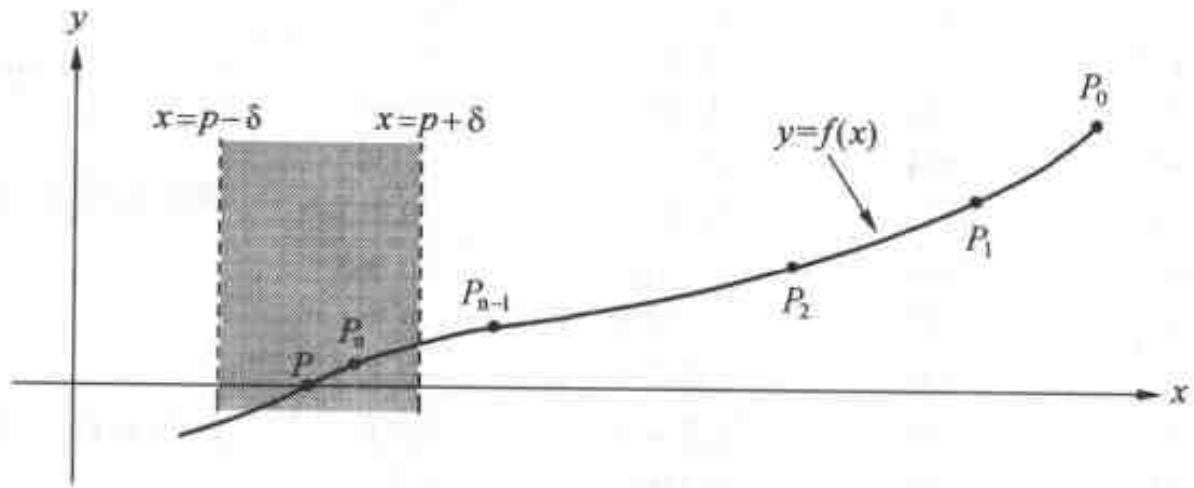


图 2.11(b)　定位函数 $f(x)=0$ 的解的纵向收敛区

收敛区域：

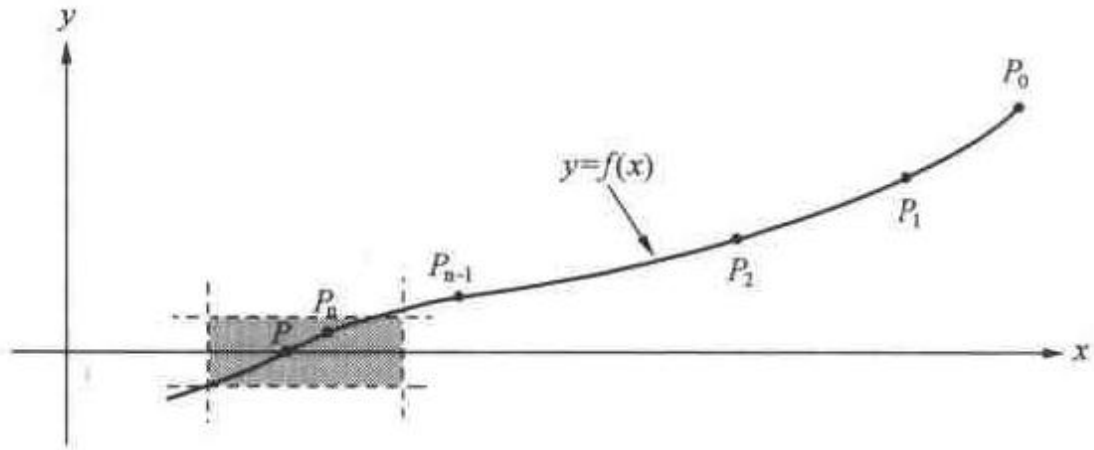


图 2.12(a)　由 $|x-p|<\delta$ 和 $|y|<\varepsilon$ 定义的矩形区域

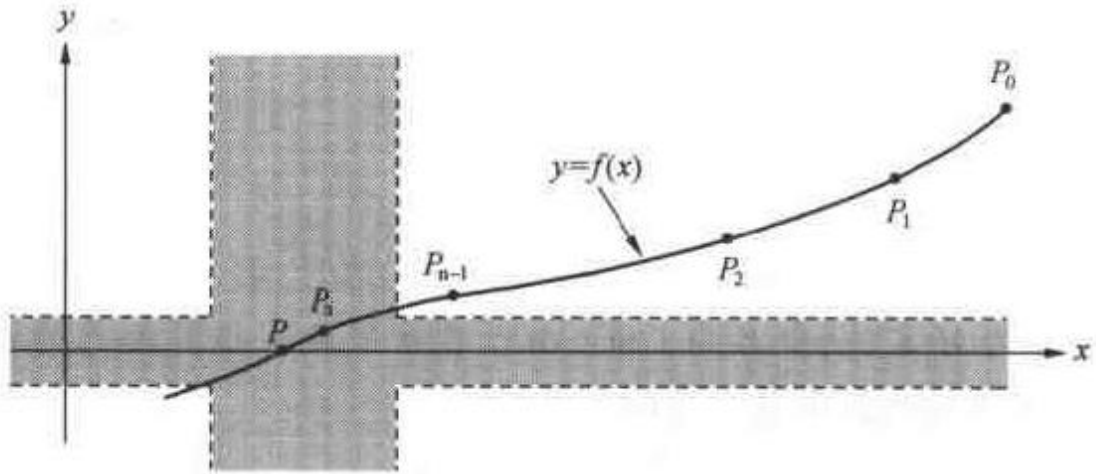


图 2.12(b)　由 $|x-p|<\delta$ 或 $|y|<\varepsilon$ 定义的无边界区域

Program 2.2 (Bisection Methods)

```
function [c,err,yc] = bisect(f,a,b,delta)
% Input  - f is the function input as a string 'f
%        - a and b are the left and right end poir
%        - delta is the tolerance
% Output- c is the zero
%        - yc = f(c)
%        - err is the error estimate for c
ya = feval(f,a);
yb = feval(f,b);
if ya * yb > 0,break,end
max1 = 1 + round((log(b - a) - log(delta))/log(2));
for k = 1:max1
    c = (a + b)/2;
    yc = feval(f,c);
    if yc == 0
        a = c;
        b = c;
elseif yb * yc > 0
        b = c;
        yb = yc;
    else
        a = c;
        ya = yc;
    end
    if b - a < delta,break,end
end
c = (a + b)/2;
err = abs(b - a);
yc = feval(f,c);
```

$$\frac{b-a}{2^{k+1}} \leqslant \varepsilon$$

Program 2.3 False Position or Regular False Method

```
function [c,err,yc] = regula(f,a,b,deltak,epsilon,max1)
% Input  - f is the function input as a string 'f'
%        - a and b are the left and right end points
%        - delta is the tolerance for the zero
%        - epsilon is the tolerance for the value of f at the zero
%        - max1 is the maximum number of iterations
% Output- c is the zero
%        - yc = f(c)
%        - err is the error estimate for c
ya = feval(f,a);
yb = feval(f,b);
if ya * yb > 0
    disp('Note: f(a) * f(b) > 0'),
    break,
end
for k = 1:max1
    dx = yb * (b - a)/(yb - ya);
    c = b - dx;
    ac = c - a;
    yc = feval(f,c);
    if yc == 0,break;
    elseif yb * yc > 0
        b = c;
        yb = yc;
    else
        a = c;
        ya = yc;
    end
    dx = min(abs(dx),ac);
    if abs(dx) < delta,break,end
    if abs(yc) < epsilon,break,end
end
c;
err = abs(b - a))/2;
yc = feval(f,c);
```

Program2.5 (Newton-Raphson Iteration)

$$f(x) = 0 \qquad p_k = p_{k-1} - \frac{f(p_{k-1})}{f'(p_{k-1})}$$

```
function [po,err,k,y] = newton(f,df,p0,delta,epsilon,max1)
% Input   - f is the object function input as a string'f'
%         - df is the derivative of f input as a string 'df'
%         - p0 is the initial approximation to a zero of f
%         - delta is the tolerance for p0
%         - epsilon is the tolerance for the function values y
%         - max1 is the maximum number of iterations
% Cutput  - p0 is the Newton - Raphson approximation to the zero
%         - err is the error estimate for p0
%         - k is the number of iterations
%         - y is the function value f(p0)
for k = 1:max1
   p1 = p0 - feval(f,p0)/feval(df,p0);
   err = abs(p1 - p0);
   relerr = 2 * err/(abs(p1) + delta);
   p0 = p1;
   y = feval(f,p0)
   if (err < delta)|(relerr < delta)|(abs(y) < epsilon),break,end
end
```

Aitken’s Process and Steffensen’s and Muller’s Methods (Optional)

Atiken’s Process

$\frac{p - p_{n+1}}{p - p_n} \approx A$ 和 $\frac{p - p_{n+2}}{p - p_{n+1}} \approx A$，其中 n 足够大

定理 2.8(Aitken 加速) 设序列$\{p_n\}_{n=0}^{\infty}$线性收敛到极限 p,而且对所有 $n \geqslant 0$,有 $p - p_n \neq 0$。如果存在实数 A,且$|A| < 1$,满足:

$$\lim_{n \to \infty} \frac{p - p_{n+1}}{p - p_n} = A \tag{3}$$

则定义为:

$$q_n = p_n - \frac{(\Delta p_n)^2}{\Delta^2 p_n} = p_n - \frac{(p_{n+1} - p_n)^2}{p_{n+2} - 2p_{n+1} + p_n} \tag{4}$$

的序列$\{q_n\}_{n=0}^{\infty}$收敛到 p,且比$\{p_n\}_{n=0}^{\infty}$快,而且:

$$\lim_{n \to \infty} \left| \frac{p - q_n}{p - p_n} \right| = 0 \tag{5}$$

证明:下面将证明如果得到式(4),并把对式(5)的证明作为练习。由于式(3)中的项是逼近一个极限,可写成:

$$\frac{p - p_{n+1}}{p - p_n} \approx A \quad 和 \quad \frac{p - p_{n+2}}{p - p_{n+1}} \approx A, \quad 其中\ n\ 足够大 \tag{6}$$

则根据式(6)中的关系式可得到:

$$(p - p_{n+1})^2 \approx (p - p_{n+2})(p - p_n) \tag{7}$$

当展开式(7)的两边并消除 p^2,可得到:

$$p \approx \frac{p_{n+2}\ \ p_n - p_{n+1}^2}{p_{n+2} - 2p_{n+1} + p_n} = q_n, \quad n = 0, 1, \cdots \tag{8}$$

例 2.2　设有收敛迭代

$$p_0 = 0.5,\ p_{k+1} = e^{-p_k},\ k = 0, 1, \cdots$$

前十项的计算结果如下所示：

$$\begin{aligned} p_1 &= e^{-0.500000} = 0.606531 \\ p_2 &= e^{-0.606531} = 0.545239 \\ p_3 &= e^{-0.545239} = 0.579703 \\ &\vdots \qquad\qquad \vdots \\ p_9 &= e^{-0.566409} = 0.567560 \\ p_{10} &= e^{-0.567560} = 0.566907 \end{aligned}$$

这个序列是收敛的，且进一步计算可发现：

$$\lim_{n \to \infty} p_n = 0.567143\cdots$$

这样，可找到函数 $y = e^{-x}$ 的固定点近似值。

列 2.18　证明例 2.2 中的序列 $\{p_n\}$ 是线性收敛。同时证明由 Aitken's Δ^2 过程得到的序列 $\{q_n\}$ 收敛得更快。

使用函数 $g(x) = e^{-x}$，从 $p_0 = 0.5$ 开始，通过固定点迭代可得到序列 $\{p_n\}$。收敛后的极限为 p_n 和 q_n 的值如表 2.10 和表 2.11 所示。例如，q_1 的值的计算过程如下：

$$\begin{aligned} q_1 &= p_1 - \frac{(p_2 - p_1)^2}{p_3 - 2p_2 + p_1} \\ &= 0.606530660 - \frac{(-0.061291448)^2}{0.095755331} = 0.567298989 \end{aligned}$$

表 2.10　线性收敛序列 $\{p_n\}$

n	p_n	$E_n = p_n - p$	$A_n = \dfrac{E_n}{E_{n-1}}$
1	0.606530660	0.039387369	−0.586616609
2	0.545239212	−0.021904079	−0.556119357
3	0.579703095	0.012559805	−0.573400269
4	0.560064628	−0.007078663	−0.563596551
5	0.571172149	0.004028859	−0.569155345
6	0.564862947	−0.002280343	−0.566002341

表 2.11　用 Aitken 过程得到的序列 $\{q_n\}$

n	q_n	$q_n - p$
1	0.567298989	0.000155699
2	0.567193142	0.000049852
3	0.567159364	0.000016074
4	0.567148453	0.000005163
5	0.567144952	0.000001662
6	0.567143825	0.000000534

尽管表 2.11 中的序列$\{q_n\}$为线性收敛，根据定理 2.8，它比$\{p_n\}$收敛得快。而通常 Aitken 方法改进后收敛速度更快。把固定点迭代和 Aitken 过程结合起来的方法称为 Steffensen 加速。

Muller’s Method

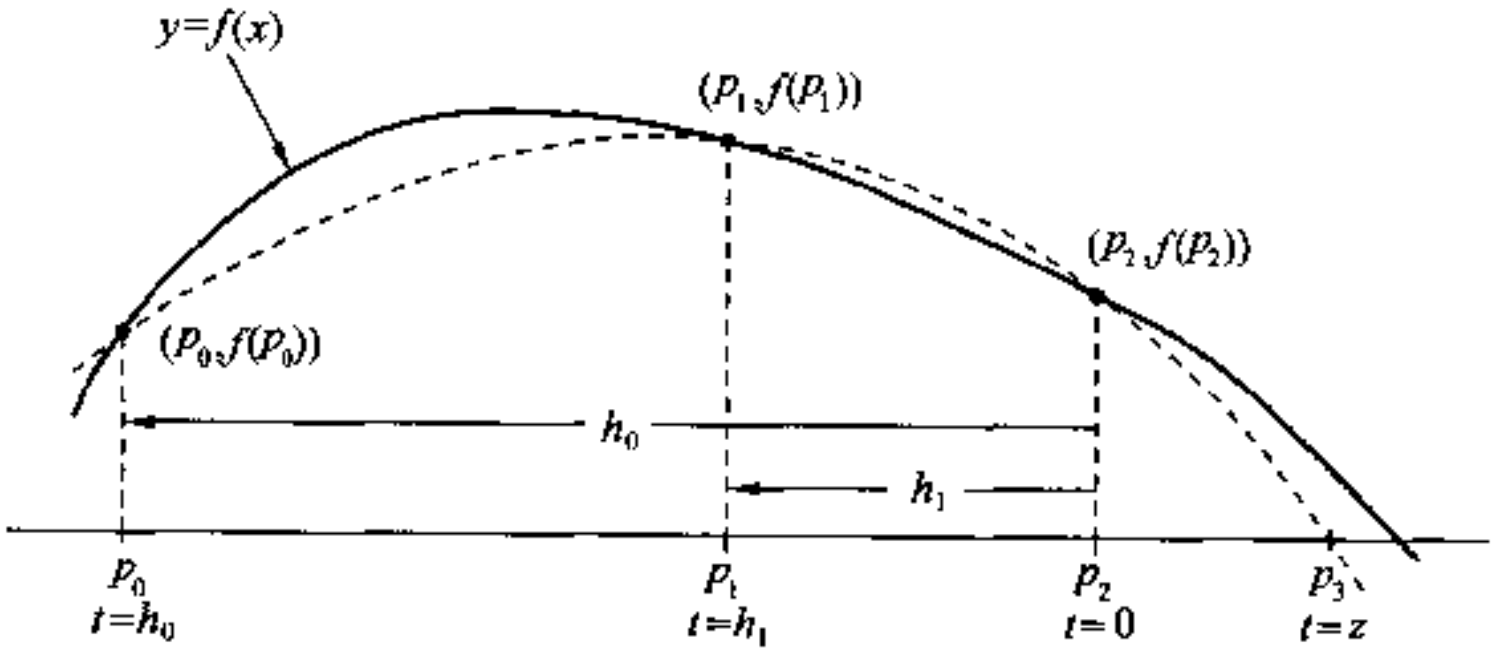


图 2.17　采用 Muller 法的初始近似值 p_0，p_1 和 p_2，以及差分 h_0 和 h_1

$$t = x - p_2 \tag{9}$$

使用差分为：

$$h_0 = p_0 - p_2 \text{ 和 } h_1 = p_1 - p_2 \tag{10}$$

设包含变量 t 的二次多项式为：

$$y = at^2 + bt + c \tag{11}$$

根据每一点可得到一个包含 a、b 和 c 的方程：

当 $t = h_0$：　　$ah_0^2 + bh_0 + c = f_0$

当 $t = h_1$：　　$ah_1^2 + bh_1 + c = f_1$ (12)

当 $t = 0$：　　$ao^2 + b0 + c = f_2$

从式(12)中的第三个方程，可看到：

$$c = f_2 \tag{13}$$

将式(13)代入式(12)中的前两个方程，并利用定义 $e_0 = f_0 - c$ 和 $e_1 = f_1 - c$，可得到线性方程组：

$$ah_0^2 + bh_0 = f_0 - c = e_0 .$$

$$ah_1^2 + bh_1 = f_1 - c = e_1 . \tag{14}$$

求解线性方程组可得：

$$a = \frac{e_0 h_1 - e_1 h_0}{h_1 h_0^2 - h_0 h_1^2}$$

$$b = \frac{e_1 h_0^2 - e_0 h_1^2}{h_1 h_0^2 - h_0 h_1^2} \tag{15}$$

下列二次式用来求解式(11)的根 $t = z_1, z_2$

$$Z = \frac{-2c}{b \pm \sqrt{b^2 - 4ac}} \tag{16}$$

式(16)等价于求二次根的标准公式，而且由于 $c = f_2$，所以它的情况更好。

为了确保方法的稳定性，需要选择式(16)中绝对值最小的根。如果 $b > 0$，使用带正号的根；如果 $b < 0$，使用带负号的根。则 p_3 如图 2.17 所示，表示为：

$$p_3 = p_2 + z \tag{17}$$

为了更新迭代，需要从 $|p_0, p_1, p_2|$ 中选择最靠近 p_3 的两点为新的 p_0 和 p_1（即放弃离 p_3 最远的一点）。然后使用新的 p_2 替代 p_3。尽管在 Muller 法中有许多辅助计算，但它每个迭代只需要计算一个函数。

直接用Matlab解非线性方程

MATLAB 命令输入格式：

```
solve('eqn1','eqn2',…,'eqnN')
```

或

```
solve('eqn1','eqn2',…,'eqnN','var1','var2',…,'varN')
```

其中 eqn$i$ 表示第 i 个方程，var$i$ 表示第 i 个变量，$i = 1, 2, \cdots, N$.

1. 一般方程

例如，求解方程 $x^2 + bx + c = 0$.

输入：

```
solve('x^2 + b * x + c')
```

输出：

```
[ -1/2 * b + 1/2 * (b^2 - 4 * c)^(1/2)]
[ -1/2 * b - 1/2 * (b^2 - 4 * c)^(1/2)]
```

2. 多项式方程

除了用上面求解一般方程的方法外，还可以直接用求解多项式方程的MATLAB函数 roots(p)，其中 p 是多项式的系数按降幂排列所形成的 n + 1 维列向量，它能够给出全部根(包含重根).

例如，求解多项式方程 $x^9 + x^8 + 1 = 0$.

输入：

```
p=[1,1,0,0,0,0,0,0,0,1];
roots(p)
```

输出：

```
-1.2131
-0.9017 + 0.5753i
-0.9017 - 0.5753i
-0.2694 + 0.9406i
-0.2694 - 0.9406i
 0.4168 + 0.8419i
 0.4168 - 0.8419i
 0.8608 + 0.3344i
 0.8608 - 0.3344i
```

注意：也可以用 solve('x^9 + 1')求解，有何区别?

思考题

1、如何改进二分法，使其收敛速度加快?

2、怎样修正牛顿迭代法可以提高其收敛速度?

3、为什么埃特肯加速法能提高线性迭代序列的收敛速度?

4、有无其他可行的解非线性方程的方法?

范例:波音公司飞机最佳定价策略

全球最大的飞机制造商——波音公司自 1955 年推出波音 707 开始,成功地开发了一系列的喷气式客机.问题:讨论该公司对一种新型客机最优定价策略的数学模型.

定价策略涉及到诸多因素,这里考虑以下主要因素:

价格、竞争对手的行为、出售客机的数量、波音公司的客机制造量、制造成本、波音公司的市场占有率等等因素.

价格记为 p,根据实际情况,对于民航飞机制造商,能够与波音公司抗衡的竞争对手只有一个,因此他们可以在价格上达成一致,具体假设如下:

1) 型号:为了研究方便,假设只有一种型号飞机;

2) 销售量:其销售量只受飞机价格 p 的影响.预测以此价格出售,该型号飞机全球销售量为 N.N 应该受到诸多因素的影响,假设其中价格是最主要的因素.根据市场历史的销售规律和需求曲线,假设该公司销售部门预测得到

$$N = N(p) = -78p^2 + 655p + 125.$$

3)市场占有率:既然在价格上达成一致,即价格的变化是同步的,因此,不同定价不会影响波音公司的市场占有率,因此场占有率是常数,记为 h.

4)制造数量:假设制造量等于销售量,记为 x.既然可以预测该型号飞机全球销售量,结合波音公司的市场占有率,可以得到

$$x = h \times N(p).$$

5)制造成本:根据波音产品分析部门的估计,制造成本为

$$C(x) = 50 + 1.5x + 8x^{\frac{3}{4}}.$$

6)利润:假设利润等于销售收入去掉成本,并且公司的最优策略原则为利润 $R(p)$最大.利润函数为

$$R(p) = px - C(x).$$

感谢聆听,欢迎讨论!