



大学数学实验

Experiments in Mathematics



第5讲

数值计算IV 非线性方程求解



非线性方程（组）

$$F(\mathbf{x}) = \begin{pmatrix} f_1(x_1, x_2, \dots, x_n) \\ \vdots \\ f_n(x_1, x_2, \dots, x_n) \end{pmatrix} = \mathbf{0}$$

满足方程（组）的未知数的取值称为方程（组）的解，或称为 $F(\mathbf{x})$ 的零点。

单变量方程（一元方程）： $f(x)=0$ ，“解”也称为“根”



非线性方程 (组)

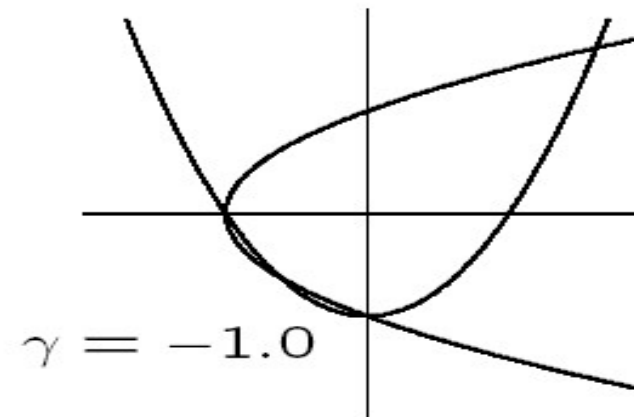
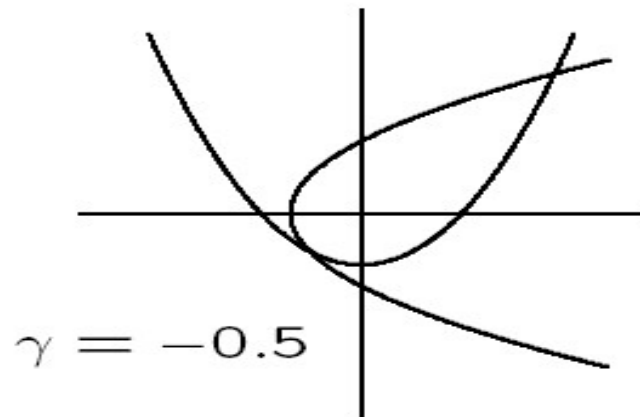
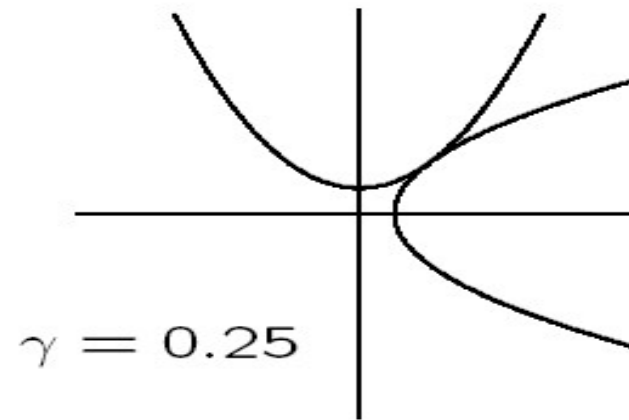
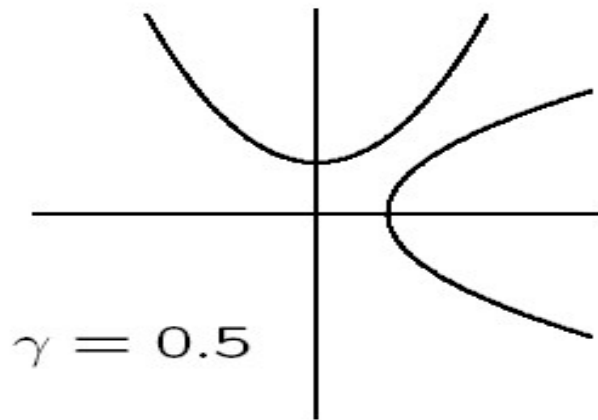
- 代数方程: $a_0x^n + a_1x^{n-1} + \dots + a_n = 0$;
- 超越方程: 包含超越函数(如 $\sin x$, $\ln x$)的方程;

方程根的特点:

- n 次代数方程有且只有 n 个根(包括复根、重根);
- 5次以上的代数方程无求根公式;
- 超越方程有无根, 有几个根通常难以判断。



$$\begin{cases} x_1^2 - x_2 + \gamma = 0 \\ -x_1 + x_2^2 + \gamma = 0 \end{cases}$$





方程的重根

x^* 是 $f(x)=0$ 的 m 重根, $m > 1$,

即 $f(x) = (x - x^*)^m g(x)$, 且 $g(x^*) \neq 0$

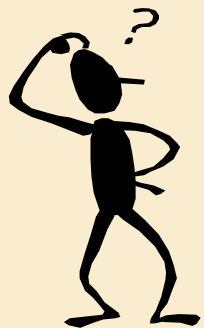
$$f'(x) = (x - x^*)^{m-1} g(x) + (x - x^*)^m g'(x)$$

重根判别法 $f^{(p)}(x^*) = 0, p = 0, 1, \dots, m-1, f^{(m)}(x^*) \neq 0$

$x^* = 0$ 是方程 $f(x) = e^{2x} - 1 - 2x - 2x^2 = 0$ 的几重根?



实验基本内容



1. 非线性方程 $f(x)=0$ 的数值解法:

- 迭代方法的基本原理;
- 局部收敛性
- 牛顿法

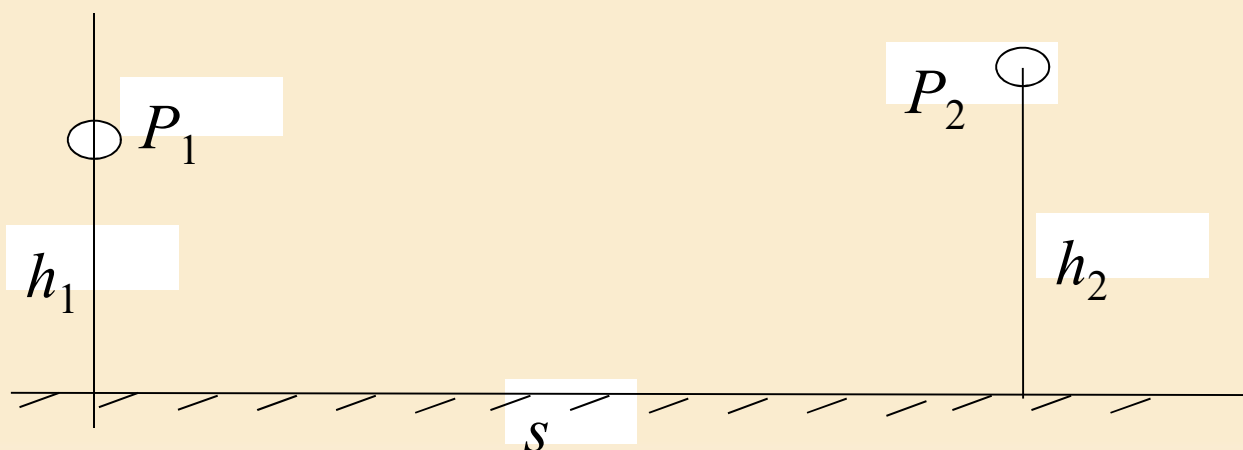
2. 非线性方程数值解的Matlab实现



实例1 路灯照明

道路两侧分别安装路灯，在漆黑的夜晚，当两只路灯开启时，两只路灯连线的路面上最暗的点和最亮的点在哪里？

- 如果 P_2 的高度可以在3米到9米之间变化，间距是20米，如何使路面上最暗点的亮度最大？
- 如果两只路灯的高度均可以在3米到9米之间变化呢？



$$s=20(\text{米})$$

$$P_1=2, P_2=3(\text{千瓦})$$

$$h_1=5, h_2=6(\text{米})$$

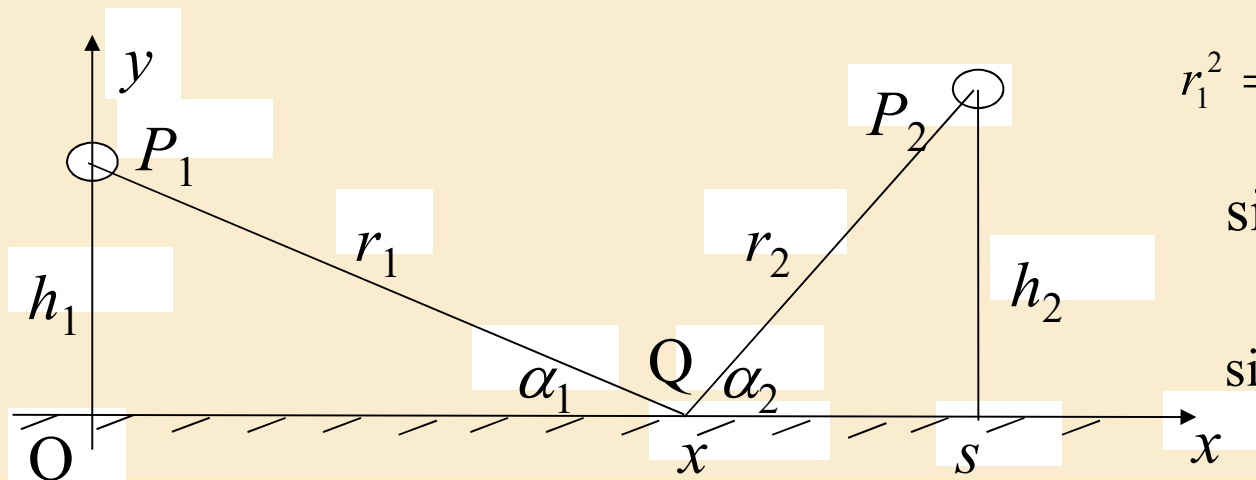


实例1 路灯照明

建立坐标系如图，两个光源在点 $Q(x,0)$ 的照度分别为

$$I_1 = k \frac{P_1 \sin \alpha_1}{r_1^2}, \quad I_2 = k \frac{P_2 \sin \alpha_2}{r_2^2} \quad (k \text{ 是由量纲单位决定的比例系数, 不妨记 } k=1)$$

$$\text{点 } Q \text{ 的照度 } C(x) = \frac{P_1 h_1}{\sqrt{(h_1^2 + x^2)^3}} + \frac{P_2 h_2}{\sqrt{(h_2^2 + (s-x)^2)^3}}$$



$$r_1^2 = h_1^2 + x^2, \quad r_2^2 = h_2^2 + (s-x)^2$$

$$\sin \alpha_1 = \frac{h_1}{r_1} = \frac{h_1}{\sqrt{h_1^2 + x^2}}$$

$$\sin \alpha_2 = \frac{h_2}{r_2} = \frac{h_2}{\sqrt{h_2^2 + (s-x)^2}}$$



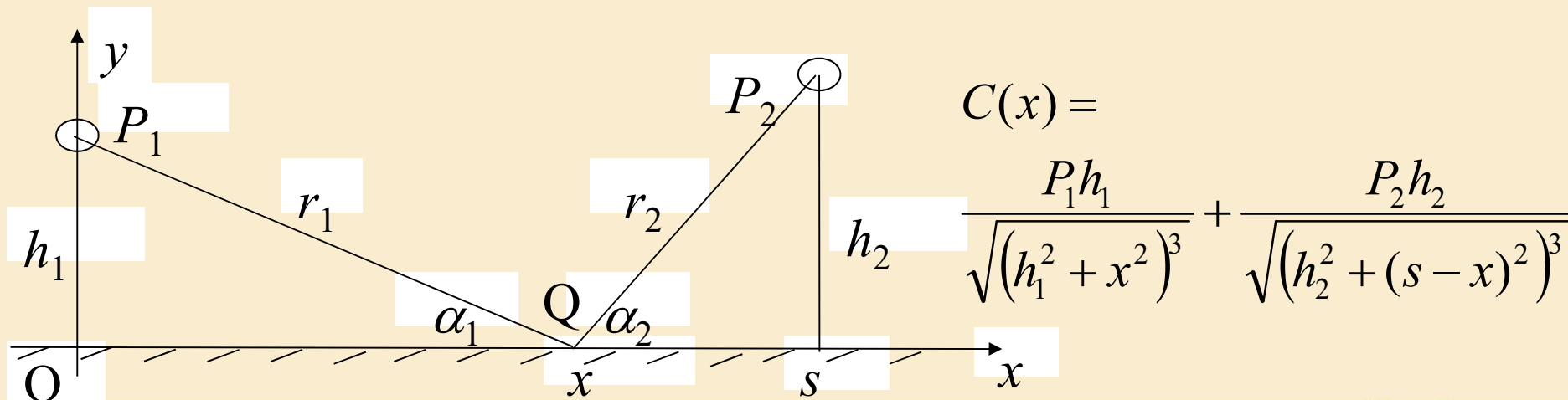


实例1 路灯照明

为求最暗点和最亮点，先求 $C(x)$ 的驻点

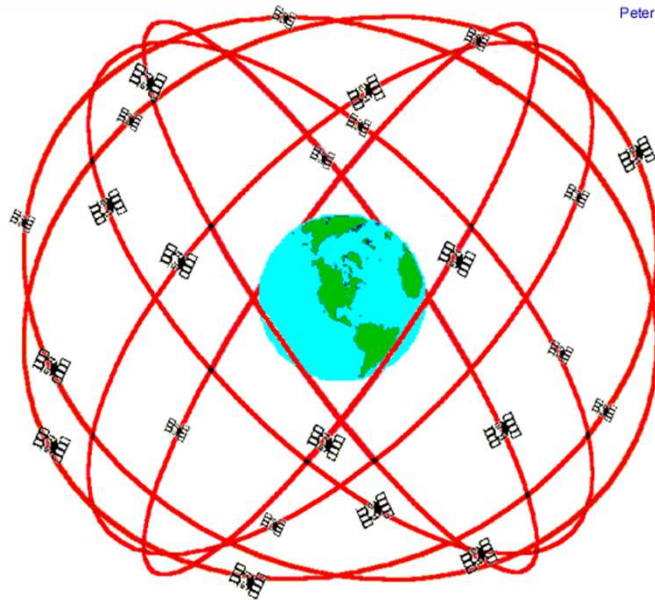
$$C'(x) = -3 \frac{P_1 h_1 x}{\sqrt{(h_1^2 + x^2)^5}} + 3 \frac{P_2 h_2 (s - x)}{\sqrt{(h_2^2 + (s - x)^2)^5}}$$

令 $C'(x)=0$ ：解析解是难以求出，需数值求解





The Global Positioning System



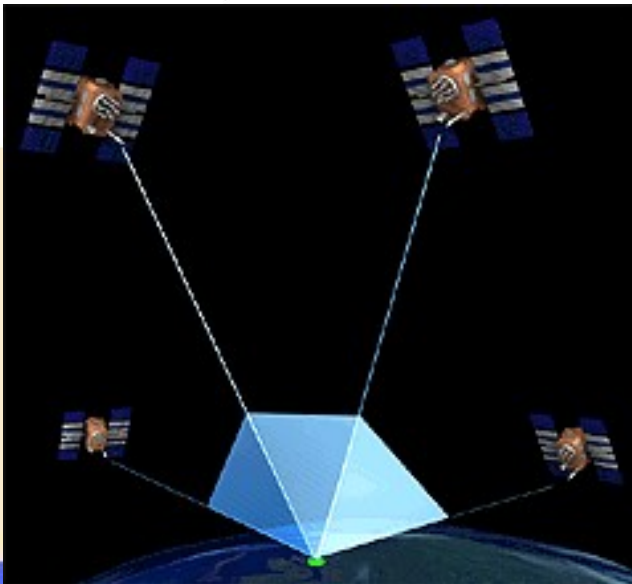
Peter H. Dana 9/22/98

$$\vec{s}_i = \begin{bmatrix} x_i \\ y_i \\ z_i \\ \rho_i \end{bmatrix} \quad (\text{data we get from satellite } i)$$

$$\vec{u} = \begin{bmatrix} x \\ y \\ z \\ b \end{bmatrix} \quad (\text{unknown data about the receiver}).$$

Thus, for each i ,

$$\sqrt{(x_i - x)^2 + (y_i - y)^2 + (z_i - z)^2} + b = \rho_i.$$





非线性方程的基本解法

解方程 $f(x)=0$ 第一步——确定根的大致范围

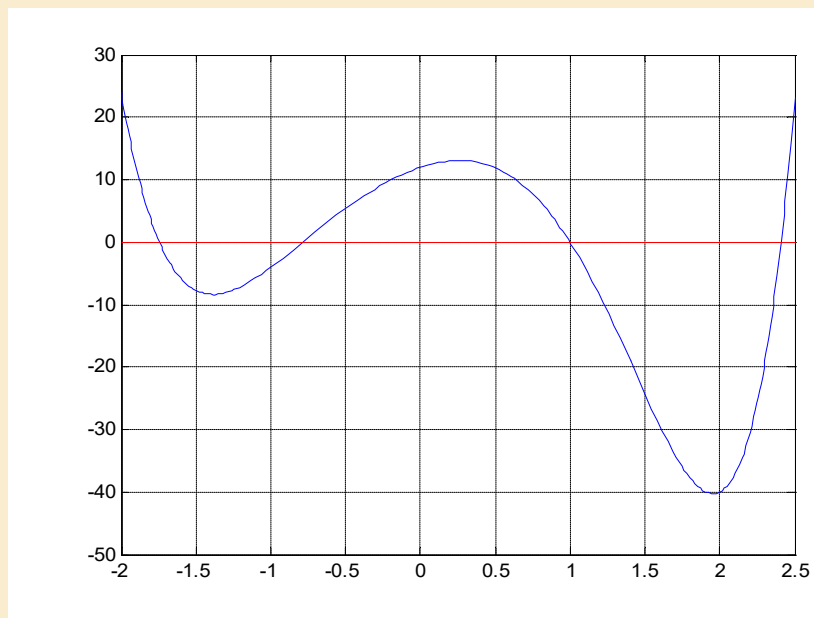
- 图形法：作 $f(x)$ 图形，观察 $f(x)$ 与 x 轴的交点
- 根的隔离：二分法

图形法

$$x^6 - 2x^4 - 6x^3 - 13x^2 + 8x + 12 = 0$$

有4个根分别位于

$x = -1.75, -0.75, 1.00, 2.40$ 附近



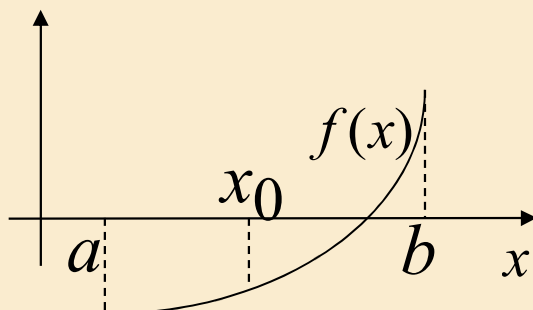
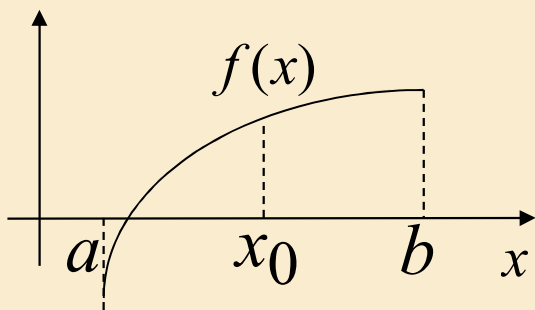


非线性方程的基本解法

二分法的原理

若对于 $a < b$, 有 $f(a) \cdot f(b) < 0$, 则在 (a, b) 内 $f(x)$ 至少有一个零点, 即 $f(x) = 0$ 至少有一个根。

二分法的实现



$$f(a) < 0$$

$$f(b) > 0$$

$x_0 : (a, b)$ 中点

$$f(x_0) > 0 \Rightarrow a_1 = a, b_1 = x_0$$

$$f(x_0) < 0 \Rightarrow a_1 = x_0, b_1 = b$$

$(a, b) \Rightarrow (a_1, b_1) \Rightarrow \cdots \Rightarrow (a_n, b_n) \Rightarrow \cdots$ 区间每次缩小一半, n 足够大时, 可确定根的范围

不足

收敛速度较慢



Matlab 二分法实现

- **function [x,k,r]=bisect(ff,a,b,err)**
- **k=0; r=[];**
-
- **while b-a>err**
- **f1=feval(ff,a); f2=feval(ff,b);**
- **r=[r;[a,b,f1,f2]];** % 记录二分过程
- **c=(a+b)/2; f3=feval(ff,c); k=k+1;**
- **if f1*f3<0 b=c; else a=c; end**
- **end**
-
- **x=(a+b)/2;**

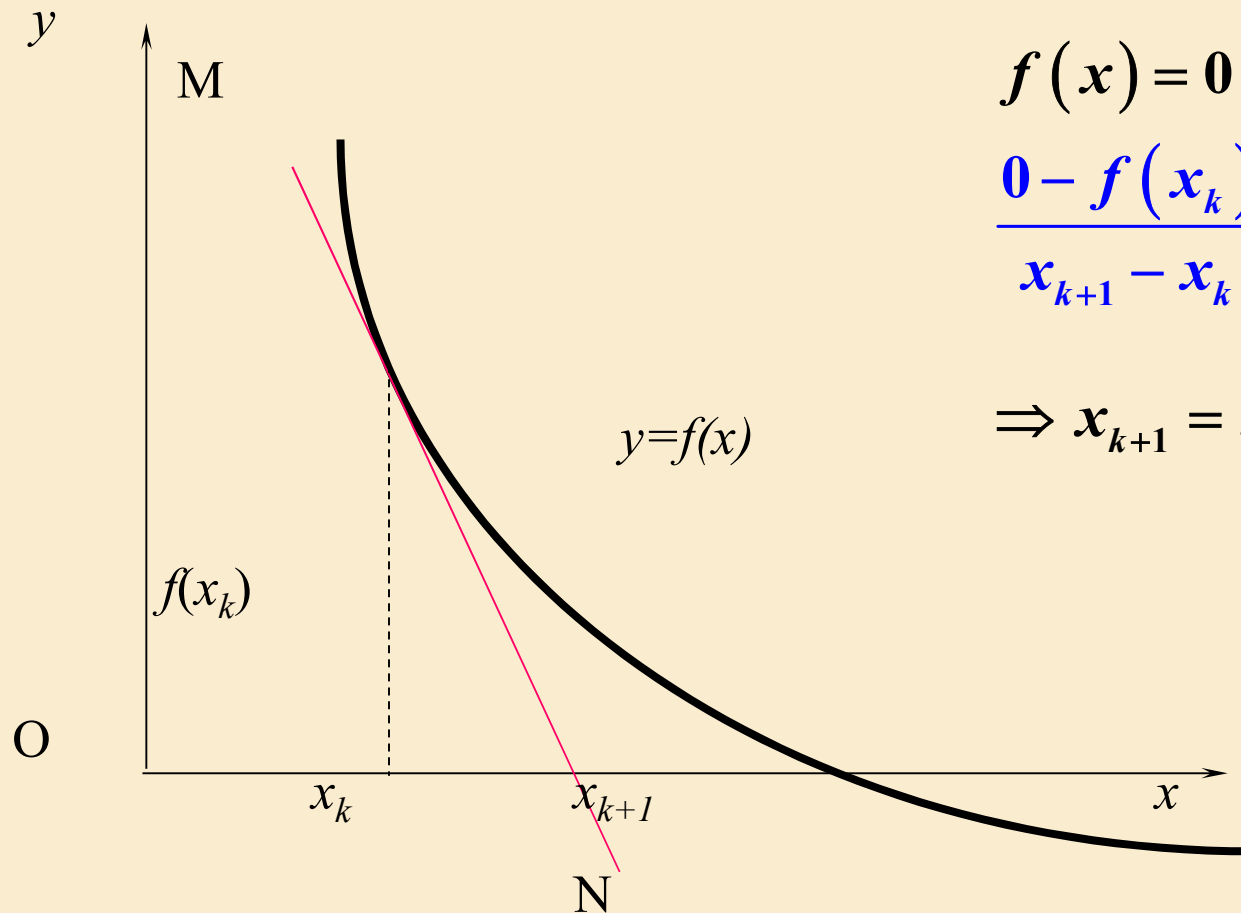


```
[x,k,r]=bisect(@ (x) x^6-2*x^4-6*x^3-13*x^2+8*x+12,-2,-1.5,1e-6);
```

-2.0000	-1.5000	24.0000	-7.7344
-1.7500	-1.5000	0.3088	-7.7344
-1.7500	-1.6250	0.3088	-5.1150
-1.7500	-1.6875	0.3088	-2.8132
-1.7500	-1.7188	0.3088	-1.3630
-1.7500	-1.7344	0.3088	-0.5558
-1.7500	-1.7422	0.3088	-0.1308
-1.7461	-1.7422	0.0872	-0.1308
-1.7461	-1.7441	0.0872	-0.0223
-1.7451	-1.7441	0.0323	-0.0223
-1.7446	-1.7441	0.0050	-0.0223



Newton 迭代法



$$f(x) = 0$$

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

$$\Rightarrow x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$



$$f(x) = 0 \Rightarrow x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

计算 $\sqrt{3}$

$$x^2 = 3 \Rightarrow x^2 - 3 = 0 \Rightarrow x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} = x_k - \frac{x_k^2 - 3}{2x_k} = \frac{x_k}{2} + \frac{3}{2x_k}$$

2.0000000000000000

1.7500000000000000

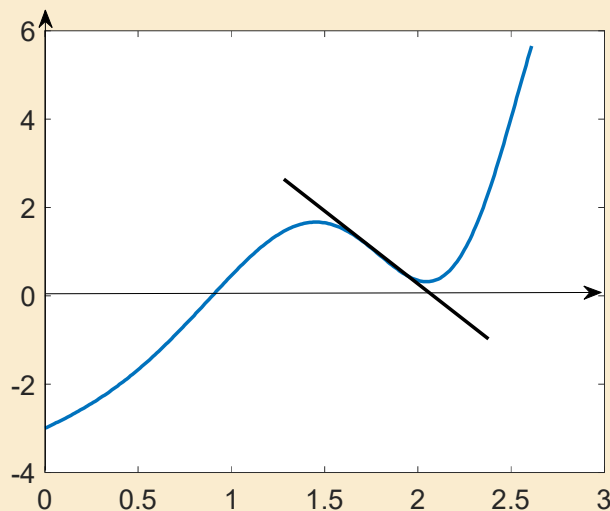
1.732142857142857

1.732050810014727

1.732050807568877

1.732050807568877

1.732050807568877



局部收敛性：依赖初值,只在解的附近收敛

全局收敛性：不依赖初值



非线性方程迭代法的基本思想

将原方程 $f(x) = 0$ 改写成容易迭代的形式 $x = \varphi(x)$ ，选合适的初值 x_0 ，进行迭代： $x_{k+1} = \varphi(x_k)$ ($k = 0, 1, 2, \dots$)

例1

$$f(x) = x^2 + x - 14 = 0 \quad \Rightarrow x = \varphi(x)$$

$$f(3) = -2, f(4) = 6 \quad \text{存在根 } x \in (3, 4)$$

$$x = \varphi_1(x) = 14 - x^2, \text{ 迭代公式: } x_{k+1} = 14 - x_k^2$$

$$x = \varphi_2(x) = 14/(x+1), \text{ 迭代公式: } x_{k+1} = 14/(x_k + 1)$$

$$x = \varphi_3(x) = x - (x^2 + x - 14)/(2x + 1),$$

$$\text{迭代公式: } x_{k+1} = x_k - (x_k^2 + x_k - 14)/(2x_k + 1)$$



非线性方程的迭代法（例）

	x_0	x_1	x_2	x_3	x_4	x_5
φ_1	3.0000	5.0000	-11.0000	-107.0000		
φ_2	3.0000	3.5000	3.1111	3.4054	3.1779	3.3510
φ_3	3.0000	3.2857	3.2749	3.2749	3.2749	3.2749

$$x_{1,2} = \frac{(-1 \pm \sqrt{1+14 \times 4})}{2}, x^* = \frac{(-1 + \sqrt{57})}{2} \approx 3.2749$$

φ_1 根本不收敛； φ_2 虽呈现收敛趋势，但很慢； φ_3 收敛很快。



局部收敛性

定义: 设 ϕ 在区间 I 有不动点 x^* , 若存在 x^* 的一个邻域 $S \subset I$, 对任意的 $x_0 \in S$, 不动点迭代法 $x_{k+1} = \phi(x_k)$, $k = 0, 1, \dots$ 产生的序列 $\{x_k\} \subset S$, 且 $\{x_k\}$ 收敛到 x^* , 称此迭代法局部收敛.

定理: 设 x^* 为 ϕ 的不动点, $\phi'(x)$ 在 x^* 的某个邻域上存在且连续, 满足 $|\phi'(x^*)| < 1$ 则迭代法 $x_{k+1} = \phi(x_k)$, $k = 0, 1, \dots$ 局部收敛.

$\phi'(x)$ 连续 \Rightarrow 存在 $\varepsilon > 0$, $\forall x \in [x^* - \varepsilon, x^* + \varepsilon]$, 有 $|\phi'(x)| \leq L < 1$,

$$|x_{k+1} - x^*| = |\phi(x_k) - x^*| = |\phi(x_k) - \phi(x^*)| \leq L |x_k - x^*| \leq \dots \leq L^{k+1} |x_0 - x^*|$$

$$x_0 \in [x^* - \varepsilon, x^* + \varepsilon], x_1 \in [x^* - \varepsilon, x^* + \varepsilon], \dots$$



迭代法的收敛速度（收敛阶）

定义：设序列 $\{x_k\}$ 收敛到 x^* ，记误差 $e_k = x_k - x^*$ 。若存在实数 $p \geq 1$ 及非零的常数 C ，使

$$\lim_{k \rightarrow \infty} \frac{|e_{k+1}|}{|e_k|^p} = C,$$

则称 $\{x_k\}$ 为 p 阶收敛， C 称为见近误差常数

$p = 1$ 线性收敛， $|C| < 1$ ，思考：如果 $C = 0$ 表示什么？

$p > 1$ 超线性收敛

$p = 2$ 平方收敛



利用泰勒(Taylor)展开分析收敛速度

定理: 设 x^* 为 ϕ 的不动点, 整数 $p > 1$, $\phi^{(p)}(x)$ 在 x^* 的邻域上连续, 且满足 $\phi^{(k)}(x^*) = 0$, $k = 1, \dots, p-1$, $\phi^{(p)}(x^*) \neq 0$

则由迭代法 $x_{k+1} = \phi(x_k)$, $k = 0, 1, \dots$ 产生的序列 $\{x_k\}$ 在 x^* 的邻域上是 p 阶收敛的, 且有

$$\lim_{k \rightarrow \infty} \frac{e_{k+1}}{e_k^p} = \frac{\phi^{(p)}(x^*)}{p!}$$

$$\phi(x_k) = \phi(x^*) + \phi'(x^*)(x_k - x^*) + \dots + \frac{\phi^{(p-1)}(x^*)}{(p-1)!}(x_k - x^*)^{p-1} + \frac{\phi^{(p)}(\xi)}{p!}(x_k - x^*)^p$$

$$e_{k+1} = x_{k+1} - x^* = \phi(x_k) - \phi(x^*) = \frac{\phi^{(p)}(\xi)}{p!}(x_k - x^*)^p = \frac{\phi^{(p)}(\xi)}{p!} e_k^p$$



迭代法的收敛速度 (例)

例题 $f(x) = x^2 + x - 14 = 0$ $\phi_2(x) = \frac{14}{x+1}$

$$\phi_2'(x) = \frac{-14}{(x+1)^2}, |\phi_2'(x^*)| < 1 \Rightarrow \{x_k\} \text{ 1阶收敛}$$

$$\phi_3(x) = x - (x^2 + x - 14)/(2x + 1)$$

$$\phi_3'(x) = \frac{2(x^2 + x - 14)}{(2x + 1)^2}, \phi_3'(x^*) = 0,$$

$$\phi_3''(x^*) \neq 0 \Rightarrow \{x_k\} \text{ 2阶收敛}$$

结论: $\phi(x)$ 的构造决定收敛速度



牛顿迭代法（切线法）

考虑函数 $f(x)$ 在 x 附近的Taylor展开式,

只取到线性项 截断 $f(x+h) \approx f(x) + h f'(x)$

Newton迭代公式:
$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x)}$$

单根情况, Newton迭代具有2阶收敛性

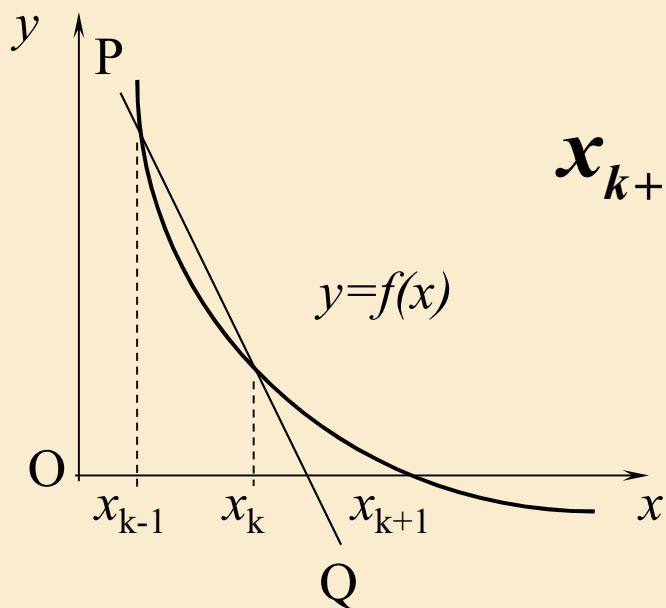


若 x^* 为重根 $f(x^*) = f'(x^*) = 0 \Leftrightarrow \varphi'(x^*) \neq 0$

牛顿切线法1阶收敛 重数越高, 收敛越慢

牛顿 割线法

用差商 $\frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}$ 代替 $f'(x_k)$



$$x_{k+1} = x_k - \frac{f(x_k)(x_k - x_{k-1})}{f(x_k) - f(x_{k-1})}$$

收敛速度比牛顿切线法稍慢

x^* 为单根时收敛阶数是 1.618



求解 $f(x)=0$ 的newton.m文件

```
function [r,k,x]=newton(fv,df,x0,n,tol)
%
x(1)=x0; b=1; k=1;
while or(k==1,abs(b)>tol*abs(x(k)))
    x(k+1)=x(k)-feval(fv,x(k))/feval(df,x(k));
    b=x(k+1)-x(k);
    k=k+1;
    if(k>n)
        error('Error: Reached maximum iteration times');
    end
end
r=x(k-1);
[x1,k1,r1]=bisection(@(x) x.^2-5,2,3,1e-8);
[x2,k2,r2]=newton(@(x) x^2-5,@(x) 2*x,2,10,1e-8);
[x3,k3,r3]=secant(@(x) x^2-5,2,2.1,10, 1e-8);
```

fv是 $f(x)$ 的函数句柄，df是 $f'(x)$ 的函数句柄



解非线性方程组的牛顿法

解方程 $f(x)=0$

的牛顿切线法

$$f(x_{k+1}) = f(x_k) + f'(x_k)(x_{k+1} - x_k)$$

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

推广到解方程组

$$F(x) = 0, \quad x = [x_1, \cdots, x_n]^T, \quad F(x) = [f_1(x), \cdots, f_n(x)]^T$$

$$f_i(x^{k+1}) = f_i(x^k) + \frac{\partial f_i(x^k)}{\partial x_1} (x_1^{k+1} - x_1^k)$$

$$+ \cdots + \frac{\partial f_i(x^k)}{\partial x_n} (x_n^{k+1} - x_n^k), \quad i = 1, 2, \cdots, n$$



解非线性方程组的牛顿法

$$F(x) = 0, \quad x = [x_1, \cdots, x_n]^T, \quad F(x) = [f_1(x), \cdots, f_n(x)]^T$$

$$F'(x) = \left[\frac{\partial f_i}{\partial x_j} \right]_{n \times n} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \cdots & & \cdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}$$

$$F(x^k) + F'(x^k)(x^{k+1} - x^k) = 0$$

$$x^{k+1} = x^k + (x^{k+1} - x^k)$$

$$\Rightarrow F'(x^k)\Delta x^k = -F(x^k) \quad \Rightarrow \quad x^{k+1} = x^k + \Delta x^k$$



例 解 $x_1^2 + x_2^2 = 4$, $x_1^2 - x_2^2 = 1$

$$F(x) = \begin{bmatrix} x_1^2 + x_2^2 - 4 \\ x_1^2 - x_2^2 - 1 \end{bmatrix}, \quad F'(x) = 2 \begin{bmatrix} x_1 & x_2 \\ x_1 & -x_2 \end{bmatrix}$$

$$F'(x^k) \Delta x^k = -F(x^k), \quad \Delta x^k = -F'(x^k)^{-1} F(x^k)$$

取 $x^0 = (1, 1)^T$

$$x^{k+1} = x^k + \Delta x^k, \quad k = 0, 1, \dots$$

精确解 $x = (\sqrt{5/2}, \sqrt{3/2})^T$

$$= (1.58113883, 1.22474487)^T$$

k	x^k
0	(1.000000, 1.000000)
1	(1.750000, 1.250000)
2	(1.589286, 1.225000)
3	(1.581160, 1.224645)
4	(1.581139, 1.224745)
5	(1.581139, 1.224745)



Newton法解非线性方程组

```
x=[1;1];
```

```
for k=1:5
```

```
    jx=2*[[x(1,k), x(2,k)]; [x(1,k), -x(2,k)]];
```

```
    fx=[x(1,k)^2+x(2,k)^2-4; x(1,k)^2-x(2,k)^2-1];
```

```
    dx=-jx\fx;
```

```
    x(:,k+1)=x(:,k)+dx;
```

```
end
```

```
x
```



拟牛顿法(Quasi-Newton)

解方程 $f(x)=0$
的牛顿割线法

$$f(x_{k+1}) = f(x_k) + a_k(x_{k+1} - x_k)$$

$$x_{k+1} = x_k - \frac{f(x_k)}{a_k} \quad a_k = \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}$$

解方程组的拟牛顿法——用 A^k 代替 $F'(x^k)$

使 A^k 满足 $A^k(x^k - x^{k-1}) = F(x^k) - F(x^{k-1})$

矩阵 A^k (n^2 个未知数) 不能由这样的 n 个方程确定

用迭代方法 $A^k = A^{k-1} + \Delta A^{k-1}$ (ΔA 有不同的构造) 计算 A^k

再求 $x^{k+1} = x^k - (A^k)^{-1} F(x^k)$



MATLAB优化工具箱解非线性方程

fzero: 单变量方程 $f(x)=0$ 求根(变号点)

最简形式 $x = \text{fzero}(@f, x0)$

必须输入: 'f'为f.m函数名, 'x0'是迭代初值(或有根区间)

输出: 'x'是变号点的近似值(函数不连续时不一定是根)

一般形式 $[x, fv, ef, out] = \text{fzero}(@f, x0, opt, P1, P2, \dots)$

可选输入: "P1,P2,..."是传给f.m的参数(如果需要的话)

'opt'是一个结构变量, 控制参数(如精度TolX)

opt可用optimset设定, 不指定或指定为' []'时将采用缺省值

如: $opt = \text{optimset}('TolX', 1e-8)$

输出: 'fv'是函数值; 'ef'是程序停止运行的原因(1,0,-1);

'out'是一个结构变量, 包含:

iterations(迭代次数), funcCount (函数调用次数), algorithm(所用算法)



- `fzero(inline('x^3-2*x-5'),0)`
- `fzero(inline('x^3-2*x-5'),[1,3])`
- `fzero(@tan,[-1,1])`
- `fzero(@tan,[1,2])`
- `fzero(inline('x^2'),1)` (**ans=NaN**)
- `[x,fv,ef,out]=fzero(inline('x^3-2*x-5'),0)`
- **out = intervaliterations: 14; iterations: 10**
- **funcCount: 39;**
- **algorithm: 'bisection, interpolation'**
- **message: 'Zero found in the interval [-2.56, 2.56]'**



MATLAB优化工具箱解非线性方程组

fsolve: 多变量方程组 $F(x)=0$ 求解

$x = \text{fsolve}(@f, x0)$

最简形式

$[x, fv, ef, out, jac] = \text{fsolve}(@f, x0, opt, P1, P2, \dots)$

一般形式

输入 ---- 与fzero类似, 但:

'x0'是迭代初值,

'opt'中控制参数更多(如MaxFunEvals, MaxIter等)

输出 ---- 与fzero类似, 但:

'out'中还输出 **'firstorderopt'**, 即结果 (x 点) 处梯度向量的范数(实际上是1-范数, 即分量按绝对值取最大的值);

'jac' 输出 x 点所对应的雅可比矩阵



MATLAB优化工具箱解非线性方程

牛顿法

需自行编制程序，如对切线法编写名为 `newton.m` 的 `m` 文件

多项式求根

当 $f(x)$ 为多项式时可用

`r=roots(c)`

输入多项式的系数 c （按降幂排列），

输出 r 为 $f(x) = 0$ 的全部根；

`c=poly(r)`

输入 $f(x) = 0$ 的全部根 r ，

输出 c 为多项式的系数（按降幂排列）；



求解 $f(x)=0$ 的newton.m文件

```
function [y,z]=newton(fv,df,x0,n,tol)
x(1)=x0; b=1; k=1;
while or(k==1,abs(b)>tol*abs(x(k)))
    x(k+1)=x(k)-feval(fv,x(k))/feval(df,x(k));
    b=x(k+1)-x(k);
    k=k+1;
    if(k>n)
        error('Error: Reached maximum iteration times');break;
    end
end
y=x(k-1);
if nargout>1
    z=k-1;
end
```



Newton.m

```
[xx,k]=newton(inline('x^3-2*x-5'),
inline('3*x^2-2'),0.5,100,1e-6)
```

fv是 $f(x)$ 的函数句柄，df是 $f'(x)$ 的函数句柄



例 解 $x_1^2 + x_2^2 = 4$, $x_1^2 - x_2^2 = 1$

$$F(x) = \begin{bmatrix} x_1^2 + x_2^2 - 4 \\ x_1^2 - x_2^2 - 1 \end{bmatrix}, \quad F'(x) = 2 \begin{bmatrix} x_1 & x_2 \\ x_1 & -x_2 \end{bmatrix}$$

$$F'(x^k) \Delta x^k = -F(x^k), \quad x^{k+1} = x^k + \Delta x^k, \quad k = 0, 1, \dots$$

取 $x^0 = (1, 1)^T$

`[x,fv,ef,out,jac]`

`=fsolve(@duoyuanfun,x0)`

精确解 $x = (\sqrt{5/2}, \sqrt{3/2})^T$

$= (1.58113883, 1.22474487)^T$

k	x^k
0	(1.000000, 1.000000)
1	(1.750000, 1.250000)
2	(1.589286, 1.225000)
3	(1.581160, 1.224645)
4	(1.581139, 1.224745)
5	(1.581139, 1.224745)



利用矩阵特征值计算多项式零点

$$p(\lambda) = a_0 + a_1\lambda + \cdots + a_{n-1}\lambda^{n-1} + \lambda^n$$

$$\begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ -a_0 & -a_1 & -a_2 & \cdots & -a_{n-1} \end{bmatrix}$$

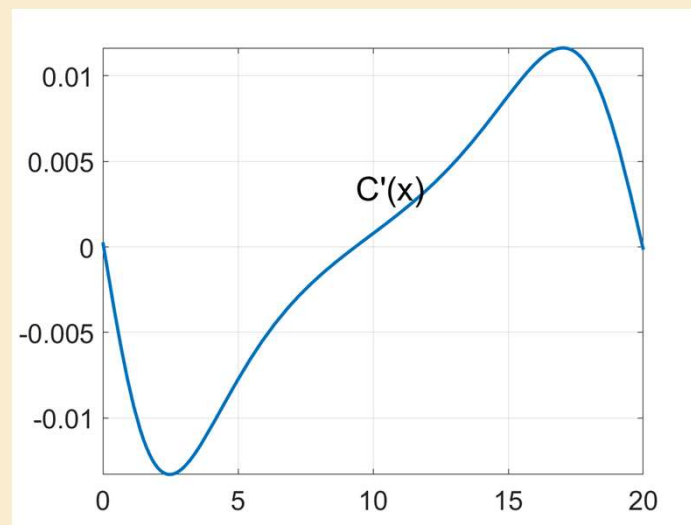
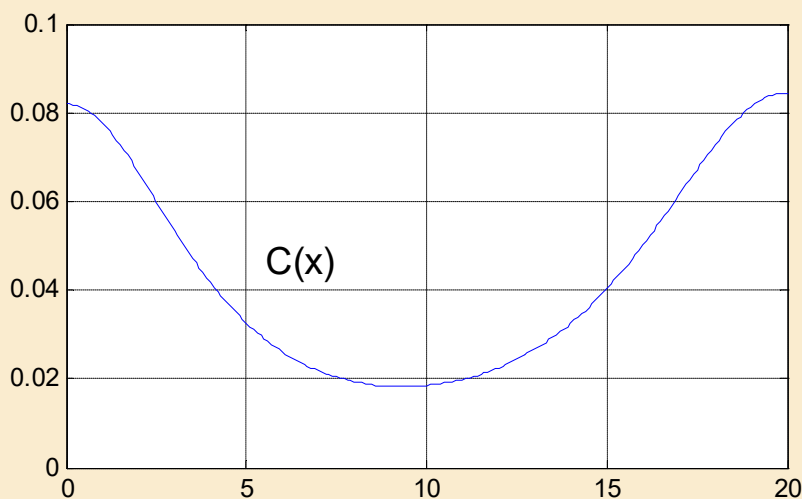
$$x^6 - 2x^4 - 6x^3 - 13x^2 + 8x + 12 = 0$$

$$\mathbf{x} = \text{roots}([1 \ 0 \ -2 \ -6 \ -13 \ 8 \ 12])$$



实例 路灯照明

$$C(x) = \frac{P_1 h_1}{\sqrt{(h_1^2 + x^2)^3}} + \frac{P_2 h_2}{\sqrt{(h_2^2 + (s-x)^2)^3}} \quad \begin{matrix} P_1 = 2, & P_2 = 3 \\ h_1 = 5, & h_2 = 6, & s = 20 \end{matrix}$$
$$C'(x) = -3 \frac{P_1 h_1 x}{\sqrt{(h_1^2 + x^2)^5}} + 3 \frac{P_2 h_2 (s-x)}{\sqrt{(h_2^2 + (s-x)^2)^5}} = 0$$



$C(x)$ 有3个驻点: (9,10)内的是最小点, 0或20附近的是最大点



实例 路灯照明

```
function y=zhaoming(x)    % C'(x)
p1=2;p2=3;h1=5;h2=6;s=20;
y=-3*p1*h1*x/(h1^2+x^2)^(5/2)+3*p2*h2*(s-x)/(h2^2+(s-x)^2)^(5/2);
x0=[0,10,20];
for k=1:3
    x(k)=fzero(@zhaoming,x0(k));
    c(k)=feval(@zhaoming_fun,x(k));
end
[x';c]
```

x	0	0.02848997	9.33829914	19.97669581	20
$C(x)$	0.08197716	0.08198104	0.01824393	0.08447655	0.08447468

$x = 9.3383$ 是 $C(x)$ 的最小值点, $x = 19.9767$ 是 $C(x)$ 的最大值点



实例 路灯照明

问题： $P_2=3$ 千瓦路灯的高度在 3~9 米变化，如何使路面上最暗点的照度最大？

$$C(x, h_2) = \frac{P_1 h_1}{\sqrt{(h_1^2 + x^2)^3}} + \frac{P_2 h_2}{\sqrt{(h_2^2 + (s-x)^2)^3}}$$

$$\frac{\partial C}{\partial h_2} = \frac{P_2}{\sqrt{(h_2^2 + (s-x)^2)^3}} - 3 \frac{P_2 h_2^2}{\sqrt{(h_2^2 + (s-x)^2)^5}} = 0 \Rightarrow h_2 = \frac{1}{\sqrt{2}}(s-x)$$

$$\frac{\partial C}{\partial x} = -3 \frac{P_1 h_1 x}{\sqrt{(h_1^2 + x^2)^5}} + 3 \frac{P_2 h_2 (s-x)}{\sqrt{(h_2^2 + (s-x)^2)^5}} = 0 \Rightarrow \frac{9\sqrt{3}P_1 h_1 x}{\sqrt{(h_1^2 + x^2)^5}} - \frac{4P_2}{(s-x)^3} = 0$$

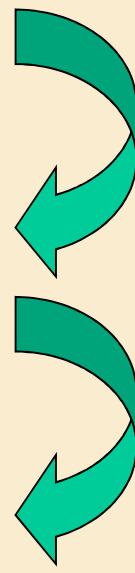
用 **fzero** 命令解方程，得到的结果是：

$x=9.5032$, $h_2=7.4224$, $C(x, h_2)=0.018556$ (最暗点的最大照度)



实例 路灯照明

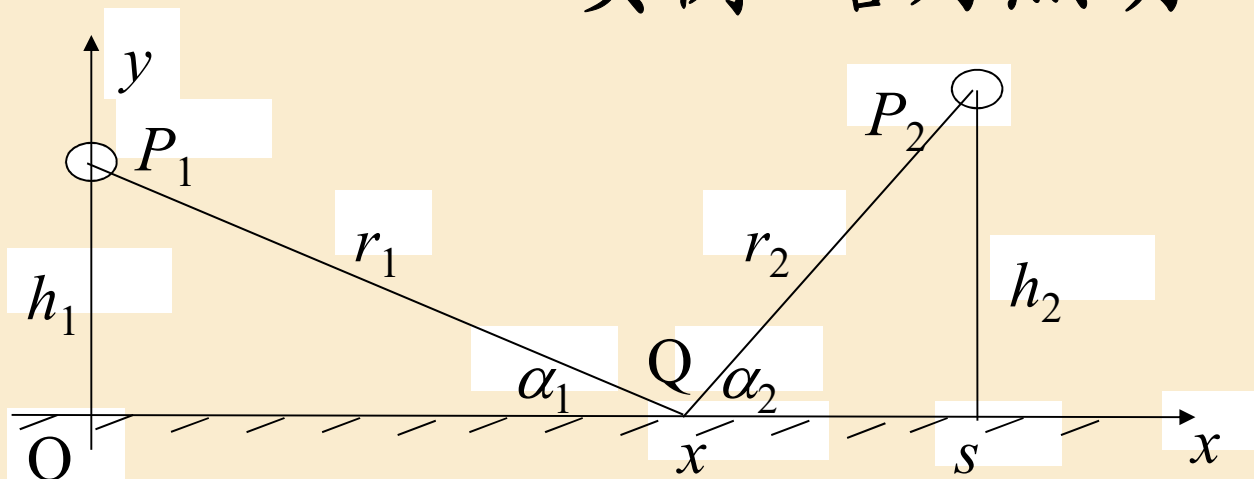
问题：讨论两只路灯的高度均可以在3~9米之间变化的情况

$$C(x, h_1, h_2) = \frac{P_1 h_1}{\sqrt{(h_1^2 + x^2)^3}} + \frac{P_2 h_2}{\sqrt{(h_2^2 + (s-x)^2)^3}}$$
$$\frac{\partial C}{\partial h_1} = 0 \Rightarrow h_1 = \frac{1}{\sqrt{2}} x$$
$$\frac{\partial C}{\partial h_2} = \frac{P_2}{\sqrt{(h_2^2 + (s-x)^2)^3}} - 3 \frac{P_2 h_2^2}{\sqrt{(h_2^2 + (s-x)^2)^5}} = 0 \Rightarrow h_2 = \frac{1}{\sqrt{2}} (s-x)$$
$$\frac{\partial C}{\partial x} = -3 \frac{P_1 h_1 x}{\sqrt{(h_1^2 + x^2)^5}} + 3 \frac{P_2 h_2 (s-x)}{\sqrt{(h_2^2 + (s-x)^2)^5}} = 0 \Rightarrow \frac{P_1}{x^3} = \frac{P_2}{(s-x)^3}$$
$$x = \frac{\sqrt[3]{P_1} s}{\sqrt[3]{P_1} + \sqrt[3]{P_2}}$$


实际数据计算，得到 $x=9.3253$ ，最暗点的照度达到最大的路灯高度 $h_1=6.5940$, $h_2=7.5482$



实例 路灯照明



$$h_1 = \frac{1}{\sqrt{2}} x$$

$$h_2 = \frac{1}{\sqrt{2}} (s - x)$$

$$x = \frac{\sqrt[3]{P_1} s}{\sqrt[3]{P_1} + \sqrt[3]{P_2}}$$

讨论1: 若 $P_1=P_2$, 则 $x=0.5s$ (中点), 与直觉符合

讨论2: $\tan \alpha_1 = \tan \alpha_2 = \sqrt{2}/2$ $\alpha_1 = \alpha_2 = 35^\circ 16'$
(这个角度与路灯的功率和道路长度均无关)

思考: 2只以上路灯的情形 (如篮球场四周安装照明灯)



对称不定问题的不精确 Newton 法^{*1)}

梁 恒 白峰杉

(清华大学数学科学系, 北京 100084)

INEXACT NEWTON METHOD FOR SYMMETRIC INDEFINITE PROBLEMS

Liang Heng Bai Fengshan

(*Department of Mathematical Sciences, Tsinghua University, Beijing, 100084*)

Abstract

Inexact Newton methods for symmetric indefinite problems are studied in this paper. Theoretical analysis and numerical computations show that better performance could be achieved if attentions are paid on the special structure of such class of problems. Newton-MINRES method behaves well among Newton-Krylov subspace methods for symmetric indefinite problems.

Keywords: inexact Newton method, Newton-Krylov method, symmetric indefinite problem

关键词: 不精确 Newton 法, Newton-Krylov 方法, 对称不定问题





1. 引言

非线性方程组 $F(x) = 0$ 的数值求解, 经典的算法是 Newton 迭代:

$$x_{k+1} = x_k + s_k, \quad k = 0, 1, 2, \dots, \quad (1.1)$$

其中的 s_k 满足

$$F'(x_k)s_k = -F(x_k), \quad k = 0, 1, 2, \dots. \quad (1.2)$$

这里 x_0 为迭代的初始点, $\{x_k\}$ 称为 Newton 迭代序列.

当变量个数比较多时, 每一步 Newton 迭代中计算 *Jacobi* 矩阵 $F'(x_k)$ 和求解线性方程组 (1.2) 的代价非常高; 特别当 x_k 远离方程组的解 x_* 时, 高精度地求解线性方程组 (1.2) 所得到的 Newton 迭代点, 带有相当的盲目性. 80 年代初开始, Dembo 等讨论了一类改进的 Newton 型方法^[3,2,6], 试图克服上述缺陷. 算法的基本思想是, 每一步 Newton 迭代, 只近似地求解线性方程组 (1.2), 要求残差向量 $r_k = F'(x_k)s_k + F(x_k)$ 满足

$$\frac{\|r_k\|}{\|F(x_k)\|} \leq \eta_k, \quad \eta_k \in [0, 1), \quad (1.3)$$



2. 不精确 Newton 法

求解非线性方程组 $F(x) = 0$ 的不精确 Newton 法, 算法描述如下:

算法 IN (不精确 Newton 法)

设 x_0 给定从 $k = 0$ 起, 步长为 1, 直到迭代收敛, 执行:

选取恰当的 $\eta_k \in [0, 1)$ 和 s_k , 使它们满足

$$\|F'(x_k)s_k + F(x_k)\| \leq \eta_k \|F(x_k)\|$$

令 $x_{k+1} = x_k + s_k$.

这里 η_k 常依赖于 x_k . 如果令 $\eta_k \equiv 0$, 则上述方法即为经典的 Newton 法.

在保持 Newton 法快速收敛性质的前提下, 每一步线性方程组的求解究竟需要精确到什么程度. [3] 给出了不精确 Newton 法的局部收敛定理.

Krylov子空间方法

$$\{r^{(0)}, r^{(1)}, r^{(2)}, \dots, r^{(k)}\} = \text{span}\{r^{(0)}, Ar^{(0)}, A^2r^{(0)}, \dots, A^k r^{(0)}\}$$

$$x^{(k+1)} = \min_{x \in \text{span}\{r^{(0)}, r^{(1)}, \dots, r^{(k)}\}} \|Ax - b\|_2$$

不精确 Newton 法的每一步迭代, 就是近似求解一个线性方程组, 所以各种线性方程组的迭代解法都可以用来实现这种算法. 注意到 Krylov 子空间方法求解线性方程组 $Ax = b$ 时, 可以不必知道矩阵 A 的显式形式, 只要能够得到 A 或 A^T 与任意向量 v 的乘积即可. 由于 Krylov 子空间方法的这一特性, 使得它在不精确 Newton 法的实现上成为首选算法. 它可以避开 *Jacobi* 矩阵 $F'(x_c)$ 的显式计算和存储, 直接利用差分可以得到 $F'(x_c)$ 与向量的乘积, 从而实现计算过程中没有显式的矩阵存储与计算, 这里称之为无矩阵算法.

$$F'(x_c)v \approx \frac{F(x_c + \sigma v) - F(x_c)}{\sigma},$$

其中 σ 为适当选取的一标量.

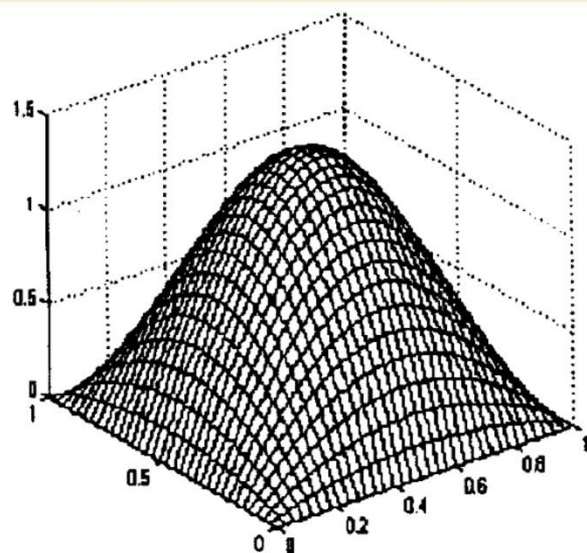
4.1 Bratu 方程的计算

首先考虑求解非线性偏微分方程

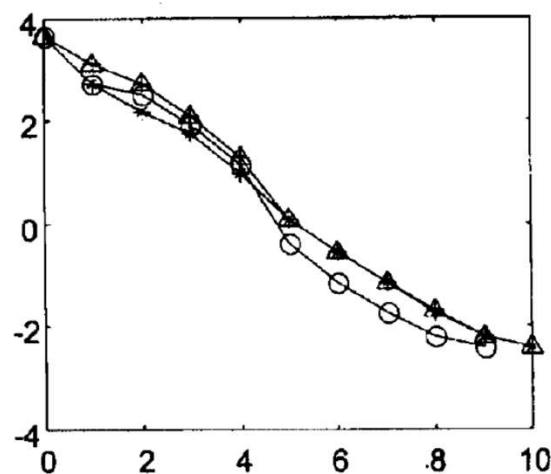
$$-\Delta u = \lambda e^u, \quad \Omega = (0, 1) \times (0, 1), \quad u|_{\partial\Omega} = 0, \quad (4.1)$$

其中 Δ 是 Laplace 算子. 这个方程被称为 Bratu 方程. 当 $\lambda < 0$ 时, 方程 (4.1) 有唯一解; 当 $\lambda > 0$ 时, (4.1) 可能有一个、多个或无解. 当 $\lambda = 6.80812$ 时, 在解的附近 *Jacobi* 矩阵的条件数很坏, 因此是一个很好的检验算法的计算实例. 构造区域 Ω 上的均匀网格, 采用标准的五点差分格式离散方程 (4.1), 并取 $n = 32$.

取一随机向量为初值, 精度要求取为 10^{-6} . 方程解与各算法的迭代过程如图 4.1 所示. 图



Bratu 方程的解



Bratu 方程的收敛过程

图 4.1 Bratu 方程的解 ($\lambda = 6.80812, n = 32$)



布置实验

- 目的**
- 1) 用MATLAB软件掌握求解非线性方程的迭代法和牛顿法，并对结果作初步分析；
理解局部收敛性和收敛速度的概念
 - 2) 通过实例练习用非线性方程求解的实际问题。

内容 见网络学堂

