第九章　插值与拟合

插值：求过已知有限个数据点的近似函数。

拟合：已知有限个数据点，求近似函数，不要求过已知数据点，只要求在某种意义下它在这些点上的总偏差最小。

插值和拟合都是要根据一组数据构造一个函数作为近似，由于近似的要求不同，二者的数学方法上是完全不同的。而面对一个实际问题，究竟应该用插值还是拟合，有时容易确定，有时则并不明显。

§1 插值方法

下面介绍几种基本的、常用的插值：拉格朗日多项式插值、牛顿插值、分段线性插值、Hermite 插值和三次样条插值。

1.1　拉格朗日多项式插值

1.1.1　插值多项式

用多项式作为研究插值的工具，称为代数插值。其基本问题是：已知函数 $f(x)$ 在区间 $[a,b]$ 上 $n+1$ 个不同点 $x_0,x_1,\cdots,x_n$ 处的函数值 $y_i=f(x_i)\ (i=0,1,\cdots,n)$，求一个至多 n 次多项式

$$\varphi_n(x)=a_0+a_1x+\cdots+a_nx^n \tag{1}$$

使其在给定点处与 $f(x)$ 同值，即满足插值条件

$$\varphi_n(x_i)=f(x_i)=y_i\quad (i=0,1,\cdots,n) \tag{2}$$

$\varphi_n(x)$ 称为插值多项式，$x_i(i=0,1,\cdots,n)$ 称为插值节点，简称节点，$[a,b]$ 称为插值区间。从几何上看，n 次多项式插值就是过 $n+1$ 个点 $(x_i,f(x_i))\ (i=0,1,\cdots,n)$，作一条多项式曲线 $y=\varphi_n(x)$ 近似曲线 $y=f(x)$。

n 次多项式（1）有 $n+1$ 个待定系数，由插值条件（2）恰好给出 $n+1$ 个方程

$$\begin{cases} a_0+a_1x_0+a_2x_0^2+\cdots+a_nx_0^n=y_0 \\ a_0+a_1x_1+a_2x_1^2+\cdots+a_nx_1^n=y_1 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_0+a_1x_n+a_2x_n^2+\cdots+a_nx_n^n=y_n \end{cases} \tag{3}$$

记此方程组的系数矩阵为 A，则

$$\det(A)=\begin{vmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ & \cdots\cdots\cdots\cdots & & & \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{vmatrix}$$

是范德蒙特(Vandermonde)行列式。当 $x_0,x_1,\cdots,x_n$ 互不相同时，此行列式值不为零。因此方程组（3）有唯一解。这表明，只要 $n+1$ 个节点互不相同，满足插值要求（2）的插值多项式（1）是唯一的。

插值多项式与被插函数之间的差

$$R_n(x)=f(x)-\varphi_n(x)$$

称为截断误差，又称为插值余项。当 $f(x)$ 充分光滑时，

$$R_n(x)=f(x)-L_n(x)=\frac{f^{(n+1)}(\xi)}{(n+1)!}\omega_{n+1}(x),\ \xi\in(a,b)$$

其中 $\omega_{n+1}(x)=\prod_{j=0}^{n}(x-x_j)$。

1.1.2　拉格朗日插值多项式

实际上比较方便的作法不是解方程（3）求待定系数，而是先构造一组基函数

$$l_i(x)=\frac{(x-x_0)\cdots(x-x_{i-1})(x-x_{i+1})\cdots(x-x_n)}{(x_i-x_0)\cdots(x_i-x_{i-1})(x_i-x_{i+1})\cdots(x_i-x_n)}$$

$$=\prod_{\substack{j=0\\ j\neq i}}^{n}\frac{x-x_j}{x_i-x_j}\quad,\qquad(i=0,1,\cdots,n)$$

$l_i(x)$ 是 n 次多项式，满足

$$l_i(x_j)=\begin{cases}0 & j\neq i\\ 1 & j=i\end{cases}$$

令

$$L_n(x)=\sum_{i=0}^{n}y_i l_i(x)=\sum_{i=0}^{n}y_i\left(\prod_{\substack{j=0\\ j\neq i}}^{n}\frac{x-x_j}{x_i-x_j}\right)\qquad(4)$$

上式称为 n 次 Lagrange 插值多项式，由方程（3）解的唯一性，$n+1$ 个节点的 n 次 Lagrange 插值多项式存在唯一。

1.1.3　用 Matlab 作 Lagrange 插值

Matlab 中没有现成的 Lagrange 插值函数，必须编写一个 M 文件实现 Lagrange 插值。

设 n 个节点数据以数组 $x0, y0$ 输入（注意 Matlat 的数组下标从 1 开始），m 个插值点以数组 x 输入，输出数组 y 为 m 个插值。编写一个名为 lagrange.m 的 M 文件：

```
function y=lagrange(x0,y0,x);
n=length(x0);m=length(x);
for i=1:m
   z=x(i);
   s=0.0;
   for k=1:n
      p=1.0;
      for j=1:n
         if j~=k
            p=p*(z-x0(j))/(x0(k)-x0(j));
         end
      end
      s=p*y0(k)+s;
   end
   y(i)=s;
end
```

1.2　牛顿（Newton）插值

在导出 Newton 公式前，先介绍公式表示中所需要用到的差商、差分的概念及性质。

1.2.1 差商

定义　设有函数 $f(x), x_0, x_1, x_2, \cdots$ 为一系列互不相等的点，称 $\dfrac{f(x_i)-f(x_j)}{x_i-x_j}(i \neq j)$ 为 $f(x)$ 关于点 x_i, x_j 一阶差商（也称均差）记为 $f[x_i, x_j]$，即

$$f[x_i, x_j] = \frac{f(x_i)-f(x_j)}{x_i-x_j}$$

称一阶差商的差商

$$\frac{f[x_i, x_j]-f[x_j, x_k]}{x_i-x_k}$$

为 $f(x)$ 关于点 x_i, x_j, x_k 的二阶差商，记为 $f[x_i, x_j, x_k]$。一般地，称

$$\frac{f[x_0, x_1, \cdots, x_{k-1}]-f[x_1, x_2, \cdots, x_k]}{x_0-x_k}$$

为 $f(x)$ 关于点 $x_0, x_1, \cdots, x_k$ 的 k 阶差商，记为

$$f[x_0, x_1, \cdots, x_k] = \frac{f[x_0, x_1, \cdots, x_{k-1}]-f[x_1, x_2, \cdots, x_k]}{x_0-x_k}$$

容易证明，差商具有下述性质：

$$f[x_i, x_j] = f[x_j, x_i]$$

$$f[x_i, x_j, x_k] = f[x_i, x_k, x_j] = f[x_j, x_i, x_k]$$

1.2.2　Newton 插值公式

线性插值公式可表成

$$\varphi_1(x) = f(x_0) + (x-x_0) f[x_0, x_1]$$

称为一次 Newton 插值多项式。一般地，由各阶差商的定义，依次可得

$$f(x) = f(x_0) + (x-x_0) f[x, x_0]$$

$$f[x, x_0] = f[x_0, x_1] + (x-x_1) f[x, x_0, x_1]$$

$$f[x, x_0, x_1] = f[x_0, x_1, x_2] + (x-x_2) f[x, x_0, x_1, x_2]$$

$$\cdots\cdots$$

$$f[x, x_0, \cdots, x_{n-1}] = f[x_0, x_1, \cdots, x_n] + (x-x_n) f[x, x_0, \cdots, x_n]$$

将以上各式分别乘以 $1, (x-x_0), (x-x_0)(x-x_1), \cdots, (x-x_0)(x-x_1)\cdots(x-x_{n-1})$，然后相加并消去两边相等的部分，即得

$$\begin{aligned} f(x) = {} & f(x_0) + (x-x_0) f[x_0, x_1] + \cdots \\ & + (x-x_0)(x-x_1)\cdots(x-x_{n-1}) f[x_0, x_1, \cdots, x_n] \\ & + (x-x_0)(x-x_1)\cdots(x-x_n) f[x, x_0, x_1, \cdots, x_n] \end{aligned}$$

记

$$N_n(x)=f(x_0)+(x-x_0)f[x_0,x_1]+\cdots$$
$$+(x-x_0)(x-x_1)\cdots(x-x_{n-1})f[x_0,x_1,\cdots,x_n]$$
$$R_n(x)=(x-x_0)(x-x_1)\cdots(x-x_n)f[x,x_0,x_1,\cdots,x_n]$$
$$=\omega_{n+1}(x)f[x,x_0,x_1,\cdots,x_n]$$

显然，$N_n(x)$ 是至多 n 次的多项式，且满足插值条件，因而它是 $f(x)$ 的 n 次插值多项式。这种形式的插值多项式称为 Newton 插值多项式。$R_n(x)$ 称为 Newton 插值余项。

Newton 插值的优点是：每增加一个节点，插值多项式只增加一项，即

$$N_{n+1}(x)=N_n(x)+(x-x_0)\cdots(x-x_n)f[x_0,x_1,\cdots,x_{n+1}]$$

因而便于递推运算。而且 Newton 插值的计算量小于 Lagrange 插值。

由插值多项式的唯一性可知，Newton 插值余项与 Lagrange 余项也是相等的，即

$$R_n(x)=\omega_{n+1}(x)f[x,x_0,x_1,\cdots,x_n]$$
$$=\frac{f^{(n+1)}(\xi)}{(n+1)!}\omega_{n+1}(x)\qquad \xi\in(a,b)$$

由此可得差商与导数的关系

$$f[x_0,x_1,\cdots,x_n]=\frac{f^{(n)}(\xi)}{n!}$$

其中 $\xi\in(\alpha,\beta),\alpha=\min\limits_{0\le i\le n}\{x_i\},\beta=\max\limits_{0\le i\le n}\{x_i\}$。

1.2.3　差分

当节点等距时，即相邻两个节点之差（称为步长）为常数，Newton 插值公式的形式会更简单。此时关于节点间函数的平均变化率（差商）可用函数值之差（差分）来表示。

定义　设有等距节点 $x_k=x_0+kh(k=0,1,\cdots,n)$，步长 h 为常数，$f_k=f(x_k)$。称相邻两个节点 x_k,x_{k+1} 处的函数值的增量 $f_{k+1}-f_k(k=0,1,\cdots,n-1)$ 为函数 $f(x)$ 在点 x_k 处以 h 为步长的一阶差分，记为 Δf_k，即

$$\Delta f_k=f_{k+1}-f_k\qquad(k=0,1,\cdots,n)$$

类似地，定义差分的差分为高阶差分。如二阶差分为

$$\Delta^2 f_k=\Delta f_{k+1}-\Delta f_k\qquad(k=0,1,\cdots,n-2)$$

一般地，m 阶差分为

$$\Delta^m f_k=\Delta^{m-1}f_{k+1}-\Delta^{m-1}f_k\qquad(k=2,3,\cdots),$$

上面定义的各阶差分又称为向前差分。常用的差分还有两种：

$$\nabla f_k=f_k-f_{k-1}$$

称为 $f(x)$ 在 x_k 处以 h 为步长的向后差分；

$$\delta f_k=f\left(x_k+\frac{h}{2}\right)-f\left(x_k-\frac{h}{2}\right)$$

称为 $f(x)$ 在 x_k 处以 h 为步长的中心差分。一般地，m 阶向后差分与 m 阶中心差分公式为

$$\nabla^m f_k = \nabla^{m-1} f_k - \nabla^{m-1} f_{k-1}$$
$$\delta^m f_k = \delta^{m-1} f_{k+\frac{1}{2}} - \delta^{m-1} f_{k-\frac{1}{2}}$$

差分具有以下性质：

（i）各阶差分均可表成函数值的线性组合，例如

$$\Delta^m f_k = \sum_{j=0}^{m} (-1)^j \binom{m}{j} f_{k+m-j}$$
$$\nabla^m f_k = \sum_{j=0}^{m} (-1)^j \binom{m}{j} f_{k-j}$$

（ii）各种差分之间可以互化。向后差分与中心差分化成向前差分的公式如下：

$$\nabla^m f_k = \Delta^m f_{k-m}$$
$$\delta^m f_k = \Delta^m f_{k-\frac{m}{2}}$$

1.2.4　等距节点插值公式

如果插值节点是等距的，则插值公式可用差分表示。设已知节点 $x_k = x_0 + kh \ (k = 0,1,2,\cdots,n)$，则有

$$\begin{aligned} N_n(x) &= f(x_0) + f[x_0, x_1](x - x_0) \\ &\quad + \cdots + f[x_0, x_1, \cdots, x_n](x - x_0)(x - x_1)\cdots(x - x_{n-1}) \\ &= f_0 + \frac{\Delta f_0}{h}(x - x_0) + \cdots + \frac{\Delta^n f_0}{n! h^n}(x - x_0)(x - x_1)\cdots(x - x_{n-1}) \end{aligned}$$

若令 $x = x_0 + th$，则上式又可变形为

$$N_n(x_0 + th) = f_0 + t\Delta f_0 + \cdots + \frac{t(t-1)\cdots(t-n+1)}{n!}\Delta^n f_0$$

上式称为 Newton 向前插值公式。

1.3　分段线性插值

1.3.1　插值多项式的振荡

用 Lagrange 插值多项式 $L_n(x)$ 近似 $f(x)(a \le x \le b)$，虽然随着节点个数的增加，$L_n(x)$ 的次数 n 变大，多数情况下误差 $|R_n(x)|$ 会变小。但是 n 增大时，$L_n(x)$ 的光滑性变坏，有时会出现很大的振荡。理论上，当 $n \to \infty$，在 $[a,b]$ 内并不能保证 $L_n(x)$ 处处收敛于 $f(x)$。Runge 给出了一个有名的例子：

$$f(x) = \frac{1}{1 + x^2}, \quad x \in [-5,5]$$

对于较大的 $|x|$，随着 n 的增大，$L_n(x)$ 振荡越来越大，事实上可以证明，仅当 $|x| \le 3.63$ 时，才有 $\lim_{n\to\infty} L_n(x) = f(x)$，而在此区间外，$L_n(x)$ 是发散的。

高次插值多项式的这些缺陷，促使人们转而寻求简单的低次多项式插值。

1.3.2　分段线性插值

简单地说，将每两个相邻的节点用直线连起来，如此形成的一条折线就是分段线性插值函数，记作 $I_n(x)$，它满足 $I_n(x_i) = y_i$，且 $I_n(x)$ 在每个小区间 $[x_i, x_{i+1}]$ 上是线性

函数$(i=0,1,\cdots,n)$。

$I_n(x)$可以表示为

$$I_n(x)=\sum_{i=0}^{n}y_i l_i(x)$$

$$l_i(x)=\begin{cases}\dfrac{x-x_{i-1}}{x_i-x_{i-1}}, & x\in[x_{i-1},x_i]\ (i=0\text{时舍去})\\ \dfrac{x-x_{i+1}}{x_i-x_{i+1}}, & x\in[x_i,x_{i+1}]\ (i=n\text{时舍去})\\ 0, & \text{其它}\end{cases}$$

$I_n(x)$有良好的收敛性，即对于$x\in[a,b]$有，

$\lim\limits_{n\to\infty}I_n(x)=f(x)$。

用$I_n(x)$计算x点的插值时，只用到x左右的两个节点，计算量与节点个数n无关。但n越大，分段越多，插值误差越小。实际上用函数表作插值计算时，分段线性插值就足够了，如数学、物理中用的特殊函数表，数理统计中用的概率分布表等。

1.3.3　用 Matlab 实现分段线性插值

用 Matlab 实现分段线性插值不需要编制函数程序，Matlab 中有现成的一维插值函数 interp1。

y=interp1(x0,y0,x,'method')

method 指定插值的方法，默认为线性插值。其值可为：

'nearest'　最近项插值

'linear'　线性插值

'spline'　逐段 3 次样条插值

'cubic'　保凹凸性 3 次插值。

所有的插值方法要求 x0 是单调的。

当 x0 为等距时可以用快速插值法，使用快速插值法的格式为'*nearest'、'*linear'、'*spline'、'*cubic'。

1.4　埃尔米特(Hermite)插值

1.4.1　Hermite 插值多项式

如果对插值函数，不仅要求它在节点处与函数同值，而且要求它与函数有相同的一阶、二阶甚至更高阶的导数值，这就是 Hermite 插值问题。本节主要讨论在节点处插值函数与函数的值及一阶导数值均相等的 Hermite 插值。

设已知函数$y=f(x)$在$n+1$个互异节点$x_0,x_1,\cdots,x_n$上的函数值$y_i=f(x_i)$ $(i=0,1,\cdots,n)$和导数值$y'_i=f'(x_i)$ $(i=0,1,\cdots,n)$，要求一个至多$2n+1$次的多项式$H(x)$，使得

$$H(x_i)=y_i \qquad H'(x_i)=y'_i \quad (i=0,1,\cdots,n)$$

满足上述条件的多项式$H(x)$称为 Hermite 插值多项式。

Hermite 插值多项式为

$$H(x)=\sum_{i=0}^{n}h_i[(x_i-x)(2a_iy_i-y'_i)+y_i]$$

其中 $h_i=\prod_{\substack{j=0\\j\neq i}}^{n}\left(\frac{x-x_j}{x_i-x_j}\right)^2$，$a_i=\sum_{\substack{j=0\\j\neq i}}^{n}\frac{1}{x_i-x_j}$。

1.4.2　用 Matlab 实现 Hermite 插值

Matlab 中没有现成的 Hermite 插值函数，必须编写一个 M 文件实现插值。

设 n 个节点的数据以数组 $x0$（已知点的横坐标），$y0$（函数值），$y1$（导数值）输入（注意 Matlat 的数组下标从 1 开始），m 个插值点以数组 x 输入，输出数组 y 为 m 个插值。编写一个名为 hermite.m 的 M 文件：

```
function y=hermite(x0,y0,y1,x);
n=length(x0);m=length(x);
for k=1:m
   yy=0.0;
   for i=1:n
      h=1.0;
      a=0.0;
      for j=1:n
         if j~=i
            h=h*((x(k)-x0(j))/(x0(i)-x0(j)))^2;
            a=1/(x0(i)-x0(j))+a;
         end
      end
      yy=yy+h*((x0(i)-x(k))*(2*a*y0(i)-y1(i))+y0(i));
   end
   y(k)=yy;
end
```

1.5　样条插值

许多工程技术中提出的计算问题对插值函数的光滑性有较高要求，如飞机的机翼外形，内燃机的进、排气门的凸轮曲线，都要求曲线具有较高的光滑程度，不仅要连续，而且要有连续的曲率，这就导致了样条插值的产生。

1.5.1　样条函数的概念

所谓样条（Spline）本来是工程设计中使用的一种绘图工具，它是富有弹性的细木条或细金属条。绘图员利用它把一些已知点连接成一条光滑曲线（称为样条曲线），并使连接点处有连续的曲率。

数学上将具有一定光滑性的分段多项式称为样条函数。具体地说，给定区间 $[a,b]$ 的一个分划

$$\Delta:\quad a=x_0<x_1<\cdots<x_{n-1}<x_n=b$$

如果函数 $s(x)$ 满足：

（i）在每个小区间 $[x_i,x_{i-1}](i=0,1,\cdots,n-1)$ 上 $s(x)$ 是 k 次多项式；

（ii）$s(x)$ 在 $[a,b]$ 上具有 $k-1$ 阶连续导数。

则称 $s(x)$ 为关于分划 Δ 的 k 次样条函数，其图形称为 k 次样条曲线。$x_0,x_1,\cdots,x_n$ 称为样条节点，$x_1,x_2,\cdots,x_{n-1}$ 称为内节点，x_0,x_n 称为边界点，这类样条函数的全体记做

$S_P(\Delta,k)$，称为k次样条函数空间。

显然，折线是一次样条曲线。

若$s(x)\in S_P(\Delta,k)$，则$s(x)$是关于分划Δ的k次多项式样条函数。k次多项式样条函数的一般形式为

$$s_k(x)=\sum_{i=0}^{k}\frac{\alpha_i x^i}{i!}+\sum_{j=1}^{n-1}\frac{\beta_j}{k!}(x-x_j)_+^k,$$

其中$\alpha_i(i=0,1,\cdots,k)$和$\beta_j(j=1,2,\cdots,n-1)$均为任意常数，而

$$(x-x_j)_+^k=\begin{cases}(x-x_j)^k, x\geq x_j\\ 0, \quad x<x_j\end{cases}，（j=1,2,\cdots,n-1）$$

在实际中最常用的是$k=2$和3的情况，即为二次样条函数和三次样条函数。

二次样条函数：对于$[a,b]$上的分划$\Delta: a=x_0<x_1<\cdots<x_n=b$，则

$$s_2(x)=\alpha_0+\alpha_1 x+\frac{\alpha_2}{2!}x^2+\sum_{j=1}^{n-1}\frac{\beta_j}{2!}(x-x_j)_+^2\in S_P(\Delta,2),\tag{5}$$

其中$(x-x_j)_+^2=\begin{cases}(x-x_j)^2, x\geq x_j\\ 0, \quad x<x_j\end{cases}$，（$j=1,2,\cdots,n-1$）。

三次样条函数：对于$[a,b]$上的分划$\Delta: a=x_0<x_1<\cdots<x_n=b$，则

$$s_3(x)=\alpha_0+\alpha_1 x+\frac{\alpha_2}{2!}x^2+\frac{\alpha_3}{3!}x^3+\sum_{j=1}^{n-1}\frac{\beta_j}{3!}(x-x_j)_+^3\in S_P(\Delta,3),\tag{6}$$

其中$(x-x_j)_+^3=\begin{cases}(x-x_j)^3, x\geq x_j\\ 0, \quad x<x_j\end{cases}$，（$j=1,2,\cdots,n-1$）。

利用样条函数进行插值，即取插值函数为样条函数，称为样条插值。例如分段线性插值是一次样条插值。下面我们介绍二次、三次样条插值。

1.5.2　二次样条函数插值

首先，我们注意到$s_2(x)\in S_P(\Delta,2)$中含有$n+2$个特定常数，故应需要$n+2$个插值条件，因此，二次样条插值问题可分为两类：

问题（1）：

已知插值节点x_i和相应的函数值$y_i(i=0,1,\cdots,n)$以及端点x_0（或x_n）处的导数值y'_0（或y'_n），求$s_2(x)\in S_P(\Delta,2)$使得

$$\begin{cases}s_2(x_i)=y_i(i=0,1,2,\cdots,n)\\ s'_2(x_0)=y_0(\text{或}s'_n(x_n)=y_n)\end{cases}\tag{7}$$

问题（2）：

已知插值节点x_i和相应的导数值$y'_i(i=0,1,2,\cdots,n)$以及端点x_0（或x_n）处的函数值y_0（或y_n），求$s_2(x)\in S_P(\Delta,2)$使得

$$\begin{cases}s'_2(x_i)=y'_i(i=0,1,2,\cdots,n)\\ s_2(x_0)=y_0(\text{或}s_2(x_n)=y_n)\end{cases}\tag{8}$$

事实上，可以证明这两类插值问题都是唯一可解的。

对于问题（1），由条件（7）

$$\begin{cases} s_2(x_0)=\alpha_0+\alpha_1x_0+\dfrac{1}{2}\alpha_2x_0^2=y_0 \\ s_2(x_1)=\alpha_0+\alpha_1x_1+\dfrac{1}{2}\alpha_2x_1^2=y_1 \\ s_2(x_j)=\alpha_0+\alpha_1x_j+\dfrac{1}{2}\alpha_2x_j^2+\dfrac{1}{2}\sum_{i=1}^{j-1}\beta_i(x_j-x_i)^2=y_j \quad (j=2,3,\cdots,n) \\ s'_2(x_0)=\alpha_1+\alpha_2x_0=y'_0 \end{cases}$$

引入记号 $X=(\alpha_0,\alpha_1,\alpha_2,\beta_1,\cdots,\beta_{n-1})^T$ 为未知向量，$C=(y_0,y_1,\cdots,y_n,y'_0)$ 为已知向量。

$$A=\begin{bmatrix} 1 & x_0 & \dfrac{1}{2}x_0^2 & 0 & \cdots & 0 \\ 1 & x_1 & \dfrac{1}{2}x_1^2 & 0 & \cdots & 0 \\ 1 & x_2 & \dfrac{1}{2}x_2^2 & \dfrac{1}{2}(x_2-x_1)^2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 1 & x_n & \dfrac{1}{2}x_n^2 & \dfrac{1}{2}(x_n-x_1)^2 & \cdots & \dfrac{1}{2}(x_n-x_{n-1})^2 \\ 0 & 1 & x_0 & 0 & \cdots & 0 \end{bmatrix}$$

于是，问题转化为求方程组 $AX=C$ 的解 $X=(\alpha_0,\alpha_1,\alpha_2,\beta_1,\cdots,\beta_{n-1})^T$ 的问题，即可得到二次样条函数 $s_2(x)$ 的表达式。

对于问题（2）的情况类似。

1.5.3　三次样条函数插值

由于 $s_3(x)\in S_P(\Delta,3)$ 中含有 $n+3$ 个待定系数，故应需要 $n+3$ 个插值条件，已知插值节点 x_i 和相应的函数值 $f(x_i)=y_i(i=0,1,2,\cdots,n)$，这里提供了 $n+1$ 个条件，还需要 2 个边界条件。

常用的三次样条函数的边界条件有 3 种类型：

（i）$s'_3(a)=y'_0, s'_3(b)=y'_n$。由这种边界条件建立的样条插值函数称为 $f(x)$ 的完备三次样条插值函数。

特别地，$y'_0=y'_n=0$ 时，样条曲线在端点处呈水平状态。

如果 $f'(x)$ 不知道，我们可以要求 $s'_3(x)$ 与 $f'(x)$ 在端点处近似相等。这时以 x_0,x_1,x_2,x_3 为节点作一个三次 Newton 插值多项式 $N_a(x)$，以 $x_n,x_{n-1},x_{n-2},x_{n-3}$ 作一个三次 Newton 插值多项式 $N_b(x)$，要求

$$s'(a)=N'_a(a), s'(b)=N'_b(b)$$

由这种边界条件建立的三次样条称为 $f(x)$ 的 Lagrange 三次样条插值函数。

（ii）$s''_3(a)=y''_0, s''_3(b)=y''_3$。特别地 $y''_n=y''_n=0$ 时，称为自然边界条件。

（iii）$s'_3(a+0)=s'_3(b-0), s''_3(a+0)=s''_3(b-0)$，(这里要求$s_3(a+0)=s_3(b-0)$)此条件称为周期条件。

1.5.4 三次样条插值在 Matlab 中的实现

在 Matlab 中数据点称之为断点。如果三次样条插值没有边界条件，最常用的方法，就是采用非扭结（not-a-knot）条件。这个条件强迫第 1 个和第 2 个三次多项式的三阶导数相等。对最后一个和倒数第 2 个三次多项式也做同样地处理。

Matlab 中三次样条插值也有现成的函数：

y=interp1(x0,y0,x,'spline')；

y=spline(x0,y0,x)；

pp=csape(x0,y0,conds)，y=ppval(pp,x)。

其中 x0,y0 是已知数据点，x 是插值点，y 是插值点的函数值。

对于三次样条插值，我们提倡使用函数 csape，csape 的返回值是 pp 形式，要求插值点的函数值，必须调用函数 ppval。

pp=csape(x0,y0)：使用默认的边界条件，即 Lagrange 边界条件。

pp=csape(x0,y0,conds)中的 conds 指定插值的边界条件，其值可为：

'complete'　　边界为一阶导数，即默认的边界条件

'not-a-knot'　非扭结条件

'periodic'　　周期条件

'second'　　边界为二阶导数，二阶导数的值[0, 0]。

'variational'　设置边界的二阶导数值为[0,0]。

对于一些特殊的边界条件，可以通过 conds 的一个1×2矩阵来表示，conds 元素的取值为 1，2。此时，使用命令

pp=csape(x0,y0_ext,conds)

其中 y0_ext=[left, y0, right]，这里 left 表示左边界的取值，right 表示右边界的取值。

conds(i)=j 的含义是给定端点 i 的 j 阶导数，即 conds 的第一个元素表示左边界的条件，第二个元素表示右边界的条件，conds=[2,1]表示左边界是二阶导数，右边界是一阶导数，对应的值由 left 和 right 给出。

详细情况请使用帮助 help csape。

例 1　机床加工

待加工零件的外形根据工艺要求由一组数据(x,y)给出（在平面情况下），用程控铣床加工时每一刀只能沿 x 方向和 y 方向走非常小的一步，这就需要从已知数据得到加工所要求的步长很小的(x,y)坐标。

表 1 中给出的 x,y 数据位于机翼断面的下轮廓线上，假设需要得到 x 坐标每改变 0.1 时的 y 坐标。试完成加工所需数据，画出曲线，并求出 $x=0$ 处的曲线斜率和 $13\le x\le 15$ 范围内 y 的最小值。

表 1

x	0	3	5	7	9	11	12	13	14	15
y	0	1.2	1.7	2.0	2.1	2.0	1.8	1.2	1.0	1.6

要求用 Lagrange、分段线性和三次样条三种插值方法计算。

解　编写以下程序：

```
clc,clear
x0=[0   3   5   7   9   11   12   13   14  15];
```

```
y0=[0  1.2  1.7  2.0  2.1  2.0  1.8  1.2   1.0  1.6];
x=0:0.1:15;
y1=lagrange(x0,y0,x);  %调用前面编写的Lagrange插值函数
y2=interp1(x0,y0,x);
y3=interp1(x0,y0,x,'spline');
pp1=csape(x0,y0); y4=ppval(pp1,x);
pp2=csape(x0,y0,'second'); y5=ppval(pp2,x);
fprintf('比较一下不同插值方法和边界条件的结果:\n')
fprintf('x    y1     y2     y3     y4    y5\n')
xianshi=[x',y1',y2',y3',y4',y5'];
fprintf('%f\t%f\t%f\t%f\t%f\t%f\n',xianshi')
subplot(2,2,1), plot(x0,y0,'+',x,y1), title('Lagrange')
subplot(2,2,2), plot(x0,y0,'+',x,y2), title('Piecewise linear')
subplot(2,2,3), plot(x0,y0,'+',x,y3), title('Spline1')
subplot(2,2,4), plot(x0,y0,'+',x,y4), title('Spline2')
dyx0=ppval(fnder(pp1),x0(1))  %求x=0处的导数
ytemp=y3(131:151);
index=find(ytemp==min(ytemp));
xymin=[x(130+index),ytemp(index)]
```

计算结果略。

可以看出，拉格朗日插值的结果根本不能应用，分段线性插值的光滑性较差（特别是在 $x=14$ 附近弯曲处），建议选用三次样条插值的结果。

1.6 B 样条函数插值方法

1.6.1 磨光函数

实际中的许多问题，往往是既要求近似函数（曲线或曲面）有足够的光滑性，又要求与实际函数有相同的凹凸性，一般插值函数和样条函数都不具有这种性质。如果对于一个特殊函数进行磨光处理生成磨光函数（多项式），则用磨光函数构造出样条函数作为插值函数，既有足够的光滑性，而且也具有较好的保凹凸性，因此磨光函数在一维插值（曲线）和二维插值（曲面）问题中有着广泛的应用。

由积分理论可知，对于可积函数通过积分会提高函数的光滑度，因此，我们可以利用积分方法对函数进行磨光处理。

定义　若 $f(x)$ 为可积函数，对于 $h>0$，则称积分

$$f_{1,h}(x)=\frac{1}{h}\int_{x-\frac{h}{2}}^{x+\frac{h}{2}} f(t)dt$$

为 $f(x)$ 的一次磨光函数，h 称为磨光宽度。

同样的，可以定义 $f(x)$ 的 k 次磨光函数为

$$f_{k,h}(x)=\frac{1}{h}\int_{x-\frac{h}{2}}^{x+\frac{h}{2}} f_{k-1,h}(t)dt \quad （k>1）$$

事实上，磨光函数 $f_{k,h}(x)$ 比 $f(x)$ 的光滑程度要高，且当磨光宽度 h 很少时 $f_{k,h}(x)$ 很接近于 $f(x)$。

1.6.2 等距 B 样条函数

对于任意的函数 $f(x)$，定义其步长为 1 的中心差分算子 δ 如下：

$$\delta f(x)=f(x+\frac{1}{2})-f(x-\frac{1}{2})$$

在此取 $f(x)=x_+^0$，则

$$\delta x_+^0=\left(x+\frac{1}{2}\right)_+^0-\left(x-\frac{1}{2}\right)_+^0$$

是一个单位方波函数（如图 1），记 $\Omega_0(x)=\delta x_+^0$。并取 $h=1$，对 $\Omega_0(x)$ 进行一次磨光得

$$\Omega_1(x)=\int_{x-\frac{1}{2}}^{x+\frac{1}{2}}\Omega_0(t)dt=\int_{x-\frac{1}{2}}^{x+\frac{1}{2}}\left[\left(t+\frac{1}{2}\right)_+^0-\left(t-\frac{1}{2}\right)_+^0\right]dt$$

$$=\int_x^{x+1}t_+^0dt-\int_{x-1}^{x}t_+^0dt=(x+1)_+-2x_++(x-1)_+$$

显然 $\Omega_1(x)$ 是连续的（如图 1）。

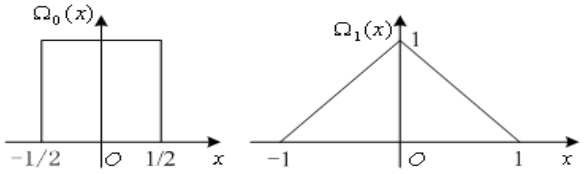

图 1　$\Omega_0(x)$ 和 $\Omega_1(x)$ 的图形

类似地可得到 k 次磨光函数为

$$\Omega_k(x)=\sum_{j=0}^{k+1}(-1)^j\frac{C_{k+1}^j}{k!}\left(x+\frac{k+1}{2}-j\right)_+^k$$

实际上，可以证明：$\Omega_k(x)$ 是分段 k 次多项式，且具有 $k-1$ 阶连续导数，其 k 阶导数有 $k+2$ 个间断点，记为 $x_j=j-\frac{k+1}{2}$（$j=0,1,2,\cdots,k+1$）。从而可知 $\Omega_k(x)$ 是对应于分划 $\Delta:-\infty<x_0<x_1<\cdots<x_{k+1}<+\infty$ 的 k 次多项式样条函数，称之为基本样条函数，简称为 k 次 B 样条。由于样条节点为 $x_j=j-\frac{k+1}{2}$（$j=0,1,2,\cdots,k+1$）是等距的，故 $\Omega_k(x)$ 又称为 k 次等距 B 样条函数。

对于任意函数 $f(x)$ 的 k 次磨光函数，由归纳法可以得到：

$$f_{k,h}(x)=\frac{1}{h}\int_{-\infty}^{+\infty}\Omega_{k-1}\left(\frac{x-t}{h}\right)f(t)dt\quad\left(x-\frac{h}{2}\le t\le x+\frac{h}{2}\right)$$

特别地，当 $f(x)=1$ 时，有 $\frac{1}{h}\int_{-\infty}^{+\infty}\Omega_{k-1}\left(\frac{x-t}{h}\right)dt=1$，从而 $\int_{-\infty}^{+\infty}\Omega_k(x)dx=1$，且当 $k\ge1$ 时有递推关系

$$\Omega_k(x)=\frac{1}{k}\left[\left(x+\frac{k+1}{2}\right)\Omega_{k-1}\left(x+\frac{1}{2}\right)-\left(\frac{k-1}{2}-x\right)\Omega_{k-1}\left(x-\frac{1}{2}\right)\right]$$

1.6.3　一维等距 B 样条函数插值

等距 B 样条函数与通常的样条有如下的关系：

定理　设有区间$[a,b]$的均匀分划$\Delta: x_j=x_0+jh$（$j=0,1,2,\cdots,n$），$h=\dfrac{b-a}{n}$，则对任意k次样条函数$s_k(x)\in S_P(\Delta,k)$都可以表示为 B 样条函数族

$$\left\{\Omega_k\left(\frac{x-x_0}{h}-j-\frac{k+1}{2}\right)\right\}_{j=-k}^{j=n-1}$$

的线性组合。

根据定理，如果已知曲线上一组点(x_j,y_j)，其中$x_j=x_0+jh$（$h>0,\ j=0,1,2,\cdots,n$），则可以构造出一条样条磨光曲线（即为 B 样条函数族的线性组合）

$$s_k(x)=\sum_{j=-k}^{n-1}c_j\Omega_k\left(\frac{x-x_0}{h}-j\right)$$

其中c_j（$j=-k,-k+1,\cdots,n-1$）为待定常数。用它来逼近曲线，既有较好的精度，又有良好的保凸性。

实际中，最常用的是$k=3$的情况，即一般形式为

$$s_3(x)=\sum_{j=-1}^{n+1}c_j\Omega_3\left(\frac{x-x_0}{h}-j\right)$$

其中$n+3$个待定系数c_j（$j=-1,0,\cdots,n+1$）可以由插值条件确定。

对于插值条件

$$\begin{cases}s_3(x_j)=y_j\ (j=0,1,2,\cdots n)\\ s'_3(x_j)=y'_j\ (j=0,n)\end{cases}$$

有

$$\begin{cases}s'_3(x_0)=\dfrac{1}{h}\displaystyle\sum_{j=-1}^{n+1}c_j\Omega'_3(-j)=y'_0\\ s_3(x_i)=\displaystyle\sum_{j=-1}^{n+1}c_j\Omega_3(i-j)=y_i,\quad i=0,1,2,\cdots,n\\ s'_3(x_n)=\dfrac{1}{h}\displaystyle\sum_{j=-1}^{n+1}c_j\Omega'_3(n-j)=y'_n\end{cases}\tag{9}$$

注意到$\Omega_3(x)$的局部非零性及其函数值：$\Omega_3(0)=\dfrac{2}{3}$，$\Omega_3(\pm1)=\dfrac{1}{6}$，当$|x|\geq2$时$\Omega_3(x)=0$；且由$\Omega'_3(x)=\Omega_2(x+\dfrac{1}{2})-\Omega_2(x-\dfrac{1}{2})$知，$\Omega'_3(0)=0$，$\Omega'_3(\pm1)=\mp\dfrac{1}{2}$，当$|x|\geq2$时$\Omega'_3(x)=0$。则（9）式的每一个方程只有三个非零系数，具体为

$$\begin{cases} -c_{-1}+c_1=2hy'_0 \\ c_{i-1}+4c_i+c_{i+1}=6y_i, \quad i=0,1,2,\cdots,n \\ -c_{n-1}+c_{n+1}=2hy'_n \end{cases} \tag{10}$$

由方程组（10）容易求解出c_j（$j=-1,0,\cdots,n+1$），即可得到三次样条函数$s_3(x)$表达式。

1.6.4 二维等距 B 样条函数插值

设有空间曲面$z=f(x,y)$（未知），如果已知二维等距节点$(x_i,y_j)=(x_0+ih,y_0+j\tau)$（$h,\tau>0$）上的值为$z_{ij}$（$i=0,1,\cdots,n;\ j=0,1,\cdots,m$），则相应的 B 样条磨光曲面的一般形式为

$$s(x,y)=\sum_{i=-k}^{n-1}\sum_{j=-l}^{m-1}c_{ij}\Omega_k\left(\frac{x-x_0}{h}-i\right)\Omega_l\left(\frac{y-y_0}{\tau}-j\right)$$

其中c_{ij}（$i=0,1,\cdots,n;\ j=0,1,\cdots,m$）为待定常数，$k,l$可以取不同值，常用的也是$k,l=2$和$3$的情形。这是一种具有良好保凸性的光滑曲面（函数），在工程设计中是常用的，但只能使用于均匀划分或近似均匀划分的情况。

1.7 二维插值

前面讲述的都是一维插值，即节点为一维变量，插值函数是一元函数（曲线）。若节点是二维的，插值函数就是二元函数，即曲面。如在某区域测量了若干点（节点）的高程（节点值），为了画出较精确的等高线图，就要先插入更多的点（插值点），计算这些点的高程（插值）。

1.7.1 插值节点为网格节点

已知$m\times n$个节点：(x_i,y_j,z_{ij})（$i=1,2,\cdots,m;\ j=1,2,\cdots,n$），且$x_1<\cdots<x_m$；$y_1<\cdots<y_n$。求点$(x,y)$处的插值$z$。

Matlab 中有一些计算二维插值的程序。如

z=interp2(x0,y0,z0,x,y,'method')

其中 x0,y0 分别为m维和n维向量，表示节点，z0 为$n\times m$维矩阵，表示节点值，x,y 为一维数组，表示插值点，x 与 y 应是方向不同的向量，即一个是行向量，另一个是列向量，z 为矩阵，它的行数为y的维数，列数为x的维数，表示得到的插值，'method' 的用法同上面的一维插值。

如果是三次样条插值，可以使用命令

pp=csape({x0,y0},z0,conds,valconds)，z=fnval(pp,{x,y})

其中 x0,y0 分别为m维和n维向量，z0 为$m\times n$维矩阵，z为矩阵，它的行数为x的维数，列数为y的维数，表示得到的插值，具体使用方法同一维插值。

例2 在一丘陵地带测量高程，x和y方向每隔100米测一个点，得高程如2表，试插值一曲面，确定合适的模型，并由此找出最高点和该点的高程。

解 编写程序如下：

```
clear,clc
x=100:100:500;
y=100:100:400;
z=[636    697    624    478   450
   698    712    630    478   420
```

```
   680    674    598    412   400
   662    626    552    334   310];
pp=csape({x,y},z')
xi=100:10:500;yi=100:10:400
cz1=fnval(pp,{xi,yi})
cz2=interp2(x,y,z,xi,yi','spline')
[i,j]=find(cz1==max(max(cz1)))
x=xi(i),y=yi(j),zmax=cz1(i,j)
```

表2

y \ x	100	200	300	400	500
100	636	697	624	478	450
200	698	712	630	478	420
300	680	674	598	412	400
400	662	626	552	334	310

1.7.2 插值节点为散乱节点

已知 n 个节点：$(x_i, y_i, z_i)(i = 1,2,\cdots,n)$，求点 (x, y) 处的插值 z 。

对上述问题，Matlab 中提供了插值函数 griddata，其格式为：

ZI = GRIDDATA(X,Y,Z,XI,YI)

其中 X、Y、Z 均为 n 维向量，指明所给数据点的横坐标、纵坐标和竖坐标。向量 XI、YI 是给定的网格点的横坐标和纵坐标，返回值 ZI 为网格（XI，YI）处的函数值。XI 与 YI 应是方向不同的向量，即一个是行向量，另一个是列向量。

例 3　在某海域测得一些点（x,y）处的水深 z 由下表给出，在矩形区域（75,200）×(−50,150) 内画出海底曲面的图形。

表 3

x	129	140	103.5	88	185.5	195	105	157.5	107.5	77	81	162	162	117.5
y	7.5	141.5	23	147	22.5	137.5	85.5	-6.5	-81	3	56.5	-66.5	84	-33.5
z	4	8	6	8	6	8	8	9	9	8	8	9	4	9

解　编写程序如下：

```
x=[129  140  103.5  88  185.5  195  105  157.5  107.5  77  81  162  162  117.5];
y=[7.5  141.5  23   147  22.5  137.5  85.5  -6.5  -81   3  56.5  -66.5  84 -33.5];
z=-[4     8     6     8     6     8     8     9     9   8     8     9     4     9];
xi=75:1:200;
yi=-50:1:150;
zi=griddata(x,y,z,xi,yi','cubic')
subplot(1,2,1), plot(x,y,'*')
subplot(1,2,2), mesh(xi,yi,zi)
```

§2 曲线拟合的线性最小二乘法

2.1 线性最小二乘法

曲线拟合问题的提法是，已知一组（二维）数据，即平面上的 n 个点 (x_i, y_i)，$i = 1,2,\cdots,n$，x_i 互不相同，寻求一个函数（曲线）$y = f(x)$，使 $f(x)$ 在某种准则下与所有数据点最为接近，即曲线拟合得最好。

线性最小二乘法是解决曲线拟合最常用的方法，基本思路是，令

$$f(x)=a_1r_1(x)+a_2r_2(x)+\cdots+a_mr_m(x) \tag{11}$$

其中 $r_k(x)$ 是事先选定的一组线性无关的函数，a_k 是待定系数 $(k=1,2,\cdots,m,m<n)$。拟合准则是使 y_i，$i=1,2,\cdots,n$，与 $f(x_i)$ 的距离 δ_i 的平方和最小，称为最小二乘准则。

2.1.1　系数 a_k 的确定

记

$$J(a_1,\cdots,a_m)=\sum_{i=1}^{n}\delta_i^2=\sum_{i=1}^{n}[f(x_i)-y_i]^2 \tag{12}$$

为求 $a_1,\cdots,a_m$ 使 J 达到最小，只需利用极值的必要条件 $\dfrac{\partial J}{\partial a_k}=0\ (k=1,\cdots,m)$，得到关于 $a_1,\cdots,a_m$ 的线性方程组

$$\sum_{i=1}^{n}r_j(x_i)[\sum_{k=1}^{m}a_kr_k(x_i)-y_i]=0,\quad (j=1,\cdots,m)$$

即

$$\sum_{k=1}^{m}a_k[\sum_{i=1}^{n}r_j(x_i)r_k(x_i)]=\sum_{i=1}^{n}r_j(x_i)y_i,\quad (j=1,\cdots,m) \tag{13}$$

记

$$R=\begin{bmatrix} r_1(x_1) & \cdots & r_m(x_1) \\ \vdots & \vdots & \vdots \\ r_1(x_n) & \cdots & r_m(x_n) \end{bmatrix}_{n\times m},$$

$A=[a_1,\cdots,a_m]^T$，$Y=(y_1,\cdots,y_n)^T$

方程组（13）可表为

$$R^TRA=R^TY \tag{14}$$

当 $\{r_1(x),\cdots,r_m(x)\}$ 线性无关时，R 列满秩，R^TR 可逆，于是方程组（14）有唯一解

$$A=(R^TR)^{-1}R^TY$$

2.1.2　函数 $r_k(x)$ 的选取

面对一组数据 $(x_i,y_i),i=1,2,\cdots,n$，用线性最小二乘法作曲线拟合时，首要的、也是关键的一步是恰当地选取 $r_1(x),\cdots,r_m(x)$。如果通过机理分析，能够知道 y 与 x 之间应该有什么样的函数关系，则 $r_1(x),\cdots,r_m(x)$ 容易确定。若无法知道 y 与 x 之间的关系，通常可以将数据 $(x_i,y_i),i=1,2,\cdots,n$ 作图，直观地判断应该用什么样的曲线去作拟合。人们常用的曲线有

（i）直线　　$y=a_1x+a_2$

（ii）多项式　$y=a_1x^m+\cdots+a_mx+a_{m+1}$（一般 $m=2,3$，不宜太高）

（iii）双曲线（一支）　$y=\dfrac{a_1}{x}+a_2$

（iv）指数曲线　$y = a_1 e^{a_2 x}$

对于指数曲线，拟合前需作变量代换，化为对 a_1, a_2 的线性函数。

已知一组数据，用什么样的曲线拟合最好，可以在直观判断的基础上，选几种曲线分别拟合，然后比较，看哪条曲线的最小二乘指标 J 最小。

2.2 最小二乘法的 Matlab 实现

2.2.1　解方程组方法

在上面的记号下，

$$J(a_1,\cdots,a_m) = \| RA - Y \|^2$$

Matlab 中的线性最小二乘的标准型为

$$\min_A \| RA - Y \|_2^2,$$

命令为　$A = R \backslash Y$ 。

例 4　用最小二乘法求一个形如 $y = a + bx^2$ 的经验公式，使它与表 4 所示的数据拟合。

表 4

x	19	25	31	38	44
y	19.0	32.3	49.0	73.3	97.8

解　编写程序如下

```
x=[19    25    31    38    44]';
y=[19.0   32.3   49.0   73.3   97.8]';
r=[ones(5,1),x.^2];
ab=r\y
x0=19:0.1:44;
y0=ab(1)+ab(2)*x0.^2;
plot(x,y,'o',x0,y0,'r')
```

2.2.2　多项式拟合方法

如果取 $\{r_1(x),\cdots,r_{m+1}(x)\} = \{1, x, \cdots, x^m\}$，即用 m 次多项式拟合给定数据，Matlab 中有现成的函数

a=polyfit(x0,y0,m)

其中输入参数 x0,y0 为要拟合的数据，m 为拟合多项式的次数，输出参数 a 为拟合多项式 $y=a_m x^m+\cdots+a_1 x+a_0$ 系数 a=[a_m, …, a_1, a_0]。

多项式在 x 处的值 y 可用下面的函数计算

y=polyval(a, x)。

例 5 某乡镇企业 1990-1996 年的生产利润如表 5。

表 5

年份	1990	1991	1992	1993	1994	1995	1996
利润（万元）	70	122	144	152	174	196	202

试预测 1997 年和 1998 年的利润。

解　作已知数据的的散点图，

```
x0=[1990  1991  1992  1993  1994  1995  1996];
y0=[70   122   144   152   174   196   202];
plot(x0,y0,'*')
```

发现该乡镇企业的年生产利润几乎直线上升。因此，我们可以用 $y=a_1x+a_0$ 作为拟合函数来预测该乡镇企业未来的年利润。编写程序如下：

```
x0=[1990  1991  1992  1993  1994  1995  1996];
y0=[70   122   144   152   174   196   202];
a=polyfit(x0,y0,1)
y97=polyval(a,1997)
y98=polyval(a,1998)
```

求得 $a_1=20$，$a_0=-4.0705\times10^4$，1997 年的生产利润 y97=233.4286，1998 年的生产利润 y98=253.9286。

§3 最小二乘优化

在无约束最优化问题中，有些重要的特殊情形，比如目标函数由若干个函数的平方和构成。这类函数一般可以写成：

$$F(x)=\sum_{i=1}^{m}f_i^2(x),x\in R^n$$

其中 $x=(x_1,\cdots,x_n)^T$，一般假设 $m\geq n$。我们把极小化这类函数的问题：

$$\min\quad F(x)=\sum_{i=1}^{m}f_i^2(x)$$

称为最小二乘优化问题。

最小二乘优化是一类比较特殊的优化问题，在处理这类问题时，Matlab 也提供了一些强大的函数。在 Matlab 优化工具箱中，用于求解最小二乘优化问题的函数有：lsqlin、lsqcurvefit、lsqnonlin、lsqnonneg，用法介绍如下。

3.1 lsqlin 函数

求解 $\min\limits_x \frac{1}{2}\|Cx-d\|_2^2$

$$\text{s.t.}\quad\begin{cases}A*x\leq b\\ Aeq*x=beq\\ lb\leq x\leq ub\end{cases}$$

其中 C,A,Aeq 为矩阵，d,b,beq,lb,ub,x 为向量。

Matlab 中的函数为：

```
x=lsqlin(C,d,A,b,Aeq,beq,lb,ub,x0)
```

例 6　用 lsqlin 命令求解例 4。

解　编写程序如下：

```
x=[19      25      31      38      44]';
y=[19.0    32.3    49.0    73.3    97.8]';
r=[ones(5,1),x.^2];
ab=lsqlin(r,y)
x0=19:0.1:44;
y0=ab(1)+ab(2)*x0.^2;
plot(x,y,'o',x0,y0,'r')
```

3.2 lsqcurvefit 函数

给定输入输出数列 $xdata, ydata$，求参量 x，使得

$$\min_x \frac{1}{2}\left\|F(x,xdata)-ydata\right\|_2^2=\frac{1}{2}\sum_i\left(F(x,xdata_i)-ydata_i\right)^2$$

Matlab 中的函数为

X=LSQCURVEFIT(FUN,X0,XDATA,YDATA,LB,UB,OPTIONS)

其中 FUN 是定义函数 $F(x,xdata)$ 的 M 文件。

例 7　用下面表 6 中的数据拟合函数 $c(t)=a+be^{-0.02kt}$ 中的参数 a,b,k。

表 6

t_j	100	200	300	400	500	600	700	800	900	1000
c_j	4.54	4.99	5.35	5.65	5.90	6.10	6.26	6.39	6.50	6.59

解　该问题即解最优化问题：

$$\min F(a,b,k)=\sum_{i=1}^{10}(a+be^{-0.02kt_j}-c_j)^2$$

（1）编写 M 文件 fun1.m 定义函数 $F(x,tdata)$：

```
function f=fun1(x,tdata);
f=x(1)+x(2)*exp(-0.02*x(3)*tdata);  %其中 x(1)=a，x(2)=b，x(3)=k
```

（2）调用函数 lsqcurvefit，编写程序如下：

```
td=100:100:1000;
cd=[4.54  4.99  5.35  5.65  5.90  6.10  6.26  6.39  6.50  6.59];
x0=[0.2  0.05  0.05];
x=lsqcurvefit(@fun1,x0,td,cd)
```

3.3 lsqnonlin 函数

已知函数向量 $F(x)=[f_1(x),\ \cdots,\ f_k(x)]^T$，求 x 使得

$$\min_x \frac{1}{2}\left\|F(x)\right\|_2^2$$

Matlab 中的函数为

X=LSQNONLIN(FUN,X0,LB,UB,OPTIONS)

其中 FUN 是定义向量函数 $F(x)$ 的 M 文件。

例 8　用 lsqnonlin 函数求解例 7。

解　这里

$$F(x)=F(x,t)=\left[a+be^{-0.02kt_1}-c_1,\ \cdots,\ a+be^{-0.02kt_{10}}-c_{10}\right]^T$$

$$x=[a,\ \ b,\ \ k]$$

（1）编写 M 文件 fun2.m 如下：

```
function f=fun2(x);
td=100:100:1000;
cd=[4.54  4.99  5.35  5.65  5.90  6.10  6.26  6.39  6.50  6.59];
f=x(1)+x(2)*exp(-0.02*x(3)*td)-cd;
```

（2）调用函数 lsqnonlin，编写程序如下：

```
x0=[0.2 0.05 0.05];  %初始值是任意取的
x=lsqnonlin(@fun2,x0)
```

3.4 lsqnonneg 函数

求解非负的 x，使得满足 $\min\limits_x \frac{1}{2}\|Cx-d\|_2^2$，

Matlab 中的函数为

```
X = LSQNONNEG(C,d,X0,OPTIONS)
```

例 9　已知 $C=\begin{bmatrix}0.0372 & 0.2869\\ 0.6861 & 0.7071\\ 0.6233 & 0.6245\\ 0.6344 & 0.6170\end{bmatrix}$，$d=\begin{bmatrix}0.8587\\ 0.1781\\ 0.0747\\ 0.8405\end{bmatrix}$，求 $x(x\ge 0)$ 满足

$$\min_x \frac{1}{2}\|Cx-d\|_2^2$$

最小。

解　编写程序如下：

```
c=[0.0372 0.2869;0.6861 0.7071;0.6233 0.6245;0.6344 0.6170];
d=[0.8587;0.1781;0.0747;0.8405];
x=lsqnonneg(c,d)
```

3.5　曲线拟合的用户图形界面求法

Matlab 工具箱提供了命令 cftool，该命令给出了一维数据拟合的交互式环境。具体执行步骤如下：

（1）把数据导入到工作空间；

（2）运行 cftool，打开用户图形界面窗口；

（3）对数据进行预处理；

（4）选择适当的模型进行拟合；

（5）生成一些相关的统计量，并进行预测。

可以通过帮助（运行 doc cftool）熟悉该命令的使用细节。

§4 曲线拟合与函数逼近

前面讲的曲线拟合是已知一组离散数据 $\{(x_i,y_i),i=1,\cdots,n\}$，选择一个较简单的函数 $f(x)$，如多项式，在一定准则如最小二乘准则下，最接近这些数据。

如果已知一个较为复杂的连续函数 $y(x),x\in[a,b]$，要求选择一个较简单的函数 $f(x)$，在一定准则下最接近 $f(x)$，就是所谓函数逼近。

与曲线拟合的最小二乘准则相对应，函数逼近常用的一种准则是最小平方逼近，即

$$J=\int_a^b[f(x)-y(x)]^2dx \tag{15}$$

达到最小。与曲线拟合一样，选一组函数 $\{r_k(x),k=1,\cdots,m\}$ 构造 $f(x)$，即令

$$f(x)=a_1r_1(x)+\cdots+a_mr_m(x)$$

代入（15）式，求$a_1,\cdots,a_m$使J达到极小。利用极值必要条件可得

$$\begin{bmatrix}(r_1,r_1) & \cdots & (r_1,r_m)\\ \vdots & \vdots & \vdots\\ (r_m,r_1) & \cdots & (r_m,r_m)\end{bmatrix}\begin{bmatrix}a_1\\ \vdots\\ a_m\end{bmatrix}=\begin{bmatrix}(y,r_1)\\ \vdots\\ (y,r_m)\end{bmatrix} \tag{16}$$

这里$(g,h)=\int_a^b g(x)h(x)dx$。当方程组（16）的系数矩阵非奇异时，有唯一解。

最简单的当然是用多项式逼近函数，即选$r_1(x)=1$，$r_2(x)=x$，$r_3(x)=x^2$，…。并且如果能使$\int_a^b r_i(x)r_j(x)dx=0$，$(i\neq j)$，方程组（16）的系数矩阵将是对角阵，计算大大简化。满足这种性质的多项式称正交多项式。

勒让得（Legendre）多项式是在$[-1,1]$区间上的正交多项式，它的表达式为

$$P_0(x)=1,\quad P_k(x)=\frac{1}{2^k k!}\frac{d^k}{dx^k}(x^2-1)^k,k=1,2,\cdots$$

可以证明

$$\int_{-1}^{1}P_i(x)P_j(x)dx=\begin{cases}0, & i\neq j\\ \dfrac{2}{2i+1}, & i=j\end{cases}$$

$$P_{k+1}(x)=\frac{2k+1}{k+1}xP_k(x)-\frac{k}{k+1}P_{k-1}(x),\quad k=1,2,\cdots$$

常用的正交多项式还有第一类切比雪夫（Chebyshev）多项式

$$T_n(x)=\cos(n\arccos x),\quad (x\in[-1,1],n=0,1,2,\cdots)$$

和拉盖尔（Laguerre）多项式

$$L_n(x)=e^x\frac{d^n}{dx^n}(x^ne^{-x}),\quad (x\in[0,+\infty),n=0,1,2,\cdots)。$$

例 10 求$f(x)=\cos x,x\in\left[-\frac{\pi}{2},\frac{\pi}{2}\right]$在$H=\mathrm{Span}\{1,x^2,x^4\}$中的最佳平方逼近多项式。

解　编写程序如下：

```
syms x
base=[1,x^2,x^4];
y1=base.'*base
y2=cos(x)*base.'
r1=int(y1,-pi/2,pi/2)
r2=int(y2,-pi/2,pi/2)
a=r1\r2
xishu1=double(a)
digits(8),xishu2=vpa(a)
```

求得 xishu1=0.9996　　-0.4964　　0.0372，即所求的最佳平方逼近多项式为

$$y=0.9996-0.4964x^2+0.0372x^4$$

§5 黄河小浪底调水调沙问题

5.1 问题的提出

2004 年 6 月至 7 月黄河进行了第三次调水调沙试验，特别是首次由小浪底、三门峡和万家寨三大水库联合调度，采用接力式防洪预泄放水，形成人造洪峰进行调沙试验获得成功。整个试验期为 20 多天，小浪底从 6 月 19 日开始预泄放水，直到 7 月 13 日恢复正常供水结束。小浪底水利工程按设计拦沙量为 75.5 亿 m^3，在这之前，小浪底共积泥沙达 14.15 亿 t。这次调水调沙试验一个重要目的就是由小浪底上游的三门峡和万家寨水库泄洪，在小浪底形成人造洪峰，冲刷小浪底库区沉积的泥沙，在小浪底水库开闸泄洪以后，从 6 月 27 日开始三门峡水库和万家寨水库陆续开闸放水，人造洪峰于 29 日先后到达小浪底，7 月 3 日达到最大流量 2700m3/s，使小浪底水库的排沙量也不断地增加。表 7 是由小浪底观测站从 6 月 29 日到 7 月 10 检测到的试验数据。

表 7 观测数据

日期	6.29		6.30		7.1		7.2		7.3		7.4	
时间	8:00	20:00	8:00	20:00	8:00	20:00	8:00	20:00	8:00	20:00	8:00	20:00
水流量	1800	1900	2100	2200	2300	2400	2500	2600	2650	2700	2720	2650
含沙量	32	60	75	85	90	98	100	102	108	112	115	116
日期	7.5		7.6		7.7		7.8		7.9		7.10	
时间	8:00	20:00	8:00	20:00	8:00	20:00	8:00	20:00	8:00	20:00	8:00	20:00
水流量	2600	2500	2300	2200	2000	1850	1820	1800	1750	1500	1000	900
含沙量	118	120	118	105	80	60	50	30	26	20	8	5

现在，根据试验数据建立数学模型研究下面的问题：

（1）给出估计任意时刻的排沙量及总排沙量的方法；

（2）确定排沙量与水流量的关系。

5.2 模型的建立与求解

已知给定的观测时刻是等间距的，以 6 月 29 日零时刻开始计时，则各次观测时刻（离开始时刻 6 月 29 日零时刻的时间）分别为

$$t_i = 3600(12i-4),\ i = 1,2,\cdots,24,$$

其中计时单位为秒。第 1 次观测的时刻 $t_1 = 28800$，最后一次观测的时刻 $t_{24} = 1022400$。

记第 i（$i = 1,2,\cdots,24$）次观测时水流量为 v_i，含沙量为 c_i，则第 i 次观测时的排沙量为 $y_i = c_i v_i$。有关的数据见表 8。

表 8 插值数据对应关系 单位：排沙量为 kg

节点	1	2	3	4	5	6	7	8
时刻	28800	72000	115200	158400	201600	244800	288000	331200
排沙量	57600	114000	157500	187000	207000	235200	250000	265200

节点	9	10	11	12	13	14	15	16
时刻	374400	417600	460800	504000	547200	590400	633600	676800
排沙量	286200	302400	312800	307400	306800	300000	271400	231000
节点	17	18	19	20	21	22	23	24
时刻	720000	763200	806400	849600	892800	936000	979200	1022400
排沙量	160000	111000	91000	54000	45500	30000	8000	4500

对于问题（1），根据所给问题的试验数据，要计算任意时刻的排沙量，就要确定出排沙量随时间变化的规律，可以通过插值来实现。考虑到实际中的排沙量应该是时间的连续函数，为了提高模型的精度，我们采用三次样条函数进行插值。

利用 MATLAB 函数，求出三次样条函数，得到排沙量 $y = y(t)$ 与时间的关系，然后进行积分，就可以得到总的排沙量

$$z = \int_{t_1}^{t_{24}} y(t)dt$$

最后求得总的排沙量为1.844×10^9 t，计算的 Matlab 程序如下：

```
clc,clear
load data.txt       %data.txt 按照原始数据格式把水流量和排沙量排成 4 行，12 列
liu=data([1,3],:);
liu=liu';liu=liu(:);
sha=data([2,4],:);
sha=sha';sha=sha(:);
y=sha.*liu;y=y';
i=1:24;
t=(12*i-4)*3600;
t1=t(1);t2=t(end);
pp=csape(t,y);
xsh=pp.coefs   %求得插值多项式的系数矩阵，每一行是一个区间上多项式的系数。
TL=quadl(@(tt)ppval(pp,tt),t1,t2)
```

也可以利用 3 次 B 样条函数进行插值，求得总的排沙量也为1.844×10^9 t,，计算的 Matlab 程序如下：

```
clc,clear
load data.txt       %data.txt 按照原始数据格式把水流量和排沙量排成 4 行，12 列
liu=data([1,3],:);
liu=liu';liu=liu(:);
sha=data([2,4],:);
sha=sha';sha=sha(:);
y=sha.*liu;y=y';
i=1:24;
t=(12*i-4)*3600;
t1=t(1);t2=t(end);
pp=spapi(4,t,y)          %三次 B 样条
pp2=fn2fm(pp,'pp')    %把 B 样条函数转化为 pp 格式
TL=quadl(@(tt)fnval(pp,tt),t1,t2)
```

对于问题（2），研究排沙量与水量的关系，从试验数据可以看出，开始排沙量是随

着水流量的增加而增长，而后是随着水流量的减少而减少。显然，变化规律并非是线性的关系，为此，把问题分为两部分，从开始水流量增加到最大值 2720m^3/s（即增长的过程）为第一阶段，从水流量的最大值到结束为第二阶段，分别来研究水流量与排沙量的关系。

画出排沙量与水流量的散点图（见图 2）。

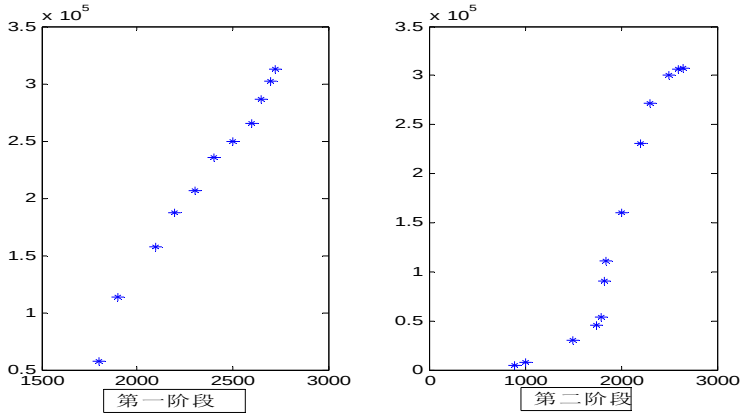

图 2　散点图

画散点图的程序如下：

```
load data.txt
liu=data([1,3],:);
liu=liu';liu=liu(:);
sha=data([2,4],:);
sha=sha';sha=sha(:);
y=sha.*liu;
subplot(1,2,1), plot(liu(1:11),y(1:11),'*')
subplot(1,2,2), plot(liu(12:24),y(12:24),'*')
```

从散点图可以看出，第一阶段基本上是线性关系，第二阶段准备依次用二次、三次、四次曲线来拟合，看哪一个模型的剩余标准差小就选取哪一个模型。最后求得第一阶段排沙量 y 与水流量 v 之间的预测模型为

$$y = 250.5655v - 373384.4661$$

第二阶段的预测模型为一个四次多项式。

$$y = -2.7693\times10^{-7}v^4 + 0.0018v^3 - 4.092v^2 + 3891.0441v - 1.32262749668\times10^6$$

计算的 Matlab 程序如下：

```
clc, clear
load data.txt       %data.txt 按照原始数据格式把水流量和排沙量排成 4 行，12 列
liu=data([1,3],:); liu=liu'; liu=liu(:);
sha=data([2,4],:); sha=sha'; sha=sha(:);
y=sha.*liu;
%以下是第一阶段的拟合
format long e
nihe1_1=polyfit(liu(1:11),y(1:11),1)     %拟合一次多项式，系数排列从高次幂到低次幂
nihe1_2=polyfit(liu(1:11),y(1:11),2)
```

```
yhat1_1=polyval(nihe1_1,liu(1:11));   %求预测值
yhat1_2=polyval(nihe1_2,liu(1:11));
%以下求误差平凡和与剩余标准差
cha1_1=sum((y(1:11)-yhat1_1).^2); rmse1_1=sqrt(cha1_1/9)
cha1_2=sum((y(1:11)-yhat1_2).^2); rmse1_2=sqrt(cha1_2/8)
%以下是第二阶段的拟合
for j=1:3
    str1=char(['nihe2_' int2str(j) '=polyfit(liu(12:24),y(12:24),' int2str(j+1) ')']);
    eval(str1)
    str2=char(['yhat2_' int2str(j) '=polyval(nihe2_' int2str(j) ',liu(12:24));']);
    eval(str2)
    str3=char(['cha2_' int2str(j) '=sum((y(12:24)-yhat2_' int2str(j) ').^2);'...
        'rmse2_' int2str(j) '=sqrt(cha2_' int2str(j) '/(11-j))']);
    eval(str3)
end
format
```

习 题 九

1. 用给定的多项式，如 $y=x^3-6x^2+5x-3$，产生一组数据 (x_i,y_i)，$i=1,2,\cdots,m$，再在 y_i 上添加随机干扰（可用 rand 产生（0, 1）均匀分布随机数，或用 randn 产生 $N(0,1)$ 分布随机数），然后用 x_i 和添加了随机干扰的 y_i 作 3 次多项式拟合，与原系数比较。如果作 2 或 4 次多项式拟合，结果如何？

2. 已知平面区域 $0\le x\le 5600$，$0\le y\le 4800$ 的高程数据见 9 表（单位：m）。

表 9

4800	1350	1370	1390	1400	1410	960	940	880	800	690	570	430	290	210	150
4400	1370	1390	1410	1430	1440	1140	1110	1050	950	820	690	540	380	300	210
4000	1380	1410	1430	1450	1470	1320	1280	1200	1080	940	780	620	460	370	350
3600	1420	1430	1450	1480	1500	1550	1510	1430	1300	1200	980	850	750	550	500
3200	1430	1450	1460	1500	1550	1600	1550	1600	1600	1600	1550	1500	1500	1550	1550
2800	950	1190	1370	1500	1200	1100	1550	1600	1550	1380	1070	900	1050	1150	1200
2400	910	1090	1270	1500	1200	1100	1350	1450	1200	1150	1010	880	1000	1050	1100
2000	880	1060	1230	1390	1500	1500	1400	900	1100	1060	950	870	900	936	950
1600	830	980	1180	1320	1450	1420	400	1300	700	900	850	810	380	780	750
1200	740	880	1080	1130	1250	1280	1230	1040	900	500	700	780	750	650	550
800	650	760	880	970	1020	1050	1020	830	800	700	300	500	550	480	350
400	510	620	730	800	850	870	850	780	720	650	500	200	300	350	320
0	370	470	550	600	670	690	670	620	580	450	400	300	100	150	250
Y/X	0	400	800	1200	1600	2000	2400	2800	3200	3600	4000	4400	4800	5200	5600

试用二维插值求 x, y 方向间隔都为 50 的高程，并画出该区域的等高线。

3. 用最小二乘法求一形如 $y=ae^{bx}$ 的经验公式拟合表 10 中的数据。

表 10

x_i	1	2	3	4	5	6	7	8
y_i	15.3	20.5	27.4	36.6	49.1	65.6	87.87	117.6

4.（水箱水流量问题）许多供水单位由于没有测量流入或流出水箱流量的设备，而只能测量水箱中的水位。试通过测得的某时刻水箱中水位的数据，估计在任意时刻（包

括水泵灌水期间）t流出水箱的流量 $f(t)$。

给出原始数据表 11，其中长度单位为 E（1E＝30.24cm）。水箱为圆柱体，其直径为 57E。

假设：

（1）影响水箱流量的唯一因素是该区公众对水的普通需要；

（2）水泵的灌水速度为常数；

（3）从水箱中流出水的最大流速小于水泵的灌水速度；

（4）每天的用水量分布都是相似的；

（5）水箱的流水速度可用光滑曲线来近似；

（6）当水箱的水容量达到 514×10^3g 时，开始泵水；达到 677.6×10^3g 时，便停止泵水。

表 11　水位数据表

时间（s）	水位（10^{-2}E）	时间（s）	水位（10^{-2}E）
0	3175	44636	3350
3316	3110	49953	3260
6635	3054	53936	3167
10619	2994	57254	3087
13937	2947	60574	3012
17921	2892	64554	2927
21240	2850	68535	2842
25223	2795	71854	2767
28543	2752	75021	2697
32284	2697	79254	泵水
35932	泵水	82649	泵水
39332	泵水	85968	3475
39435	3550	89953	3397
43318	3445	93270	3340