第三十章　偏最小二乘回归

在实际问题中，经常遇到需要研究两组多重相关变量间的相互依赖关系，并研究用一组变量（常称为自变量或预测变量）去预测另一组变量（常称为因变量或响应变量），除了最小二乘准则下的经典多元线性回归分析（MLR），提取自变量组主成分的主成分回归分析（PCR）等方法外，还有近年发展起来的偏最小二乘（PLS）回归方法。

偏最小二乘回归提供一种多对多线性回归建模的方法，特别当两组变量的个数很多，且都存在多重相关性，而观测数据的数量（样本量）又较少时，用偏最小二乘回归建立的模型具有传统的经典回归分析等方法所没有的优点。

偏最小二乘回归分析在建模过程中集中了主成分分析，典型相关分析和线性回归分析方法的特点，因此在分析结果中，除了可以提供一个更为合理的回归模型外，还可以同时完成一些类似于主成分分析和典型相关分析的研究内容，提供更丰富、深入的一些信息。

本章介绍偏最小二乘回归分析的建模方法;通过例子从预测角度对所建立的回归模型进行比较。

§1　偏最小二乘回归

考虑 p 个变量 $y_1,y_2,\cdots,y_p$ 与 m 个自变量 $x_1,x_2,\cdots,x_m$ 的建模问题。偏最小二乘回归的基本作法是首先在自变量集中提出第一成分 t_1（t_1 是 $x_1,\cdots,x_m$ 的线性组合，且尽可能多地提取原自变量集中的变异信息）；同时在因变量集中也提取第一成分 u_1，并要求 t_1 与 u_1 相关程度达到最大。然后建立因变量 $y_1,\cdots,y_p$ 与 t_1 的回归，如果回归方程已达到满意的精度，则算法中止。否则继续第二对成分的提取，直到能达到满意的精度为止。若最终对自变量集提取 r 个成分 $t_1,t_2,\cdots,t_r$，偏最小二乘回归将通过建立 $y_1,\cdots,y_p$ 与 $t_1,t_2,\cdots,t_r$ 的回归式，然后再表示为 $y_1,\cdots,y_p$ 与原自变量的回归方程式，即偏最小二乘回归方程式。

为了方便起见，不妨假定 p 个因变量 $y_1,\cdots,y_p$ 与 m 个自变量 $x_1,\cdots,x_m$ 均为标准化变量。因变量组和自变量组的 n 次标准化观测数据阵分别记为

$$F_0=\begin{bmatrix} y_{11} & \cdots & y_{1p} \\ \vdots & & \vdots \\ y_{n1} & \cdots & y_{np} \end{bmatrix},\quad E_0=\begin{bmatrix} x_{11} & \cdots & x_{1m} \\ \vdots & & \vdots \\ x_{n1} & \cdots & x_{nm} \end{bmatrix}$$

偏最小二乘回归分析建模的具体步骤如下：

（1）分别提取两变量组的第一对成分，并使之相关性达最大。

假设从两组变量分别提出第一对成分为t_1和u_1，t_1是自变量集$X=(x_1,\cdots,x_m)^T$的线性组合：$t_1=w_{11}x_1+\cdots+w_{1m}x_m=w_1^TX$，$u_1$是因变量集$Y=(y_1,\cdots,y_p)^T$的线性组合：$u_1=v_{11}y_1+\cdots+v_{1p}y_p=v_1^TY$。为了回归分析的需要，要求：

① t_1和u_1各自尽可能多地提取所在变量组的变异信息；

② t_1和u_1的相关程度达到最大。

由两组变量集的标准化观测数据阵E_0和F_0，可以计算第一对成分的得分向量，记为$\hat{t}_1$和$\hat{u}_1$：

$$\hat{t}_1=E_0w_1=\begin{bmatrix}x_{11}&\cdots&x_{1m}\\ \vdots& &\vdots\\ x_{n1}&\cdots&x_{nm}\end{bmatrix}\begin{bmatrix}w_{11}\\ \vdots\\ w_{1m}\end{bmatrix}=\begin{bmatrix}t_{11}\\ \vdots\\ t_{n1}\end{bmatrix}$$

$$\hat{u}_1=F_0v_1=\begin{bmatrix}y_{11}&\cdots&y_{1p}\\ \vdots& &\vdots\\ y_{n1}&\cdots&y_{np}\end{bmatrix}\begin{bmatrix}v_{11}\\ \vdots\\ v_{1p}\end{bmatrix}=\begin{bmatrix}u_{11}\\ \vdots\\ u_{n1}\end{bmatrix}$$

第一对成分t_1和u_1的协方差$\mathrm{Cov}(t_1,u_1)$可用第一对成分的得分向量$\hat{t}_1$和$\hat{u}_1$的内积来计算。故而以上两个要求可化为数学上的条件极值问题：

$$\begin{cases}<\hat{t}_1,\hat{u}_1>=<E_0w_1,Y_0v_1>=w_1^TE_0^TF_0x_1\Rightarrow\max\\ w_1^Tw=\|w_1\|^2=1,\quad v_1^Tv_1=\|v_1\|^2=1\end{cases}$$

利用Lagrange乘数法，问题化为求单位向量w_1和v_1，使$\theta_1=w_1^TE_0^TF_0v_1\Rightarrow$最大。问题的求解只须通过计算$m\times m$矩阵$M=E_0^TF_0F_0^TE_0$的特征值和特征向量，且$M$的最大特征值为$\theta_1^2$，相应的单位特征向量就是所求的解$w_1$，而$v_1$可由$w_1$计算得到$v_1=\dfrac{1}{\theta_1}F_0^TE_0w_1$。

（2）建立$y_1,\cdots,y_p$对t_1的回归及$x_1,\cdots,x_m$对t_1的回归。

假定回归模型为

$$\begin{cases} E_0 = \hat{t}_1 \alpha_1^T + E_1 \\ F_0 = \hat{u}_1 \beta_1^T + F_1 \end{cases}$$

其中 $\alpha_1 = (\alpha_{11}, \cdots, \alpha_{1m})^T$， $\beta_1 = (\beta_{11}, \cdots, \beta_{1p})^T$ 分别是多对一的回归模型中的参数向量，E_1 和 F_1 是残差阵。回归系数向量 α_1, β_1 的最小二乘估计为

$$\begin{cases} \alpha_1 = E_0^T \hat{t}_1 \Big/ \left\| \hat{t}_1 \right\|^2 \\ \beta_1 = F_0^T \hat{t}_1 \Big/ \left\| \hat{t}_1 \right\|^2 \end{cases},$$

称 α_1, β_1 为模型效应负荷量。

（3）用残差阵 E_1 和 F_1 代替 E_0 和 F_0 重复以上步骤。

记 $\hat{E}_0 = \hat{t}_1 \alpha_1^T$， $\hat{F}_0 = \hat{t}_1 \beta_1^T$，则残差阵 $E_1 = E_0 - \hat{E}_0$， $F_1 = F_0 - \hat{F}_0$。如果残差阵 F_1 中元素的绝对值近似为 0，则认为用第一个成分建立的回归式精度已满足需要了，可以停止抽取成分。否则用残差阵 E_1 和 F_1 代替 E_0 和 F_0 重复以上步骤即得：

$w_2 = (w_{21}, \cdots, w_{2m})^T$； $v_2 = (v_{21}, \cdots, v_{2p})^T$ 分别为第二对成分的权数。而 $\hat{t}_2 = E_1 w_2$， $\hat{u}_2 = F_1 v_2$ 为第二对成分的得分向量。

$\alpha_2 = \dfrac{E_1^T \hat{t}_2}{\left\| \hat{t}_2 \right\|^2}$， $\beta_2 = \dfrac{F_1^T \hat{t}_2}{\left\| \hat{t}_2 \right\|^2}$ 分别为 X, Y 的第二对成分的负荷量。这时有

$$\begin{cases} E_0 = \hat{t}_1 \alpha_1^T + \hat{t}_2 \alpha_2^T + E_2 \\ F_0 = \hat{t}_1 \beta_1^T + \hat{t}_2 \beta_2^T + F_2 \end{cases}$$

（4）设 $n \times m$ 数据阵 E_0 的秩为 $r \le \min(n-1, m)$，则存在 r 个成分 $t_1, t_2, \cdots, t_r$，使得

$$\begin{cases} E_0 = \hat{t}_1 \alpha_1^T + \cdots + \hat{t}_r \alpha_r^T + E_r \\ F_0 = \hat{t}_1 \beta_1^T + \cdots + \hat{t}_r \beta_r^T + F_r \end{cases}$$

把 $t_k = w_{k1} x_1 + \cdots + w_{km} x_m$（$k = 1, 2, \cdots, r$），代入 $Y = t_1 \beta_1 + \cdots + t_r \beta_r$，即得 p 个因变量的偏最小二乘回归方程式

$y_j = a_{j1}x_1 + \cdots + a_{jm}x_m$，（$j = 1,2,\cdots,m$）

（5）交叉有效性检验。

一般情况下，偏最小二乘法并不需要选用存在的r个成分$t_1, t_2, \cdots, t_r$来建立回归式，而像主成分分析一样，只选用前l个成分（$l \le r$），即可得到预测能力较好的回归模型。对于建模所需提取的主成分个数l，可以通过交叉有效性检验来确定。

每次舍去第i个观测（$i = 1,2,\cdots,n$），用余下的$n-1$个观测值按偏最小二乘回归方法建模，并考虑抽取h个成分后拟合的回归式，然后把舍去的第i个观测点代入所拟合的回归方程式，得到$y_j(j = 1,2,\cdots,p)$在第i个观测点上的预测值$\hat{y}_{(i)j}(h)$。对$i = 1,2,\cdots,n$重复以上的验证，即得抽取h个成分时第j个因变量$y_j(j = 1,2,\cdots,p)$的预测误差平方和为

$$\mathrm{PRESS}_j(h) = \sum_{i=1}^{n}(y_{ij} - \hat{y}_{(i)j}(h))^2 \quad （j = 1,2,\cdots,p）$$

$Y = (y_1,\cdots,y_p)^T$的预测误差平方和为

$$\mathrm{PRESS}(h) = \sum_{i=1}^{p}\mathrm{PRESS}_j(h)。$$

另外，再采用所有的样本点，拟合含h个成分的回归方程。这时，记第i个样本点的预测值为$\hat{y}_{ij}(h)$，则可以定义y_j的误差平方和为

$$\mathrm{SS}_j(h) = \sum_{i=1}^{n}(y_{ij} - \hat{y}_{ij}(h))^2$$

定义Y的误差平方和为

$$\mathrm{SS}(h) = \sum_{j=1}^{p}\mathrm{SS}_j(h)$$

当$\mathrm{PRESS}(h)$达到最小值时，对应的h即为所求的成分个数。通常，总有$\mathrm{PRESS}(h)$大于$\mathrm{SS}(h)$，而$\mathrm{SS}(h)$则小于$\mathrm{SS}(h-1)$。因此，在提取成分时，总希望比值$\mathrm{PRESS}(h)/\mathrm{SS}(h-1)$越小越好；一般可设定限制值为0.05，即当

$$\mathrm{PRESS}(h)/\mathrm{SS}(h-1) \le (1-0.05)^2 = 0.95^2$$

时，增加成分t_h有利于模型精度的提高。或者反过来说，当

$$\mathrm{PRESS}(h)/\mathrm{SS}(h-1) > 0.95^2$$

时，就认为增加新的成分t_h，对减少方程的预测误差无明显的改善作用。

为此，定义交叉有效性为$Q_h^2 = 1-\mathrm{PRESS}(h)/\mathrm{SS}(h-1)$，这样，在建模的每一步计算结束前，均进行交叉有效性检验，如果在第h步有$Q_h^2 < 1-0.95^2 = 0.0985$，则模型达到精度要求，可停止提取成分；若$Q_h^2 \ge 0.0975$，表示第$h$步提取的$t_h$成分的边际贡献显著，应继续第$h+1$步计算。

§2　一种更简洁的计算方法

上节介绍的算法原则和推导过程的思路在目前的文献中是最为常见的。然而，还有一种更为简洁的计算方法，即直接在$E_0,\cdots,E_{r-1}$矩阵中提取成分$t_1,\cdots,t_r\,(r \le m)$。要求$t_h$能尽可能多地携带$X$中的信息，同时，$t_h$对因变量系统$F_0$有最大的解释能力。注意，无需在$F_0$中提取成分得分$u_h$，这可以使计算过程大为简化，并且对算法结论的解释也更为方便。

偏最小二乘法的简记算法的步骤如下：

（1）求矩阵$E_0^T F_0 F_0^T E_0$最大特征值所对应的特征向量w_1，求得成分$t_1 = w_1^T X$，计算成分得分向量$\hat{t}_1 = E_0 w_1$，和残差矩阵$E_1 = E_0 - \hat{t}_1 \alpha_1^T$，其中$\alpha_1 = E_0^T \hat{t}_1 \Big/ \left\|\hat{t}_1\right\|^2$。

（2）求矩阵$E_1^T F_0 F_0^T E_1$最大特征值所对应的特征向量w_2，求得成分$t_2 = w_2^T X$，计算成分得分向量$\hat{t}_2 = E_1 w_2$，和残差矩阵$E_2 = E_1 - \hat{t}_2 \alpha_2^T$，其中$\alpha_2 = E_1^T \hat{t}_2 \Big/ \left\|\hat{t}_2\right\|^2$。

⋮

（r）至第r步，求矩阵$E_{r-1}^T F_0 F_0^T E_{r-1}$最大特征值所对应的特征向量$w_r$，求得成分$t_r = w_r^T X$，计算成分得分向量$\hat{t}_r = E_{r-1} w_r$。

如果根据交叉有效性，确定共抽取 r 个成分 $t_1,\cdots,t_r$ 可以得到一个满意的预测模型，则求 F_0 在 $\hat{t}_1,\cdots,\hat{t}_r$ 上的普通最小二乘回归方程为

$$F_0=\hat{t}_1\beta_1^T+\cdots+\hat{t}_r\beta_r^T+F_r$$

把 $t_k=w_{k1}^*x_1+\cdots+w_{km}^*x_m$（$k=1,2,\cdots,r$），代入 $Y=t_1\beta_1+\cdots+t_r\beta_r$，即得 p 个因变量的偏最小二乘回归方程式

$$y_j=a_{j1}x_1+\cdots+a_{jm}x_m,\ (\ j=1,2,\cdots,m\ )。$$

这里的 w_h^* 满足 $\hat{t}_h=E_0w_h^*$，$w_h^*=\prod_{j=1}^{h-1}(I-w_j\alpha_j^T)w_h$。

§3 案例分析

本节采用兰纳胡德（Linnerud）给出的关于体能训练的数据进行偏最小二乘回归建模。在这个数据系统中被测的样本点，是某健身俱乐部的 20 位中年男子。被测变量分为两组。第一组是身体特征指标 X，包括：体重、腰围、脉搏。第二组变量是训练结果指标 Y，包括：单杠、弯曲、跳高。原始数据见表 1。

表 1　体能训练数据

No	体重(x_1)	腰围(x_2)	脉搏(x_3)	单杠(y_1)	弯曲(y_2)	跳高(y_3)
1	191	36	50	5	162	60
2	189	37	52	2	110	60
3	193	38	58	12	101	101
4	162	35	62	12	105	37
5	189	35	46	13	155	58
6	182	36	56	4	101	42
7	211	38	56	8	101	38
8	167	34	60	6	125	40
9	176	31	74	15	200	40
10	154	33	56	17	251	250
11	169	34	50	17	120	38
12	166	33	52	13	210	115
13	154	34	64	14	215	105
14	247	46	50	1	50	50
15	193	36	46	6	70	31
16	202	37	62	12	210	120

17	176	37	54	4	60	25
18	157	32	52	11	230	80
19	156	33	54	15	225	73
20	138	33	68	2	110	43
均值	178.6	35.4	56.1	9.45	145.55	70.3
标准差	24.6905	3.202	7.2104	5.2863	62.5666	51.2775

表 2 给出了这 6 个变量的简单相关系数矩阵。从相关系数矩阵可以看出，体重与腰围是正相关的；体重、腰围与脉搏负相关；而在单杠、弯曲与跳高之间是正相关的。从两组变量间的关系看，单杠、弯曲和跳高的训练成绩与体重、腰围负相关，与脉搏正相关。

表 2　相关系数矩阵

	1	0.8702	−0.3658	−0.3897	−0.4931	−0.2263
	0.8702	1	−0.3529	−0.5522	−0.6456	−0.1915
	−0.3658	−0.3529	1	0.1506	0.225	0.0349
	−0.3897	−0.5522	0.1506	1	0.6957	0.4958
	−0.4931	−0.6456	0.225	0.6957	1	0.6692
	−0.2263	−0.1915	0.0349	0.4958	0.6692	1

利用如下的 MATLAB 程序：

```
clc,clear
load pz.txt     %原始数据存放在纯文本文件 pz.txt 中
mu=mean(pz);sig=std(pz); %求均值和标准差
rr=corrcoef(pz);     %求相关系数矩阵
data=zscore(pz); %数据标准化
n=3;m=3;     %n 是自变量的个数,m 是因变量的个数
x0=pz(:,1:n);y0=pz(:,n+1:end);
e0=data(:,1:n);f0=data(:,n+1:end);
num=size(e0,1);%求样本点的个数
chg=eye(n);   %w 到 w*变换矩阵的初始化
for i=1:n
%以下计算 w，w*和 t 的得分向量，
    matrix=e0'*f0*f0'*e0;
    [vec,val]=eig(matrix); %求特征值和特征向量
    val=diag(val);     %提出对角线元素
    [val,ind]=sort(val,'descend');
    w(:,i)=vec(:,ind(1));       %提出最大特征值对应的特征向量
```

```
    w_star(:,i)=chg*w(:,i);    %计算 w*的取值
    t(:,i)=e0*w(:,i);          %计算成分 ti 的得分
    alpha=e0'*t(:,i)/(t(:,i)'*t(:,i));    %计算 alpha_i
    chg=chg*(eye(n)-w(:,i)*alpha');       %计算 w 到 w*的变换矩阵
    e=e0-t(:,i)*alpha';       %计算残差矩阵
    e0=e;
%以下计算 ss(i)的值
    beta=[t(:,1:i),ones(num,1)]\f0;     %求回归方程的系数
    beta(end,:)=[];     %删除回归分析的常数项
    cancha=f0-t(:,1:i)*beta;       %求残差矩阵
    ss(i)=sum(sum(cancha.^2));    %求误差平方和
%以下计算 press(i)
    for j=1:num
        t1=t(:,1:i);f1=f0;
        she_t=t1(j,:);she_f=f1(j,:);    %把舍去的第 j 个样本点保存起来
        t1(j,:)=[];f1(j,:)=[];          %删除第 j 个观测值
        beta1=[t1,ones(num-1,1)]\f1;    %求回归分析的系数
        beta1(end,:)=[];                %删除回归分析的常数项
        cancha=she_f-she_t*beta1;    %求残差向量
        press_i(j)=sum(cancha.^2);
    end
    press(i)=sum(press_i);
    if i>1
        Q_h2(i)=1-press(i)/ss(i-1);
    else
        Q_h2(1)=1;
    end
    if Q_h2(i)<0.0975
        fprintf('提出的成分个数 r=%d',i);
        r=i;
        break
    end
end
beta_z=[t(:,1:r),ones(num,1)]\f0;     %求 Y 关于 t 的回归系数
beta_z(end,:)=[];          %删除常数项
xishu=w_star(:,1:r)*beta_z;     %求 Y 关于 X 的回归系数，且是针对标准数据的回归系数，
每一列是一个回归方程
mu_x=mu(1:n);mu_y=mu(n+1:end);
sig_x=sig(1:n);sig_y=sig(n+1:end);
```

```
for i=1:m
    ch0(i)=mu_y(i)-mu_x./sig_x*sig_y(i)*xishu(:,i);  %计算原始数据的回归方程的常数项
end
for i=1:m
    xish(:,i)=xishu(:,i)./sig_x'*sig_y(i);  %计算原始数据的回归方程的系数，每一列是一个回归方程
end
sol=[ch0;xish]          %显示回归方程的系数，每一列是一个方程，每一列的第一个数是常数项
save mydata x0 y0 num xishu ch0 xish
```

计算得只要提出两个成分t_1, t_2即可，交叉有效性$Q_2^2 = -0.1969$。w_h与w_h^*的取值见表 3，成分t_h的得分$\hat{t}_h$见表 4。

表 3　w_h与w_h^*的取值

自变量	w_1	w_2	w_1^*	w_2^*
x_1	−0. 5899	-0.4688	−0. 5899	-0.3679
x_2	−0. 7713	0.5680	−0. 7713	0.6999
x_3	0. 2389	0.6765	0. 2389	0.6356

表 4　成分t_h的得分$\hat{t}_h$

No	1	2	3	4	5	6	7	8	9	10
$\hat{t}_1$	−0. 6429	−0. 7697	−0. 9074	0. 6884	−0. 4867	−0. 2291	−1. 4037	0. 7436	1. 7151	1. 1626
$\hat{t}_2$	−0. 5914	−0. 1667	0. 5212	0. 68	−1. 1328	0. 0717	0. 0767	0. 2106	0. 6549	−0. 1668
No	11	12	13	14	15	16	17	18	19	20
$\hat{t}_1$	0. 3645	0. 7433	1. 1867	−4. 3898	−0. 8232	−0. 749	−0. 3929	1. 1993	1. 0485	1. 9424
$\hat{t}_2$	−0. 7007	−0. 6983	0. 757	0. 76	−0. 9738	0. 5211	0. 2034	−0. 7827	−0. 3729	1. 1294

标准化变量$\tilde{y}_k$关于成分t_1的回归模型如下

$$\tilde{y}_k = r_{1k}t_1 + r_{2k}t_2,\quad k = 1,2,3$$

由于成分 t_h 可以写成原变量的标准化变量 $\tilde{x}_j$ 的函数，即有

$$t_h = w_{1h}^* \tilde{x}_1 + w_{2h}^* \tilde{x}_2 + w_{3h}^* \tilde{x}_3$$

由此可得由成分 t_1 所建立的偏最小二乘回归模型为

$$\begin{aligned}\tilde{y}_k &= r_{1k}(w_{11}^* \tilde{x}_1 + w_{21}^* \tilde{x}_2 + w_{31}^* \tilde{x}_3) + r_{2k}(w_{12}^* \tilde{x}_1 + w_{22}^* \tilde{x}_2 + w_{32}^* \tilde{x}_3) \\ &= (r_{1k} w_{11}^* + r_{2k} w_{12}^*)\tilde{x}_1 + (r_{1k} w_{21}^* + r_{2k} w_{22}^*)\tilde{x}_2 + (r_{1k} w_{31}^* + r_{2k} w_{32}^*)\tilde{x}_3\end{aligned}$$

有关 $r_h = (r_{h1}, r_{h2}, r_{h3})$ 的计算结果见表 5。

表 5　回归系数 r_h

k	1	2	3
r_1	0.3416	0.4161	0.1430
r_2	−0. 3364	−0. 2908	−0. 0652

所以，有

$$\tilde{y}_1 = -0.0778\tilde{x}_1 - 0.4989\tilde{x}_2 - 0.1322\tilde{x}_3$$

$$\tilde{y}_2 = -0.1385\tilde{x}_1 - 0.5244\tilde{x}_2 - 0.0854\tilde{x}_3$$

$$\tilde{y}_3 = -0.0604\tilde{x}_1 - 0.1559\tilde{x}_2 - 0.0073\tilde{x}_3$$

将标准化变量 $\tilde{y}_k, \tilde{x}_k (k = 1,2,3)$ 分别还原成原始变量 $y_k, x_k (k = 1,2,3)$，则回归方程为

$$y_1 = 47.0197 - 0.0167x_1 - 0.8237x_2 - 0.0969x_3$$

$$y_2 = 612.5671 - 0.3509x_1 - 10.2477x_2 - 0.7412x_3$$

$$y_3 = 183.9849 - 0.1253x_1 - 2.4969x_2 - 0.0518x_3$$

为了更直观、迅速地观察各个自变量在解释 $y_k (k = 1,2,3)$ 时的边际作用，可以绘制回归系数图，见图 1。这个图是针对标准化数据的回归方程的。

从回归系数图中可以立刻观察到，腰围变量在解释三个回归方程时起到了极为重要的作用。然而，与单杠及弯曲相比，跳高成绩的回归方程显然不够理想，三个自变量对它的解释能力均很低。

为了考察这三个回归方程的模型精度，我们以 $(\hat{y}_{ik}, y_{ik})$ 为坐标值，对所有的样本点绘制预测图。$\hat{y}_{ik}$ 是第 k 个变量，第 i 个样本点 (y_{ik}) 的预测值。在这个预测图上，如果所有点都能在图的对角线附近均匀分布，则方程的拟合值与原值差异很小，这个方程的拟合效果就是满意的。体能训练的预测图见图 2。

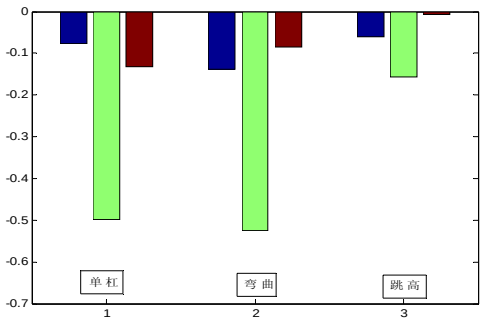

图 1　回归系数的直方图

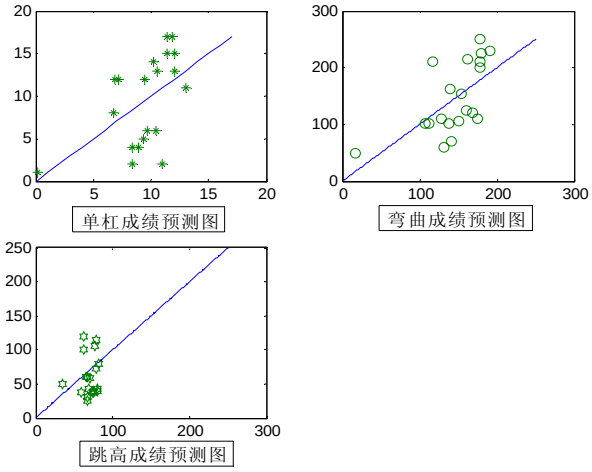

图 2　体能训练预测图

画直方图的 MATLAB 程序为：bar(xishu')
画体能训练的预测图的 MATLAB 程序如下：

```
load mydata
num
ch0=repmat(ch0,num,1);
yhat=ch0+x0*xish; %计算 y 的预测值
y1max=max(yhat);
y2max=max(y0);
```

```
ymax=max([y1max;y2max])
cancha=yhat-y0; %计算残差
subplot(2,2,1)
plot(0:ymax(1),0:ymax(1),yhat(:,1),y0(:,1),'*')
subplot(2,2,2)
plot(0:ymax(2),0:ymax(2),yhat(:,2),y0(:,2),'O')
subplot(2,2,3)
plot(0:ymax(3),0:ymax(3),yhat(:,3),y0(:,3),'H')
```