PubH 7405 Biostat Regression - Fall 2005

Baolin Wu, Ph.D.
Division of Biostatistics
School of Public Health
University of Minnesota
baolin@biostat.umn.edu

Lecture 10/11 - Oct 10/12, 2005
Matrix Approach to Simple Linear Regression Analysis

- Matrix Overview
- Matrix Approach to Linear Regression
Matrix Definition

A matrix is a \textit{rectangular} array of elements arranged in \textit{rows} and \textit{columns}. The \textit{dimension} of the matrix is $n \times p$, where $n$ is the number of rows and $p$ the number of columns.

A matrix with $n$ rows and $p$ columns is usually represented using \textbf{boldface} letters, say $A$, which can be represented as

$$A = \begin{bmatrix}
a_{11} & a_{12} & \cdots & a_{1j} & \cdots & a_{1p} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
a_{i1} & a_{i2} & \cdots & a_{ij} & \cdots & a_{ip} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
a_{n1} & a_{n2} & \cdots & a_{nj} & \cdots & a_{np}
\end{bmatrix}.$$
or

\[ A = [a_{ij}] \quad i = 1, \cdots, n; \quad j = 1, \cdots, p, \]

where \( a_{ij} \) is the element in the \( i \)th row and \( j \)th column.

Two matrices \( A = B \), if their corresponding elements are equal, \( a_{ij} = b_{ij} \).

When \( n = p \), matrix \( A \) is called a square matrix. When \( p = 1 \), \( A \) is called a column vector or simply a vector. When \( n = 1 \), \( A \) is called a row vector.

column vector: \[
\begin{bmatrix}
Y_1 \\
Y_2 \\
Y_3
\end{bmatrix},
\begin{bmatrix}
X_1 \\
X_2 \\
X_3
\end{bmatrix}
\]

row vector: \[
\begin{bmatrix}
1 & X_1
\end{bmatrix}
\]

square matrix: \[
\begin{bmatrix}
3 & \sum_{i=1}^{3} X_i \\
\sum_{i=1}^{3} X_i & \sum_{i=1}^{3} X_i^2
\end{bmatrix}
\]

(design) matrix: \[
\begin{bmatrix}
1 & X_1 \\
1 & X_2 \\
1 & X_3
\end{bmatrix}
\]
R commands for creating matrix

## simulate data for a regression model
n = 25; p = 2
x = rnorm(n); y = 1 + 2*x + rnorm(n)
## or use real data
toluca = read.table("toluca.txt", head=TRUE)
x = toluca$LotSize; y = toluca$WorkHours
### create a matrix
A = matrix( c(rep(1,n),x), n,p, byrow=FALSE )
dim(A); nrow(A); ncol(A)
### column/row vector: R indexing cmd
A[,1]; A[,2] ## jth column
A[1,]; A[n,] ## ith row
### Create matrix by binding columns/rows together
cbind(1, x); rbind(1, x)

### Extract the (i,j)th element

## Matrix is a special vector with "dim" attribute

A[1,1]; A[n,p]; A[1]; A[n*p]
Matrix Transpose

For $A = [a_{ij}]$, the transpose

$$A' = [a_{ji}], \ i = 1, \cdots, n; \ j = 1, \cdots, p.$$  \hspace{1cm} (1)

column vector: $Y = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix}$

(transpose) row vector: $Y' = \begin{bmatrix} Y_1 & Y_2 & \cdots & Y_n \end{bmatrix}$
A is called a symmetric matrix if $A = A'$, which implies $n = p$ and $a_{ij} = a_{ji}$.

symmetric matrix: $\begin{bmatrix}
    \sum_{i=1}^{n} X_i & \sum_{i=1}^{n} X_i \\
    \sum_{i=1}^{n} X_i & \sum_{i=1}^{n} X_i^2
\end{bmatrix}$

R commands for transposing a matrix

### t() for matrix transpose

t(x); t(y)
Ap = t(A)

### checking the dim and individual elements

dim(A); dim(Ap)
A[2,1]; Ap[1,2]
A[n,p]; Ap[p,n]
Matrix Summation and Subtraction

Element-wise summation and subtraction

\[ C = A \pm B : c_{ij} = a_{ij} \pm b_{ij}. \]  \hspace{1cm}(2)

R commands

## create a random matrix B
B = matrix( rnorm(n*p), n, p )
## Summation and subtraction
A + B; A - B
Matrix Multiplication

Inner product of two $\mathbb{R}^n$ vectors,

$$x = (x_1, \cdots, x_n), \quad y = (y_1, \cdots, y_n)$$

is defined as

$$\langle x, y \rangle = \sum_{k=1}^{n} x_k y_k.$$  \hfill (3)

The product of a scalar $\lambda$ (an ordinary number) and matrix $A$ is

$$\lambda A = [\lambda a_{ij}].$$

The product of matrix $A$ and $B$ is determined by the inner products of
matrix rows and columns

\[ C = AB : \quad c_{ij} = \langle (a_{i1}, \cdots, a_{in}), (b_{1j}, \cdots, b_{nj}) \rangle = \sum_{k=1}^{n} a_{ik} b_{kj}, \quad (4) \]

i.e. the \( ij \)th element of the product matrix is the inner product of the \( i \)th row of \( A \) and the \( j \)th column of \( B \) (viewed as vectors in \( \mathbb{R}^n \)). So \( A \) must have the same number of columns as the number of rows of \( B \). Generally \( AB \neq BA \).

The \( ij \)th element of product matrix \( A' A \) is the inner product of \( i \)th and \( j \)th columns of \( A \).
\[ Y'Y = \begin{bmatrix} Y_1 & Y_2 & \cdots & Y_n \end{bmatrix} \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix} = [Y_1^2 + Y_2^2 + \cdots + Y_n^2] = [\sum Y_i^2] \]

\[ X\beta = \begin{bmatrix} 1 & X_1 \\ 1 & X_2 \\ \vdots & \vdots \\ 1 & X_n \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} = \begin{bmatrix} \beta_0 + X_1\beta_1 \\ \beta_0 + X_2\beta_1 \\ \vdots \\ \beta_0 + X_n\beta_1 \end{bmatrix} \]
\[ X'X = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ X_1 & X_2 & \cdots & X_n \end{bmatrix} \begin{bmatrix} 1 & X_1 \\ 1 & X_2 \\ \vdots & \vdots \\ 1 & X_n \end{bmatrix} = \begin{bmatrix} n & \sum X_i \\ \sum X_i & \sum X_i^2 \end{bmatrix} \]

\[ X'Y = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ X_1 & X_2 & \cdots & X_n \end{bmatrix} \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix} = \begin{bmatrix} \sum Y_i \\ \sum X_iY_i \end{bmatrix} \]
R commands for matrix product

## create matrix X and Y
X = cbind(1,x); Y = cbind(y)
Bcoef = cbind( c(1,2) )
## "%%" is matrix product operator
t(Y)%*%Y
t(X)%*%X; X%*%t(X)
t(X)%*%Y; X%*%Bcoef
Special Types of Matrix

- **Diagonal Matrix**: square matrix with off-diagonal elements being zero.

- **Identify Matrix** $I$: diagonal matrix with diagonal elements being 1.

- **Scalar Matrix**: diagonal matrix with diagonal elements being the same, $\lambda I$.

- **1**: a column vector with all elements being 1. $J$: a square matrix with all elements being 1. $0$: a column vector containing only zeros.

$$1 = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}, \quad J = \begin{bmatrix} 1 & \ldots & 1 \\ 1 & \ldots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \ldots & 1 \end{bmatrix}$$
We have

\[ 1'1 = n, \quad 11' = J. \]

**R commands**

```r
## symmetric matrix
Xs = t(X) %*% X; t(Xs) - Xs
## create diagonal matrix
diag(rnorm(n))
diag(n) ## n x n identity matrix
## extract diagonal elements of a square matrix
diag(Xs)
## special matrix
matrix(1, nrow=n, ncol=1); matrix(1, nrow=1, ncol=p)
matrix(1, ncol=p, nrow=p)
```
Rank of Matrix

The rank of a matrix is defined to be the maximum number of linearly independent columns in the matrix.

\[
\text{Rank}(AB) \leq \min(\text{Rank}(A), \text{Rank}(B))
\]
\[
\text{Rank}(A) = \text{Rank}(A') \leq \min(n, p)
\]
\[
\text{Rank}(A'A) = \text{Rank}(AA') = \text{Rank}(A)
\]

where \((n, p)\) are the number of rows and columns of matrix \(A\).

The \(p\) columns of matrix \(A\) is called linearly dependent if there exists \(p\) scalars \(\lambda_1, \cdots, \lambda_p\) not all zero and

\[
\lambda_1 C_1 + \lambda_2 C_2 + \cdots + \lambda_p C_p = 0,
\]
otherwise they are called independent, where $C_i$ is the $i$th column of $A$.

**R commands**

```r
## several ways to get the ranks
qr(A,LAPACK=FALSE)$rank
svd(A)$d

## rank of matrix product
D1 = t(A) %*% A; D2 = A %*% t(A)
qr(D1)$rank; qr(D2)$rank
```
Matrix Inverse

The inverse of matrix $A$ is another matrix, denoted by $A^{-1}$, such that

$$AA^{-1} = A^{-1}A = I. \quad (6)$$

$A^{-1}$ exists if $\text{Rank}(A)$ equals to the number of rows/columns, when $A$ is said to be nonsingular or of full rank (equivalent to the determinant not being zero).

Sample inverse matrix

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}, \quad A^{-1} = \begin{bmatrix} \frac{d}{D} & \frac{-c}{D} \\ \frac{-b}{D} & \frac{a}{D} \end{bmatrix},$$
where $D = ab - cd$ is the **determinant** of the matrix $A$.

\[
X'X = \begin{bmatrix}
\sum X_i & \sum X_i^2 \\
\sum X_i & \sum X_i^2
\end{bmatrix}, \quad D = n \sum X_i^2 - (\sum X_i)^2 = n \sum (X_i - \bar{X})^2.
\]

\[
(X'X)^{-1} = \begin{bmatrix}
\frac{1}{n} + \frac{\bar{X}^2}{\sum (X_i - \bar{X})^2} & \frac{-\bar{X}}{\sum (X_i - \bar{X})^2} \\
\frac{-\bar{X}}{\sum (X_i - \bar{X})^2} & \frac{1}{\sum (X_i - \bar{X})^2}
\end{bmatrix}
\]

(R) commands

```r
## matrix determinant
Xs = t(X) %*% X; det(Xs)
## matrix inverse
Xs.inv = solve(Xs)
Xs%*%Xs.inv; Xs.inv%*%Xs
```
Some Basic Theorems for Matrix

\[ A + B = B + A, \quad (A + B) + C = A + (B + C) \]
\[ (AB)C = A(BC), \quad C(A + B) = CA + CB, \quad \lambda(A + B) = \lambda A + \lambda B \]
\[ (A')' = A, \quad (A + B)' = A' + B', \quad (AB)' = B'A', \quad (ABC)' = C'B'A' \]
\[ (A^{-1})^{-1} = A, \quad (AB)^{-1} = B^{-1}A^{-1}, \quad (ABC)^{-1} = C^{-1}B^{-1}A^{-1} \]
\[ (A')^{-1} = (A^{-1})' \]

(8)
R commands

## create matrix
A = matrix(rnorm(4),2,2); B = matrix(rnorm(4),2,2)
C = matrix(rnorm(4),2,2); lambda = 1.0

## sum and product
\{(A+B)+C\} - \{A+(B+C)\}
\{(A B) C\} - \{A (B C)\}; \{C (A+B)\} - \{C A+C B\}
\{lambda (A+B)\} - \{lambda A+lambda B\}

## transpose and inverse
\{t(t(A))\} - A; \{t(t(A+B))\} - \{t(A)+t(B)\}
\{t(A B)\}-\{t(B) t(A)\}; \{t(A B C)\}-\{t(C) t(B) t(A)\}
\{solve(solve(A))\} - A; \{solve(A B)\} - \{solve(B) solve(A)\}
\{solve(A B C)\} - \{solve(C) solve(B) solve(A)\}
\{solve(t(A))\} - \{t(solve(A))\}
Random Vector and Matrix

A random vector or a random matrix contains elements that are random variables. Their expectations are defined as the element-wise expectations

\[ E(A) = [E(a_{i,j})]. \]  \hspace{1cm} (9)

The \textit{variance-covariance matrix} of random vector

\[ Y = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix} \]
are defined as

\[
\sigma^2(Y) = \begin{bmatrix}
\sigma^2(Y_1) & \sigma(Y_1, Y_2) & \cdots & \sigma(Y_1, Y_n) \\
\sigma(Y_2, Y_1) & \sigma^2(Y_2) & \cdots & \sigma(Y_2, Y_n) \\
\vdots & & & \vdots \\
\sigma(Y_n, Y_1) & \sigma(Y_n, Y_2) & \cdots & \sigma^2(Y_n)
\end{bmatrix} = [\sigma(Y_i, Y_j)], \quad (10)
\]

which is symmetric because \(\sigma(Y_i, Y_j) = \sigma(Y_j, Y_i)\).

Suppose \(A\) is a constant matrix, define \(W = AY\). We have

\[
E(A) = A \\
E(W) = E(AY) = AE(Y) \\
\sigma^2(W) = \sigma^2(AY) = A\sigma^2(Y)A'
\]  

(11)
R commands

## create two random samples
Y = matrix( rnorm(100*2), 100,2 )

## covariance and correlation matrix
cov(Y); cor(Y)

## product matrix
## product matrix
A = matrix( sample(1:9, size=4), 2, 2 )
W = Y%*%t(A)
cov(W); A%*%cov(Y)%*%t(A)
Derivative in Matrix Function

Consider the linear function of parameters

$$\beta' = \begin{bmatrix} \beta_1 & \beta_2 & \cdots & \beta_p \end{bmatrix}$$

in the matrix form

$$F(\beta) = A'\beta = \beta'A, \ A = [a_{i1}], i = 1, \cdots, p.$$  

It’s obvious that

$$\frac{\partial F}{\partial \beta_i} = a_{i1}.$$
So the vector of derivatives

\[ \frac{\partial F}{\partial \beta} = \begin{bmatrix} \frac{\partial F}{\partial \beta_1} \\ \frac{\partial F}{\partial \beta_2} \\ \vdots \\ \frac{\partial F}{\partial \beta_p} \end{bmatrix} = A. \quad (12) \]

Consider the following \textit{quadratic function} of \( \beta \)

\[ Q(\beta) = \beta'Q\beta, \quad Q = [q_{ij}], i = 1, \cdots, p, j = 1, \cdots, p \quad (13) \]

In equation (13), the terms involving \( \beta_i \) are

\[ q_{ii}\beta_i^2 + \left\{ \sum_{j \neq i} (q_{ij} + q_{ji}) \beta_j \right\} \beta_i. \]
therefore

\[
\frac{\partial Q}{\partial \beta_i} = 2q_{ii}\beta_i + \sum_{j \neq i} (q_{ij} + q_{ji})\beta_j
\]

\[= \sum_j (q_{ij} + q_{ji})\beta_j, \quad (14)\]

and, hence we have the vector of derivatives

\[
\frac{\partial Q}{\partial \beta} = \begin{bmatrix}
\frac{\partial Q}{\partial \beta_1} \\
\frac{\partial Q}{\partial \beta_2} \\
\vdots \\
\frac{\partial Q}{\partial \beta_p}
\end{bmatrix} = (Q + Q')\beta 
\]

\[= 2Q\beta \quad \text{if } q_{ij} = q_{ji}. \quad (15)\]
For quadratic function (13), we can always write it using a symmetric matrix

\[ \tilde{Q} = \frac{1}{2}(Q + Q'), \quad Q(\beta) = \beta' \tilde{Q} \beta, \quad \frac{\partial Q}{\partial \beta} = 2\tilde{Q} \beta. \]
Simple Linear Regression in Matrix Terms

The normal error regression model

\[ Y_i = \beta_0 + X_i \beta_1 + \epsilon_i \quad i = 1, \cdots, n, \]

can be compactly written in matrix terms

\[ \mathbf{Y} = \mathbb{E}(\mathbf{Y}) + \mathbf{\epsilon} = \mathbf{X}\mathbf{\beta} + \mathbf{\epsilon}, \quad (16) \]

where

\[ \mathbf{Y} = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} 1 & X_1 \\ 1 & X_2 \\ \vdots & \vdots \\ 1 & X_n \end{bmatrix}, \quad \mathbf{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}, \quad \mathbf{\epsilon} = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix} \]
Our assumption for the error term $\epsilon$ can be summarized as $\epsilon$ is a vector of independent normal variables with $\mathbb{E}(\epsilon) = 0$ and $\sigma^2(\epsilon) = \sigma^2 I$, where $I$ is a $n \times n$ identity matrix.
**Least Squares Estimation**

Our goal is to minimize the quadratic difference

\[
Q = \| Y - X\beta \|^2 = (Y - X\beta)'(Y - X\beta) \\
= \beta'X'X\beta - \beta'X'Y - Y'X\beta + Y'Y \\
= \beta'X'X\beta - 2\beta'X'Y + Y'Y \tag{17}
\]

Refer to equation (12) and (15), we have

\[
\frac{\partial Q}{\partial \beta} = 2X'X\beta - 2X'Y = 2X'(X\beta - Y), \tag{18}
\]

So the least squares estimator \( b = [b_0 \ b_1]' \) satisfy the following normal
equation

\[ X'(Xb - Y) = 0, \] (19)

which says the residuals

\[ e = Y - Xb \]

are orthogonal to the columns of \( X \), i.e.

\[ 1'e = \sum_{i=1}^{n} e_i = 0, \quad \sum_{i=1}^{n} X_i e_i = 0. \]

We can also derive the solution in matrix term as

\[ b = (X'X)^{-1}X'Y. \] (20)
According to (11) we have

\[
\sigma^2(b) = (X'X)^{-1}X'Cov(Y)\{(X'X)^{-1}X'\}'
\]

\[
= (X'X)^{-1}X'\sigma^2IX(X'X)^{-1}
\]

\[
= \sigma^2(X'X)^{-1}
\]

(21)
Fitted Values and Residuals

The fitted values

\[ \hat{Y} = \begin{bmatrix} \hat{Y}_1 \\ \hat{Y}_2 \\ \vdots \\ \hat{Y}_n \end{bmatrix} = Xb = \begin{bmatrix} 1 & X_1 \\ 1 & X_2 \\ \vdots & \vdots \\ 1 & X_n \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \end{bmatrix} \]

According to (20) we have

\[ \hat{Y} = Xb = X(X'X)^{-1}X'Y, \]

where

\[ H = X(X'X)^{-1}X' \quad (22) \]
is called the hat matrix or projection matrix. It is obvious that $H$ is symmetric and

$$HH = H.$$ 

In general, a matrix $M$ is said to be idempotent if $MM = M$.

According to (11) we have

$$\sigma^2(\hat{Y}) = HCov(Y)H' = H\sigma^2 IH$$

$$= \sigma^2 H$$

(23)

Similarly we can write residuals as

$$e = Y - \hat{Y} = Y - HY = (I - H)Y.$$ 

(24)

The matrix $I - H$ is also symmetric and idempotent.
According to (11) we have

\[
\sigma^2(e) = (I - H)Cov(Y)(I - H)'
\]

\[
= (I - H)\sigma^2 I(I - H)
\]

\[
= \sigma^2(I - H)
\]
Analysis of Variance Results

Some useful facts

\[ J = \begin{bmatrix}
1 & 1 & \cdots & 1 \\
1 & 1 & \cdots & 1 \\
\vdots & \vdots & \ddots & \vdots \\
1 & 1 & \cdots & 1 \\
\end{bmatrix} = \begin{bmatrix}
1 \\
1 \\
\vdots \\
1 \\
\end{bmatrix} \begin{bmatrix}
1 & 1 & \cdots & 1 \\
\end{bmatrix} = 11'
\]

We have

\[ Y'Y = \sum_{i=1}^{n} Y_i^2, \quad Y'1 = \sum_{i=1}^{n} Y_i \]
and
\[
\left( \sum_{i=1}^{n} Y_i \right)^2 = Y'1(Y'1)' = Y'11'Y = Y'JY
\]

So
\[
\begin{align*}
SSTO & = \sum_{i=1}^{n} Y_i^2 - \left( \sum_{i=1}^{n} Y_i \right)^2 / n = Y'Y - \frac{1}{n} Y'JY \\
SSE & = e'e = Y'(I - H)Y = Y'Y - Y'HY \\
& = e'(Y - Xb) = e'Y = (Y - Xb)'Y = Y'Y - b'X'Y \\
SSR & = SSTO - SSE = b'X'Y - \frac{1}{n} Y'JY \\
& = Y'HY - \frac{1}{n} Y'JY,
\end{align*}
\]
therefore we have a unified formula for these three sum of squares

\[ Y'AY, \]

where \( A \) are

\[
\begin{align*}
SSTO &: I - \frac{1}{n}J \\
SSE &: I - H \\
SSR &: H - \frac{1}{n}J,
\end{align*}
\]

so they are all quadratic functions of \( Y \).
Statistical Inferences

As shown in (21) and (7)

\[ \sigma^2(b) = \sigma^2(X'X)^{-1} = \begin{bmatrix}
\frac{\sigma^2}{n} + \frac{\sigma^2 \bar{X}^2}{\sum(X_i - \bar{X})^2} & \frac{-\bar{X} \sigma^2}{\sum(X_i - \bar{X})^2} \\
\frac{\sum(X_i - \bar{X})^2}{\sum(X_i - \bar{X})^2} & \frac{\sum(X_i - \bar{X})^2}{\sum(X_i - \bar{X})^2}
\end{bmatrix}, \]

replace \( \sigma^2 \) with \( MSE \) we can derive the estimated variance matrix \( s^2(b) \).

The mean response at \( X_h \) can be written as

\[ \hat{Y}_h = X'_h b, \text{ where } X_h = [1 \ X_h]', \]
so

\[ \sigma^2(\hat{Y}_h) = X'_h \sigma^2(b) X_h = \sigma^2 X'_h (X'X)^{-1} X_h \]

\[ = \sigma^2 \left\{ \frac{1}{n} + \frac{(X_h - \bar{X})^2}{\sum (X_i - \bar{X})^2} \right\} , \quad (27) \]

replace \( \sigma^2 \) with \( MSE \) we can derive the estimated variance

\[ s^2(\hat{Y}_h) = MSE \left\{ \frac{1}{n} + \frac{(X_h - \bar{X})^2}{\sum (X_i - \bar{X})^2} \right\} . \]

When predicting a new observation, we have

\[ \sigma^2(\text{pred}) = \sigma^2 + \sigma^2(\hat{Y}_h) = \sigma^2(1 + X'_h (X'X)^{-1} X_h) , \]
replace $\sigma^2$ with $MSE$ we can derive the estimated variance

$$s^2(pred) = MSE \left\{ 1 + \frac{1}{n} + \frac{(X_h - \bar{X})^2}{\sum(X_i - \bar{X})^2} \right\}.$$
R commands

```r
### matrix approach to OLS
n = 25; p = 2
x = rnorm(n); y = 1 + 2*x + rnorm(n)
reg = lm(y~x)

bY = matrix(y, ncol=1) ## cbind(y)
bX = cbind(1,x)
Xs = t(bX)%*%bX
Xs.inv = solve(Xs)

## LS estimates
bhat = Xs.inv%*%t(bX)%*%bY
## bhat = solve(Xs,t(bX)%*%bY)
```
## put estimates row by row for comparison
rbind(reg$coef, t(bhat))

## prediction and residuals
Hmat = bX%*%Xs.inv%*%t(bX)
## Hmat = bX%*%solve(Xs,t(bX))
Yhat = Hmat%*%bY  ## bX%*%bhat
res = bY-Yhat  ## (diag(p)-Hmat)%*%bY

rbind(reg$fit, t(Yhat))
rbind(reg$res, t(res))

## MSE
sig2 = sum(res^2)/(n-p)
## normal equation
round( t(bX) %*% res, 7 )

## variance estimation
s2b = sig2 * Xs.inv
s2yhat = sig2 * Hmat
s2res = sig2 * (diag(n) - Hmat)

## ANOVA
## special matrix J
J = matrix(1, n, n)
Assto = diag(n) - J/n
Asse = diag(n) - Hmat
Assr = Hmat - J/n
ssto = t(bY) %*% Assto %*% bY
sse = t(bY)%*%Asse%*%bY
ssr = t(bY)%*%Assr%*%bY

anova(reg)
c(ssr,sse,ssto)

## mean estimation and prediction
Xh = matrix(c(1,1.5), ncol=1)
Yhhat = t(Xh)%*%bhat
predict(reg, data.frame(x=1.5))
s2yh = sig2*t(Xh)%*%Xs.inv%*%Xh
sig2*(1/n+(Xh[2]-mean(x))^2/sum((x-mean(x))^2))