Chapter 3

Multiple Linear Regression

Chapter 3g

consider all the regression coeffs together

Simultaneous confidence intervals on regression coefficients

(k regressor variables) $y = X\beta + \varepsilon$ where $\beta = \begin{pmatrix} \beta_0 \\ \vdots \\ \beta_k \end{pmatrix}$

$p \times 1$

Simultaneous confidence intervals on regression coefficients

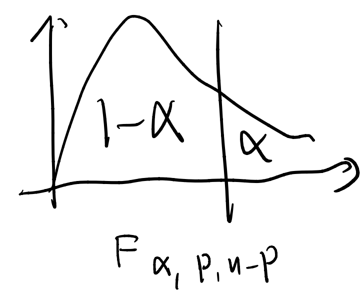
A simultaneous confidence interval for $\boldsymbol{\beta}$ is a joint region that applies simultaneously to the entire set of regression coefficients.

(a) Ellipsoidal confidence region

1. Linear model theory: $\dfrac{(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})' \boldsymbol{X}' \boldsymbol{X} (\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})}{p\, MS_{Res}} \sim F_{p,n-p}$

measure of distance between $\hat{\beta}$ and β

$$P\left\{ \frac{(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})' \boldsymbol{X}' \boldsymbol{X} (\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})}{p\, MS_{Res}} \leq F_{p,n-p} \right\} = 1 - \alpha$$



A $100(1-\alpha)\%$ joint confidence region for all of the parameters in $\boldsymbol{\beta}$ is

$$\frac{(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})' \boldsymbol{X}' \boldsymbol{X} (\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})}{p\, MS_{Res}} \leq F_{\alpha,p,n-p}$$

2. The joint confidence region is an ellipsoidal region.

3. Consider a simple linear regression model $y = \beta_0 + \beta_1 x + \epsilon$.
A $100(1-\alpha)\%$ joint confidence region for β_0 and β_1 is

find from normal equation

$$\frac{(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})' \boldsymbol{X}'\boldsymbol{X} (\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})}{2\ MS_{Res}} \leq F_{\alpha,2,n-2}$$

numerator reduced to a scalar quantity

$$\frac{(\hat{\beta}_0 - \beta_0, \hat{\beta}_1 - \beta_1) \begin{bmatrix} n & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} x_i^2 \end{bmatrix} \begin{bmatrix} \hat{\beta}_0 - \beta_0 \\ \hat{\beta}_1 - \beta_1 \end{bmatrix}}{2\ MS_{Res}} \leq F_{\alpha,2,n-2}$$

$$\frac{n(\hat{\beta}_0 - \beta_0)^2 + 2(\hat{\beta}_0 - \beta_0)(\hat{\beta}_1 - \beta_1) \sum_{i=1}^{n} x_i + (\hat{\beta}_1 - \beta_1)^2 \sum_{i=1}^{n} x_i^2}{2\ MS_{Res}} \leq F_{\alpha,2,n-2}$$

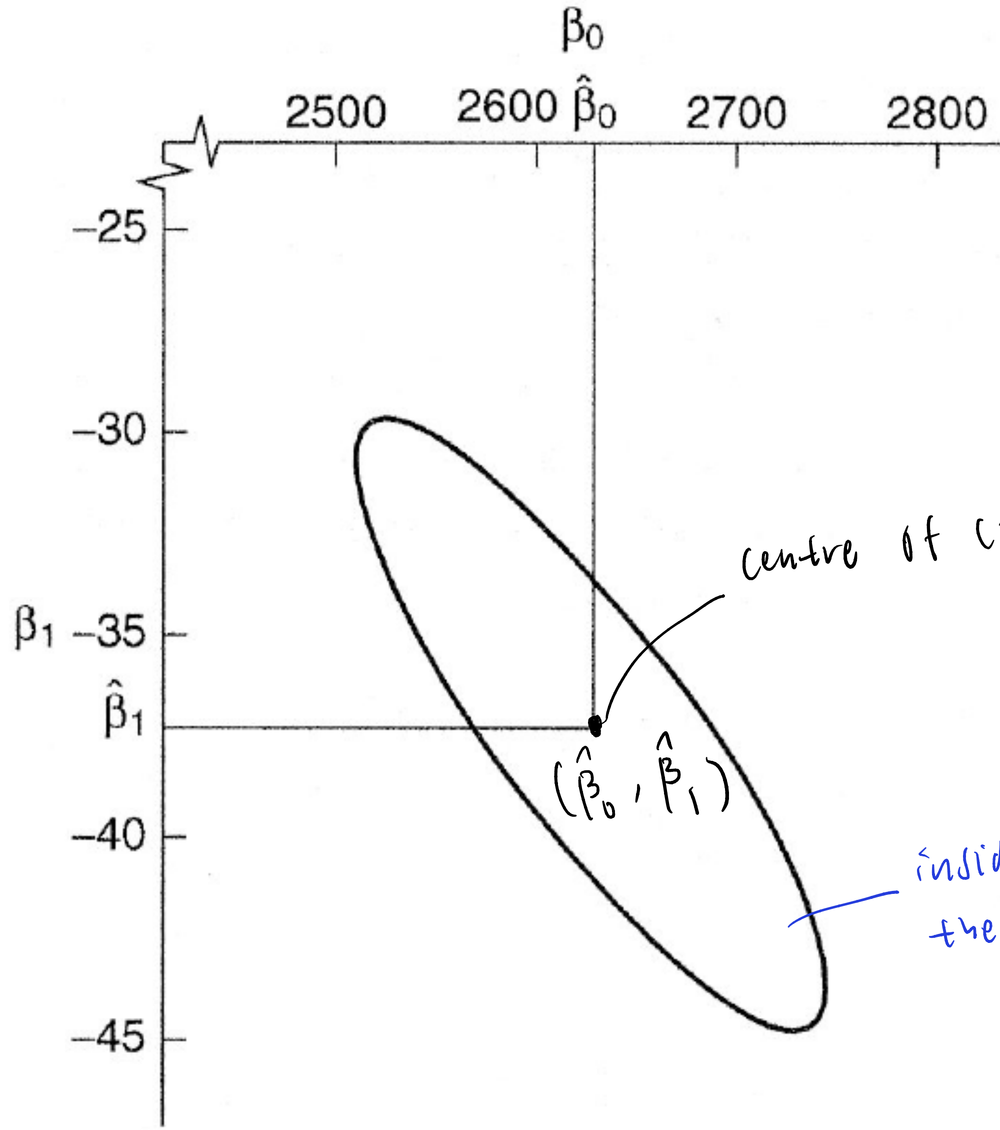
Example - Rocket propellant data

$\hat{\beta}_0 = 2627.82$, $\hat{\beta}_1 = -37.15$, $\sum_{i=1}^{20} x_i = 267.25$, $\sum_{i=1}^{20} x_i^2 = 4677.69$,
$MS_{Res} = 9244.59$, $F_{0.05,2,18} = 3.55$

A 95% joint confidence region for β_0 and β_1 is

$$\frac{20(2627.82 - \beta_0)^2 + 2(2627.82 - \beta_0)(-37.15 - \beta_1)267.25 + (-37.15 - \beta_1)^2 4677.69}{2(9244.59)}$$
$$\leq \ 3.55.$$

The joint confidence region is displayed here.



$\hat{\beta}_1 \pm t_{\frac{\alpha}{2}} \sqrt{\hat{\sigma}^2 \cdot c_{11}}$

centre of confidence region

inside ellipse: satisfies the inequality

(b) Bonferroni intervals

1. The Bonferroni approach is based on the Bonferroni inequality:

$$P(\bigcup_{i=1}^{m} A_i^c) \leq \sum_{i=1}^{m} P(A_i^c)$$

$P(A_1 \cup A_2) \leq P(A_1) + P(A_2)$

$P(A_1) + P(A_2) - \underbrace{P(A_1 \cap A_2)}_{\geq 0}$

We can then show that

$$1 - P(\bigcup_{i=1}^{m} A_i^c) \geq 1 - \sum_{i=1}^{m} P(A_i^c)$$

$$P(\bigcap_{i=1}^{m} A_i) \geq 1 - \sum_{i=1}^{m} P(A_i^c)$$

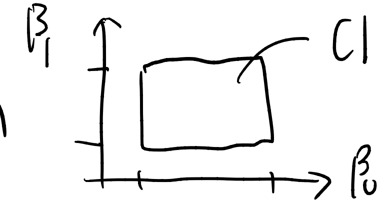
A_i: event that CI contains the parameter (eg. β_i)

Suppose $P(A_i) = 1 - \alpha/m$, then $P(A_i^c) = \alpha/m$, and $P(\bigcap_{i=1}^{m} A_i) \geq 1 - \alpha$.

considering m regression coeff (β_i)

In order to construct $(1-\alpha)100\%$ simultaneous confidence intervals for m parameters, we will construct $(1-\alpha/m)100\%$ confidence interval for each of the m parameters, the joint coverage probability of the m intervals will then be **at least** $1-\alpha$.

instead of ellipsical region



2. The Bonferroni confidence region is a rectangular region.

$p = k+1$

3. In order to construct a $(1-\alpha)100\%$ joint confidence region for $\beta_0, \beta_1, ..., \beta_k$, we will construct $(1-\alpha/p)100\%$ confidence intervals for $\beta_0, \beta_1, ..., \beta_k$,

$$\hat{\beta}_j - t_{\alpha/(2p),n-p}\sqrt{\hat{\sigma}^2 C_{jj}} \leq \beta_j \leq \hat{\beta}_j + t_{\alpha/(2p),n-p}\sqrt{\hat{\sigma}^2 C_{jj}}, j = 0, 1, 2, ..., k.$$

The p intervals form the Bonferroni confidence region. This region contains $(\beta_0, \beta_1, ..., \beta_k)'$ with a probability of **at least** $1-\alpha$, this means the intervals are wider than what they are supposed to be, therefore the Bonferroni confidence region is conservative.

Example - Rocket propellant data

only need to adjust p percentile

The R code

```
confint(lm(y~x),level=1-0.05/2)
```

can be used to construct Bonferroni confidence region by using $1 - \alpha/p$ instead of $1 - \alpha$ where $\alpha = 0.05$.

The Bonferroni confidence region for $\overbrace{\beta_0 \text{ and } \beta_1}^{m=2=p}$ is here.

$$\hat{\beta}_j \pm t_{\alpha/(2p),n-p}\,\mathrm{se}\left(\hat{\beta}_j\right)$$

$\alpha/(2p)$ = 0.05/(2x2) = 0.0125

$$\hat{\beta}_0 - t_{0.0125,18}\,\mathrm{se}\left(\hat{\beta}_0\right) \le \beta_0 \le \hat{\beta}_0 + t_{0.0125,18}\,\mathrm{se}\left(\hat{\beta}_0\right)$$
$$2627.822 - (2.445)(44.184) \le \beta_0 \le 2627.822 + (2.445)(44.184)$$
$$2519.792 \le \beta_0 \le 2735.852$$

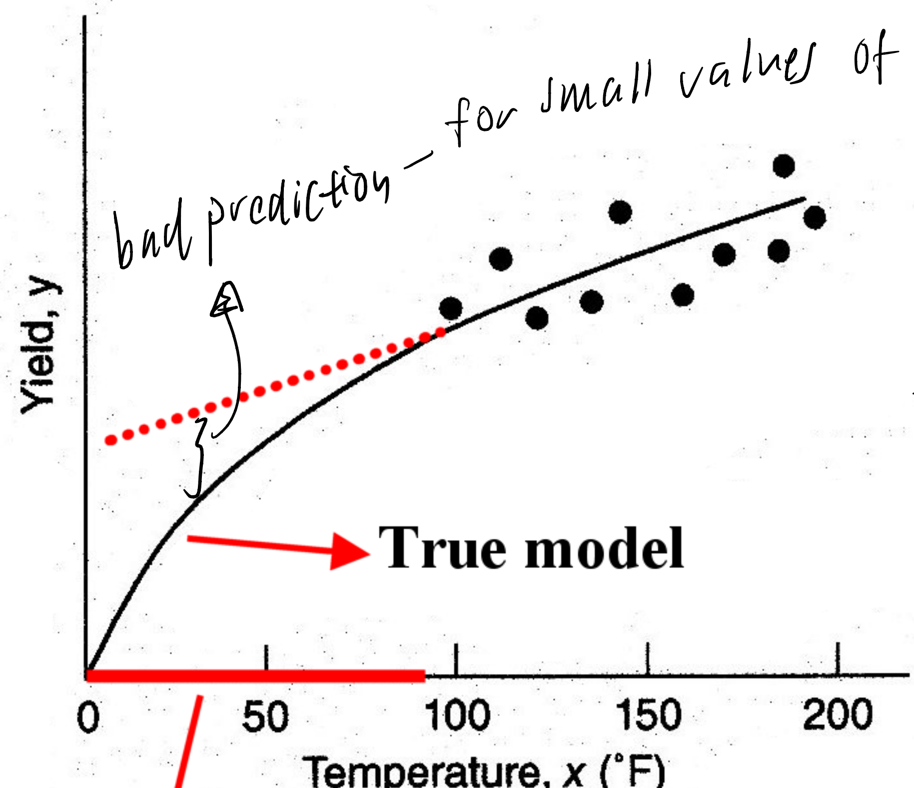
$$\hat{\beta}_1 - t_{0.0125,18}\,\mathrm{se}\left(\hat{\beta}_1\right) \le \beta_1 \le \hat{\beta}_1 - t_{0.0125,18}\,\mathrm{se}\left(\hat{\beta}_1\right)$$
$$-37.154 - (2.445)(2.889) \le \beta_1 \le -37.154 + (2.445)(2.889)$$
$$-44.218 \le \beta_1 \le -30.090$$

```
> confint(lm(y~x),level=1-0.05/2)
                     1.25 %    98.75 %
```

Hidden extrapolation in multiple regression

In predicting new responses and in estimating the mean response at a given point $x_{01}, x_{02}, \ldots, x_{0k}$ one must be careful about **extrapolating** beyond the region containing the original observations.

bad prediction — for small values of displacement, not linear

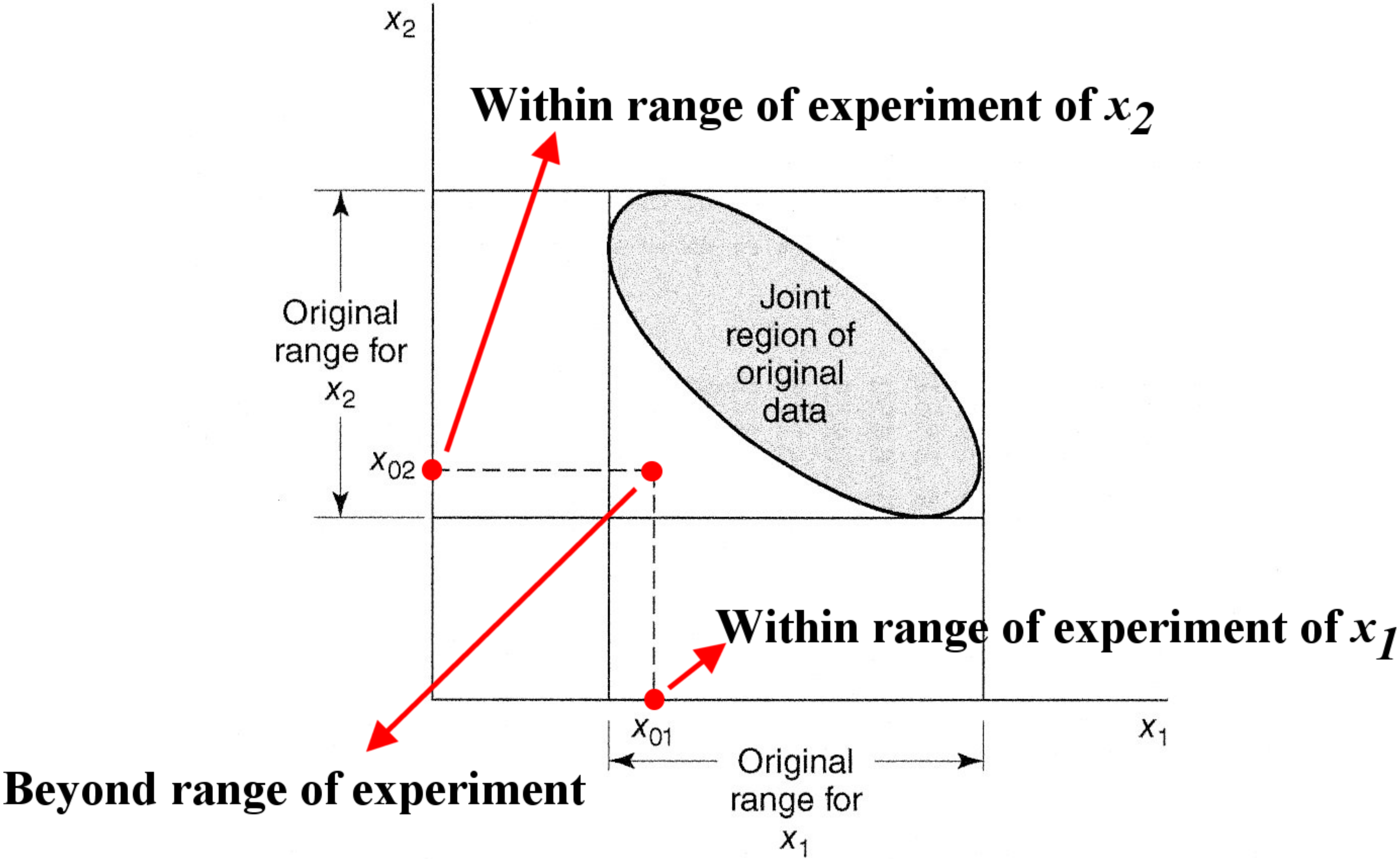


not good idea to force regression line thru origin

Extrapolation beyond range of experiment

must look at 2 variables to extrapolate

Extrapolation in multiple regression



If there are more than two regressor variables, it is difficult to check whether a point is an extrapolation point. We need a non-graphical approach.

Hidden extrapolation in multiple regression

1. Suppose we have the multiple linear regression model

$$\boldsymbol{y_{n\times 1}} = \boldsymbol{X_{n\times p}\beta_{p\times 1}} + \boldsymbol{\epsilon_{n\times 1}}$$

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ . \\ y_n \end{bmatrix}_{n\times 1} \qquad \boldsymbol{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & ... & x_{1k} \\ 1 & x_{21} & x_{22} & ... & x_{2k} \\ . & . & . & & . \\ . & . & . & & . \\ . & . & . & & . \\ 1 & x_{n1} & x_{n2} & ... & x_{nk} \end{bmatrix}_{n\times p} = \begin{bmatrix} \boldsymbol{x_1'} \\ \boldsymbol{x_2'} \\ . \\ . \\ . \\ \boldsymbol{x_n'} \end{bmatrix}_{n\times p}$$

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ . \\ . \\ \beta_k \end{bmatrix}_{p\times 1} \qquad \boldsymbol{\epsilon} = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ . \\ . \\ . \\ \epsilon_n \end{bmatrix}_{n\times 1}$$

where $\boldsymbol{\beta'} = (\beta_0, \beta_1, ..., \beta_k)$, $p = k+1$, $\boldsymbol{E(\epsilon) = 0}$ and $\boldsymbol{Var(\epsilon)} = \sigma^2\boldsymbol{I_n}$.

Assume $\boldsymbol{\epsilon \sim N(0}, \sigma^2\boldsymbol{I)}$.

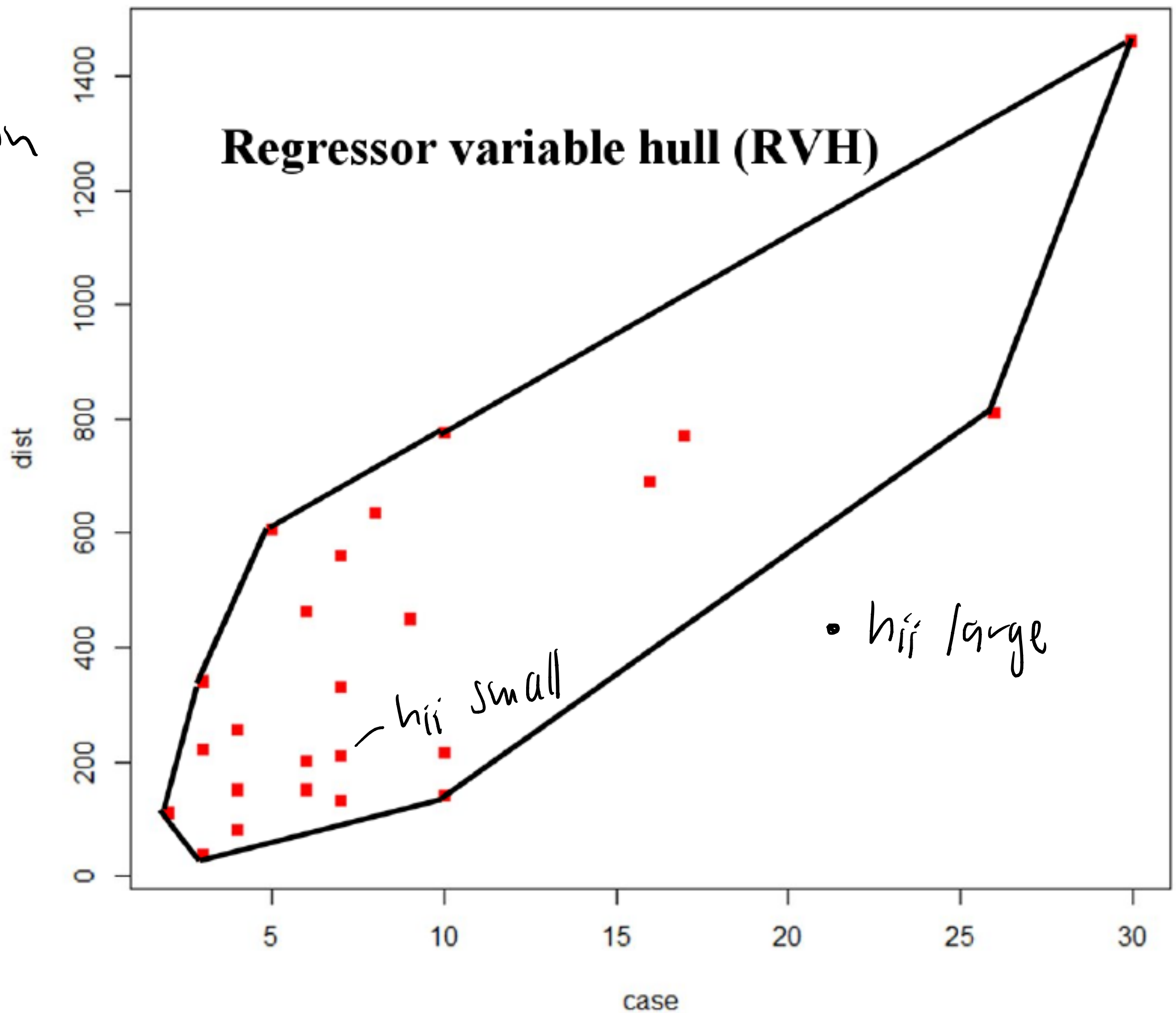
Note that $\boldsymbol{x_i'} = (1, x_{i1}, x_{i2}, ..., x_{ik})$.

2. We first define the smallest convex set containing all of the original n data point $(x_{i1}, x_{i2}, \ldots, x_{ik})$, $i = 1, 2, \ldots, n$, as the regressor variable hull (RVH).

point outside
is extrapolation

Point inside
is interpolation

Example - Delivery time data



look at diagonal elements?

measure of observation i distance from centroid of data

$H = \begin{pmatrix} h_{11} \cdots & & \\ & h_{ii} \cdots & \\ & & h_{nn} \end{pmatrix}$

every observation has a h value

3. If a point $(x_{01}, x_{02}, ..., x_{0k})$ lies inside or on the boundary of the RVH, then prediction involves interpolation. If this point lies outside the RVH, then the prediction is based on extrapolation.

4. The diagonal elements of the hat matrix $\boldsymbol{H}_{\boldsymbol{n}\times\boldsymbol{n}} = \boldsymbol{X}_{\boldsymbol{n}\times\boldsymbol{p}}(\boldsymbol{X}'\boldsymbol{X})^{-1}_{\boldsymbol{p}\times\boldsymbol{p}}\boldsymbol{X}'_{\boldsymbol{p}\times\boldsymbol{n}}$ are useful in detecting hidden extrapolation. The value of h_{ii} depends on the Euclidean distance of the point $\boldsymbol{x_i}$ from the centroid and on the density of the points in the RVH. In general, the point that has the largest value of h_{ii} (denoted by h_{max}) will lie on the boundary of the RVH in a region of the x space where the density of the observations is relatively low.

far out – h_{ii} large

close to centre – h_{ii} small

use diagonal to check if point is extrapolation or interpolation

5. The set of point $\boldsymbol{x}$'s (not necessarily data points used to fit the model) that satisfy

$$\boldsymbol{x}'(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{x} \leq h_{max}$$

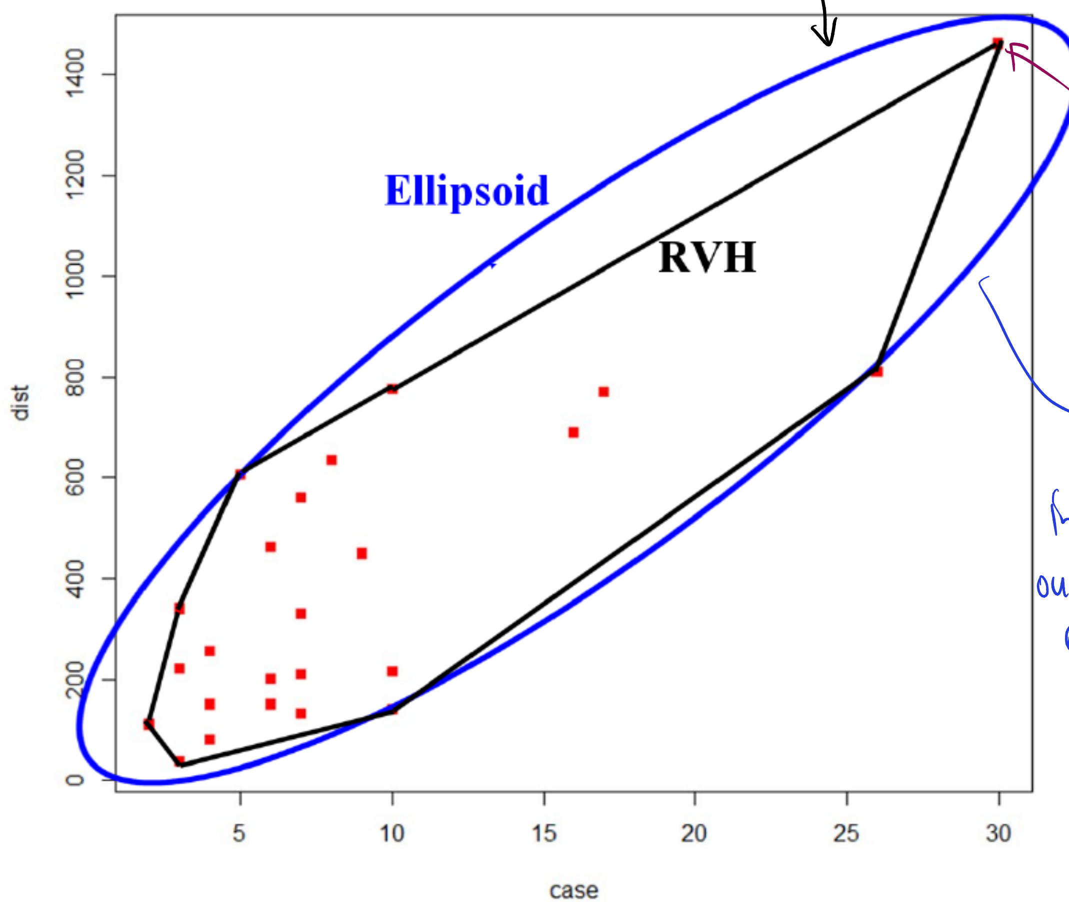
max diagonal element of hat matrix

$H = X'(X'X)^{-1}X$

$X' = (1, X_1, \ldots X_K)$

is an ellipsoid enclosing all points inside the RVH.

Example -Delivery time data



highest value hmax

within ellipsoid is interpolation while outside is extrapolation

6. Suppose we are interested in prediction at the point $\boldsymbol{x}_0' = (1, x_{01}, x_{02}, ..., x_{0k})$, a measure of the distance of this point to the centroid of the data is given by

distance of x_0 to centroid

$$h_{00} = \boldsymbol{x}_0'(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{x}_0$$

extrapolation

If $h_{00} > h_{max}$, the point $\boldsymbol{x}_0'$ is outside the ellipsoid enclosing the RVH.

If $h_{00} < h_{max}$, the point $\boldsymbol{x}_0'$ is inside the ellipsoid enclosing the RVH, and possibly inside the RVH.

(not sure)

Example - Delivery time data

Values of h_{ii} for the Delivery Time Data

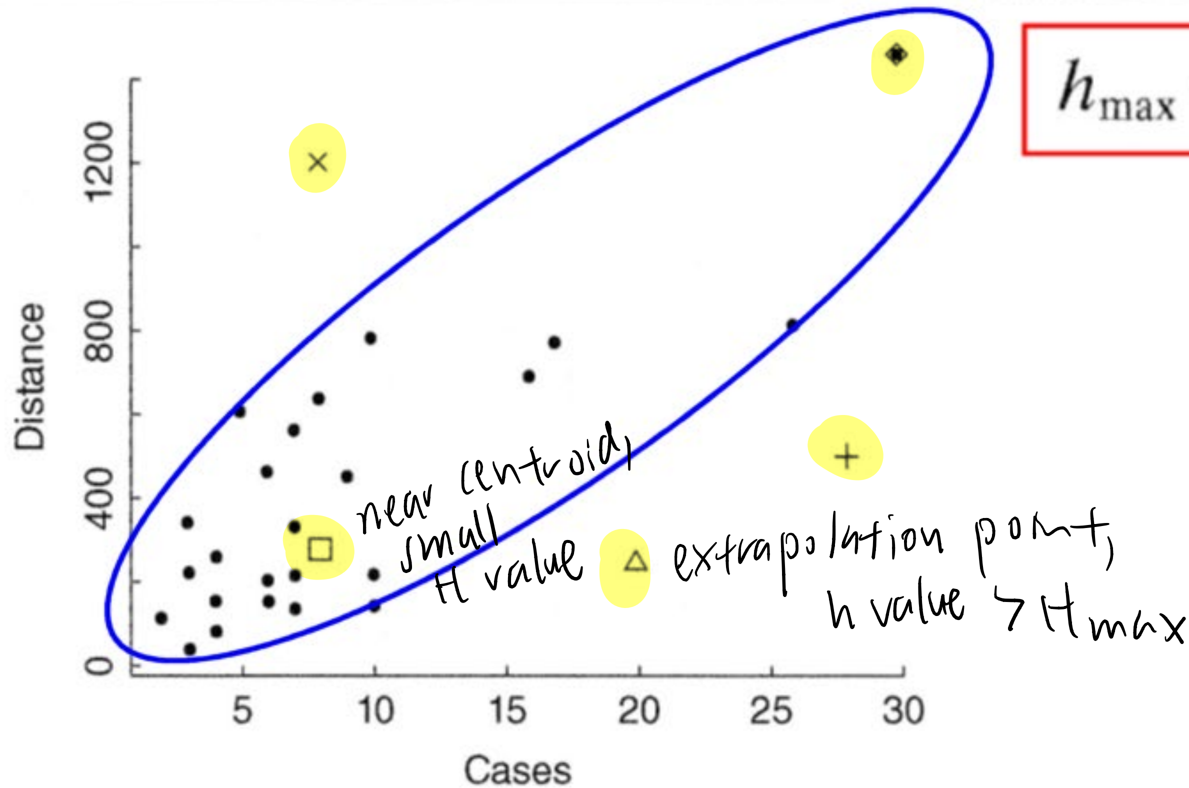
Observation, i	Cases, x_{i1}	Distance, x_{i2}	h_{ii}
1	7	560	0.10180
2	3	220	0.07070
3	3	340	0.09874
4	4	80	0.08538
5	6	150	0.07501
6	7	330	0.04287
7	2	110	0.08180
8	7	210	0.06373
9	30	1460	$0.49829 = h_{max}$
10	5	605	0.19630
11	16	688	0.08613
12	10	215	0.11366
13	4	255	0.06113
14	6	462	0.07824
15	9	448	0.04111
16	10	776	0.16594
17	6	200	0.05943
18	7	132	0.09626
19	3	36	0.09645
20	17	770	0.10169
21	10	140	0.16528
22	26	810	0.39158
23	9	450	0.04126
24	8	635	0.12061
25	4	150	0.06664

furthest from centroid

The Delivery Time Data

Point	Symbols in Figure 3.10	Cases	Distance	h_{00}
a	□	8	275	0.05346
b	△	20	250	0.58917
c	+	28	500	0.89874
d	×	8	1200	0.86736

$$\boxed{h_{max} = 0.49829}$$



Scatterplot of cases and distance for the delivery time data.

```r
# diagonal elements of hat matrix
rm(list = ls())
dat <- read.csv("D:\\nus_teaching\\st3131\\data\\Delivery_Time.csv",
                header = T, sep=",")
names(dat) <- c("Obs", "time", "case", "dist")
dat <- dat[,2:4]
time <- dat[,1]
case <- dat[,2]
dist <- dat[,3]
one <- rep(1,length(time))
X <- array(c(one, case,dist), dim=c(length(time),3))
XPX <-t(X) %*% X
XPXI <- solve(XPX)
```

get $(X'X)^{-1}$

```r
beta <- XPXI %*% t(X) %*% time
```

$(X'X)^{-1}X'Y$

```r
H <- X %*% XPXI %*% t(X)
```

$H = X(X'X)^{-1}X'$

```r
Hii <- diag(H)
Hii

x0 <- array(c(1,20,250), dim=3)
x0
t(x0) %*% XPXI %*% x0

x0 <- array(c(1,8,275), dim=3)
x0
t(x0) %*% XPXI %*% x0
```

```
> rm(list = ls())
> dat <- read.csv("D:\\nus_teaching\\st3131\\data\\Delivery_Time.csv",
+                 header = T, sep=",")
> names(dat) <- c("Obs", "time", "case", "dist")
> dat <- dat[,2:4]
> time <- dat[,1]
> case <- dat[,2]
> dist <- dat[,3]
> one <- rep(1,length(time))
> X <- array(c(one, case,dist), dim=c(length(time),3))
> XPX <-t(X) %*% X
> XPXI <- solve(XPX)
> beta <- XPXI %*% t(X) %*% time
> H <- X %*% XPXI %*% t(X)
> Hii <- diag(H)
> Hii
```

diagonal

$0 \le h_{ii} \le 1$

h_{max}

```
 [1] 0.10180178 0.07070164 0.09873476 0.08537479 0.07501050 0.04286693
 [7] 0.08179867 0.06372559 0.49829216 0.19629595 0.08613260 0.11365570
[13] 0.06112463 0.07824332 0.04111077 0.16594043 0.05943202 0.09626046
[19] 0.09644857 0.10168486 0.16527689 0.39157522 0.04126005 0.12060826
[25] 0.06664345
>
```

h_{00}

```
> x0 <- array(c(1,20,250), dim=3)
> x0
[1]   1  20 250
> t(x0) %*% XPXI %*% x0
          [,1]
[1,] 0.5891742
>
> x0 <- array(c(1,8,275), dim=3)
> x0
[1]   1   8 275
> t(x0) %*% XPXI %*% x0
```

greater than h_{max} ⇒ extrapolation point

Standardized regression coefficients

1. It is usually not meaningful to compare the magnitudes of regression coefficients because the regressor variables are measured in different units of measurement. If we standardized the regressor variables, the regression coefficients can be compared meaningfully in terms of magnitude.

2. There are two ways to standardized the variables: (a) unit normal scaling, and (b) unit length scaling.

3. Suppose the multiple linear regression model is

   $y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_k x_{ik} + \epsilon_i,\ i = 1, 2, ..., n.$

4. Let $y, x_1, x_2, ..., x_k$ be scaled to unit normal, that is

(optional)

$$y_i^* = \frac{y_i - \bar{y}}{\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(y_i - \bar{y})^2}}, \quad z_{i1} = \frac{x_{i1} - \bar{x}_1}{\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_{i1} - \bar{x}_1)^2}},$$

to scale response but not needed, just scale regression variables for comparison

$$..., z_{ik} = \frac{x_{ik} - \bar{x}_k}{\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_{ik} - \bar{x}_k)^2}}.$$

Can now compare the regression coefficients

The model becomes

$$y_i^* = b_1 z_{i1} + b_2 z_{i2} + ... + b_k z_{ik} + \epsilon_i, \; i = 1, 2, ..., n.$$

$\hookrightarrow b_0 = 0$

i is the observation number

Note that the least-squares estimate of b_0 is zero if all the variables are unit normal scaled.

The least-squares regression coefficients $\hat{\boldsymbol{b}} = (\boldsymbol{Z}'\boldsymbol{Z})^{-1}\boldsymbol{Z}'\boldsymbol{y}^*$.

5. Let $y, x_1, x_2, ..., x_k$ be scaled to unit length, that is

$$y_i^* = \frac{y_i - \bar{y}}{\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}, \quad w_{i1} = \frac{x_{i1} - \bar{x}_1}{\sqrt{\sum_{i=1}^{n}(x_{i1} - \bar{x}_1)^2}}, ..., w_{ik} = \frac{x_{ik} - \bar{x}_k}{\sqrt{\sum_{i=1}^{n}(x_{ik} - \bar{x}_k)^2}}.$$

The model becomes

$$y_i^{\#} = b_1 w_{i1} + b_2 w_{i2} + ... + b_k w_{ik} + \epsilon_i, \; i = 1, 2, ..., n.$$

Note that the least-squares estimate of b_0 is zero if all the variables are unit length scaled.

X'X matrix

The least-squares regression coefficients $\hat{\boldsymbol{b}} = (\boldsymbol{W}'\boldsymbol{W})^{-1}\boldsymbol{W}'\boldsymbol{y}^{\#}$.

For unit length scaled variables, the off-diagonal elements of $\boldsymbol{W}'\boldsymbol{W}$ are the correlation coefficients of the regressor variables. Also, r_{ij} is the correlation coefficient of x_i and x_j.

correlation coeff between x_1 and x_2

$$\boldsymbol{W'W} = \begin{bmatrix} 1 & r_{12} & r_{13} & \dots & r_{1k} \\ r_{21} & 1 & r_{23} & \dots & r_{2k} \\ r_{31} & x_{32} & 1 & \dots & r_{3k} \\ \vdots & \vdots & & & \vdots \\ r_{k1} & r_{k2} & r_{k3} & \dots & 1 \end{bmatrix}$$

Example - Delivery time data

```
# standardized regression coefficients
library(psychometric) # load package
dat <- read.csv("D:\\nus_teaching\\st3131\\data\\Delivery_Time.csv",
                header = T, sep=",")
names(dat) <-c("Obs", "time", "case", "dist")
dat <- dat[,2:4]
corr <- cor(dat)
corr
WTW <- corr[c(2:3),c(2,3)]
WTW
WTy0 <- corr[c(2:3),1]
WTy0
dim(WTW)
length(WTy0)
solve(WTW,WTy0)

#standardized the variables directly and then use lm
stdmod <- lm(scale(time) ~ scale(case) + scale(dist))
stdmod
```

1 2 3 4

W'W : correlation matrix

W'y

y(time)

```
> dat <- read.csv("D:\\nus_teaching\\st3131\\data\\Delivery_Time.csv",
+                 header = T, sep=",")
> names(dat) <-c("Obs", "time", "case", "dist")
> dat <- dat[,2:4]
> corr <- cor(dat)
> corr
          time      case      dist
time 1.0000000 0.9646146 0.8916701
case 0.9646146 1.0000000 0.8242150
dist 0.8916701 0.8242150 1.0000000
```

→ $W'W$

```
> WTW <- corr[c(2:3),c(2,3)]
> WTW
         case     dist
case 1.000000 0.824215
dist 0.824215 1.000000
> WTy0 <- corr[c(2:3),1]
```

column 2-3

column 1 is the correlation matrix?

```
> WTy0
     case      dist
0.9646146 0.8916701
> dim(WTW)
[1] 2 2
> length(WTy0)
[1] 2
> solve(WTW,WTy0)
     case      dist
0.7162722 0.3013078
```

solve for regression coeff

$(W'W)^{-1} W'y$

```
>
> #standardized the variables directly and then use lm
> stdmod <- lm(scale(time) ~ scale(case) + scale(dist))
> stdmod

Call:
lm(formula = scale(time) ~ scale(case) + scale(dist))

Coefficients:
(Intercept)  scale(case)  scale(dist)
 -1.118e-16    7.163e-01    3.013e-01
```

β_1 β_2

$\beta_0 = 0$

fit model directly by scaling?

The End