Applied Regression Analysis using R

Code

L01-Basic

use '_' in name, to avoid confusion with the member function

Hide

```
# help
rm(list = ls()) # clean environment
?is.logical()
library(help = "stats") # for function list in a package
is.integer(7L) # integer, with 'L' after number
```

```
[1] TRUE
```

Hide

```
is.integer(7)
```

```
[1] FALSE
```

Hide

```
x <- 1 # assignment
1 -> x1
x = 1

# vector c()
month <- c('Jan','Feb','Dec')
class(month)
```

```
[1] "character"
```

Hide

```
num <- 1:50 # return a vector
num_seq <- seq(1, 50, by = 2) # create sequence: seq(from, to, by)
seq(stats::rnorm(20))
```

```
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
```

Hide

```
num_seq[which(num_seq>9)]
```

```
 [1] 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49
```

Hide

```
# categorical data
month.fac <- factor(month, order=TRUE, levels = c("Jan", "Feb", "Dec"))
month.fac.in <- factor(month) # default: mathematical order

# matrix, special vectors in R
A = matrix(month, nrow = 6, byrow = TRUE, ncol = 9)
B = matrix(month, nrow = 6, byrow = FALSE, ncol = 9)

# list, combine different data type
listAB = list(num = num_seq, Amatrix = A, Bmatrix = B)
class(listAB)
```

```
[1] "list"
```

Hide

```
# data.frame
df.A <- data.frame(A)
df.A$X1
```

```
[1] "Jan" "Jan" "Jan" "Jan" "Jan" "Jan"
```

Hide

```
class(df.A)
```

```
[1] "data.frame"
```

Hide

```
class(df.A$X3)
```

```
[1] "character"
```

Hide

```
df.A$X1 <- factor(df.A$X1)

# function
myfuc = function(x) { # key word function
    s = 0
    for (eps in x) {
      if (eps <= 0){
        next
      }
      s = s + eps
      if (s > 20) {
        break
      }
    }
    s
}

# recycling rule
c(1, 2, 3, 4) + c(1, 2) # equal to c(1, 2, 3, 4) + c(1, 2, 1, 2)
```

```
[1] 2 4 4 6
```

e ## implicit coercion to mixed types logical -> integer -> numeric -> complex -> character

as transform from logical to integer is easier

`c(11, month)` will convert integer to a character

L02-slr

simple linear regression basic

- regression analysis: statistical model that involve one dependent variable and one or more independent variables
- regression: find a rule of picking distribution for *Y* from a space of infinitely many distribution that agrees with the data

Primary Assumption: (for now) a sequence of random variables is independent and identically distributed (i.i.d.)

Estimation:

The process of using data to suggest a value for a parameter

Estimate: the value suggested is called the estimate of the parameter (depend on data)

Estimator: a function of the data, that gives estimates of the parameter $\hat{\theta} = h(X)$, the distribution for $\hat{\theta}$ is called **sampling distribution** of the estimator.

Hide

```
?pbinom
```

Given X~Normal(n, p)

Pr(X = x) = dbinom(x, size = n, prob = p)

Pr(X <= x) = pbinom(x, size = n, prob = p)

qbinom: The quantile is defined as the smallest value x such that F(x) ≥ p, where F is the distribution function. with p∈ [0,1], returns the smallest value of q s.t. Pr(X<=returnvalue) >= p

dbinom(x, size, prob, log = FALSE) pbinom(q, size, prob, lower.tail = TRUE, log.p = FALSE) qbinom(p, size, prob, lower.tail = TRUE, log.p = FALSE) rbinom(n, size, prob)

Binomial Example

L02-16/55 pages, US presidents

- if we become US presidents, we are more likely to have sons – T/F depend on the p-value
- US presidents are more likely to have sons – independent of the p-value, because the current data has exhausted all the info of presidents

Hide

```
rm(list = ls())  # clean working environment

## set up hypothesis
theta <- 1/2  ## assume equal chance 50-50
n <- 158  ## total number of trials
x <- 0:n  ## all possible number
x1 <- seq(0, n, by = 1)  # alternative way of set up sequence
fX <- dbinom(x, size = n, prob = theta)  ## pdf of x
# ? dbinom  ## getting help with ?
obs <- 67
p.lowerTail <- pbinom(obs, size = n, prob = theta)
## under the assumption of theta = 0.5
p.value <- 2 * p.lowerTail

## visualization
tmpL <- fX[x <= 67]  ## as extreme as what we observed
x.upperTail <- 1 + qbinom(p.lowerTail,
                          size = n, prob = theta, lower.tail = FALSE) ## quantile functi
on, different from pbinom
tmpU <- fX[x >= x.upperTail]

M <- x[x > 67 & x < x.upperTail]
tmpM <- fX[x > 67 & x < x.upperTail]

plot(x, fX, type = "n",
     xlab = "observed number of daughter", ylab = "probability mass")
lines(0:obs, tmpL,type = "h", col = "red")
lines(x.upperTail:n, tmpU,type = "h", col = "red")
lines(M, tmpM, tmpM,type = "h")
points(0:obs, tmpL, col = "red")
points(x.upperTail:n, tmpU, col = "red")
points(M, tmpM)
legend("topright", legend = "P-value", col = "red", lty = 1, pch = 1)
```

Hide

```
## function to calculate the density of a binomial distribution
## modified with two more attributes: obs and n
myp_func <- function(n, obs, p){
  ## n: total number of binomial trials
  ## obs: observed number of events
  ## p: probability of success
  pbinom(obs, size = n, prob = p) - pbinom(obs - 1, size = n, prob = p)
  ## exactly the same as dbinom(obs, size = n, prob = p)
}

myp_func(190, 125, 0.5)
```

```
[1] 3.972689e-06
```

Hide

```
dbinom(125, size = 190, prob = 0.5)
```

```
[1] 3.972689e-06
```

Hide

```
myp_func(190, 125, 0.6)/myp_func(190, 125, 0.5)
```

```
[1] 3967.39
```

Hide

```
dbinom(125, size = 190, prob = 0.6)/dbinom(125, size = 190, prob = 0.5)
```

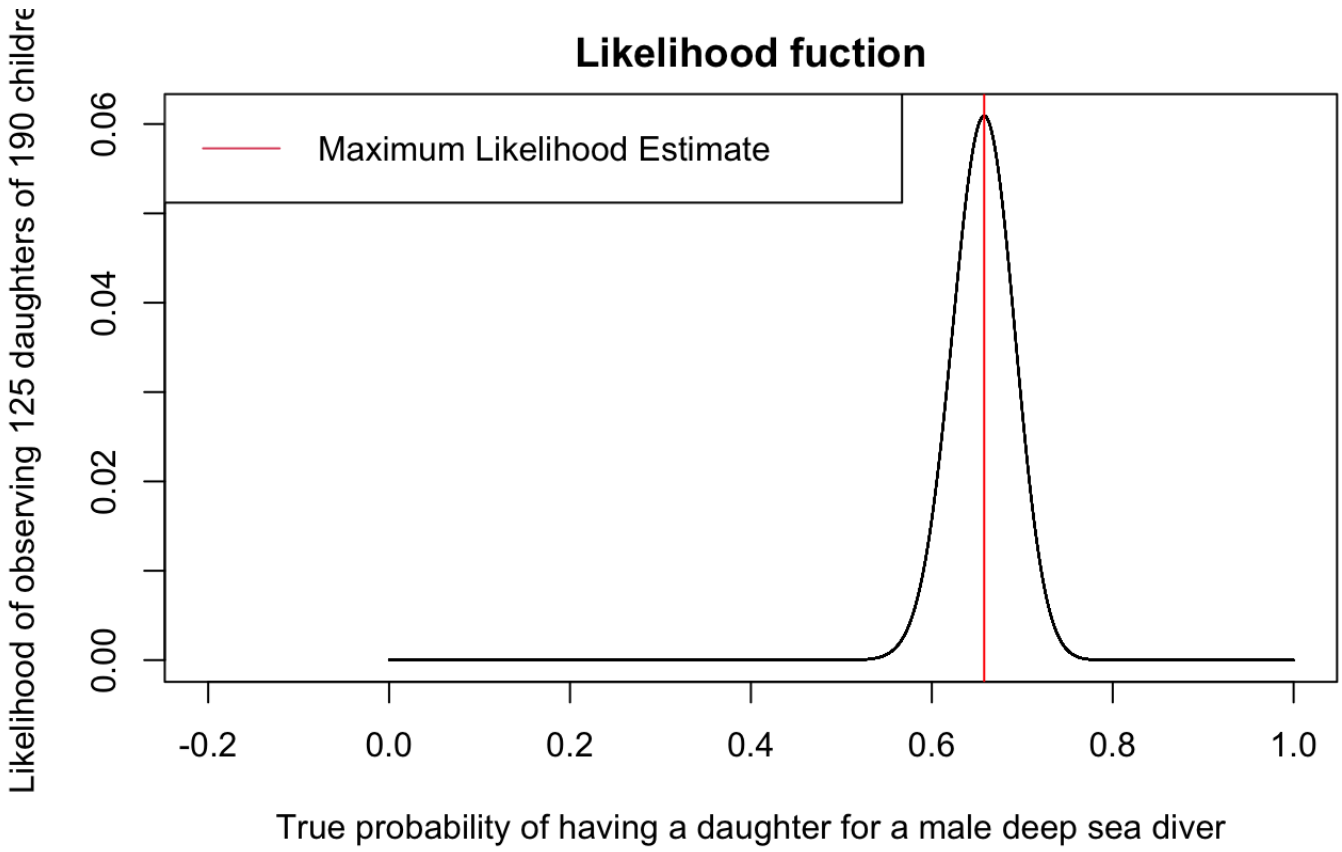
```
[1] 3967.39
```

Hide

```
## to find the maximum likelihood estimate based on the obs
## we need to find the p that maximise the underlying pdf/pmf
## myp_func is here to show how to write a simple function
## we will use the proper dbinom function here after


## to visually examine the mle of deep sea diver case study
## observations: 125 daughter of 190 children
pvec <- seq (0, 1, length.out = 10000000)  ## set the seq of all possible prob
lvec <- dbinom(125, size = 190, prob = pvec)  ## find the likelihood
plot (pvec, lvec, type = "l",  ## visualisation of the likelihood
      xlab = "True probability of having a daughter for a male deep sea diver",
      ylab = "Likelihood of observing 125 daughters of 190 children",
      main = "Likelihood fuction",
      xlim = c(-0.2, 1))

# common sense: best guess: the estimator of p is 125/190
# get from the maximum likelihood function
phat <- pvec[which.max(lvec)]
abline(v = phat, col = "red")
```

Hide

```
legend("topleft", legend = "Maximum Likelihood Estimate",
       lty = 1, col = 2)
```



Confidence Interval

Confidence interval of binomal distribution - When says

We are 95% confident that the true probability of having a daughter is between (0.586, 0.725) for a male deep-sea diver

it is actually means

The method used to obtain the interval (0.586, 0.725) will contain the true probability 95% of time if it is to be repeated

Hide

```
binom.test(125, n = 190, p = 0.5)
```

```
    Exact binomial test

data:  125 and 190
number of successes = 125, number of
trials = 190, p-value = 1.603e-05
alternative hypothesis: true probability of success is not equal to 0.5
95 percent confidence interval:
 0.5857408 0.7250337
sample estimates:
probability of success
             0.6578947
```

Hide

```
binom::binom.confint(125, n = 190)
```

	method	x	n	mean	lower	upper
	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	agresti-coull	125	190	0.6578947	0.5878351	0.7216963
2	asymptotic	125	190	0.6578947	0.5904374	0.7253521
3	bayes	125	190	0.6570681	0.5895875	0.7236296
4	cloglog	125	190	0.6578947	0.5857332	0.7205325
5	exact	125	190	0.6578947	0.5857408	0.7250337
6	logit	125	190	0.6578947	0.5876377	0.7218476
7	probit	125	190	0.6578947	0.5882529	0.7225372
8	profile	125	190	0.6578947	0.5886447	0.7229128
9	lrt	125	190	0.6578947	0.5886470	0.7229437
10	prop.test	125	190	0.6578947	0.5852078	0.7240821

1-10 of 11 rows Previous **1** 2 Next

Hide

```
rm(list = ls())
num = 1e3
n = 190
runif(1)
```

```
[1] 0.552725
```

Law of large numbers

Suppose that $X_1, X_2, .. X_n$ all have the same expected value $\mathbb{E}[X_i] = \mu$, the same variance $Var[X_i] = \sigma^2$ and zero covariance with each other $Cov[X_i, X_j] = 0$, when $i \neq j$

or could say if X_i are i.i.d., then the following holds

$$\bar{X_n} = \frac{1}{n}\sum_{i=1}^{n} X_i \to \mu, when \quad n \to \infty$$

reduce the spread of the distribution to 0 effectively

$$\lim_{n\to\infty} Pr(|\bar{X_n} - \mu| > \varepsilon) = 0, \text{for any } \varepsilon > 0$$

Illustration of LLN

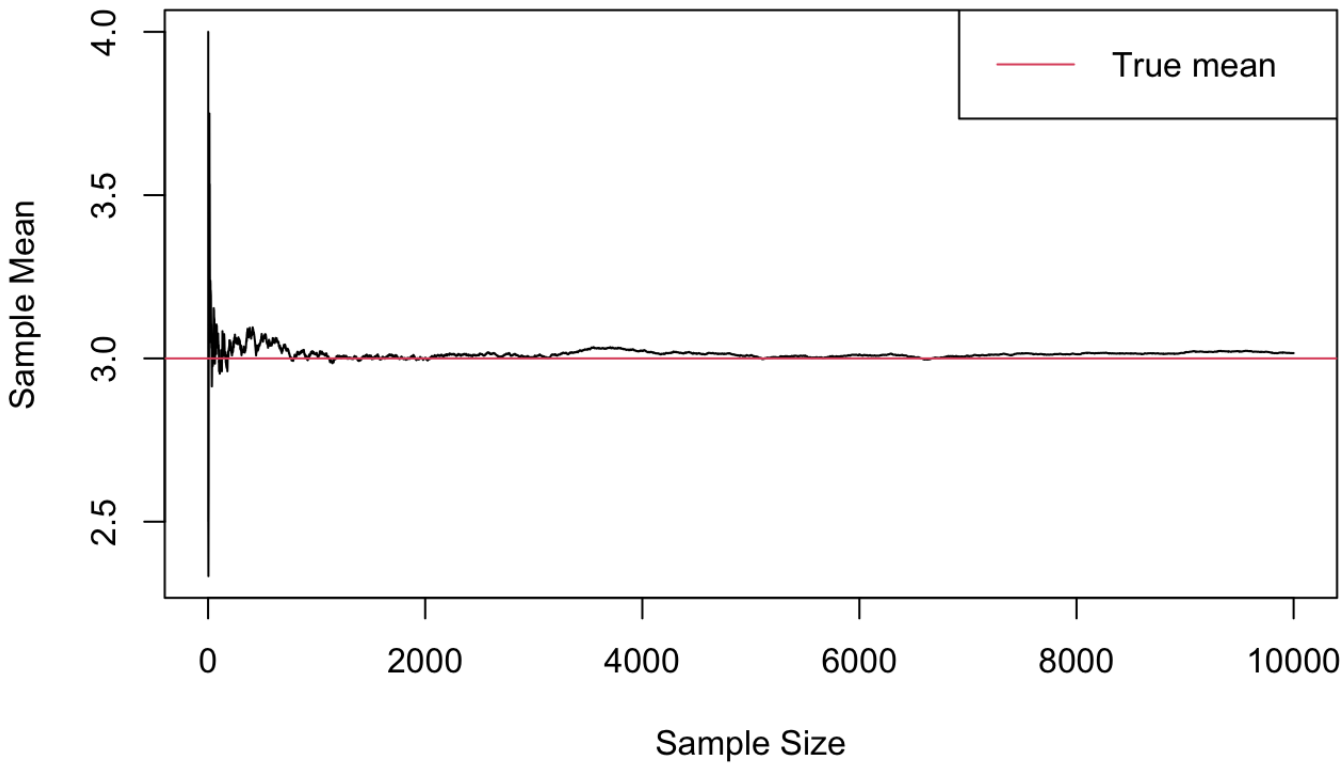
As n increases, the mean of X approaches the true mean and the error approaches 0.

Hide

```
rm(list = ls())
n <- 1e4
lambda <- 3

xpois <- rpois(n, lambda)  ## consider poisson random variable
xexp <- lambda  ## true underlying mean

nseq <- 1:n
# investigate Xbar when n increases, `cumsum` is  cumulative sum
xcbar <- cumsum(xpois)/nseq  ## sample mean as n increases, at various stage
error <- abs(xcbar - xexp)  ## relative error
plot(nseq, xcbar, type = "l", xlab = "Sample Size",
     ylab = "Sample Mean")
abline(h = xexp, col = 2)
```

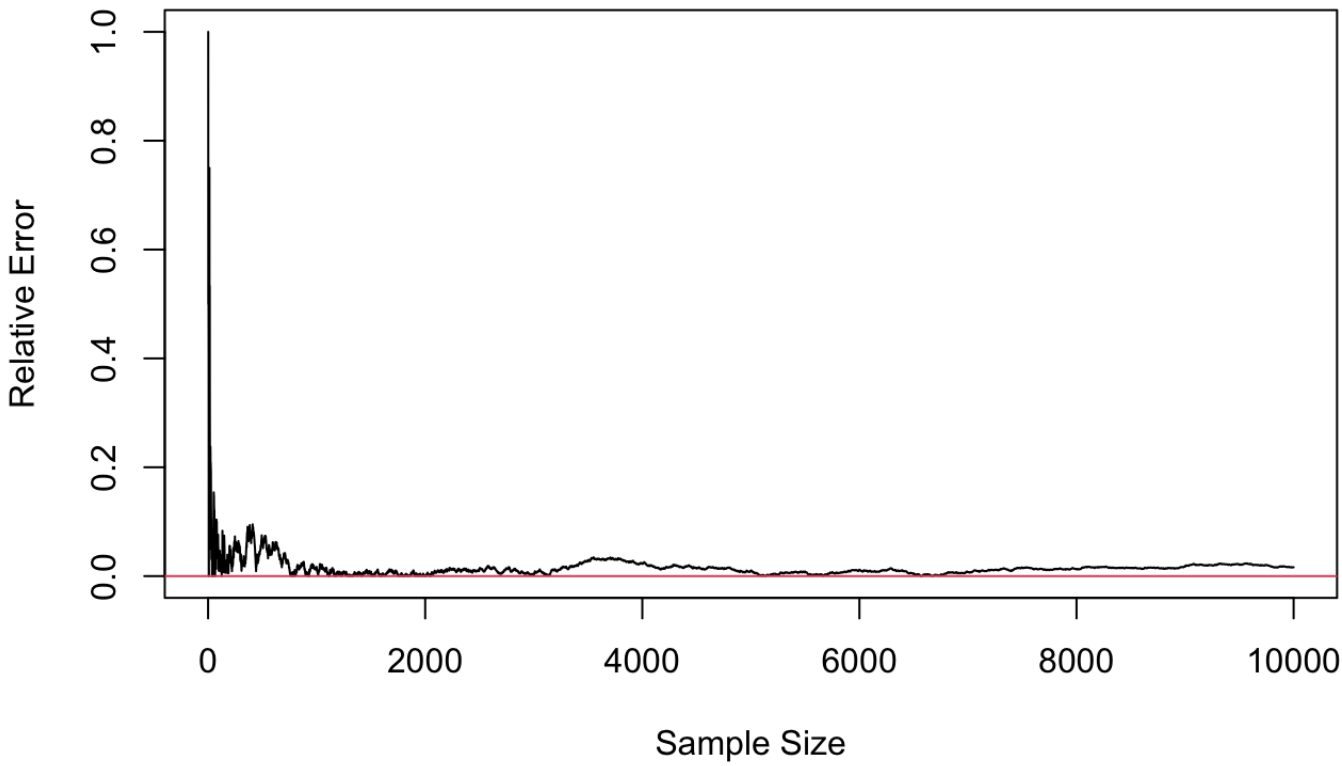
Hide

```
legend("topright", legend = "True mean", lty = 1, col = 2)
```



Hide

```
plot(nseq, error, type = "l", xlab = "Sample Size", ylab = "Relative Error")
abline(h = 0, col = 2)
```



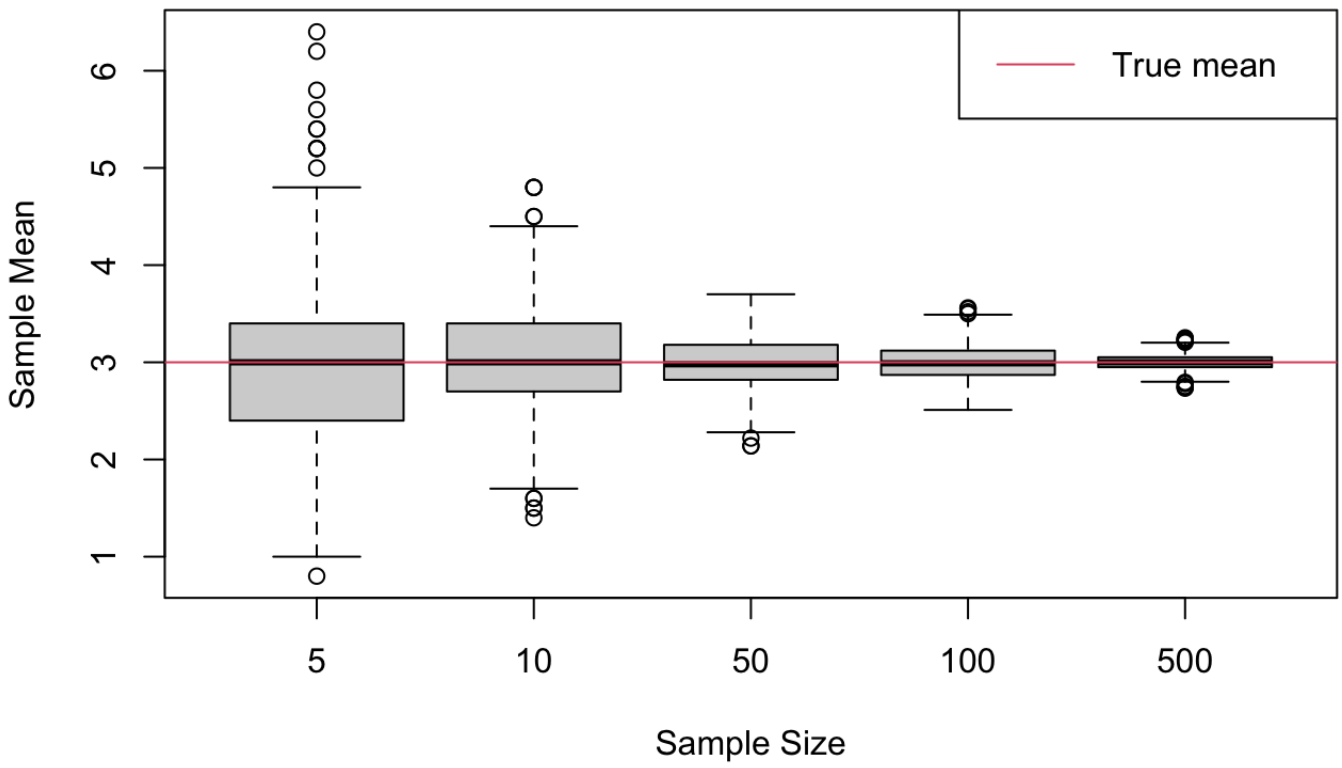
A better look:

Hide

```
## a better look at LLN
sample.size.vec <- c(5, 10, 50, 100, 500)  ## 5 cases
ncases <- length(sample.size.vec)
num <- 1e3  ## number of repetitions
xbar.vec <- double()  ## x bar for all simulations
error.vec <- double()  ## error for all simulations
n.vec <- integer()  ## sample size for each case

for(j in 1:ncases){
  n <- sample.size.vec[j]  ## current sample size
  s.vec <- double(num)  ## all x bar for this n
  e.vec <- double(num)  ## all error for this n

  for (i in 1:num){  ## repeat num number of times
    x <- rpois(n, lambda)  ## sample of x by sample size 5, 10 etc.
    s.vec[i] <- sum(x)/n
    e.vec[i] <- abs(s.vec[i] - xexp)/abs(xexp)
  }

  xbar.vec <- c(xbar.vec, s.vec)
  error.vec <- c(error.vec, e.vec)
  n.vec <- c(n.vec, rep(sample.size.vec[j], num))
  }

x.df <- data.frame(xbar = xbar.vec, error = error.vec, n = n.vec)
boxplot(xbar~n, data = x.df, xlab = "Sample Size", ylab = "Sample Mean")
abline(h = xexp, col = 2)
```

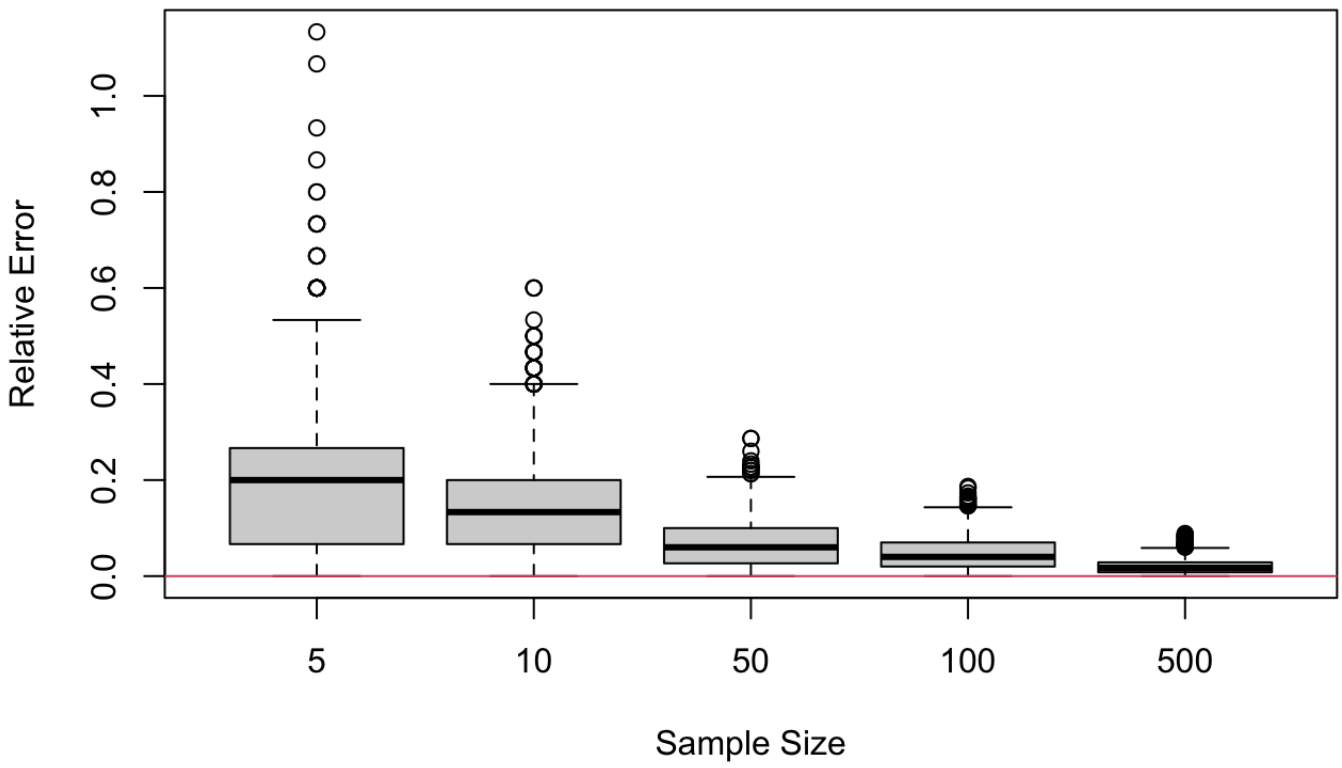
Hide

```
legend("topright", legend = "True mean",
       lty = 1, col = 2)
```



Hide

```
boxplot(error~n, data = x.df,
        xlab = "Sample Size", ylab = "Relative Error")
abline(h = 0, col = 2)
```



Hide

```
## as we increase the sample size
## the error gets closer to 0 and more constant
```

Central Limit Theorm

Central Limit Theorem $\sim N(\mu, \frac{\sigma^2}{n})$

Suppose $X_1, X_2, \ldots, X_n$ are i.i.d. with mean μ and variance σ^2, then

$$\sqrt{n}\frac{\bar{X}_n - \mu}{\sigma}$$

becomes more and more likely a standard normal random variable as $n \to \infty$

$$Z \sim \text{Normal}(0, 1)$$

in the sense that for every real number z

$$\lim_{n\to\infty} \Pr\left(\sqrt{n}\frac{\bar{X}_n - \mu}{\sigma} \leq z\right) = \Pr\left(Z \leq z\right)$$

Illustration

Hide

```
############################################################################## centra
l limit therom
sample.size.vec
```

```
[1]   5  10  50 100 500
```

Hide

```
lambda; xexp
```

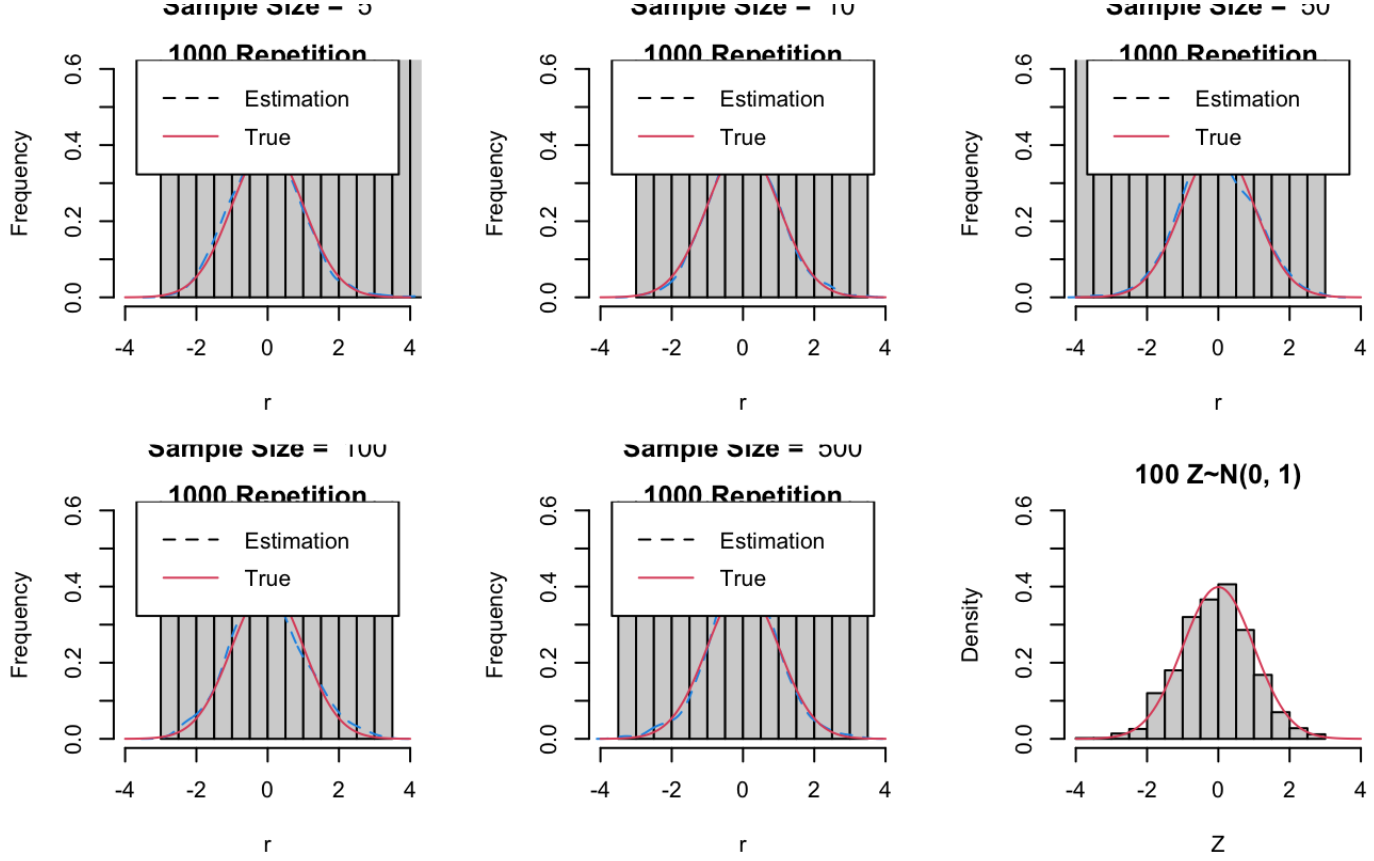
```
[1] 3
[1] 3
```

Hide

```
xvar <- lambda  ## true variance for Poisson
sqrt.n.vec <- sqrt(n.vec)  ## last for loop

sample_error <- xbar.vec - xexp
sample_variance <- sqrt(xvar)
r.vec <- sqrt.n.vec*sample_error/sample_variance  ## normalised error

x.hist <- seq(-4, 4, length.out = 100)
par(mfrow = c(2, 3))

for(eps in sample.size.vec){
  ss <- r.vec[n.vec == eps]  ## subset by n
  tname <- bquote(bold(atop("Sample Size = "~.(eps), "1000 Repetition")))

  hist(ss, feq = FALSE, xlab = "r", main = tname,
       ylim = c(0, 0.6), xlim = c(-4, 4))

  lines(density(ss), col = 4, lty = 2)  ## kernel denity estimation
  lines(x.hist, dnorm(x.hist), col = 2, lty = 1)  ## true density function
  legend("top", col = c(1, 2), lty = c(2, 1),
         legend = c("Estimation", "True"))
}

true.norm <- rnorm(num, mean = 0, sd = 1)
hist(true.norm, freq = FALSE, ylim = c(0, 0.6),
     xlim = c(-4, 4), xlab = "Z", main = "100 Z~N(0, 1)")
```

Hide

```
lines(x.hist, dnorm(x.hist), col = 2, lty = 1)
par(mfrow = c(1, 1))
```



Hide

```
sample_quantile_5 = sort(r.vec[n.vec == 5])
sample_quantile_10 = sort(r.vec[n.vec == 10])
sample_quantile_50 = sort(r.vec[n.vec == 50])
sample_quantile_100 = sort(r.vec[n.vec == 100])
sample_quantile_500 = sort(r.vec[n.vec == 500])
sample_cdf = (1:num)/num
plot(sample_quantile_5, sample_cdf, type = "s",
     xlab = "", ylab = "",
     main = "Cumulative distribution function n = 500")
lines(sample_quantile_10, sample_cdf, col = 3, lty = 4)
```

Hide

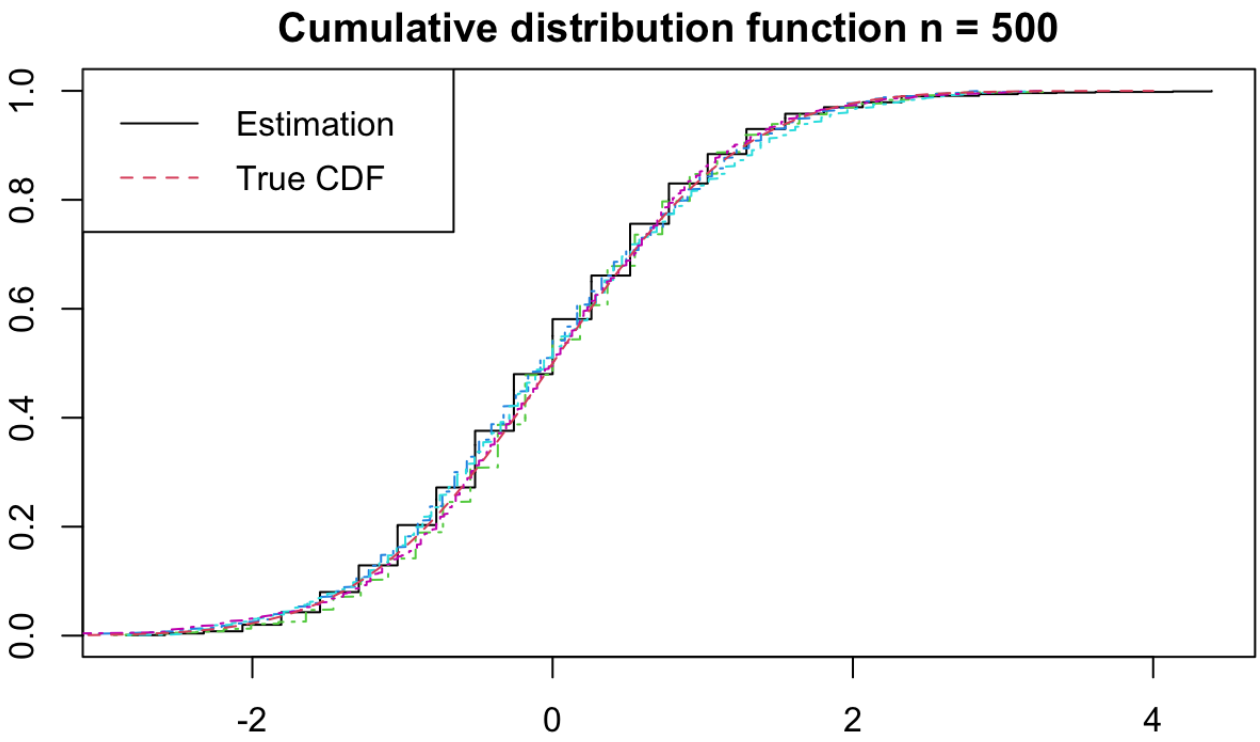
```
lines(sample_quantile_50, sample_cdf, col = 4, lty = 4)
lines(sample_quantile_100, sample_cdf, col = 5, lty = 4)
```

Hide

```
lines(sample_quantile_500, sample_cdf, col = 6, lty = 4)
lines(x.hist, pnorm(x.hist), col = 2, lty = 2)
```

Hide

```
legend("topleft", lty = c(1, 2), col = c(1, 2),
       legend = c("Estimation", "True CDF"))
```



Hide

```
NA
NA
```

L03 - LSE & MLE

$MSE(m) = \mathbb{E}[(Y-m)^2] = Var[Y] + (\mathbb{E}[Y-m])^2$

-> bias variance decomposition

The common model assumptions for slr

bias of an estimator $\hat{\theta}$, θ is a fixed population parameter $Bias(\hat{\theta}, \theta) = \mathbb{E}[\hat{\theta} - \theta]$

- residual

the estimate $\hat{e_i}$ is known as **residual** .

observed y_i and x_i, $\hat{e_i} = y_i - (b_0 + b_1 x_i)$, where b_0 and b_1 are estimates of β_0 and β_1; $\hat{y_i} = b_0 + b_1 x_i$ is known as the predicted/fitted value.

Then $y_i = \hat{y_i} + \hat{e_i}$

- consistent

An estimator is **consistent** if

$$\hat{\theta_n} \to \theta \quad \text{when} \quad n \to \infty$$

Least Square Estimation

SS_E : error sum of squares

$SS_E := e_1^2 + e_2^2 + \ldots + e_n^2 = \sum_{i=1}^{n} (y_i - (b_0 + b_1 x_i))^2$

model assumption for slr

1. The conditional mean is linear for all i, i.e.

$$\mathbb{E}\left[Y_i \mid X_i = x_i\right] = \beta_0 + \beta_1 x_i$$

2. The error term

$$\varepsilon_i = Y_i - \beta_0 - \beta_1 X_i$$

has a zero mean and a constant but unknown variance for all i, i.e.

$$\mathbb{E}\left[\varepsilon_i \mid X_i\right] = 0 \qquad \text{and} \qquad \operatorname{Var}\left[\varepsilon_i \mid X_i\right] = \sigma^2$$

3. The error terms are uncorrelated with X_i for all i, i.e.

$$\operatorname{Cov}\left[\varepsilon_i, X_i\right] = 0$$

and are uncorrelated to each other for all $i \neq j$, i.e.

$$\operatorname{Cov}\left[\varepsilon_i, \varepsilon_j\right] = 0$$

assumptions about the error term

Overview of statistics

In true universe, we have random variable Y;

In real world, we observe value y;

Then to link them together, we use **estimator**. However, error exist in this process, two types

- sampling error (improve when data collection)
- estimation error (improve using statistics)

Two main things - parameter estimation - hypothesis testing

To test the model sufficiency

- F-Test for significance of regression: whether all predictors are zero
- T-test: whether a single predictor is necessary for a given model
- Partial F-test: can be applied to an arbitrary subset of predictors

L04 - diagnostics

residual plot

$\hat{e_i}$ $vs.$ $\hat{y_i}$ fitted value of y

the variance of ε in true world is expected to be a constant, the variance of the residuals could be shown it is also a constant.

To check the assumptions of the residual

- 1. In residual plot, it should be random scatter, no underlying non-linear relationship
- 2. In residual plot, randomly scatter around mean zero, with basically unchanged variance
- 3. check they are independent of each other by ACF
- 4. check normality by QQplot (Quantile-Quartile normal plot)

autocorrelation function

If errors are independent, the estimates should be small and patternless

time series variable - very likely to be underlying correlation among residuals

do the residual plot after fit the tesla price model

observe some high price - after days are likely to be large

based on ACF plot: determine the underlying lag correlation

bad impact - negative on the following days

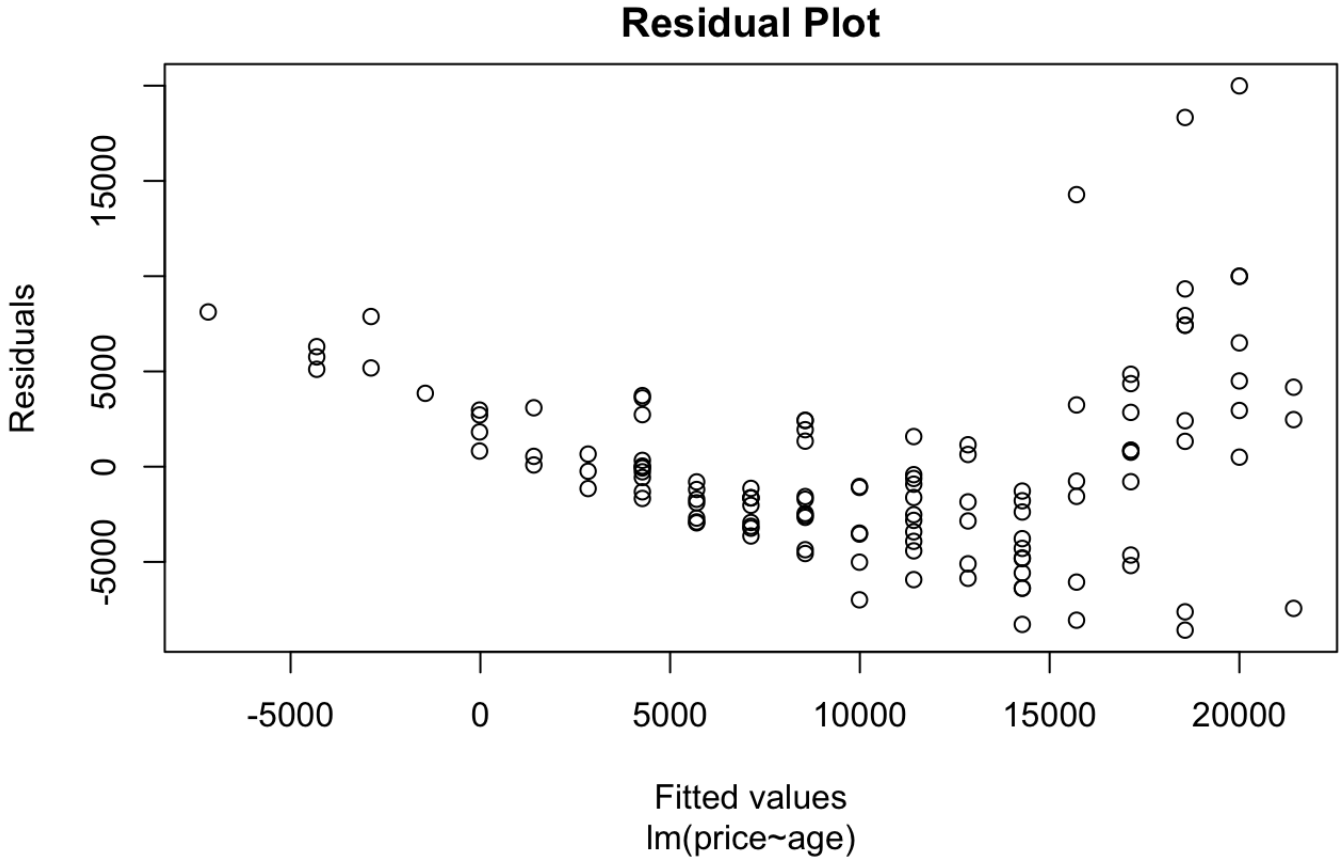
tesla price today ~ tomorrow? ~ next week? :**autocorrelation function**

hypothesis -> colloect data -> describe the data use (sample mean/…/plot) -> fit the model -> use diagnostics to check the assumption (through residual ~ plot residual vs. fitted value / residuali vs. residualj 或者 autocorralation check coevariance / residuals vs. X) — if the assumption is violated -> do the transformation (constant variance ~ log trandformation of y / normality -> box-cox to find additional variable, 可以同时对于y与x transform)

Check normality

plotting the estimated Vs assumed distribution

The error should follow normal distribution N(0, σ^2)



L05 - transformation

$Y_i|X_i$ ~ Normal $(\beta_0 + \beta_1 x_i, \sigma^2)$

Box-Cox transformation

- When we detect a possible violation of those three assumptions,

$$Y_i \mid X_i \sim \text{Normal}\left(\beta_0 + \beta_1 x_i, \sigma^2\right)$$

it means we have evidence that $Y \mid X$ does not follow a normal distribution.

- The idea is to assume the response can be transformed by

$$Y_i^*(\lambda) = \begin{cases} \dfrac{Y_i^\lambda - 1}{\lambda}, & \text{if} \quad \lambda \neq 0; \\ \ln Y_i, & \text{if} \quad \lambda = 0. \end{cases}$$

$\lim_{\lambda \to 0} \frac{y_i^\lambda - 1}{\lambda} = \lim_{\lambda \to 0} \frac{(\ln y_i) y_i^\lambda - 0}{1} = \ln y_i$

so that the new response is conditionally normal, that is

$$Y_i^*(\lambda) \mid X_i \sim \text{Normal}\left(\beta_0 + \beta_1 x_i, \sigma^2\right)$$

Estimation

Data ← $f_X(x, \tau)$

Q: Can you see the transformation when $\lambda = 0$ is the limiting case of $\lambda \neq 0$?

Q: Do you notice it implicitly assumes Y is positive?

Box-cox λ Data* → normal

want to be consistent

Jing Liu (UM-SJTU Joint Institute) Lecture 5 September 19, 2019 36 / 56

- For responses that might be negative, the following was proposed

$$Y_{i(\lambda_1,\lambda_2)}^* = \begin{cases} \dfrac{(Y_i + \lambda_2)^{\lambda_1} - 1}{\lambda_1}, & \text{if} \quad \lambda \neq 0; \\ \ln\left(Y_i + \lambda_2\right), & \text{if} \quad \lambda = 0. \end{cases}$$

the idea is essentially the same, so we will consider the positive case here.

cout

When y and x both do not follow normal dist., use transformation find the λ that maximize the likelihood function

如果log transformation是可以的，则会得到lambda的范围基本在0

而都不可以的情况，可以用提供的box-cox，可以同时转化x和y

Hide

```
oc.LM = lm(price~age, data = oc.df)
```

```
Error in eval(predvars, data, env) : object 'price' not found
```

L06 Prediction

to predict based on the current data

TSS

variability in the response variable

$TSS = \sum_{i=1}^{n}(y_i - \bar{y})^2 = \sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2 + \sum_{i=1}^{n}\hat{e}_i^2$ total sum of squares = explained sum of squares + residual sum of squares

TSS = ESS + RSS

L07 Multiple linear regression

- In multiple linear regression, we explore the relation between the response

$$Y_i \qquad i = 1, 2, \ldots, n$$

and two or more predictors/explanatory variables, k of them in general,

$$X_{i1}, \quad X_{i2}, \quad \ldots \quad X_{ij}, \quad \ldots \quad X_{ik} \qquad i = 1, 2, \ldots, n$$

where we are assuming that we observed n cases, and

1. The conditional mean of the response is given by the following for all i

$$\mathbb{E}\left[Y_i \mid X_{i1}, X_{i2}, \ldots, X_{ik}\right] = \mathbb{E}\left[Y_i \mid \mathbf{X}_i\right] = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_k x_{ik}$$

2. The errors have zero mean and constant variance

$$\mathbb{E}\left[\varepsilon_i \mid \mathbf{X}\right] = 0 \quad \text{and} \quad \mathrm{Var}\left[\varepsilon_i \mid \mathbf{X}\right] = \sigma^2 \qquad \text{where} \quad \varepsilon_i = Y_i - \mathbb{E}\left[Y_i \mid \mathbf{X}_i\right]$$

3. The errors are independent of $\mathbf{X}_i$, and of each other, for all i.
4. The errors follow a normal distribution.

regression coefficients in multiple regression: partial regression coefficients, as their interpretation requires other variables should be held fixed.

How least square works

$$\begin{aligned} \mathbf{y} &= \mathbf{X}\hat{\beta} + \hat{e} \\ \text{minimize residual sum of squrares} \quad & \hat{e}^T\hat{e} \\ \text{obtain the estimator} \quad & \hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \\ \text{fitted value} \quad & \hat{\mathbf{y}} = \mathbf{X}\hat{\beta} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} = \mathbf{P}\mathbf{y} \\ \text{residual} \quad & \hat{e} = \mathbf{y} - \hat{\mathbf{y}} = (\mathbf{I} - \mathbf{P})\mathbf{y} \end{aligned}$$

Projection Matrix$\mathbf{P}, \mathbf{P}^T = \mathbf{P}, \mathbf{P}^2 = \mathbf{P}$

Identity Matrix$\boldsymbol{I}$

unbiased $\hat{\beta}, \hat{\beta} \sim \mathrm{Normal}(\beta, \sigma^2(\mathbf{X}^\mathbf{T}\mathbf{X})^{-1})$

Case study: remember the small data set, we do not have strong evidence
three variables: y as gain; X as driven-in time and dose

1. drive-in time or dose influences gain linearly O -> F-test
2. drive-in time influences gain for a given dose linearly -> t-test
3. dose influences gain for a given drive-in time linearly -> t-test
4. there is any reason to use both drive-in time and dose to explain gain -> compare with the submodel using adjusted R-squared

Hide

```
# shapiro-wilk
shapiro.test(res) # small p-value: reject
```

```
    Shapiro-Wilk normality test

data:  res
W = 0.95149, p-value = 0.584
```

For question 4, using adjusted R squared which takes the model complexity into account. However, it is only meaningful when **all the model assumptions are met**.

L08 - poly

Example1: only drop variables after check residual plots (all the assumptions are satisfied)

Hide

```
# plot the residuals, find non-random, indicate nonlinearity
# The nonlinearity would be clearer if add estimated conditional mean to the residual pl
ot
c.mean.plot = function(tmp.x, tmp.y, n.each, pos = "topleft"){ n.x = rounod(length(uniqu
e(tmp.x)) / n.each)
x.group = cut(tmp.x, breaks = n.x, labels = FALSE)
x.vec = tapply(tmp.x, x.group, mean) y.vec = tapply(tmp.y, x.group, mean)
```

```
Error: unexpected symbol in:
"x.group = cut(tmp.x, breaks = n.x, labels = FALSE)
x.vec = tapply(tmp.x, x.group, mean) y.vec"
```

str(LifeCycleSavings)

need normally distributed y but do not need normally distributed x, because error term relates to y two mode / clear gap in the residual plot -> two distributions

当ACF的蓝色线很宽的时候，说明没有太多的data

L10

- stability problem

L 11

Case Study shows how to address multicollinearity and unusual points (data cleaning)

L12

Heteroskedasticity

Hide

```
v.vec
```

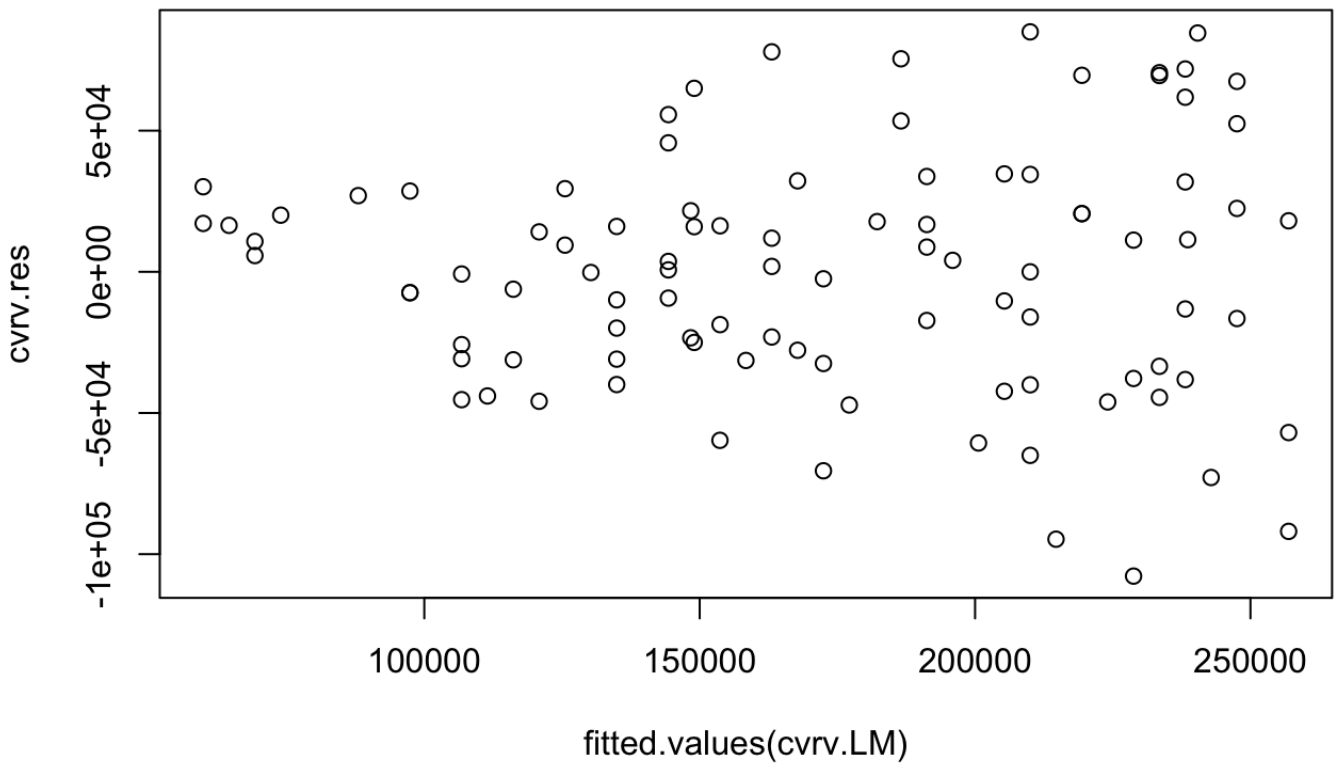
```
       16        10        12        16        16         4
144.01111  62.83929  53.14286 144.01111 144.01111  24.66667
        2         4         6         2        12         8
  9.00000  24.66667  22.00000   9.00000  53.14286  44.12500
       16         2         2         2         6        10
144.01111   9.00000   9.00000   9.00000  22.00000  62.83929
       16        16        10         6         2         6
144.01111 144.01111  62.83929  22.00000   9.00000  22.00000
       10        12         4         4        16         8
 62.83929  53.14286  24.66667  24.66667 144.01111  44.12500
       10        16         6        10        12         8
 62.83929 144.01111  22.00000  62.83929  53.14286  44.12500
       10         8         8         2        16         8
 62.83929  44.12500  44.12500   9.00000 144.01111  44.12500
        8        12        10        16         2         2
 44.12500  53.14286  62.83929 144.01111   9.00000   9.00000
        8        12         4         4        12
 44.12500  53.14286  24.66667  24.66667  53.14286
```

as the $\hat{e_i}$ might be very small/large, work with

$z_i = ln(\hat{e_i}^2) = 2ln|\hat{e_i}|$, the log-scale provides extra numerical stability

Hide

```
cvrv.df = read.table("/Users/yuxinmiao/Documents/JI/JI2020Fall/VE406/R code/in-class dem
o/Data/cvrv.csv", sep = ",", header = TRUE)
cvrv.LM = lm(capital~rental, data = cvrv.df)
cvrv.res = residuals(cvrv.LM)
plot(fitted.values(cvrv.LM), cvrv.res) # the variance is clearly unequal, and rental var
ies continuously., not in the same scale
```



Hide

```
z = 2 * log(abs(cvrv.LM$residuals)) #  z is the auxiliary response,
auxiliary.LM = lm(z~rental, data = cvrv.df)  # Perform the auxiliary regression
v.vec = exp(auxiliary.LM$fitted.values) # transform back to have the estimated \sigma
w.vec = 1/v.vec
cvrv.WLS = lm(capital~rental, weights = w.vec, data = cvrv.df) # weight according to the
estimated \sigma
```

L13

correlated errors, check of independence

Hide

```
sales.df =ead.table("/Users/yuxinmiao/Documents/JI/JI2020Fall/VE406/R code/in-class dem
o/Data/sales.txt",  sep = "", header = TRUE)
```

```
Error in ead.table("/Users/yuxinmiao/Documents/JI/JI2020Fall/VE406/R code/in-class demo/
Data/sales.txt",  :
  could not find function "ead.table"
```

add lag do not help

fit the residuals (among previous and current residuals)

$\varepsilon_i = p\varepsilon_{i-1} + v_i$, the true error will add the new v_i. So mainly useful for prediction. Bias and variance should be reconsidered.

v_i is normal, so ε_i should be normal

structure : first-order autoregressive process / AR(1) (important in time series)

improve the model by fitting the residuals with time series issue

L14 variable selection

The validity of the model is a priority especially when the goal is **to explain**.

to predict, different modern approach: data splitting / cross-validation / bootstrapping

to regulazition:

ridge regression: shrinks the number of coefficients by imposing a penalty, encourages the coefficients to be zero. higher λ, more penalization. drop the inefficient coefficient. L2norm regulazition *do the normal transformations to it*

lasso regression:

L15 nonlinear regression

Least square estimation for solving non linear regression is same as for linear regression, but no closed-form exists. Numerically solve, start with a good initial value.

MLE and LSE still identical. in order to solve MLE, has the score equation.

Example

Hide

```
str(puromycin.df)
```

```
'data.frame':   12 obs. of  2 variables:
 $ Concentration: num  0.02 0.02 0.06 0.06 0.11 0.11 0.22 0.22 0.56 0.56 ...
 $ Velocity     : int  47 76 97 107 123 139 152 159 191 201 ...
```

Based on the obtained contour of RSS, observe the better region (RSS close to 0), then zoom in (210, 0.07 as the initial guess)

Hide

```
# Gauss-Newton is the default algorithm in R
parameter_each = capture.output({ puromycin.NL = nls(
Velocity~ par1*Concentration/(par2+Concentration),
start = initial, data = puromycin.df,
trace = TRUE) }) # with capture.output, we could get RSS and parameter in each iteration
```

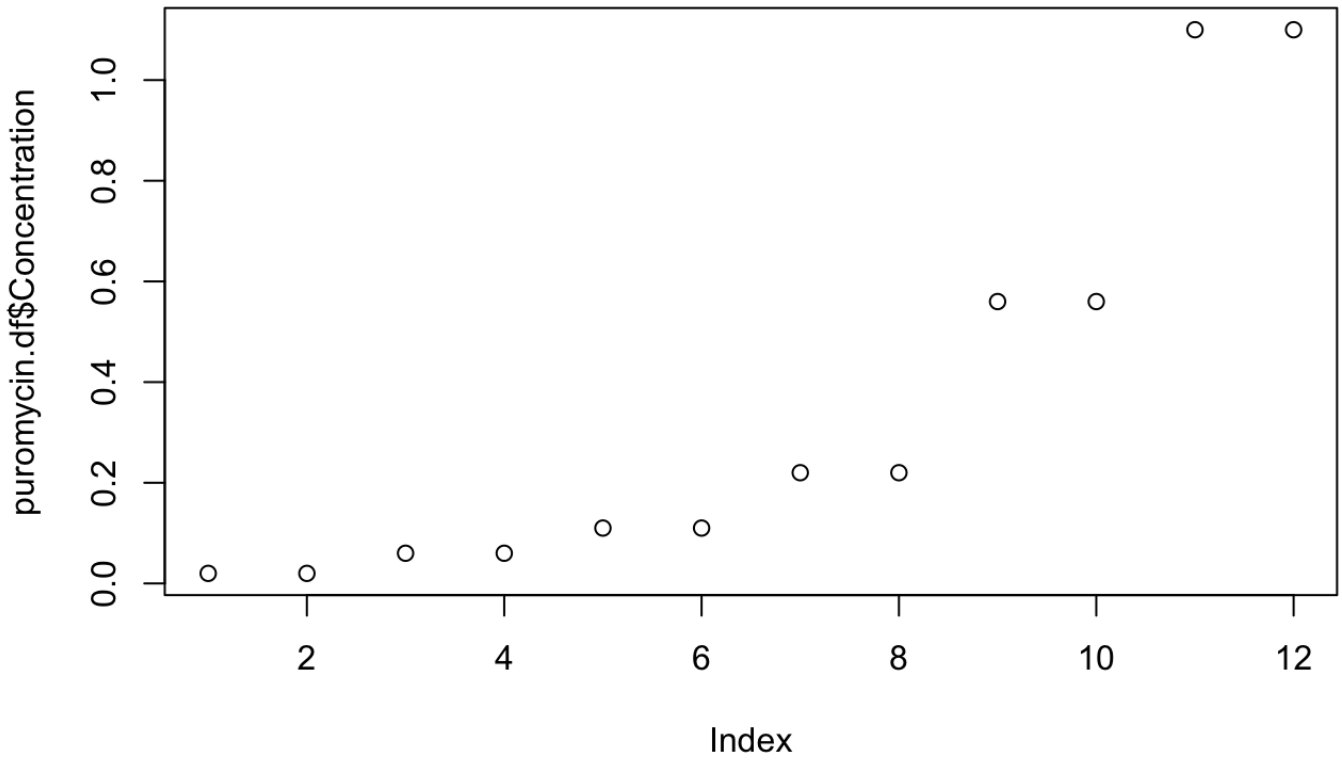
```
Error in nls(Velocity ~ par1 * Concentration/(par2 + Concentration), start = initial,  :
  object 'puromycin.df' not found
```

Hide

```
parameter_each
```

```
[1] "1480.134 :  210.00   0.07"
[2] "1195.633 :  212.95331922   0.06431621"
[3] "1195.449 :  212.69569008   0.06413995"
[4] "1195.449 :  212.68489661   0.06412308"
[5] "1195.449 :  212.68385411   0.06412145"
```

“Nonlinear regression uses an iterative algorithm to reduce the error sums of squares (SSE). For each iteration, the algorithm adjusts the parameter estimates in a manner that it predicts should reduce the SSE compared to the previous iteration. The iterations continue until the algorithm converges on the minimum SSE,”



(? Puromycin.NL Fit not finished)

For this particular nonlinear model, it could be transformed into a linear model

$\frac{\beta_1 c}{\beta_2+c} = \frac{\beta_2}{\beta_1 c} + \frac{1}{\beta_1} + \varepsilon^*$ not a same error now

(? transformed graph not finished)

Diagnostics

For nonlinear regression, large sample size, the sampling distribution approaches normal

Wald test and Wald interval

For hypothesis testing and confidence interval respectively

in summary, t-statistic and p-values are those for Wald test, only meant to be used for a really big sample.

bootstrap

used at least to check whether large-sample inference is appropriate. “construct the sampling distribution of the estimators in a nonlinear regression model, which then can be used to compute estimated standard errors, thus produce confidence intervals and prediction intervals.”

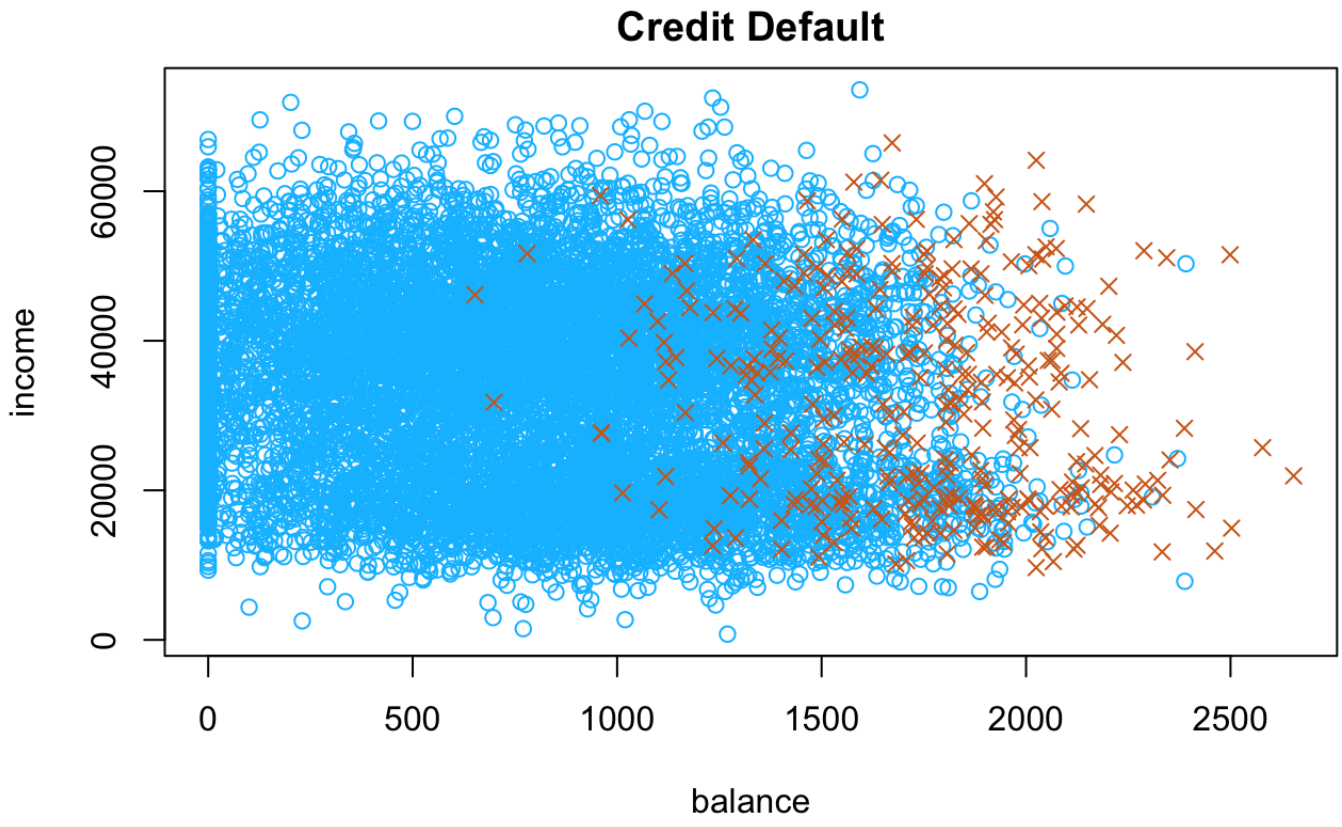
用原有的X，把estimate的beta作为true parameter，并且error$\delta \sim^{i.i.d} Normal(0, \hat{\sigma}^2)$. Then generate M sets of size-n samples of Y_i based on the model with original X. Each of simulated dataset gives one β. Then the M samples of β are used to approximate the sampling distribution of $\hat{\beta}$

Until now, we assumed that the errors following a normal distribution # L16 Logistic regression Mathematical equation / induction, see $Notes.md$

Hide

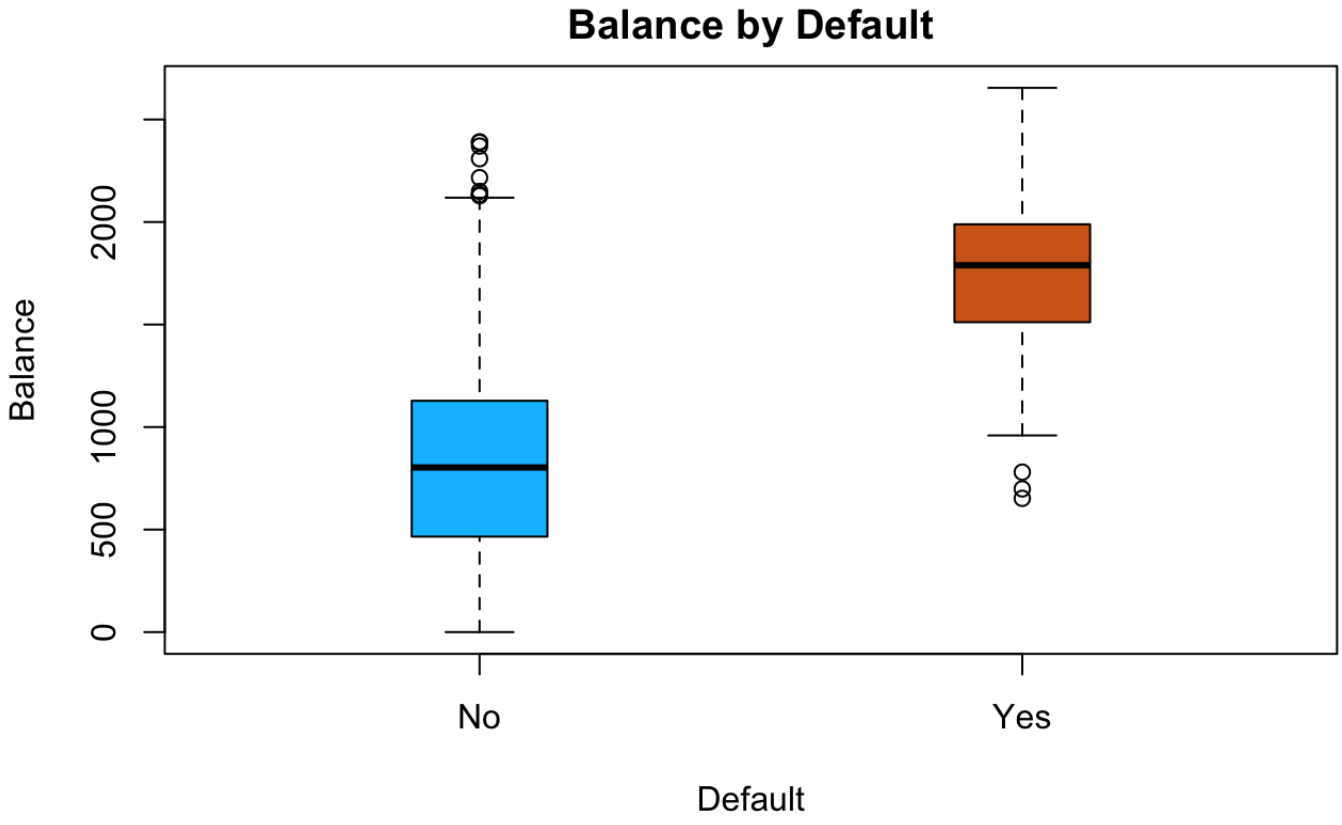
```
plot(income~balance, type="n",  data = credit.df, xlab = "balance", ylab = "income", mai
n = "Credit Default") + # type n: no display
points(credit.df[index0, 3], credit.df[index0, 4], col = "deepskyblue", pch = 1) + point
s(credit.df[index1, 3], credit.df[index1, 4],
col = "chocolate", pch = 4)
```

```
integer(0)
```



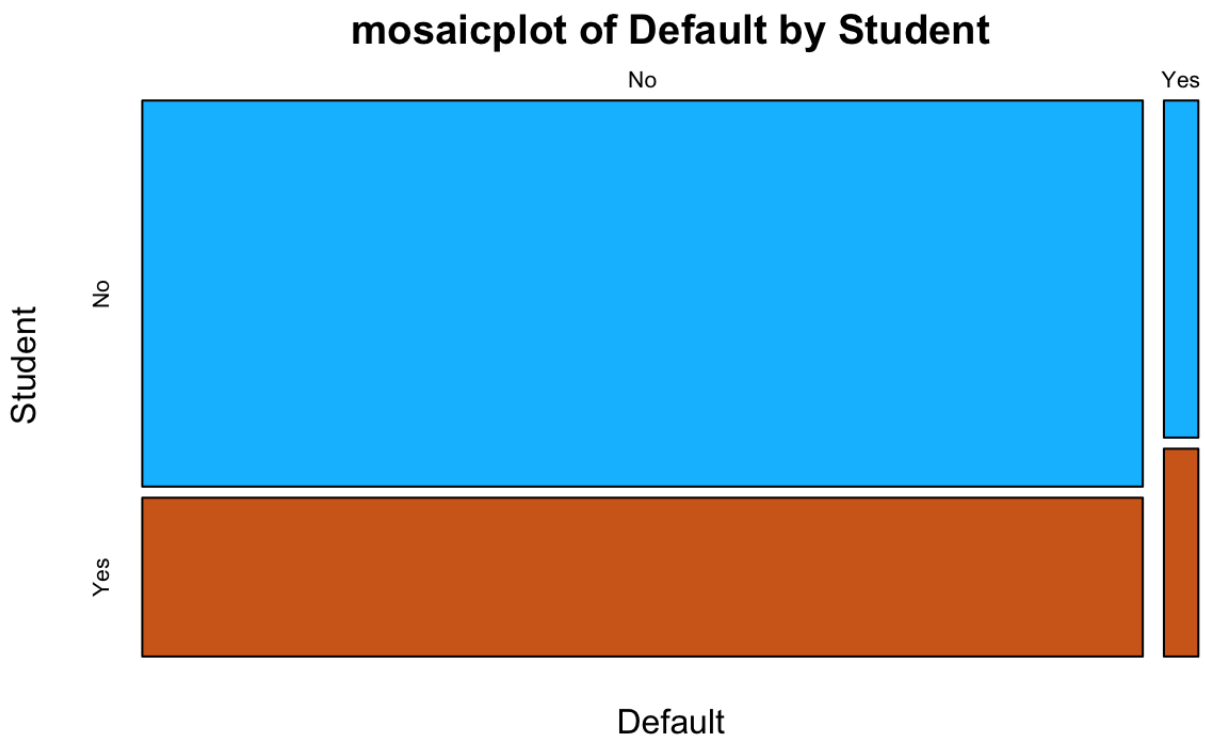
Hide

```
# Boxplot, further examine the difference caused by balance
boxplot(balance~default, data = credit.df,
col = c("deepskyblue", "chocolate"), boxwex = 0.25, xlab = "Default",
ylab = "Balance", names = c("No", "Yes"), main = "Balance by Default")
```



Hide

```
# Mosaicplot->student more likely to default
mosaicplot(
  table(list(
    Default = factor(credit.df$default,
                     labels = c("No", "Yes")),
    Student = credit.df$student)),
    col = c("deepskyblue", "chocolate"),
  main = "mosaicplot of Default by Student"
)
```



Use the logistic function

Hide

```
summary(credit.LG)
```

```
Call:
glm(formula = default ~ balance, family = binomial, data = credit.df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.2697  -0.1465  -0.0589  -0.0221   3.7589

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.065e+01  3.612e-01  -29.49   <2e-16 ***
balance      5.499e-03  2.204e-04   24.95   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1596.5  on 9998  degrees of freedom
AIC: 1600.5

Number of Fisher Scoring iterations: 8
```

Hide

```
# LR test of significance, similar to F-test
credit.null.LG = glm(default~1, family = binomial, data = credit.df)
LR.test = 2*(logLik(credit.all3.LG)[1] -logLik(credit.null.LG)[1]) # compare full with n
ull model, so three beta reduced to zero, degree of freedom should be 3
1-pchisq(LR.test, 3) # significantly strong evidence to reject that the reduced model is
adequate
```

```
[1] 0
```

However, no way to check the validity of the model, and appropriateness of using asymptotic approximation, care with **small n** !

Group dataset

Group: means there are groups of data points that have the same $\mathbf{X}$

Hide

```
# group logistic regression
ingots.df = credit.df = read.table("/Users/yuxinmiao/Documents/JI/JI2020Fall/VE406/R cod
e/in-class demo/Data/ingots.csv", header = TRUE, sep = ",")
ingots.df # 20 binomial distributions
```

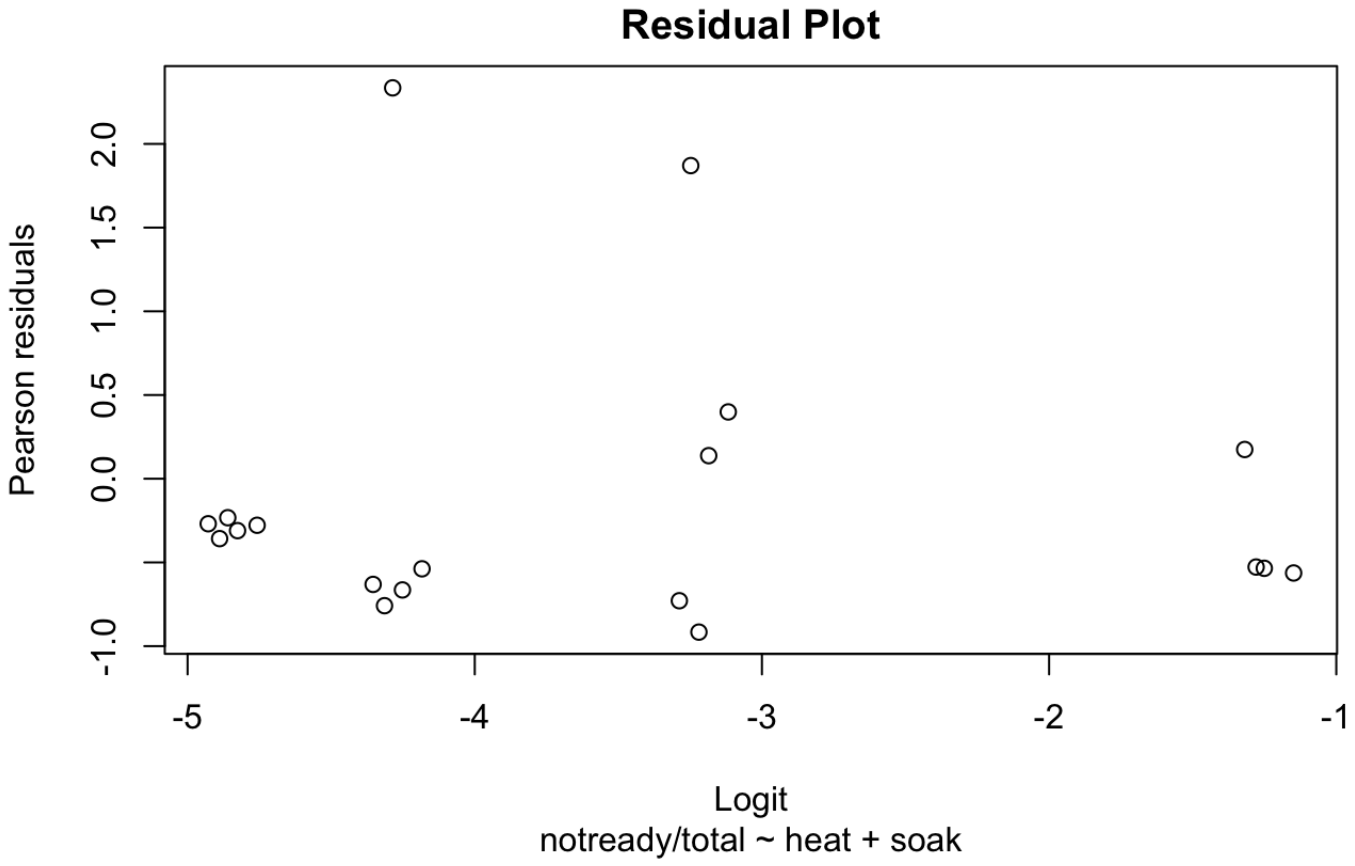
heat <int>	soak <dbl>	notready <int>	total <int>
7	1.0	0	10
14	1.0	0	31
27	1.0	1	56
51	1.0	3	13
7	1.7	0	17
14	1.7	0	43
27	1.7	4	44
51	1.7	0	1
7	2.2	0	7
14	2.2	2	33

1-10 of 20 rows Previous **1** 2 Next

Hide

```
ingots.LG = glm(notready/total~heat+soak, family = binomial, data = ingots.df, weights =
total)
summary(ingots.LG)
```

```
Call:
glm(formula = notready/total ~ heat + soak, family = binomial,
    data = ingots.df, weights = total)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.28311  -0.78183  -0.50514  -0.09701   1.71923

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -5.55917    1.11969  -4.965 6.87e-07 ***
heat         0.08203    0.02373   3.456 0.000548 ***
soak         0.05677    0.33121   0.171 0.863906
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 25.395  on 18  degrees of freedom
Residual deviance: 13.753  on 16  degrees of freedom
  (1 observation deleted due to missingness)
AIC: 34.08

Number of Fisher Scoring iterations: 5
```

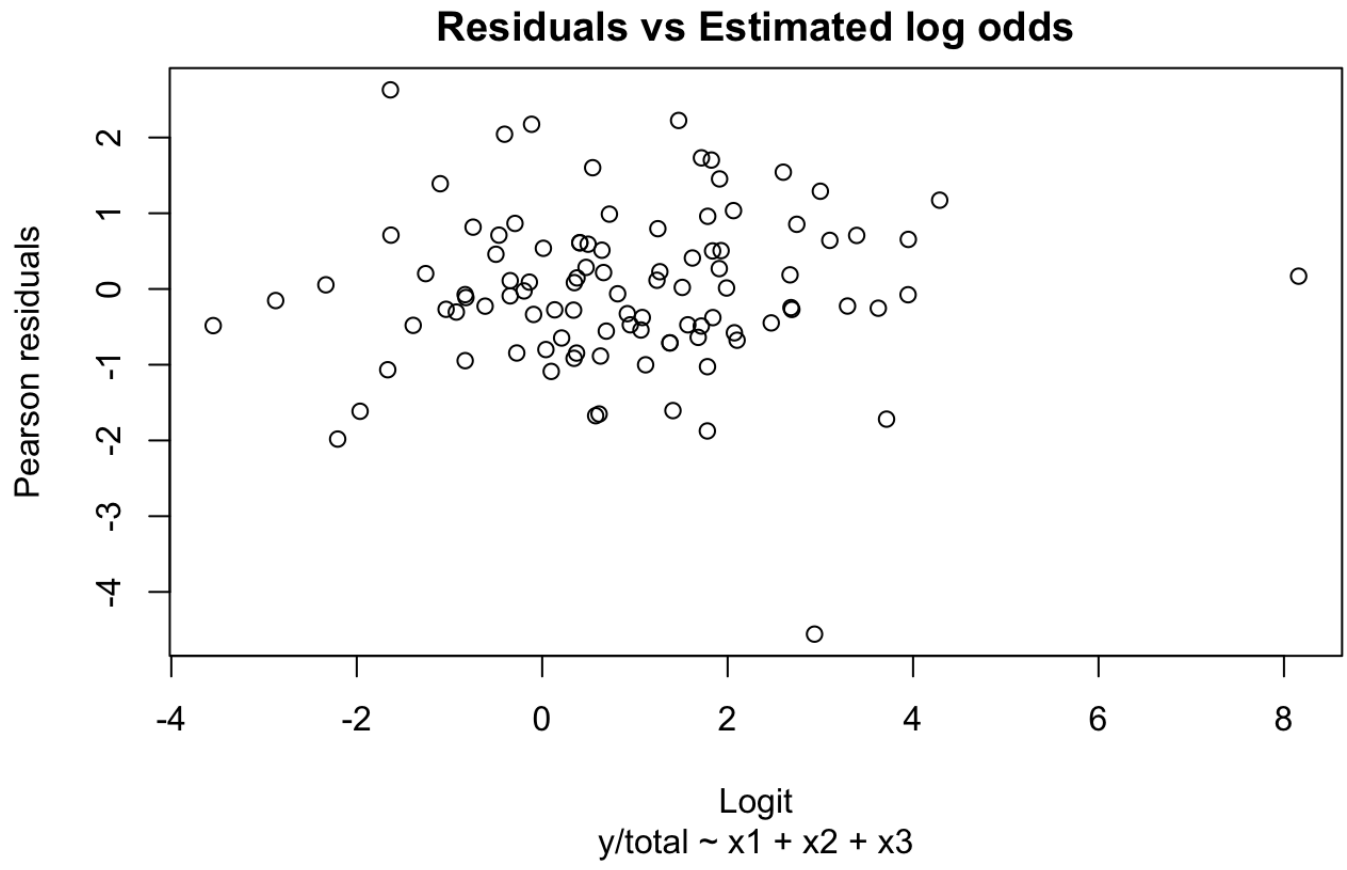


Simulation study

for grouped dataset, shows finding the outliers / leverage point using residual plot, how the model change after removing those points.

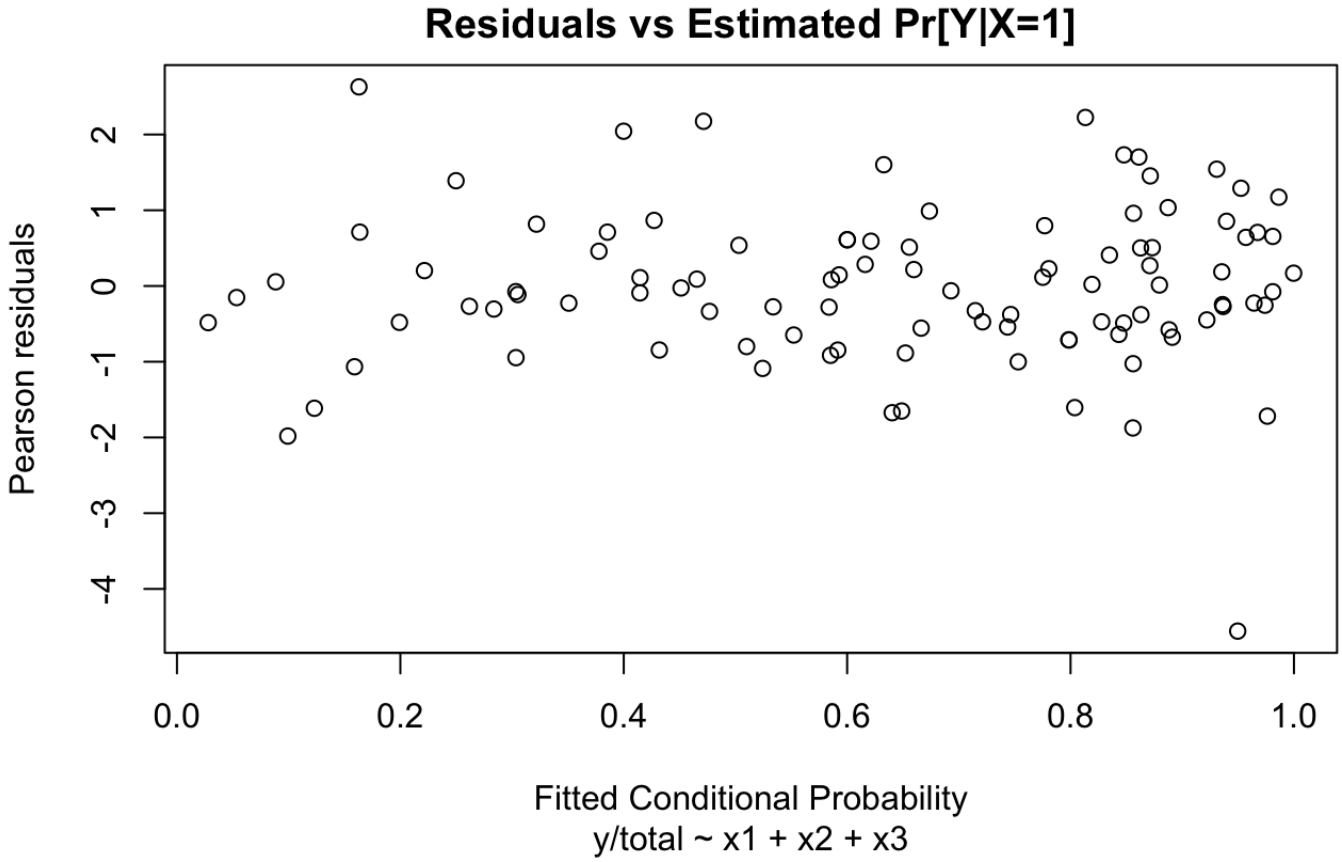
Hide

```
# simulation study
set.seed(1)
n = 100
x1 = rnorm(n, mean = 0, sd = 6)
x2 = rt(n, df = 1) # random student t distribution
x3 = rbinom(n, size = 1, prob = 0.45)
beta0 = 0.6
beta1 = 0.3
beta2 = 0.1
beta3 = 0.1
logit = beta0 + beta1*x1 + beta2*x2 + beta3*x3
true.pi = exp(logit)/(1+exp(logit))
m = 100
y = integer(n)
for (i in 1:n){
  y[i] = rbinom(1, size = m, prob = true.pi[i])
} # simulated y
sim.df = data.frame(y = y, x1 = x1, x2 = x2, x3 = x3, total = rep(m, n)) # simulated y
sim.LG = glm(y/total~x1+x2+x3, family = binomial, weights = total , data = sim.df)
res = residuals(sim.LG, type="pearson")
logit = predict(sim.LG) # default logit
prob = predict(sim.LG, type = "response") # Prob
plot(logit, res, xlab="Logit", ylab = "Pearson residuals", main="Residuals vs Estimated
 log odds", sub = deparse(sim.LG$formula))
```



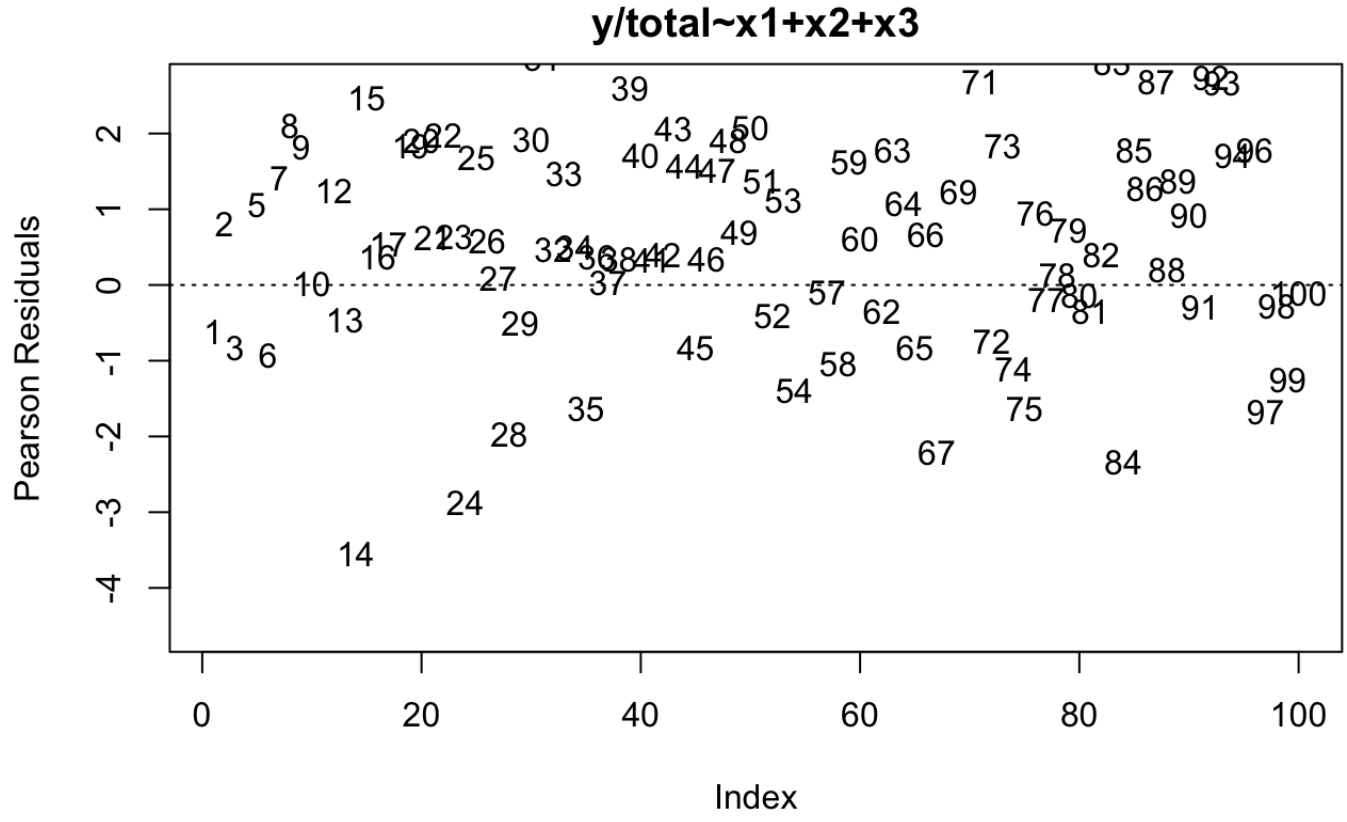
Hide

```
plot(prob, res, xlab="Fitted Conditional Probability", ylab = "Pearson residuals", main=
"Residuals vs Estimated Pr[Y|X=1]", sub = deparse(sim.LG$formula))
```



Hide

```
plot(res, type="n",
ylab = "Pearson Residuals", main="y/total~x1+x2+x3")
text(logit); abline(h=0,lty=3)
```

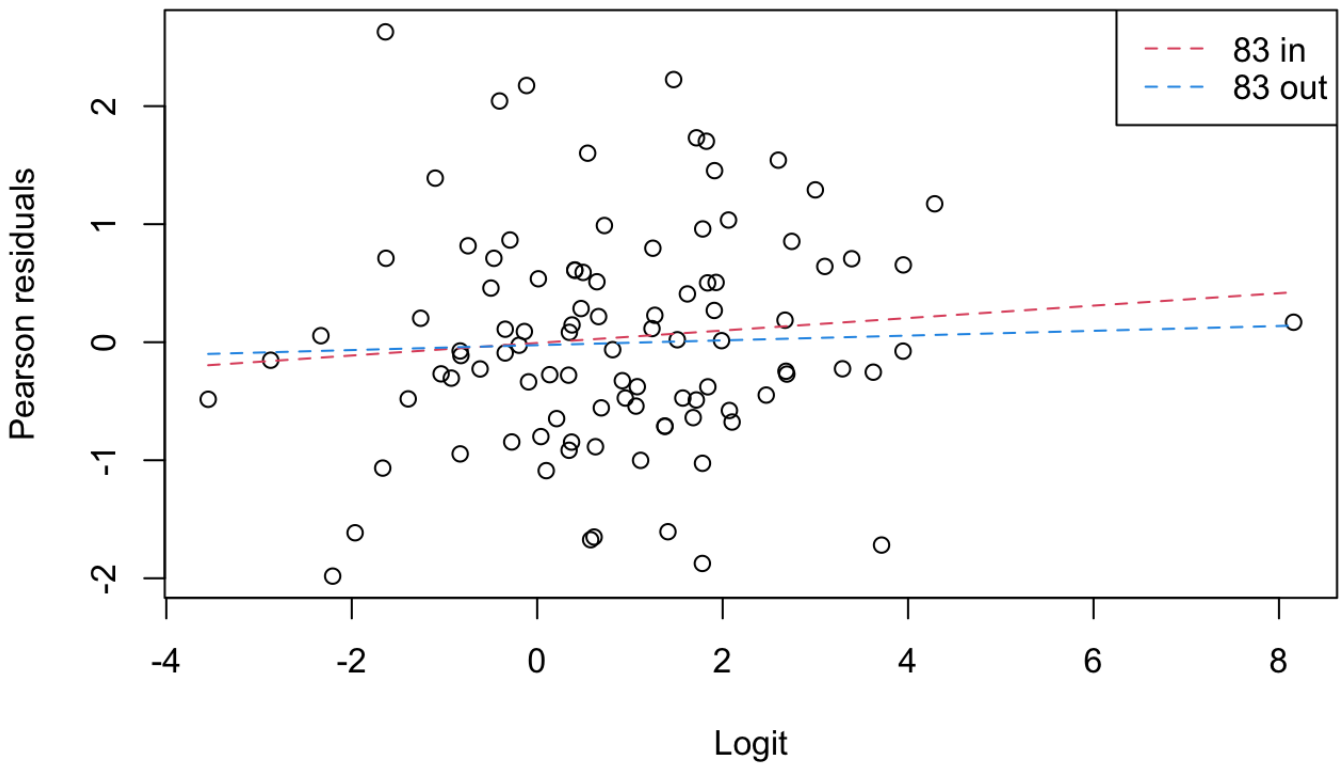


Hide

```
plot(logit[-c(83)], res[-c(83)], xlab="Logit", ylab="Pearson residuals")

# what happens if exclude point index 83 ?
lines(smooth.spline(logit[-c(83)], res[-c(83)]), col = 2, lty = 2)
```

Hide

```
lines(smooth.spline(logit, res), col = 4, lty = 2)
legend("topright", c("83 in", "83 out"), col = c(2, 4), lty = 2)
```



Hide

```
# high leverage points
head(sort(hatvalues(sim.LG), decreasing = TRUE)) # hatvalues: compute some of the regres
sion (leave-one-out deletion) diagnostics for linear and generalized linear models
```

```
       21         46         62         36         71         45
0.37676714 0.15760469 0.13174284 0.07640166 0.06673067 0.06273533
```

Hide

```
# highest: 21

# influential points
head(sort(cooks.distance(sim.LG), decreasing = TRUE))
```

```
       21         83         75         67         52         46
0.19014483 0.11674260 0.08128077 0.06087118 0.05277872 0.04652510
```

Hide

```
# highest: 21 > 83

# 21: high leverage influential point, 83: influential outlier, exclude them in the fina
l model
sim.final.LG = glm(y/total~x1+x2+x3, family=binomial, weights = total, data=sim.df[-c(83
, 21),])
# which gives the approximate input beta 0-3: 0.6, 0.3, 0.1, 0.1


# R_d^2 is
1-sim.final.LG$deviance/sim.final.LG$null.deviance
```

```
[1] 0.9747189
```

Hide

```
# =0.97, model capture most of the deviation in the date
```

L17 Poisson

Hide

```
bomber.df = read.table("/Users/yuxinmiao/Documents/JI/JI2020Fall/VE406/R code/in-class d
emo/Data/aircraft.csv", header = TRUE, sep = ",")
str(bomber.df)
```

```
'data.frame':   30 obs. of  4 variables:
 $ Damage: int  0 1 0 0 0 0 1 0 0 2 ...
 $ Type  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Month : int  4 4 4 5 5 5 6 6 6 7 ...
 $ Load  : num  91.5 84 76.5 69 61.5 80 72.5 65 57.5 50 ...
```

Hide

```
# the MLE of beta
bomber.PS = glm(Damage ~ Type + Month + Load, family=poisson, data=bomber.df)
bomber.PS
```

```

Call:  glm(formula = Damage ~ Type + Month + Load, family = poisson,
    data = bomber.df)

Coefficients:
(Intercept)         Type        Month         Load
   -0.40602      0.56877      0.16543     -0.01352

Degrees of Freedom: 29 Total (i.e. Null);  26 Residual
Null Deviance:      53.88
Residual Deviance: 25.95    AIC: 87.65
```

Hide

```
s1 <- data.frame(Type = 0, Month = mean(bomber.df$Month), Load = mean(bomber.df$Load))
predict(bomber.PS, s1)  # log of count
```

```
         1
-0.1582296
```

Hide

```
exp(predict(bomber.PS, s1))
```

```
        1
0.8536538
```

Hide

```
predict(bomber.PS, s1, type='response')  # count
```

```
        1
0.8536538
```

Hide

```
# the deviance
1-bomber.PS$deviance/bomber.PS$null.deviance # 0.51: not a very good model, only explain
s 0.51
```

```
[1] 0.5183429
```

Hide

```
# LR-test
1-pchisq(bomber.PS$deviance,bomber.PS$df.residual) # 0.46: large p-value -> no evidence
 of lack of fit
```

```
[1] 0.4656818
```

Hide

```
  # significance
1-pchisq(bomber.PS$null.deviance,bomber.PS$df.null) # 0.0033: small p-value: at least on
e of the regressors is needed
```

```
[1] 0.003337052
```

Hide

```
# for small sample & lack of distribution, do bootstrap (check the validity of using asy
mptotic approximation)
M = 100000 # bootstrap simulation
n = nrow(bomber.df)
fvs = fitted.values(bomber.PS) # mhat = mean
dam.vec = rpois(M, lambda = fvs) # specify paramter of poisson to be the estimate, vecto
r of means
dam.mat = matrix(dam.vec, nrow = M,
ncol = n, byrow = TRUE)
my.func = function(x){
  tmp.PS = glm(x ~ Type + Month + Load,
               family=poisson, data=bomber.df)
  return(c(tmp.PS$deviance, tmp.PS$null.deviance))
}

parameter.sim = apply(dam.mat, MARGIN = 1, FUN = my.func)
```

Hide

```
p = 1
hist(parameter.sim[p,], probability = TRUE, xlab = bquote(hat(beta)[.(p)]), breaks = 30,
main = bquote(paste(
"Bootstrap Sampling distribution of ", hat(beta)[.(p)])) )
```

```
Error in hist(parameter.sim[p, ], probability = TRUE, xlab = bquote(hat(beta)[.(p)]),  :
  object 'parameter.sim' not found
```

? poisson bootstrap

Model Selection

1. get the models
2. small datasize, use bootstrap simulation
3. at this case, determine based on the estimated MSE

Hide

```
rowMeans(mse.sim)
```

```
[1] 2.458989 1.550359 3.024559 1.641191
```

L18 GLM

The GLM is a model with model structure

L19 GEE

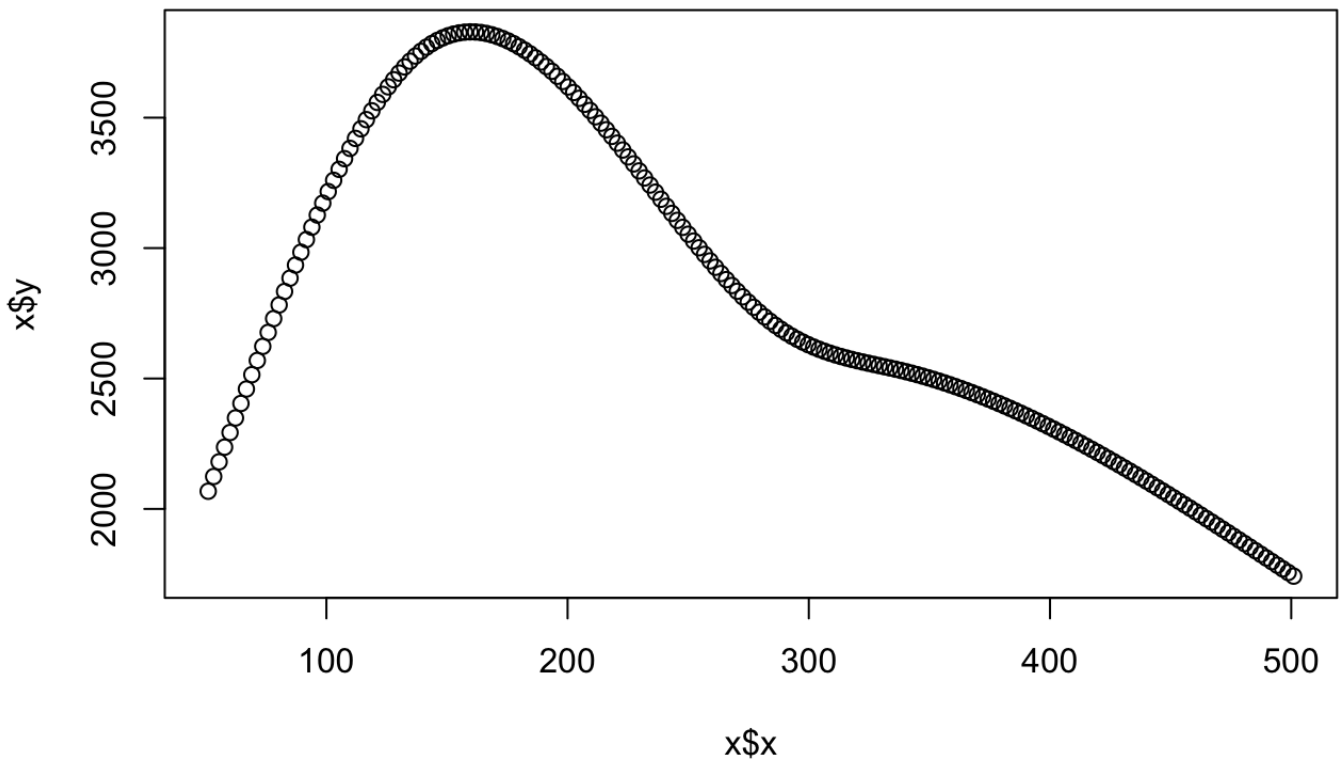
Hide

```
semi.GEE # the model does not give strong correlation, check glm
```

```
 GEE:  GENERALIZED LINEAR MODELS FOR DEPENDENT DATA
 gee S-function, version 4.13 modified 98/01/27 (1998)

Model:
 Link:                      Logarithm
 Variance to Mean Relation: Gamma
 Correlation Structure:     AR-M , M = 1

Call:
gee(formula = Camber ~ . - Run, id = Run, data = semiconductor.df,
    family = Gamma(link = "log"), corstr = "AR-M", Mv = 1)

Number of observations :  64

Maximum cluster size   :  16


Coefficients:
  (Intercept) Ltemperature1        Ltime1    Lpressure1 Ftemperature1
  -4.57245347    0.35862963    0.01178501    0.59939347   -0.04090419
       Ftime1       Fpoint1
  -0.39396896   -0.76706214

Estimated Scale Parameter:  0.1381516
Number of Iterations:  7

Working Correlation[1:4,1:4]
             [,1]        [,2]        [,3]         [,4]
[1,]  1.000000000 -0.12867472  0.01655718 -0.002130491
[2,] -0.128674720  1.00000000 -0.12867472  0.016557184
[3,]  0.016557184 -0.12867472  1.00000000 -0.128674720
[4,] -0.002130491  0.01655718 -0.12867472  1.000000000


Returned Error Value:
[1] 0
```

L20 Nonparametric

simple smoothing



Kernel Smoothing

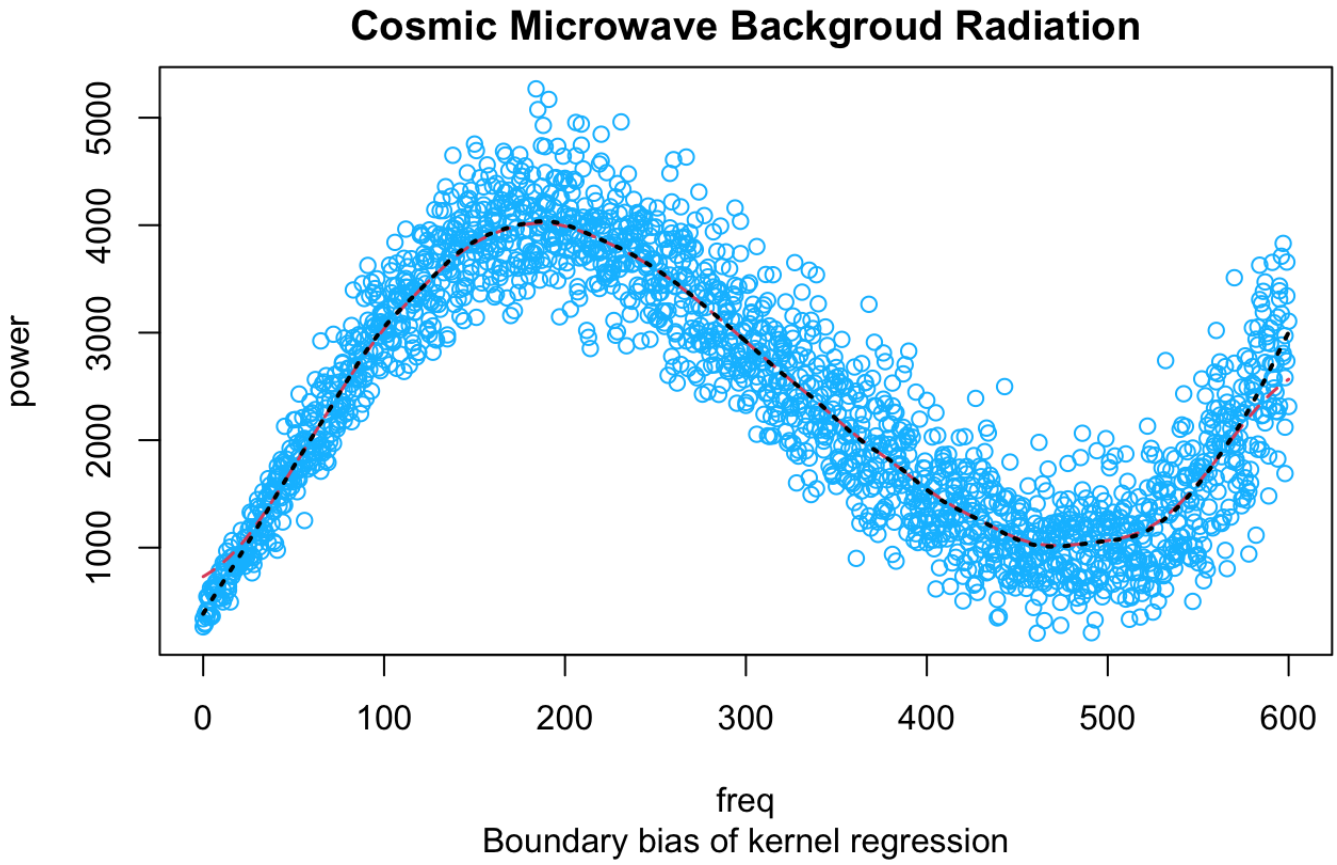
Penalised Regression

Hide

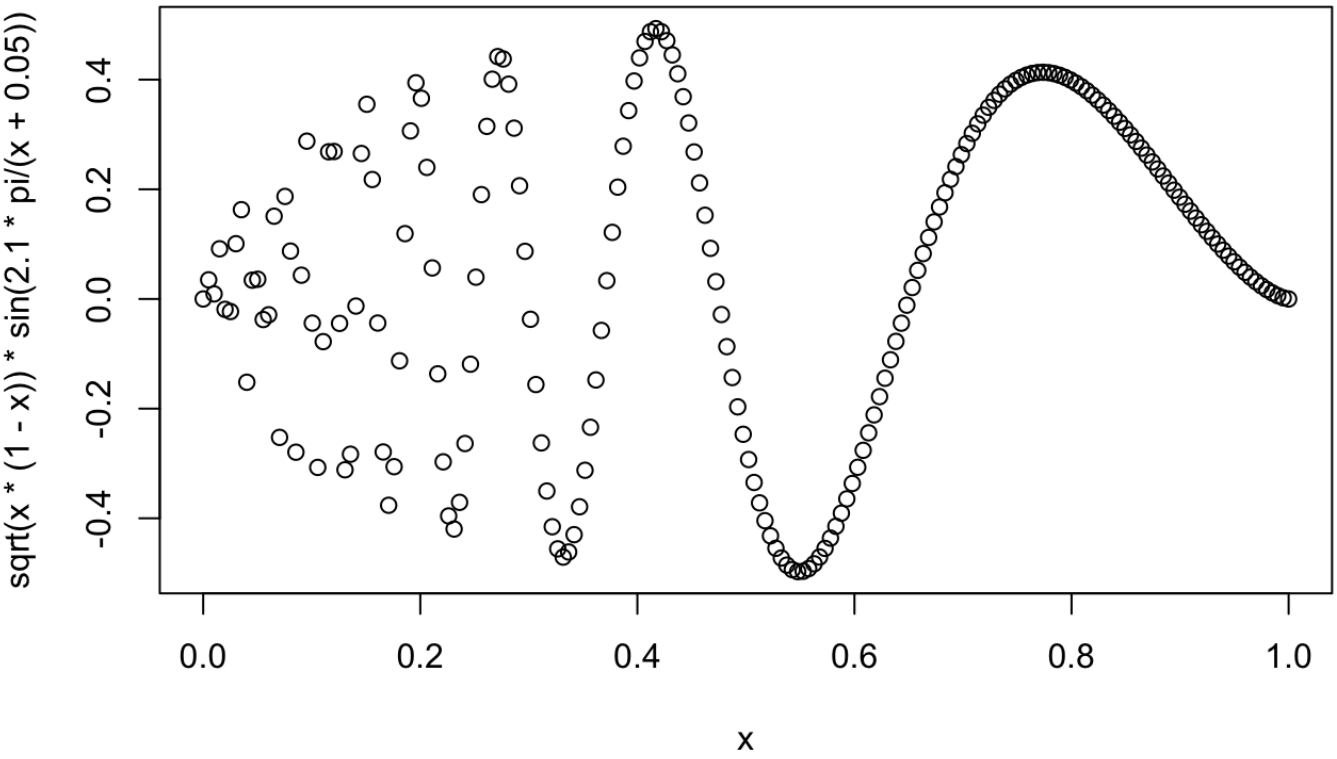
```
kreg = ksmooth(cmb.df$freq, cmb.df$power, kernel = 'normal', bandwidth = 44)
library(KernSmooth)
lreg = locpoly(cmb.df$freq,cmb.df$power,degree=1, bandwidth=dpill(cmb.df$freq, cmb.df$po
wer)) # Estimates a probability density function, regression function or their derivativ
es using local polynomials.
plot(power~freq, data = cmb.df, col = "deepskyblue", main ="Cosmic Microwave Backgroud R
adiation", sub = "Boundary bias of kernel regression")
lines(kreg, col=2, lwd = 1.5, lty = 2)
```

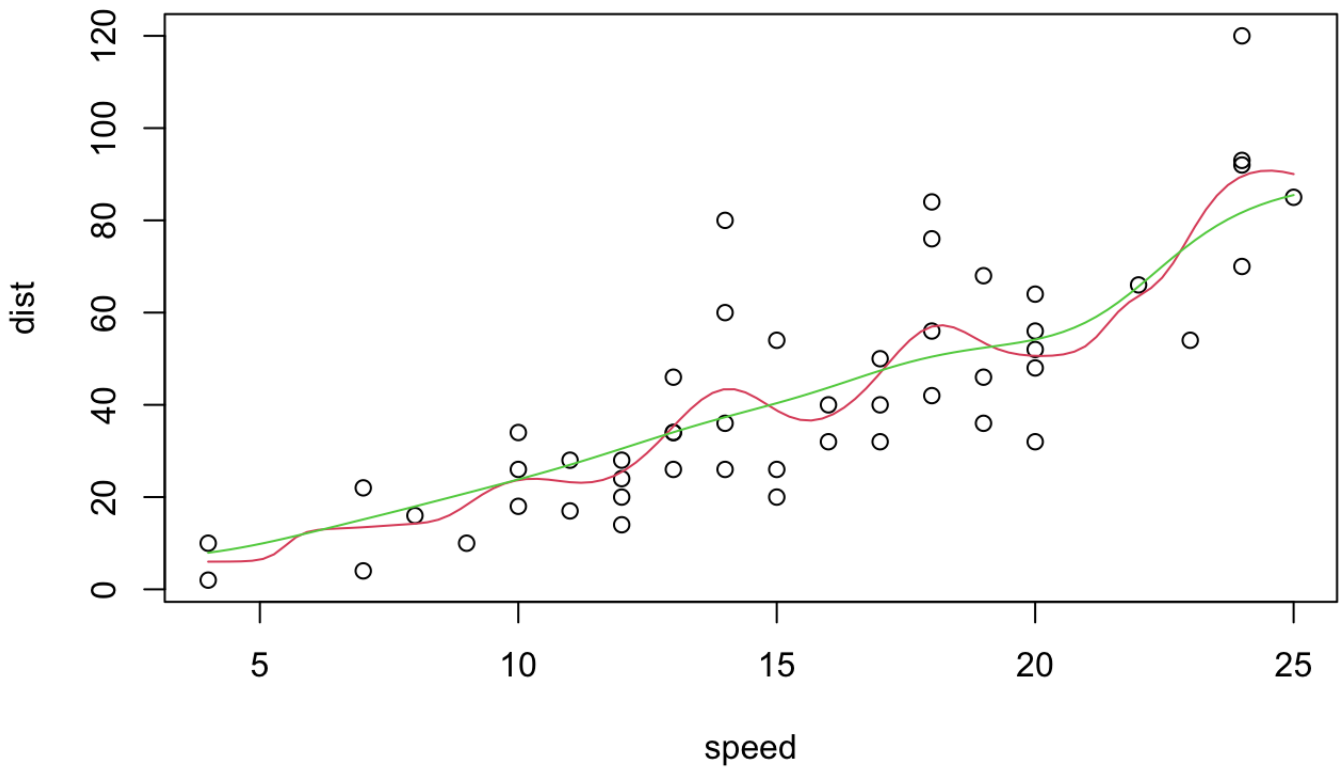
Hide

```
lines(lreg, col= 1, lwd = 2, lty = 3)
```



Boundary bias of kernel regression





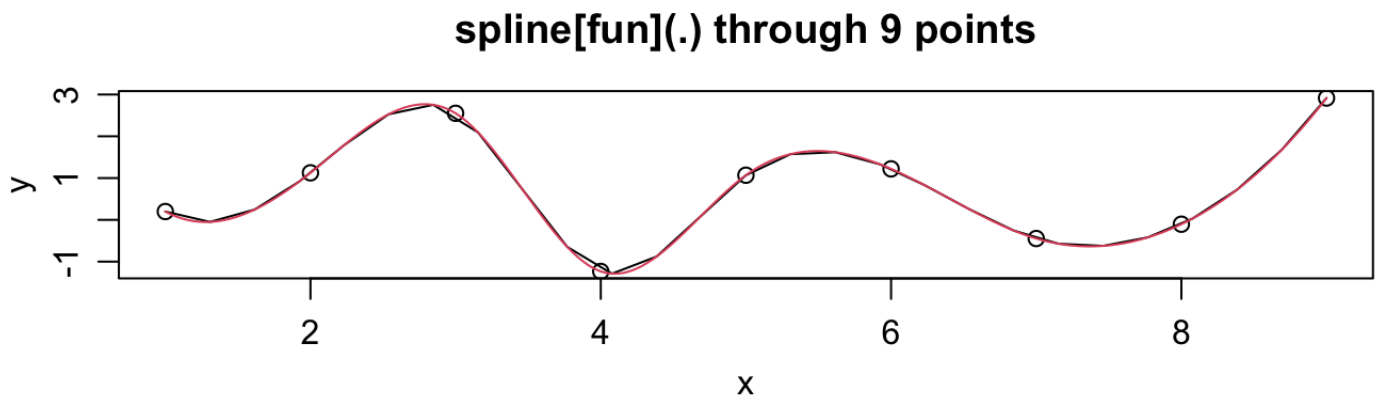
no data provided for the later courses, so no reproducing of plots in class

Hide

```
# interpolating spline from web
op <- par(mfrow = c(2,1), mgp = c(2,.8,0), mar = .1+c(3,3,3,1))
n <- 9
x <- 1:n
y <- rnorm(n)
plot(x, y, main = paste("spline[fun](.) through", n, "points"))
lines(spline(x, y))
```

Hide

```
lines(spline(x, y, n = 201), col = 2)
```



Hide

```
y <- (x-6)^2
plot(x, y, main = "spline(.) -- 3 methods")
lines(spline(x, y, n = 201), col = 2)
```

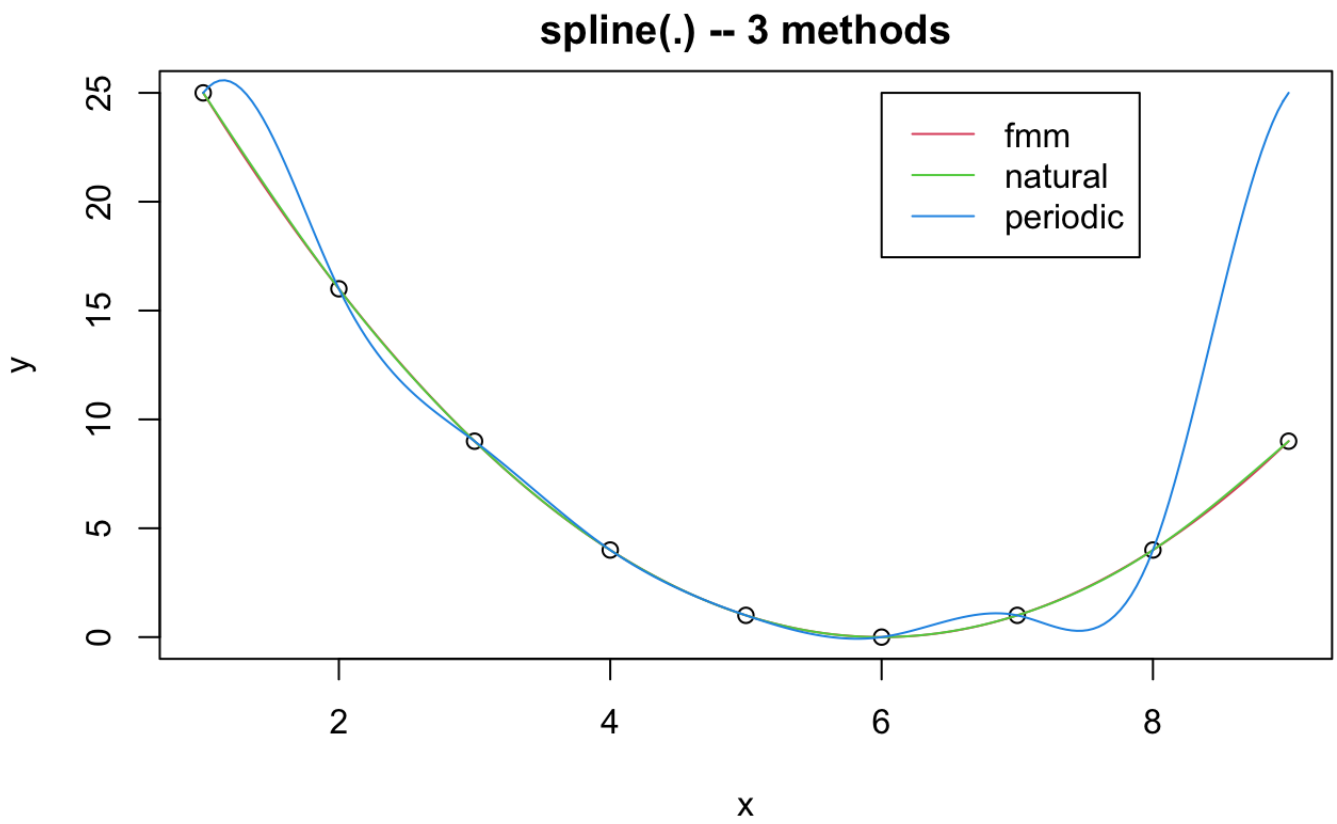
Hide

```
lines(spline(x, y, n = 201, method = "natural"), col = 3)
lines(spline(x, y, n = 201, method = "periodic"), col = 4)
```

```
spline: first and last y values differ - using y[1] for both
```

Hide

```
legend(6,25, c("fmm","natural","periodic"), col=2:4, lty=1)
```



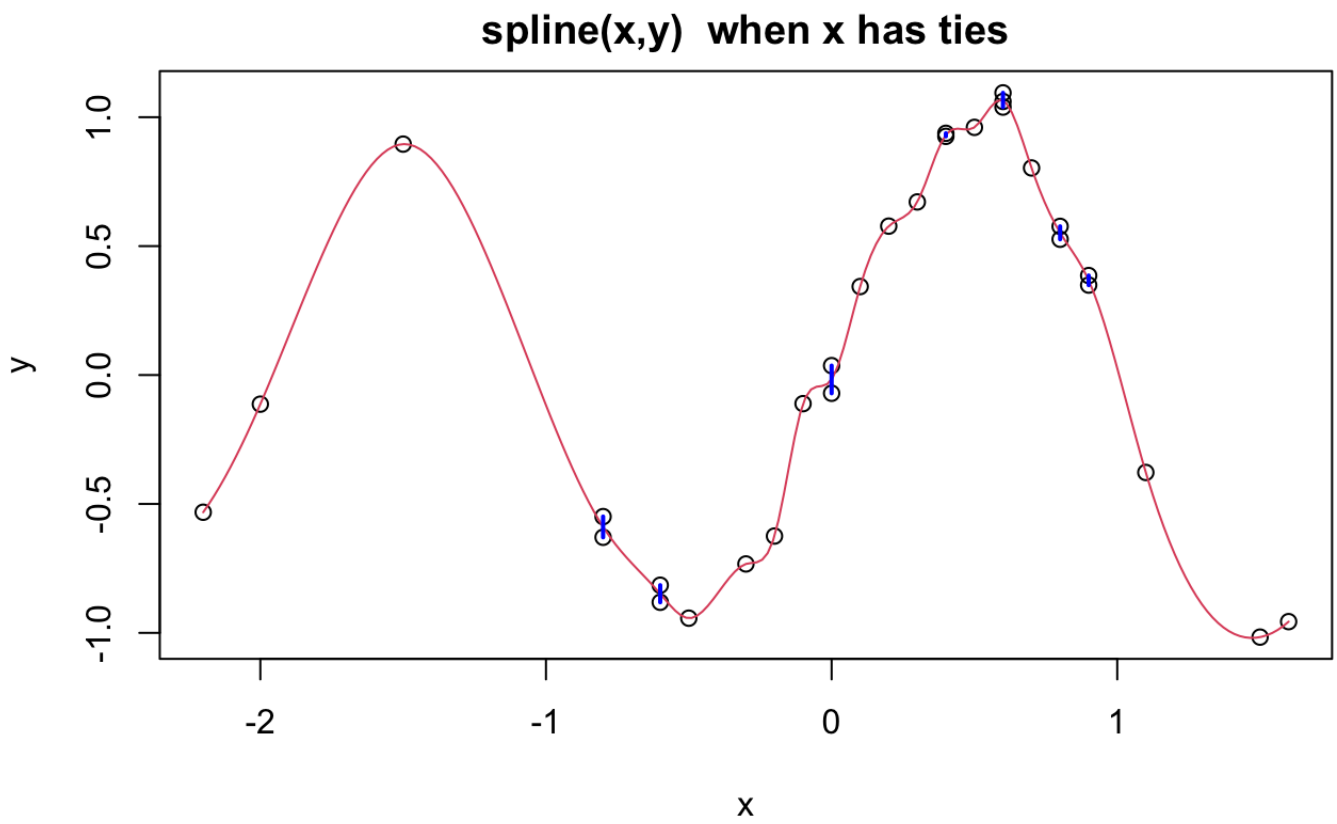
Hide

```
rm(list=ls())
## An example with ties (non-unique  x values):
set.seed(1); x <- round(rnorm(30), 1); y <- sin(pi * x) + rnorm(30)/10
plot(x,y, main="spline(x,y)  when x has ties")
lines(spline(x,y, n= 201), col = 2)
```

```
collapsing to unique 'x' values
```

Hide

```
## visualizes the non-unique ones:
tx <- table(x); mx <- as.numeric(names(tx[tx > 1]))
ry <- matrix(unlist(tapply(y, match(x,mx), range, simplify=FALSE)),
             ncol=2, byrow=TRUE)
segments(mx, ry[,1], mx, ry[,2], col = "blue", lwd = 2)
```

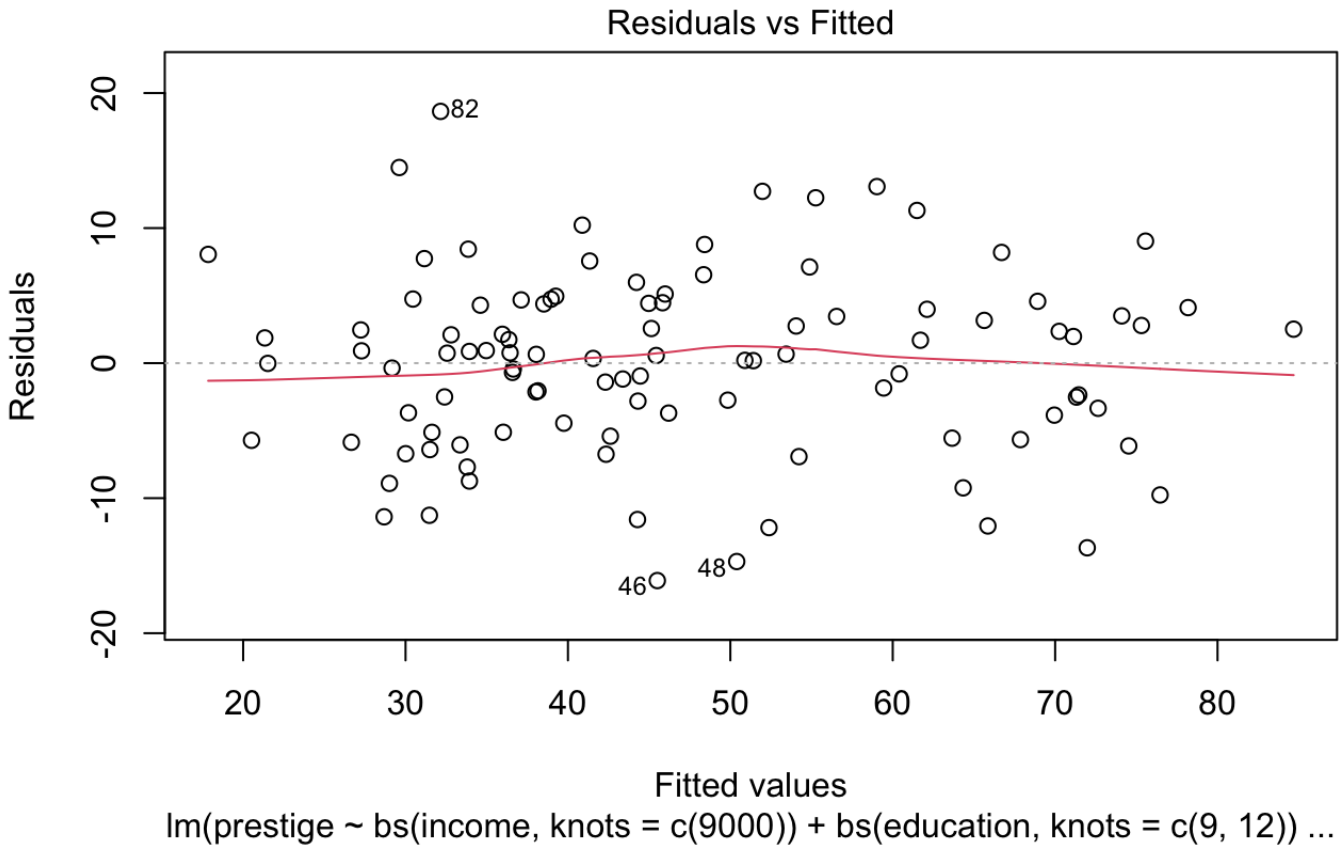


Hide

```
summary(fit2)
```

```
Call:
lm(formula = Sales ~ poly(Quantity, 2) + Quantity, data = data)

Residuals:
   Min     1Q Median     3Q    Max
-696.8 -452.1  -22.8  293.1  788.0

Coefficients: (1 not defined because of singularities)
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)          3414.2      139.8  24.425 3.02e-10 ***
poly(Quantity, 2)1   2298.5      504.0   4.560  0.00104 **
poly(Quantity, 2)2  -3211.8      504.0  -6.373 8.11e-05 ***
Quantity                NA         NA      NA       NA
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 504 on 10 degrees of freedom
Multiple R-squared:   0.86, Adjusted R-squared:  0.8319
F-statistic:  30.7 on 2 and 10 DF,  p-value: 5.386e-05
```

#L21Generalised Additive Model Additive models that use splines



Using smoothing splines and a linear function

Hide

```
# install.packages("ISLR")
library(ISLR) # The Wage dataset is a part of it
attach(Wage) # Variables in Wage become global
```

```
The following object is masked from Prestige (pos = 3):

    education

The following object is masked from Prestige (pos = 6):

    education

The following objects are masked from Wage (pos = 19):

    age, education, health, health_ins, jobclass, logwage,
    maritl, race, region, wage, year

The following objects are masked from Wage (pos = 20):

    age, education, health, health_ins, jobclass, logwage,
    maritl, race, region, wage, year

The following object is masked from Prestige (pos = 22):

    education
```

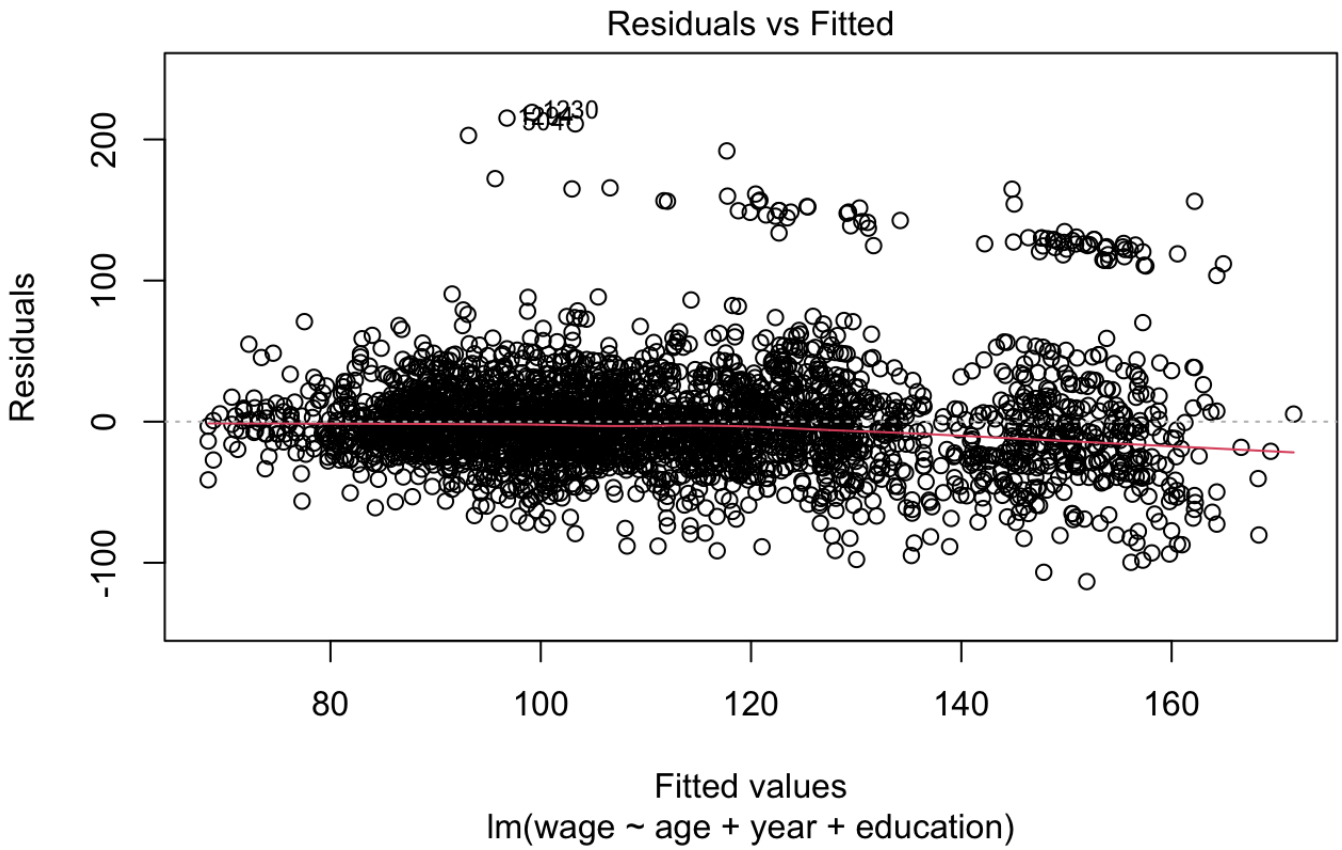
Hide

```
wage.LM = lm(wage~age+year+education)
summary(wage.LM)
```

```

Call:
lm(formula = wage ~ age + year + education)

Residuals:
     Min       1Q   Median       3Q      Max
-113.323  -19.521   -3.964   14.438  219.172

Coefficients:
                             Estimate Std. Error t value Pr(>|t|)
(Intercept)                -2.058e+03  6.493e+02  -3.169  0.00154 **
age                         5.621e-01  5.714e-02   9.838  < 2e-16 ***
year                        1.056e+00  3.238e-01   3.262  0.00112 **
education2. HS Grad         1.140e+01  2.476e+00   4.603 4.34e-06 ***
education3. Some College    2.423e+01  2.606e+00   9.301  < 2e-16 ***
education4. College Grad    3.974e+01  2.586e+00  15.367  < 2e-16 ***
education5. Advanced Degree 6.485e+01  2.804e+00  23.128  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 35.89 on 2993 degrees of freedom
Multiple R-squared:  0.2619,    Adjusted R-squared:  0.2604
F-statistic:   177 on 6 and 2993 DF,  p-value: < 2.2e-16
```

Hide

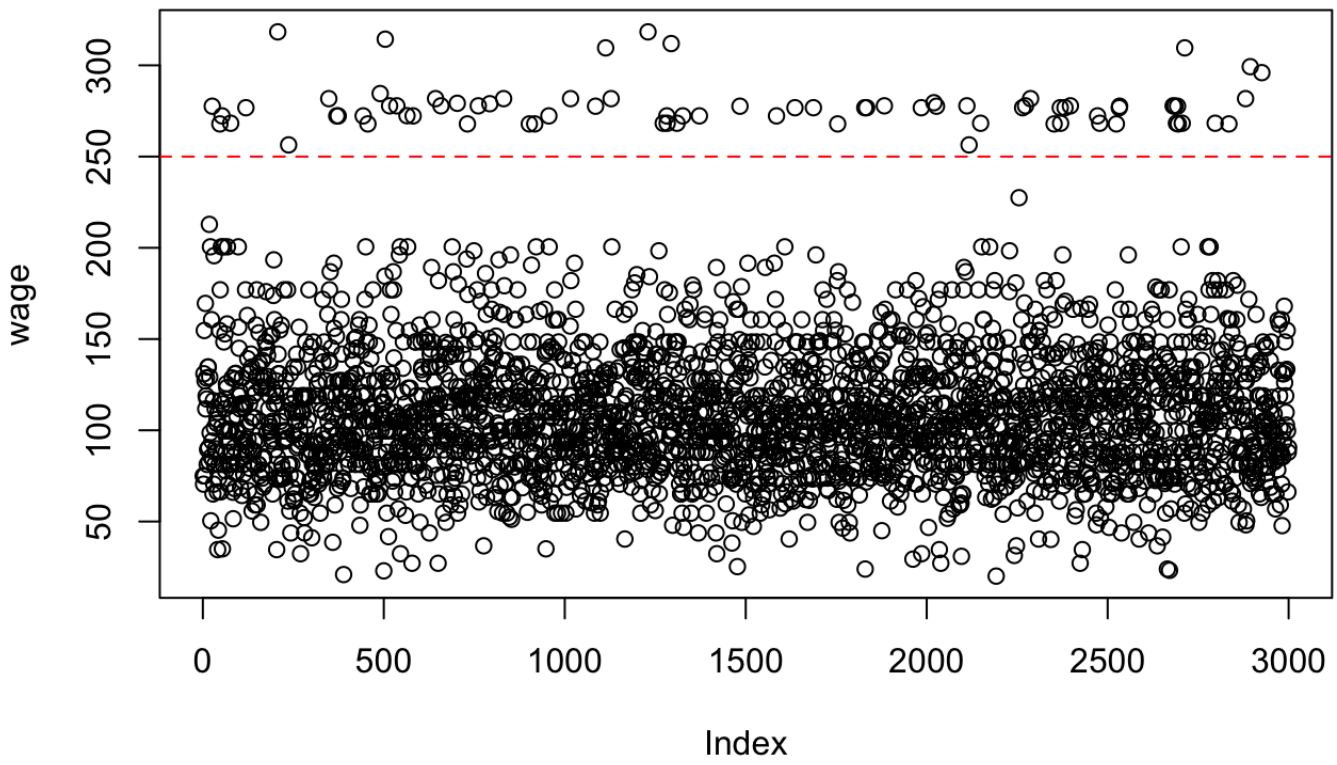
```
plot(wage.LM, which = 1) # ugly !
```



Hide

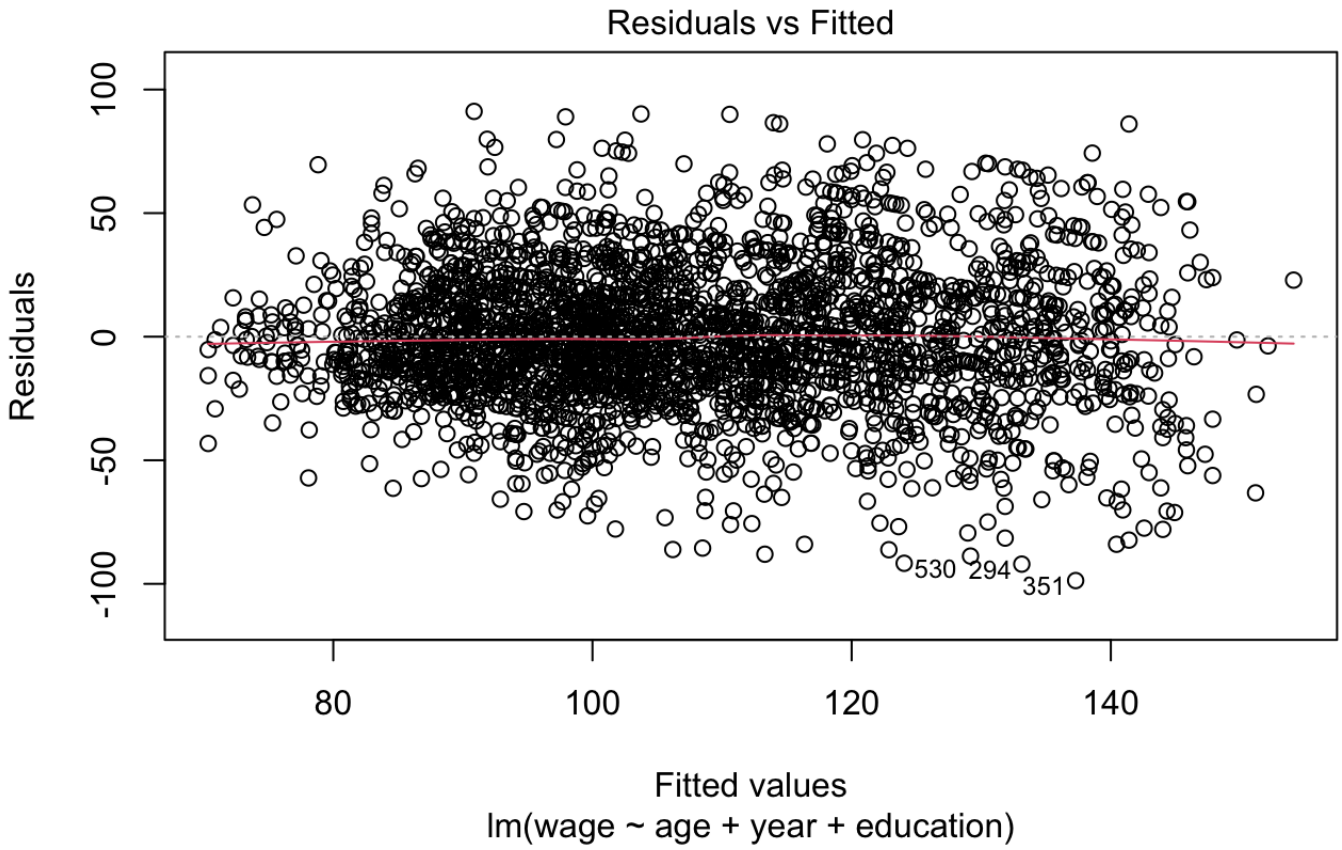
```
plot(wage) + abline(h=250, col="red", lty=2) # treat the high wages as outliers
```

```
integer(0)
```



Hide

```
# splitting the data and investigate them indiviadaully
# to compare the difference between portions of the dataset
wage.l250.df = subset(Wage, wage<250, select = c(wage, age, year, education)) # less tha
n 250
attributes(wage.l250.df)$row.names = 1:nrow(wage.l250.df)
wage.g250.df = subset(Wage, wage>250, select = c(wage, age, year, education)) # greater
 than 250
attributes(wage.g250.df)$row.names = 1:nrow(wage.g250.df)

## the first portion
wage.l250.LM = lm(wage~age+year+education, data=wage.l250.df)
# good model assumptions
plot(wage.l250.LM, which = 1)
```



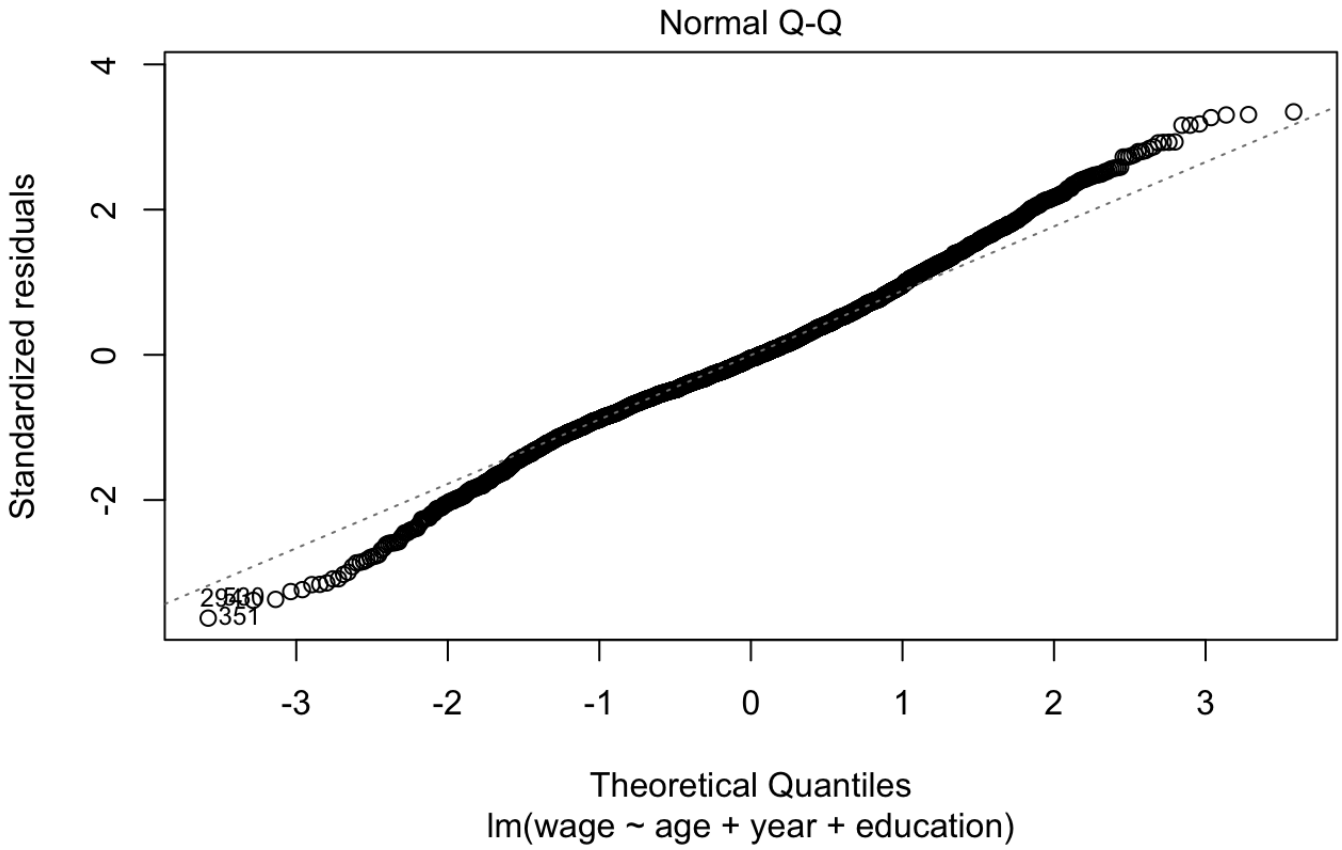
Hide

```
shapiro.test(wage.l250.LM$residuals)
```

```
    Shapiro-Wilk normality test

data:  wage.l250.LM$residuals
W = 0.99288, p-value = 8.495e-11
```

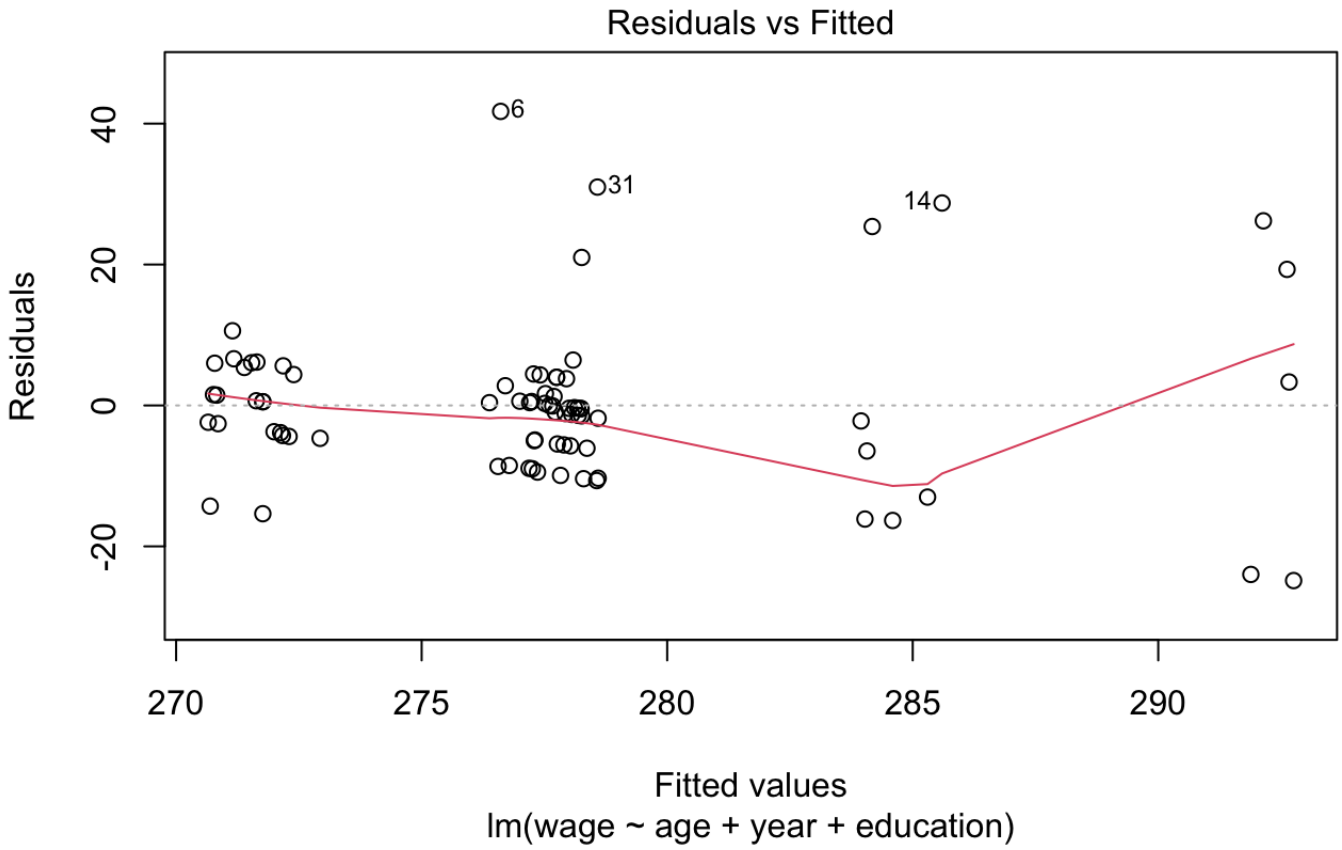
Hide

```
plot(wage.l250.LM, which = 2)
```



Hide

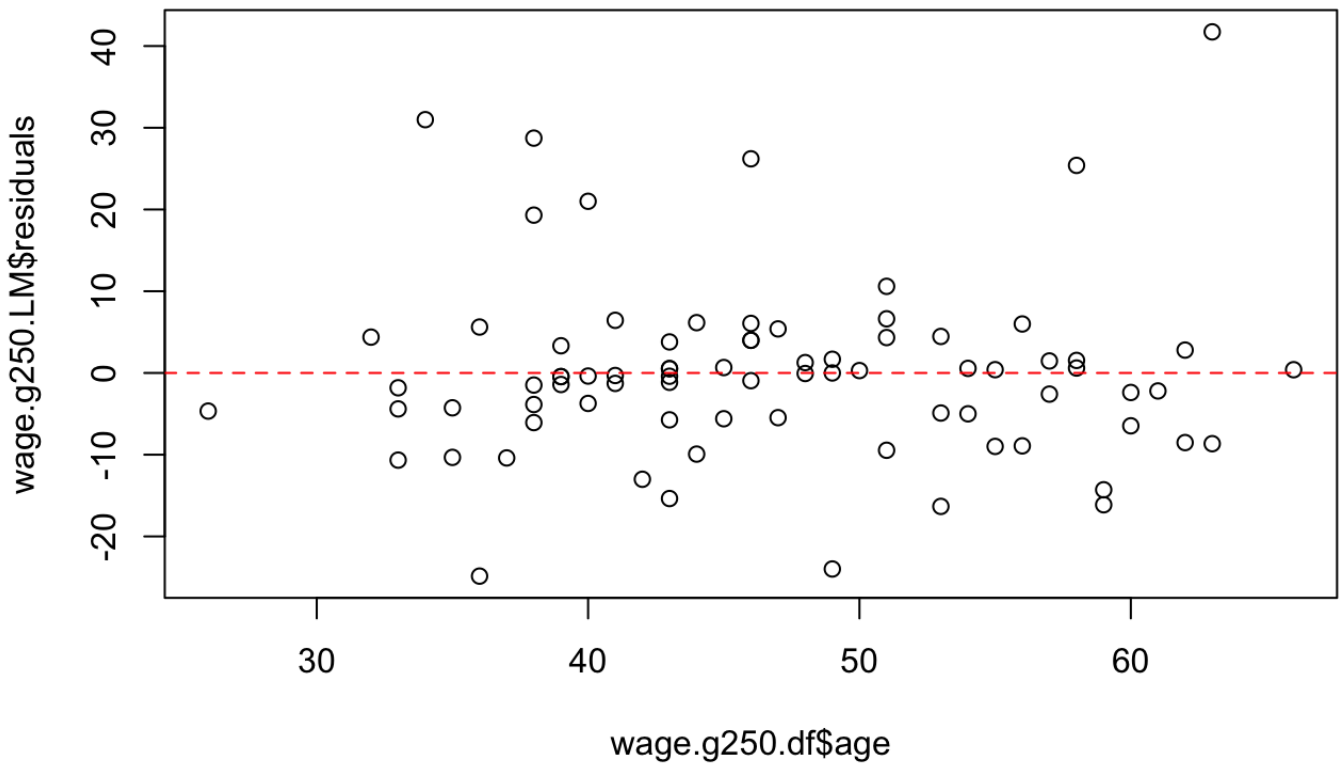
```
## the second portion
wage.g250.LM = lm(wage~age+year+education, data=wage.g250.df)
plot(wage.g250.LM, which = 1)
```



Hide

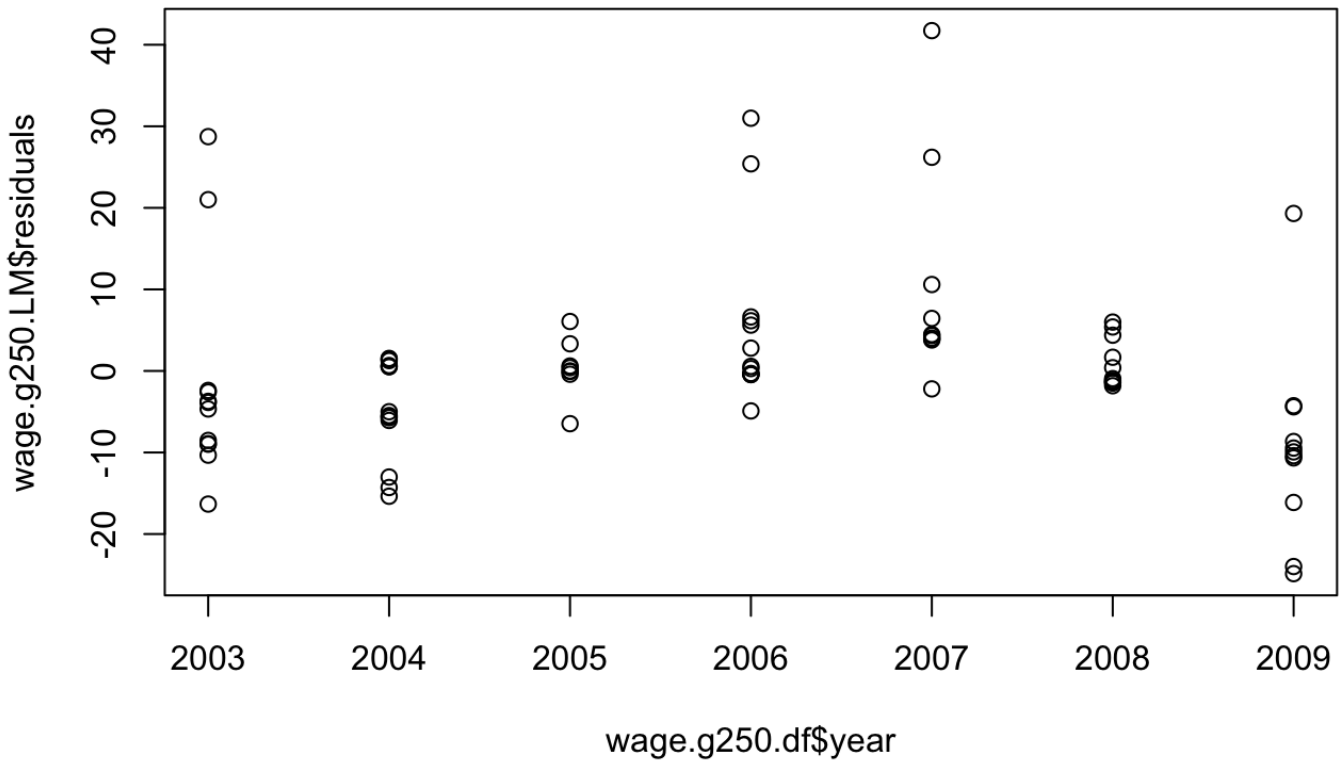
```
plot(wage.g250.df$age, wage.g250.LM$residuals)  + abline(h=0, col="red", lty=2)
```

```
integer(0)
```



Hide

```
plot( wage.g250.df$year, wage.g250.LM$residuals) # not good !
```



Hide

```
# use scaled deviance to check goodness of fit
# not nested model, can;t use deviance based test, likelihood test, to select model
summary(wage.inv.GAM)
```

```
Call: gam(formula = wage ~ s(age) + s(year) + education, data = wage.df[-index,
    ])
Deviance Residuals:
    Min      1Q  Median      3Q     Max
-119.57  -19.62   -3.42   14.01  213.24

(Dispersion Parameter for gaussian family taken to be 1238.62)

    Null Deviance: 5189634 on 2969 degrees of freedom
Residual Deviance: 3662600 on 2957 degrees of freedom
AIC: 29595.07

Number of Local Scoring Iterations: NA

Anova for Parametric Effects
            Df  Sum Sq Mean Sq F value    Pr(>F)
s(age)       1  200711  200711 162.044 < 2.2e-16 ***
s(year)      1   22792   22792  18.401 1.847e-05 ***
education    4 1067366  266842 215.435 < 2.2e-16 ***
Residuals 2957 3662600    1239
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
            Npar Df Npar F  Pr(F)
(Intercept)
s(age)            3 42.672 <2e-16 ***
s(year)           3  1.098 0.3486
education
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Hide

```
1 - pchisq(wage.LG.GAM$deviance, wage.LG.GAM$df.residual)
```

```
[1] 1
```

#L22 PCA PCA influenced by the magnitude of each variable (the results obtained when we perform PCA will also depend on whether the variables have been individually scaled)

Hide

```
library(tidyverse)  # data manipulation and visualization
library(gridExtra)  # plot arrangement
data("USArrests")
head(USArrests, 10)
```

	Murder <dbl>	Assault <int>	UrbanPop <int>	Rape <dbl>
Alabama	13.2	236	58	21.2
Alaska	10.0	263	48	44.5
Arizona	8.1	294	80	31.0
Arkansas	8.8	190	50	19.5
California	9.0	276	91	40.6
Colorado	7.9	204	78	38.7
Connecticut	3.3	110	77	11.1
Delaware	5.9	238	72	15.8
Florida	15.4	335	80	31.9
Georgia	17.4	211	60	25.8

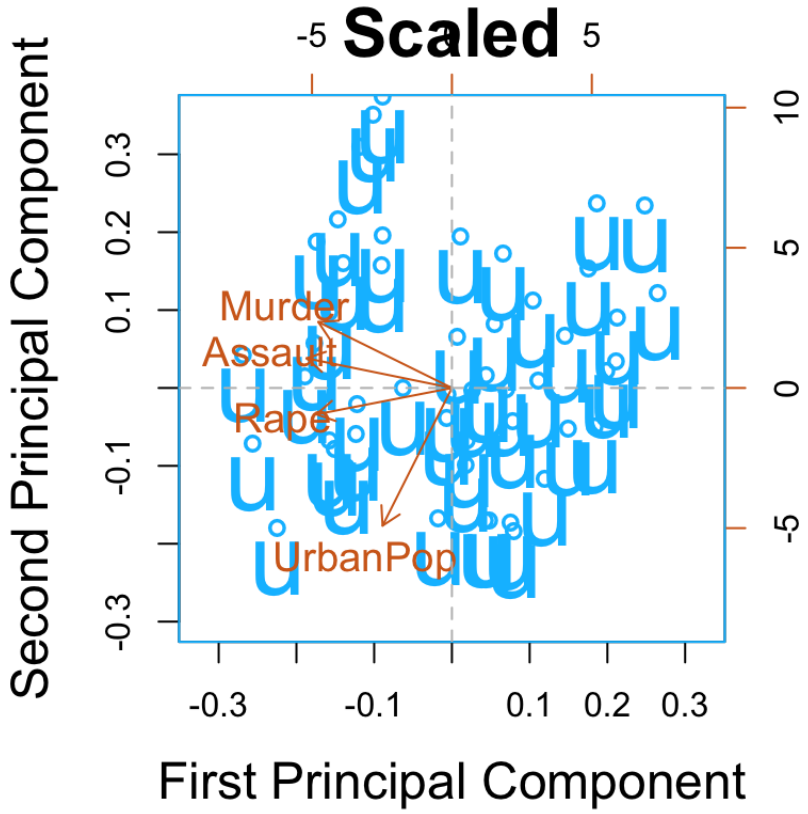
1-10 of 10 rows

Hide

```
USArrests.PCA = prcomp(USArrests, center = TRUE, scale = TRUE)
biplot(USArrests.PCA, main = "Scaled", xlab = "First Principal Component", ylab = "Secon
d Principal Component", asp = TRUE, cex.lab = 1.5, cex.main = 2, cex=c(3, 1.2), xlim = c
(-0.3, 0.3), ylim = c(-0.3, 0.35), col=c("deepskyblue", "chocolate"), xlabs=rep("ů", nro
w(USArrests)))
abline(h = 0, lty = 2, col = "grey")
```

Hide

```
abline(v = 0, lty = 2, col = "grey")
```



Hide

```
USArrests.PCA$center # means
```

```
  Murder  Assault UrbanPop     Rape
   7.788  170.760   65.540   21.232
```

Hide

```
USArrests.PCA$scale # standard deviations
```

```
   Murder   Assault  UrbanPop      Rape
 4.355510 83.337661 14.474763  9.366385
```

Hide

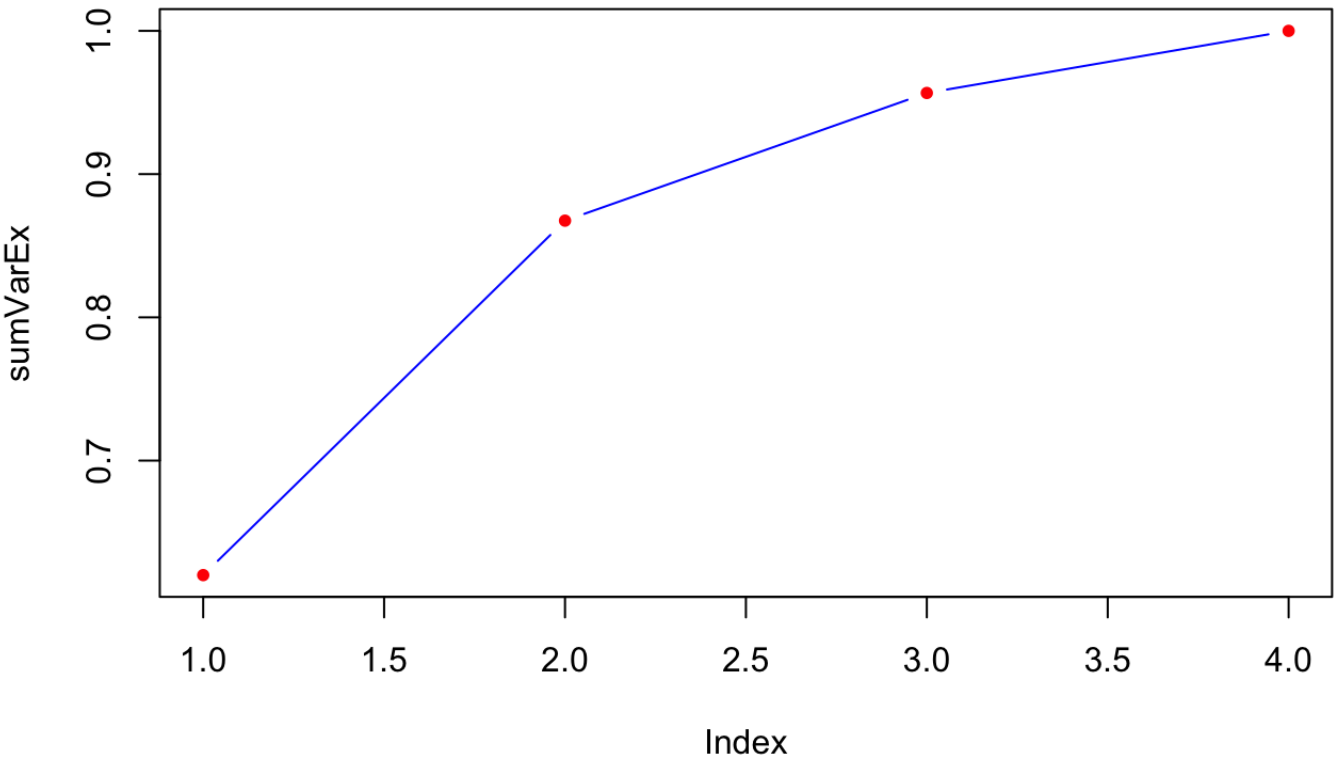
```
USArrests.PCA$sdev  # standard deviation of each principal component
```

```
[1] 1.5748783 0.9948694 0.5971291 0.4164494
```

Hide

```
varEx = USArrests.PCA$sdev^2 # variance explained by each principal component
proVarEx = varEx / sum(varEx)
sumVarEx = cumsum(proVarEx)
plot(sumVarEx, type = 'c', col = 'blue') + points(sumVarEx, pch = 20, col = 'red')
```

```
integer(0)
```



Hide

```
USArrests.PCA$rotation # each column contains the corresponding principal component load
ing vector
```

```
                PC1        PC2        PC3         PC4
Murder   -0.5358995  0.4181809 -0.3412327  0.64922780
Assault  -0.5831836  0.1879856 -0.2681484 -0.74340748
UrbanPop -0.2781909 -0.8728062 -0.3780158  0.13387773
Rape     -0.5434321 -0.1673186  0.8177779  0.08902432
```

Hide

```
phi
```

```
                PC1        PC2
Murder   -0.5358995  0.4181809
Assault  -0.5831836  0.1879856
UrbanPop -0.2781909 -0.8728062
Rape     -0.5434321 -0.1673186
```

L23 Factor Analysis