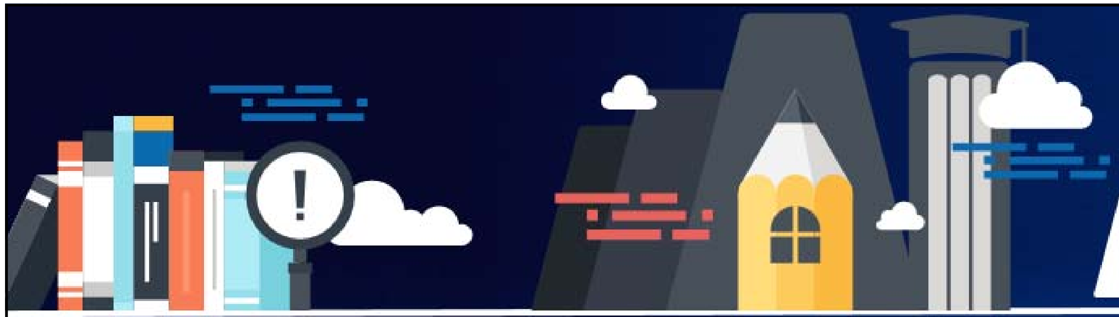
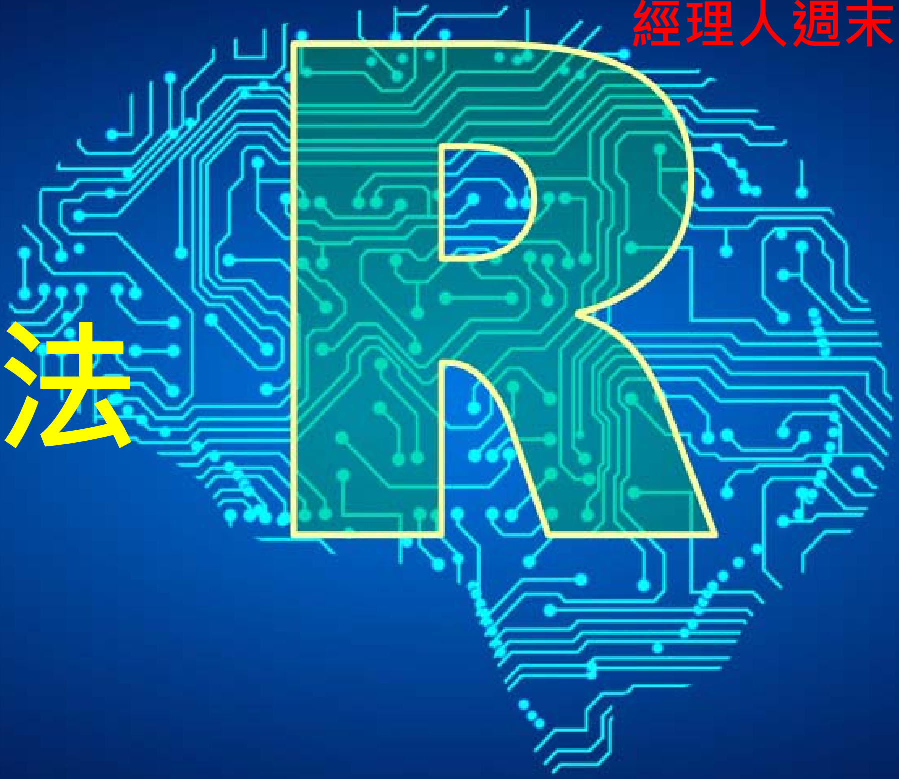
M3



台灣人工智慧學校

經理人週末研修班



資料處理方法

平滑技巧/遺失值處理
資料轉換/重抽法則

吳漢銘
國立臺北大學 統計學系

http://www.hmwu.idv.tw

資料處理方法- 大綱

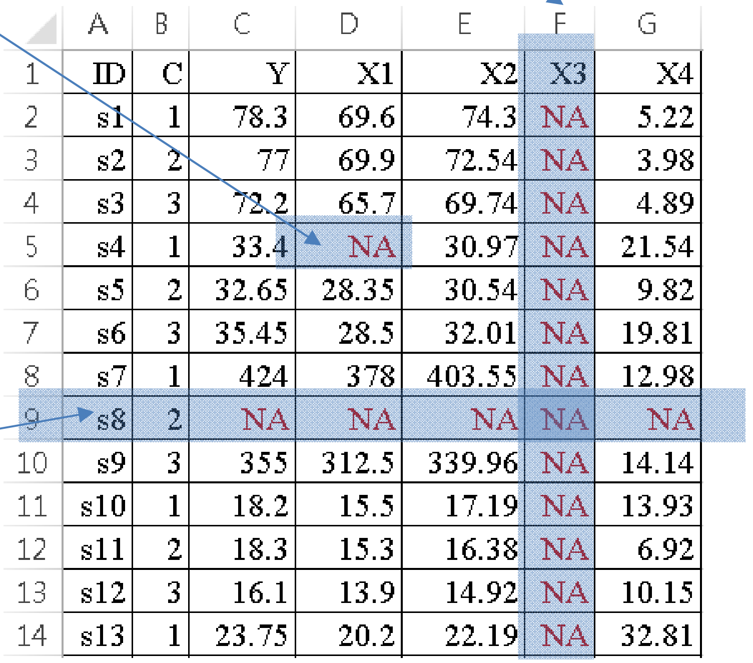
- 主題1 [進階選讀]
  - 移動平均 (Moving Average)
  - 曲線配適 (Fitting Curves): lowess
  - 核密度估計 (Kernel Density Estimation)
  - 三次樣條插值 (Cubic Spline Interpolation)
- 主題2
  - 具缺失值資料 (Missing Data)
  - 缺失機制 (Missingness Mechanism)
    - Missing by Design
    - Missing Completely at Random (MCAR)
    - Missing at Random (MAR)
    - Missing Not at Random (MNAR)
- 主題3
  - R Packages for Dealing With Missing Values: VIM, MICE
  - Visualizing the Pattern of Missing Data
  - Traditional Approaches to Handling Missing Data
  - Imputation Methods: KNN
  - Which Imputation Method?
- 主題4
  - 為什麼要做資料轉換?
  - 常見的資料轉換方式
  - 對數轉換 (Log Transformation)
  - Box-Cox Transformation
  - 標準化 (Standardization)
  - 要使用哪一種資料轉換方式?
- 主題5
  - **Training data and Testing data**
  - **Resampling methods**
    - Jackknife (leave-one-out)
    - Bootstrapping
  - **Ensemble Learning**
    - bagging
    - boosting
- 主題6
  - **Imbalanced Data Problem**
    - under-sampling
    - over-sampling

具缺失值資料 (Missing Data)

When data are missing for a variable for all cases: **latent** or **unobserved**.

Missing data (missing values for certain variables for certain cases): **item non-response**.

When data are missing for all variables for a given case: **unit non-response**.



	A	B	C	D	E	F	G
1	ID	C	Y	X1	X2	X3	X4
2	s1	1	78.3	69.6	74.3	NA	5.22
3	s2	2	77	69.9	72.54	NA	3.98
4	s3	3	72.2	65.7	69.74	NA	4.89
5	s4	1	33.4	NA	30.97	NA	21.54
6	s5	2	32.65	28.35	30.54	NA	9.82
7	s6	3	35.45	28.5	32.01	NA	19.81
8	s7	1	424	378	403.55	NA	12.98
9	s8	2	NA	NA	NA	NA	NA
10	s9	3	355	312.5	339.96	NA	14.14
11	s10	1	18.2	15.5	17.19	NA	13.93
12	s11	2	18.3	15.3	16.38	NA	6.92
13	s12	3	16.1	13.9	14.92	NA	10.15
14	s13	1	23.75	20.2	22.19	NA	32.81

缺失值的處理

- The missing values may give clues to systematic aspects of the problem.

- **How to deal with missing values:**
  - Use a global constant to fill the value will misguide the mining process. (例如: 缺考給0分; 影像訊號=前景-背景)
  - Use the attribute mean or median for all samples belonging to the same class as the given tuple.
  - **補值 (Missing value imputation)** (most popular)

缺失機制
Missingness Mechanism

- The presence of missing data can
  - effect the properties of the estimates
    (e.g. means, percentages, percentiles, variances, ratios, regression parameters, etc.).
  - affect inferences.
    (e.g., the properties of tests and confidence intervals.)
- **The missingness mechanism** (Little and Rubin, 1987)
  - The way in which the probability of an item missing depends on other observed or non-observed variables as well as on its own value.
- It helpful to classify missing values on the basis of the **stochastic mechanism** that produces them.

缺失機制
Missingness Mechanism

collected data

$$\begin{tikzcd} & X = \{X_o, X_m\} \arrow[dl] \arrow[dr] & \\ \text{observed elements} & & \text{missing elements} \end{tikzcd}$$

The missingness indicator matrix R corresponds X,

and each element of R is 1 if the corresponding element of X is missing,

and 0 otherwise.

define the missingness mechanism as

the probability of R conditional on

the values of the observed and missing elements of X:

$$Pr(R|X_o, X_m)$$

Missing by Design
Missing Completely at Random

- Missing by Design
  - **Excluded** some participants from the analysis because they are not part of the population under investigation.
  - missingness codes: (i) refused to answer; (ii) answered don't know; (iii) had a valid skip or (iv) was skipped by an enumerator error.

- Missing Completely at Random (MCAR)
  - missingness is independent of their own unobserved values and the observed data.

$$Pr(R|X) = Pr(R)$$

  - *Example*: Miscoding or forgetting to log in answer.
  - **Imputation methods** rely on the missingness being of the MCAR type.

Missing at Random (MAR) Missing Not at Random (MNAR)

- Missing at Random (MAR) $Pr(R|X) = Pr(R|X_o)$
  - missingness does not depend on their unobserved value but does dependent on the observed data.
  - *Example 1*: male participants (observed data) are more likely to refuse to fill out the **depression survey**, but it does not depend on the level of their depression (unobserved value).
  - *Example 2*: if men are more likely to tell you their weight than women, **weight** is MAR.
  - We can ignore missing data (= omit missing observations) if we have MAR or MCAR.

- Missing Not at Random (MNAR)
  - Missingness that depends on the missing value itself.
  - *Example*: question about **income**, where the high rate of missing values (usually 20%~50%) is related to the value of the income itself (very high and very low values will not be answered).
  - MNAR data is a more serious issue. (not ignorable)

Some Notes

- Assuming data is **MCAR**, too much missing data can be a problem.
  - Usually a safe maximum threshold is 5% of the total for large datasets.
  - If missing data for a certain feature or sample is more than 5% then you probably should leave that feature or sample out.
- If some variable is missing almost 25% of the data points.
  - Consider either dropping it from the analysis or gather more measurements.
  - Keep the other variables are below the 5% threshold.
- For categorical variables, replacing categorical variables is usually not advisable.
- Some common practice include replacing missing categorical variables with the mode of the observed ones (questionable).

Missing Values in R

- **NA**: a missing value ("not available"), **"NA"**: a string.
- **x[1]== NA** is not a valid logical expression and will not return **FALSE** as one would expect but will return **NA**.

```
> myvector <- c(10, 20, NA, 30, 40)
> myvector
[1] 10 20 NA 30 40
> mycountry <- c("Austria", "Australia", NA, NA, "Germany", "NA")
> mycountry
[1] "Austria"   "Australia" NA          NA          "Germany"   "NA"
> is.na(myvector)
[1] FALSE FALSE  TRUE FALSE FALSE
> which(is.na(myvector))
[1] 3
> x <- c(1, 4, 7, 10)
> x[4] <- NA # sets the 4th element to NA
> x
[1]  1  4  7 NA
> is.na(x) <- 1 # sets the first element t
> x
[1] NA  4  7 NA
```

```
> set.seed(12345)
> mydata <- matrix(round(rnorm(20), 2), ncol=5)
> mydata[sample(1:20, 3)] <- NA
> mydata
      [,1]  [,2]  [,3]  [,4]  [,5]
[1,]  0.59  0.61    NA  0.37    NA
[2,]  0.71 -1.82 -0.92  0.52 -0.33
[3,] -0.11  0.63 -0.12 -0.75  1.12
[4,] -0.45 -0.28  1.82    NA  0.30
> which(colSums(is.na(mydata)) > 0)
[1] 3 4 5
```

NOTE: **NULL** denotes something which never existed and cannot exist at all.

NA in Summary Functions

- Most of the statistical summary functions (**`mean`**, **`var`**, **`sum`**, **`min`**, **`max`**, etc.) accept an argument called **`na.rm`**, which can be set to **`TRUE`** if you want missing values to be removed before the summary is calculated. (default : **`FALSE`**)

```
> x <- c(1, 4, NA, 10)
>  summary(x)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
    1.0     2.5     4.0     5.0     7.0    10.0       1 
>  mean(x)
[1] NA
>  sd(x)
[1] NA
>  mean(x, na.rm=TRUE)
[1] 5
>  sd(x, na.rm=TRUE)
[1] 4.582576
> x[!is.na(x)]
[1]  1  4 10
```

NA in Modeling Functions

```
> mydata <- as.data.frame(matrix(sample(1:20, 8), ncol = 2))
> mydata[4, 2] <- NA
> names(mydata) <- c("y", "x")
> mydata
   y  x
1  1 19
2  6 12
3 10  2
4  4 NA
> lm(y~x, data = mydata)

Call:
lm(formula = y ~ x, data = mydata)

Coefficients:
(Intercept)            x
    11.3927      -0.5205

> lm(y~x, data = mydata, na.action = na.omit)

Call:
lm(formula = y ~ x, data = mydata, na.action = na.omit)

Coefficients:
(Intercept)            x
    11.3927      -0.5205

> lm(y~x, data = mydata, na.action = na.fail)
Error in na.fail.default(list(y = c(1L, 6L, 10L, 4L), x = c(19L, 12L,  :
  missing values in object
```

Other Special Values in R

- **NaN** : "not a number" which can arise for example when we try to compute the undeterminate 0/0.

```
> x <- c(1, 0, 10)
> x/x
[1]   1 NaN   1
> is.nan(x/x)
[1] FALSE  TRUE FALSE
```

- **Inf** which results from computations like 1/0.
- Using the functions **is.finite()** and **is.infinite()** we can determine whether a number is finite or not.

```
> 1/x
[1] 1.0 Inf 0.1
> is.finite(1/x)
[1]  TRUE FALSE  TRUE
>
> -10/x
[1]  -10 -Inf   -1
> is.infinite(-10/x)
[1] FALSE  TRUE FALSE
```

```
> exp(-Inf)
[1] 0
> 0/Inf
[1] 0
> Inf - Inf
[1] NaN
> Inf/Inf
[1] NaN
```

R Packages for Dealing With Missing Values

- `Amelia (Amelia II)`: A Program for Missing Data
- `hot.deck:` Multiple Hot-Deck Imputation
- `HotDeckImputation`: Hot Deck Imputation Methods for Missing Data
- `impute`: (Bioconductor) Imputation for Microarray Data
- `mi`: Missing Data Imputation and Model Checking
- `mice`: Multivariate Imputation by Chained Equations
- `missForest:` Nonparametric Missing Value Imputation using Random Forest
- `missMDA`: Handling Missing Values with Multivariate Data Analysis (e.g., imputePCA, imputeMCA,)
- `mitools`: Tools for Multiple Imputation of Missing Data
- `norm:` Analysis of Multivariate Normal Datasets with Missing Values
- `VIM`: Visualization and Imputation of Missing Values
- R packages support for missing values imputation.
  - `Hmisc:` Harrell Miscellaneous
  - `survey`: analysis of complex survey samples
  - `Zelig`: Everyone's Statistical Software
  - `rfImpute{randomForest}`: Imputations by randomForest
  - `imputation{rminer}`: Data Mining Classification and Regression Methods, Missing data imputation (e.g. substitution by value or hotdeck method).
  - `impute.svd{bcv}`: Cross-Validation for the SVD (Bi-Cross-Validation), Missing value imputation via a low-rank SVD approximation estimated by the EM algorithm.
  - `mlr`: Machine Learning in R provides several imputation methods.
    https://mlr-org.github.io/mlr-tutorial/release/html/index.html

Package "`imputation`" was removed from the CRAN. (Archived on 2014-01-14)

R Package: MICE

- mice: Multivariate Imputation by Chained Equations in R by Stef van Buuren.

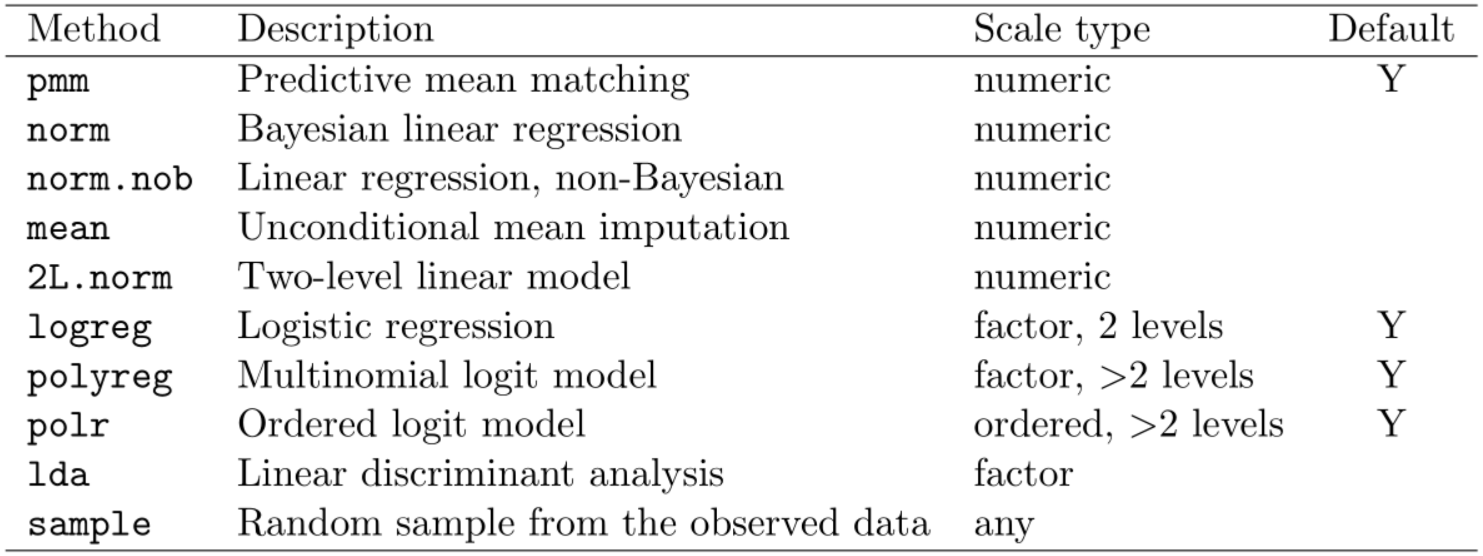
- Imputing missing values on:
  - Continuous data: Predictive mean matching, Bayesian linear regression, Linear regression ignoring model error, Unconditional mean imputation etc.
  - Binary data: Logistic Regression, Logistic regression with bootstrap
  - Categorical data (More than 2 categories) - Polytomous logistic regression, Proportional odds model etc.
  - Mixed data (Can work for both Continuous and Categorical) - CART, Random Forest, Sample (Random sample from the observed values).

Source: http://www.listendata.com/2015/08/missing-imputation-with-mice-package-in.html

Generates Multivariate Imputations by Chained Equations (MICE)

```
mice(data, m = 5, method = vector("character", length = ncol(data)),
  predictorMatrix = (1 - diag(1, ncol(data))),
  visitSequence = (1:ncol(data))[apply(is.na(data), 2, any)],
  form = vector("character", length = ncol(data)),
  post = vector("character", length = ncol(data)), defaultMethod = c("pmm",
  "logreg", "polyreg", "polr"), maxit = 5, diagnostics = TRUE,
  printFlag = TRUE, seed = NA, imputationMethod = NULL,
  defaultImputationMethod = NULL, data.init = NULL, ...)
```

```
> methods(mice)
 [1] mice.impute.2l.norm      mice.impute.2l.pan       mice.impute.2lonly.mean
 [4] mice.impute.2lonly.norm  mice.impute.2lonly.pmm   mice.impute.cart
 [7] mice.impute.fastpmm      mice.impute.lda          mice.impute.logreg
[10] mice.impute
[13] mice.impute
[16] mice.impute
[19] mice.impute
[22] mice.impute
[25] mice.theme
see '?methods' f
> ? mice
```



Method	Description	Scale type	Default
pmm	Predictive mean matching	numeric	Y
norm	Bayesian linear regression	numeric	
norm.nob	Linear regression, non-Bayesian	numeric	
mean	Unconditional mean imputation	numeric	
2L.norm	Two-level linear model	numeric	
logreg	Logistic regression	factor, 2 levels	Y
polyreg	Multinomial logit model	factor, >2 levels	Y
polr	Ordered logit model	ordered, >2 levels	Y
lda	Linear discriminant analysis	factor	
sample	Random sample from the observed data	any	

Exploring Missing Data

```
> head(airquality)
  Ozone Solar.R Wind Temp Month Day
1    41     190  7.4   67     5   1
2    36     118  8.0   72     5   2
3    12     149 12.6   74     5   3
4    18     313 11.5   62     5   4
5    NA      NA 14.3   56     5   5
6    28      NA 14.9   66     5   6
> dim(airquality)
[1] 153   6
> mydata <- airquality
> mydata[4:10, 3] <- rep(NA, 7)
> mydata[1:5, 4] <- NA
>
> # Use numerical variables as examples here.
> # Ozone is the variable with the most missing datapoints.
> summary(mydata)
     Ozone             Solar.R            Wind              Temp            Month            Day
 Min.   :  1.00   Min.   :  7.0   Min.   : 1.700   Min.   :57.00   Min.   :5.000   Min.   : 1.0
 1st Qu.: 18.00   1st Qu.:115.8   1st Qu.: 7.400   1st Qu.:73.00   1st Qu.:6.000   1st Qu.: 8.0
 Median : 31.50   Median :205.0   Median : 9.700   Median :79.00   Median :7.000   Median :16.0
 Mean   : 42.13   Mean   :185.9   Mean   : 9.806   Mean   :78.28   Mean   :6.993   Mean   :15.8
 3rd Qu.: 63.25   3rd Qu.:258.8   3rd Qu.:11.500   3rd Qu.:85.00   3rd Qu.:8.000   3rd Qu.:23.0
 Max.   :168.00   Max.   :334.0   Max.   :20.700   Max.   :97.00   Max.   :9.000   Max.   :31.0
 NA's   :37       NA's   :7       NA's   :7        NA's   :5
```

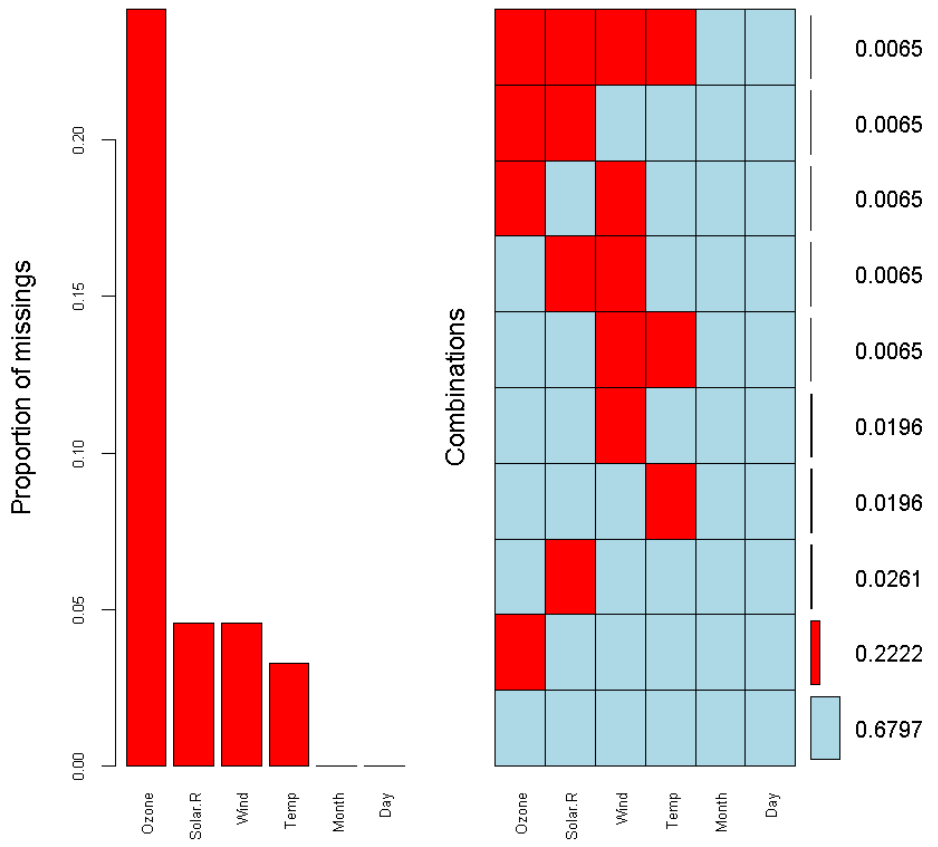
Sourec: http://www.r-bloggers.com/imputing-missing-data-with-r-mice-package/

Visualizing the Pattern of Missing Data

```
> library(mice)
> md.pattern(mydata)
    Month Day Temp Solar.R Wind Ozone
104     1   1    1       1    1     1  0
 34     1   1    1       1    1     0  1
  4     1   1    1       0    1     1  1
  3     1   1    1       1    0     1  1
  3     1   1    0       1    1     1  1
  1     1   1    1       0    1     0  2
  1     1   1    1       1    0     0  2
  1     1   1    1       0    0     1  2
  1     1   1    0       1    0     1  2
  1     1   1    0       0    0     0  4
        0   0    5       7    7    37 56
```

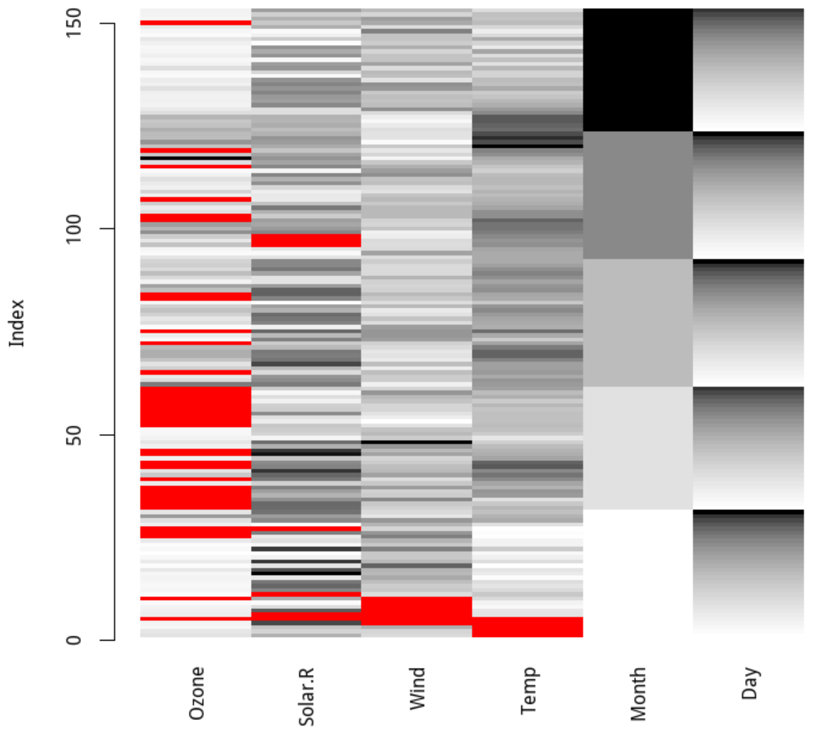
```
> library(VIM)
> mydata.aggrplot <- aggr(mydata,
col=c('lightblue','red'), numbers=TRUE,
prop = TRUE, sortVars=TRUE,
labels=names(mydata), cex.axis=.7, gap=3)

 Variables sorted by number of missings:
 Variable      Count
    Ozone 0.24183007
  Solar.R 0.04575163
     Wind 0.04575163
     Temp 0.03267974
    Month 0.00000000
      Day 0.00000000
```

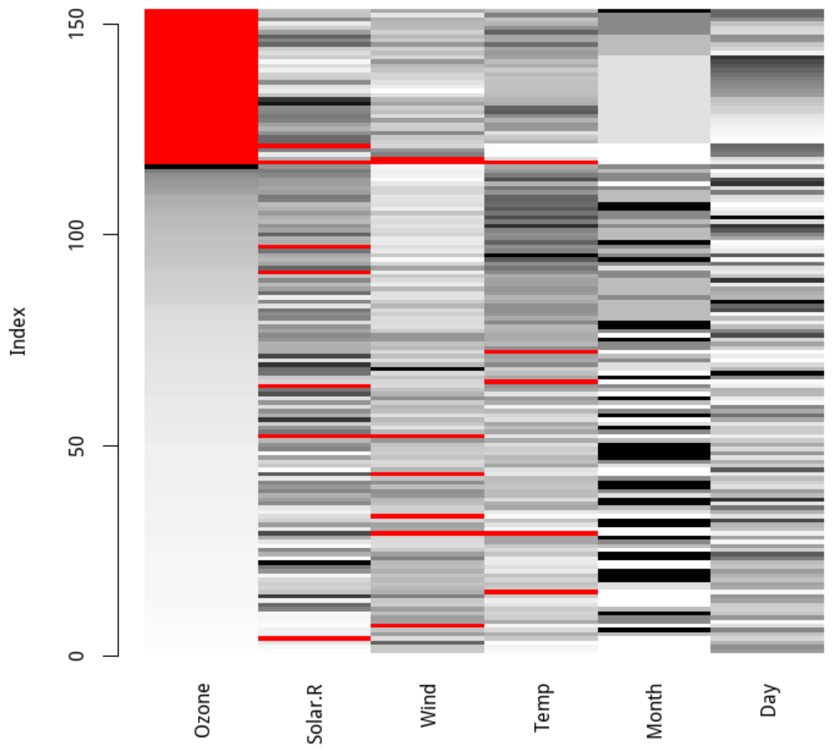
Aggregation Plot



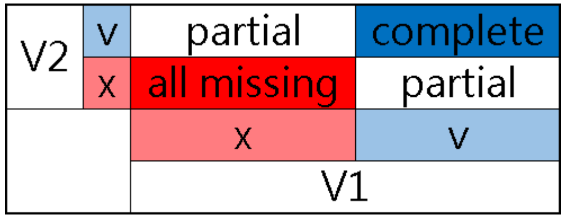
Matrix Plot

```
> matrixplot(mydata)

Click in a column to sort by the corresponding variable.
To regain use of the VIM GUI and the R console, click outside the plot region.

Matrix plot sorted by variable 'Ozone'.
```





Number of Observations Per Patterns for All Pairs of Variables



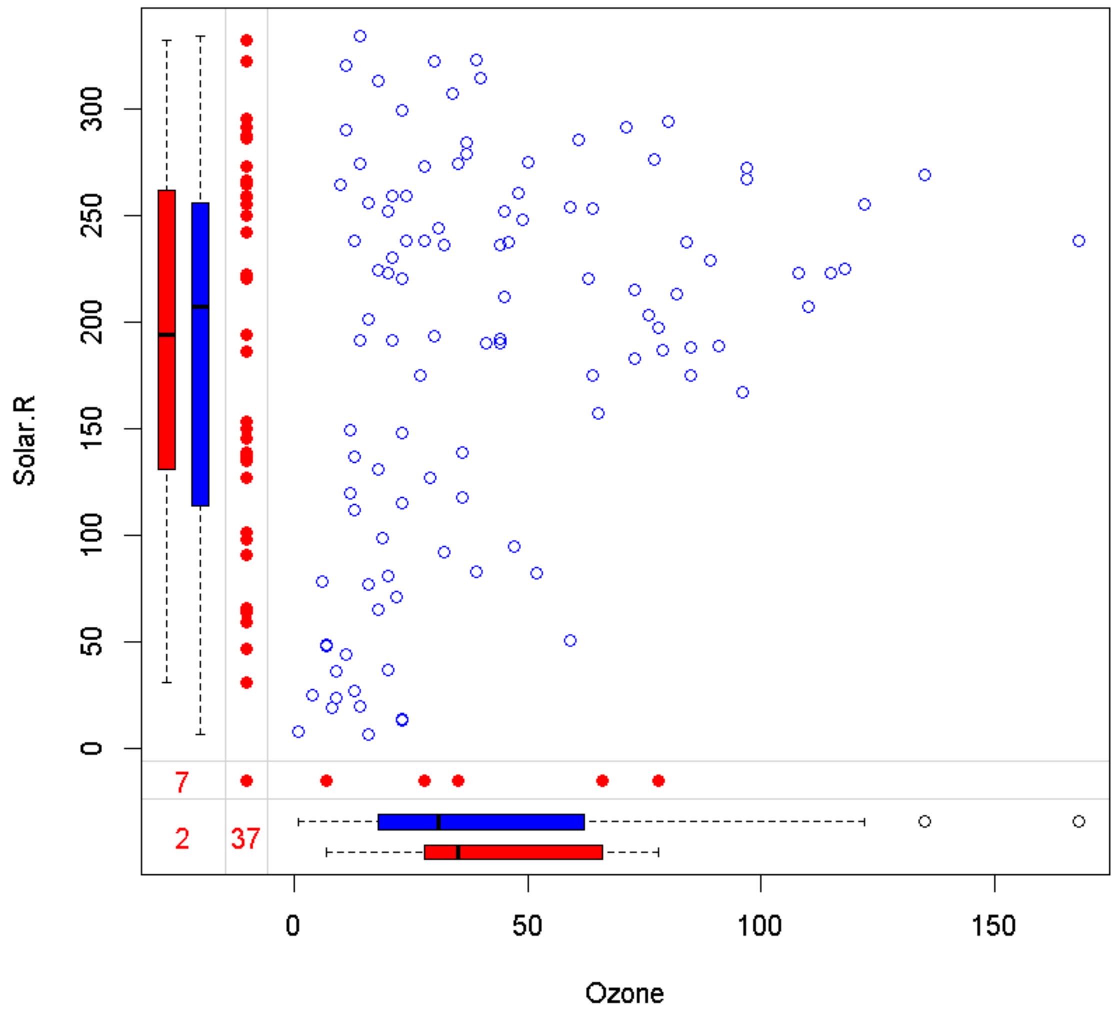
- `rr`: response-response, both variables are observed
- `rm`: response-missing, row observed, column missing
- `mr`: missing-response, row missing, column observed
- `mm`: missing-missing, both variables are missing

```
> md.pairs(mydata)
$rr
        Ozone Solar.R Wind Temp Month Day
Ozone     116     111  111  112   116 116
Solar.R   111     146  141  142   146 146
Wind      111     141  146  143   146 146
Temp      112     142  143  148   148 148
Month     116     146  146  148   153 153
Day       116     146  146  148   153 153

$rm
        Ozone Solar.R Wind Temp Month Day
Ozone       0       5    5    4     0   0
Solar.R    35       0    5    4     0   0
Wind       35       5    0    3     0   0
Temp       36       6    5    0     0   0
Month      37       7    7    5     0   0
Day        37       7    7    5     0   0
```

```
$mr
        Ozone Solar.R Wind Temp Month Day
Ozone       0      35   35   36    37  37
Solar.R     5       0    5    6     7   7
Wind        5       5    0    5     7   7
Temp        4       4    3    0     5   5
Month       0       0    0    0     0   0
Day         0       0    0    0     0   0

$mm
        Ozone Solar.R Wind Temp Month Day
Ozone      37       2    2    1     0   0
Solar.R     2       7    2    1     0   0
Wind        2       2    7    2     0   0
Temp        1       1    2    5     0   0
Month       0       0    0    0     0   0
Day         0       0    0    0     0   0
```

Marginplot

```
> marginplot(mydata[,c("Ozone", "Solar.R")], col = c("blue", "red"))
```



- The blue box plot located on the left and bottom margins shows the distribution of the non-missing datapoints.
- The red box plot on the left shows the distribution of Solar.R with Ozone missing.
- If our assumption of MCAR data is correct, then we expect the red and blue box plots to be very similar.

List-wise Deletion

- Also called the **complete case analysis**.
- The use of this method is only justified if the missing data generation mechanism is **MCAR**.

```
> mdata <- matrix(rnorm(15), nrow=5)
> mdata[sample(1:15, 4)] <- NA
> mdata <- as.data.frame(mdata)
> mdata
          V1         V2          V3
1 -0.62222501  1.0807983          NA
2  0.07124865  0.5216675 -0.08334454
3  1.70707399  0.1004917  0.88197789
4          NA -0.6595201 -0.08387860
5          NA  1.6138847          NA
> (x1 <- na.omit(mdata))
          V1        V2          V3
2 0.07124865 0.5216675 -0.08334454
3 1.70707399 0.1004917  0.88197789
> (x2 <- mdata[complete.cases(mdata),])
          V1        V2          V3
2 0.07124865 0.5216675 -0.08334454
3 1.70707399 0.1004917  0.88197789
```

```
> mdata[!complete.cases(mdata),]
         V1         V2         V3
1 -0.622225  1.0807983         NA
4        NA -0.6595201 -0.0838786
5        NA  1.6138847         NA
```

快速分析一下，得知資料大概狀況

Pairwise Deletion

- To compute a covariance matrix, each two cases will be used for which the values of both corresponding variables are available.
- This can result in covariance or correlation matrices which are not positive semi-definite, as well as NA entries if there are no complete pairs for the given pair of variables.

```
> mdata
           V1        V2          V3
1 -0.62222501 1.0807983          NA
2  0.07124865 0.5216675 -0.08334454
3  1.70707399 0.1004917  0.88197789
4          NA -0.6595201 -0.08387860
5          NA 1.6138847          NA
> cov(mdata)
   V1        V2 V3
V1 NA        NA NA
V2 NA 0.7694197 NA
V3 NA        NA NA
> cov(mdata, use = "all.obs")
Error in cov(mdata, use = "all.obs") :
missing observations in cov/cor
> cov(mdata, use = "complete.obs")
           V1          V2         V3
V1  1.3379623 -0.34448500  0.7895494
V2 -0.3444850  0.08869452 -0.2032852
V3  0.7895494 -0.20328521  0.4659237
```

```
> cov(mdata, use = "na.or.complete")
           V1          V2         V3
V1  1.3379623 -0.34448500  0.7895494
V2 -0.3444850  0.08869452 -0.2032852
V3  0.7895494 -0.20328521  0.4659237
> cov(mdata, use = "pairwise")
           V1          V2         V3
V1  1.4304107 -0.56002326 0.78954945
V2 -0.5600233  0.76941970 0.05468712
V3  0.7895494  0.05468712 0.31078774
```

Mean Substitution

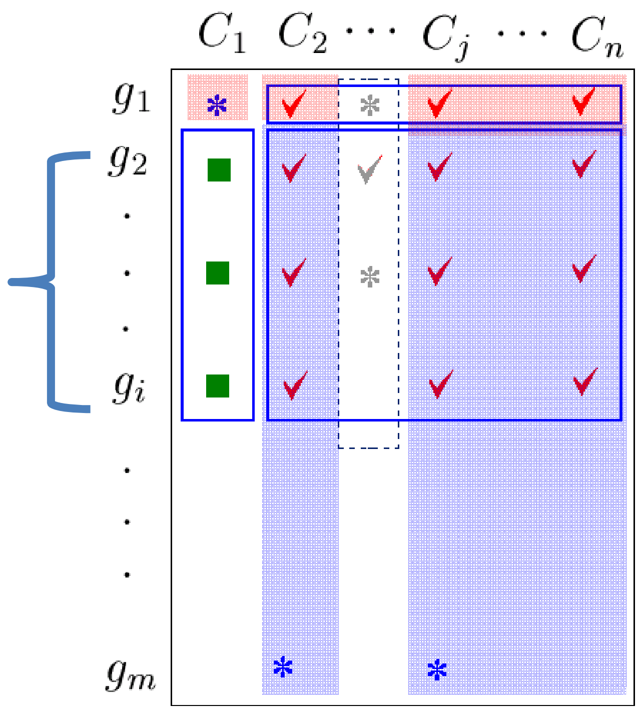
- A very simple but popular approach is to substitute means for the missing values.
- This method produces biased estimates and can severely distort the distribution of the variable in which missing values are substituted.
- Due to these **distributional problems**, it is often recommended to ignore missing values rather than impute values by mean substitution (Little and Rubin, 1989.)

```
mean.subst <- function(x) {
   x[is.na(x)] <- mean(x, na.rm = TRUE)
   x
}
```

```
> mdata
           V1         V2          V3
1 -0.62222501  1.0807983          NA
2  0.07124865  0.5216675 -0.08334454
3  1.70707399  0.1004917  0.88197789
4          NA -0.6595201 -0.08387860
5          NA  1.6138847          NA
> mdata.mip <- apply(mdata, 2, mean.subst)
> mdata.mip
              V1         V2          V3
[1,] -0.62222501  1.0807983  0.23825158
[2,]  0.07124865  0.5216675 -0.08334454
[3,]  1.70707399  0.1004917  0.88197789
[4,]  0.38536588 -0.6595201 -0.08387860
[5,]  0.38536588  1.6138847  0.23825158
```

K-Nearest Neighbour Imputation

- KNN imputation searches for the k-nearest observations (respective to the observation which has to be imputed) and replaces the missing value with the mean of the found *k* observations.
- It is recommended to use the (weighted) median instead of the arithmetic mean.
- **KNN** minimize data modeling assumptions and take advantage of the correlation structure of the data.



KNNimpute

Model:

$$\{g_{(k)}, k = 1, 2, \cdots, K\} = \underset{k}{\text{args}} \ \max_{i \in \mathrm{C}} \mathrm{Corr}(g_1, g_i)$$

$$\{g_{(k)}, k = 1, 2, \cdots, K\} = \underset{k}{\text{args}} \ \min_{i \in \mathrm{C}} \mathrm{Dist}(g_1, g_i)$$

C: Observed C_i's without missing values

Imputation:

Average $\widehat{C_1(g_1)} = \frac{1}{K}\sum_{k=1}^{K} C_1(g_k)$

Weighted Average $\widehat{C_1(g_1)} = \frac{\sum_{k=1}^{K} w_k C_1(g_k)}{\sum_{k=1}^{K} w_k}$

$$w_k = \frac{1}{\sum_{j \in \mathrm{C}} [C_j(g_k) - C_1(g_1)]^2}$$

kNN {VIM}:
k-Nearest Neighbour Imputation

Description

k-Nearest Neighbour Imputation based on a variation of the Gower Distance for numerical, categorical, ordered and semi-continous variables.

Usage

```
kNN(data, variable = colnames(data), metric = NULL, k = 5,
  dist_var = colnames(data), weights = NULL, numFun = median,
  catFun = maxCat, makeNA = NULL, NAcond = NULL, impNA = TRUE,
  donorcond = NULL, mixed = vector(), mixed.constant = NULL,
  trace = FALSE, imp_var = TRUE, imp_suffix = "imp", addRandom = FALSE,
  useImputedDist = TRUE, weightDist = FALSE)
```

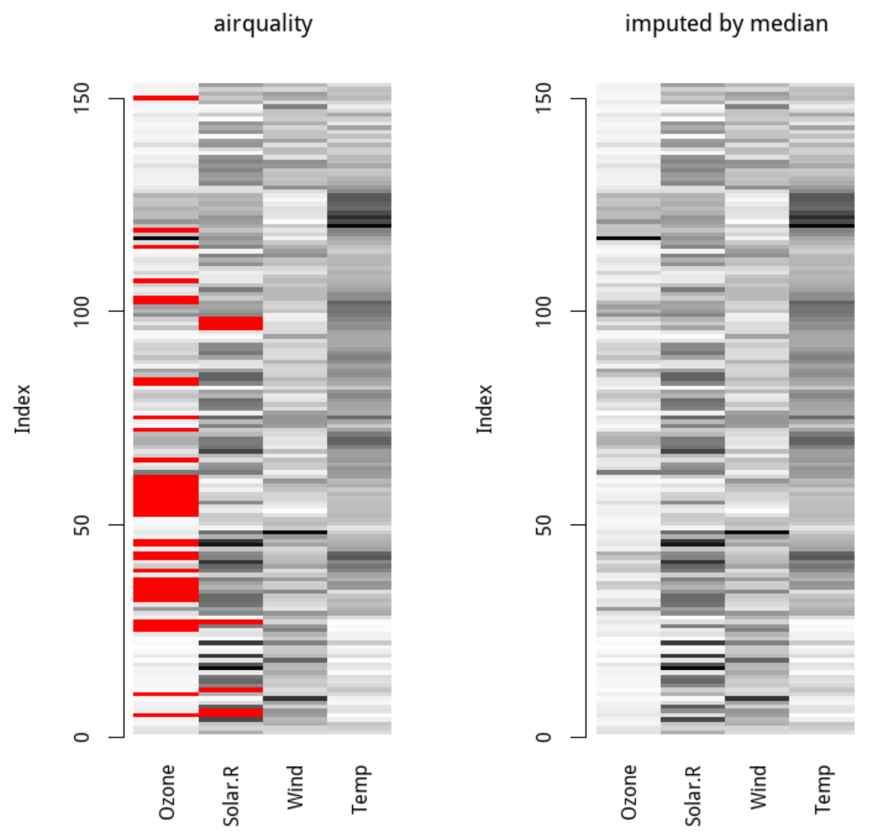
```
mean
weightedMean
```

```
> names(airquality)
[1] "Ozone"   "Solar.R" "Wind"    "Temp"    "Month"   "Day"
> airquality.imp.median <- kNN(airquality[1:4], k=5)
> head(airquality.imp.median)
  Ozone Solar.R Wind Temp Ozone_imp Solar.R_imp Wind_imp Temp_imp
1    41     190  7.4   67     FALSE       FALSE    FALSE    FALSE
2    36     118  8.0   72     FALSE       FALSE    FALSE    FALSE
3    12     149 12.6   74     FALSE       FALSE    FALSE    FALSE
4    18     313 11.5   62     FALSE       FALSE    FALSE    FALSE
5    35      92 14.3   56      TRUE        TRUE    FALSE    FALSE
6    28     242 14.9   66     FALSE        TRUE    FALSE    FALSE
```

- Gower JC, 1971, A General Coefficient of Similarity and Some of Its Properties. Biometrics, 857–871.
- Alexander Kowarik and Matthias Templ, 2016, Imputation with the R Package VIM, Journal of Statistical Software, Volume 74, Issue 7.

matrixplot、自定平均函數

```
> matrixplot(airquality[1:4], interactive = F, main="airquality")
> matrixplot(airquality.imp.median[1:4], interactive = F, main="imputed by median")
```



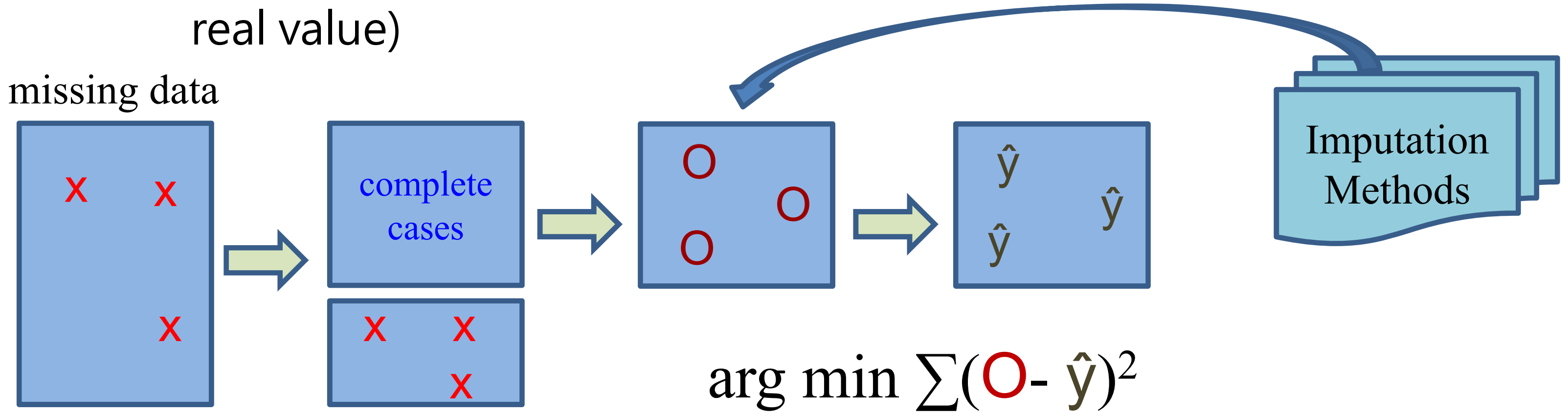
自定平均函數

```
trim_mean <- function(x){
   mean(x, trim = 0.1)
}
```

```
> airquality.imp.tmean <- kNN(airquality[1:4], k=5, numFun=trim_mean)
```

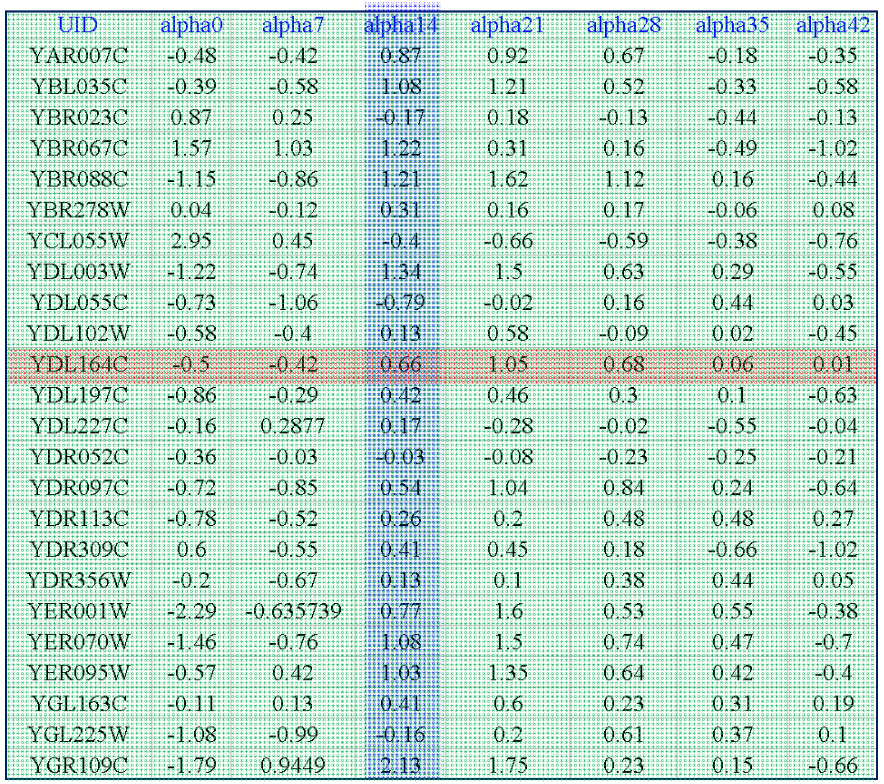
Which Imputation Method?

- KNN is the most widely-used.
- Characteristics of data that may affect choice of imputation method:
  - dimensionality.
  - percentage of values missing.
  - experimental design (time series, case/control, etc.)
  - patterns of correlation in data.
- Suggestion:
  - add (**same percentage**) artificial missing values to your (**complete cases**) data set.
  - impute them with various methods, see which is best (since you know the real value)



Classical (Numerical) Data Table

*j*th variable



UID	alpha0	alpha7	alpha14	alpha21	alpha28	alpha35	alpha42
YAR007C	-0.48	-0.42	0.87	0.92	0.67	-0.18	-0.35
YBL035C	-0.39	-0.58	1.08	1.21	0.52	-0.33	-0.58
YBR023C	0.87	0.25	-0.17	0.18	-0.13	-0.44	-0.13
YBR067C	1.57	1.03	1.22	0.31	0.16	-0.49	-1.02
YBR088C	-1.15	-0.86	1.21	1.62	1.12	0.16	-0.44
YBR278W	0.04	-0.12	0.31	0.16	0.17	-0.06	0.08
YCL055W	2.95	0.45	-0.4	-0.66	-0.59	-0.38	-0.76
YDL003W	-1.22	-0.74	1.34	1.5	0.63	0.29	-0.55
YDL055C	-0.73	-1.06	-0.79	-0.02	0.16	0.44	0.03
YDL102W	-0.58	-0.4	0.13	0.58	-0.09	0.02	-0.45
YDL164C	-0.5	-0.42	0.66	1.05	0.68	0.06	0.01
YDL197C	-0.86	-0.29	0.42	0.46	0.3	0.1	-0.63
YDL227C	-0.16	0.2877	0.17	-0.28	-0.02	-0.55	-0.04
YDR052C	-0.36	-0.03	-0.03	-0.08	-0.23	-0.25	-0.21
YDR097C	-0.72	-0.85	0.54	1.04	0.84	0.24	-0.64
YDR113C	-0.78	-0.52	0.26	0.2	0.48	0.48	0.27
YDR309C	0.6	-0.55	0.41	0.45	0.18	-0.66	-1.02
YDR356W	-0.2	-0.67	0.13	0.1	0.38	0.44	0.05
YER001W	-2.29	-0.635739	0.77	1.6	0.53	0.55	-0.38
YER070W	-1.46	-0.76	1.08	1.5	0.74	0.47	-0.7
YER095W	-0.57	0.42	1.03	1.35	0.64	0.42	-0.4
YGL163C	-0.11	0.13	0.41	0.6	0.23	0.31	0.19
YGL225W	-1.08	-0.99	-0.16	0.2	0.61	0.37	0.1
YGR109C	-1.79	0.9449	2.13	1.75	0.23	0.15	-0.66

*i*th subject
(*i*th sample)

transformation
for each row

transformation
for each column

transformation
for both rows
and columns

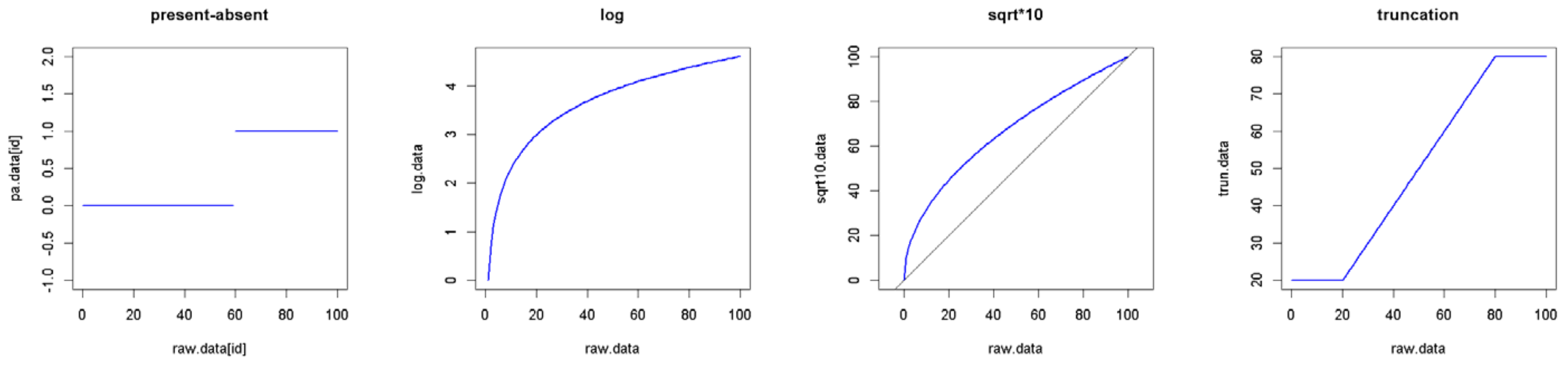
為什麼要做資料轉換?

- to make it more closely **the assumptions** of a statistical inference procedure,
- to make it **easier to visualize** (appearance of graphs),
- to improve **interpretability**,
- to make descriptors that have been measured in **different units comparable**,
- to make the relationships among **variables linear**,
- to modify the **weights** of the variables or objects (e.g. give the same length (or norm) to all object vectors)
- to **code** categorical variables into dummy binary variables.

常見的資料轉換方式

```
par(mfrow=c(1,4))
raw.data <- 0:100
pa.data <- ifelse(raw.data >= 60, 1, 0)
id <- which(pa.data==1)
plot(raw.data[id], pa.data[id], main="present-absent",
+ type="l", lwd=2, col="blue", ylim=c(-1, 2), xlim=c(0, 100))
points(raw.data[-id], pa.data[-id], type="l", lwd=2, col="blue")

log.data <- log(raw.data)
plot(raw.data, log.data, main="log", type="l", lwd=2, col="blue")

sqrt10.data <- sqrt(raw.data)*10
plot(raw.data, sqrt10.data, main="sqrt*10", type="l", lwd=2, col="blue", asp=1)
abline(a=0, b=1)

trun.data <- ifelse(raw.data >= 80, 80, ifelse(raw.data < 20, 20, raw.data))
plot(raw.data, trun.data, main="truncation", type="l", lwd=2, col="blue")
```

NOTE:
```
apply(raw.data.matrix, 2, log)
apply(raw.data.matrix, 2, function(x) sqrt(x)*10)
apply(raw.data.matrix, 2, myfun)
```



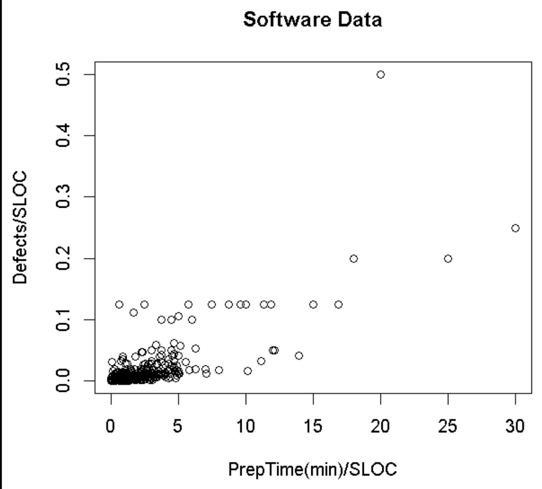
範例: Software Inspection Data

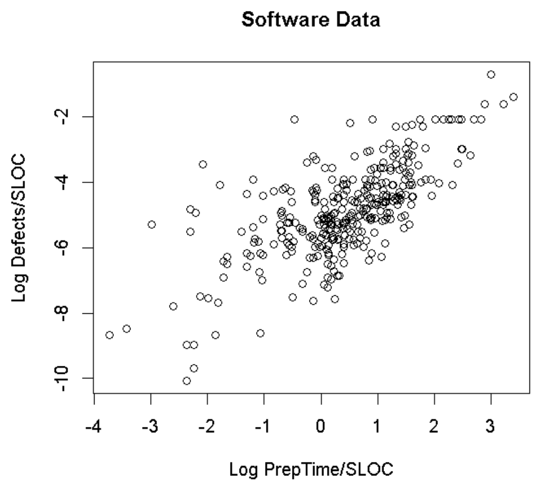
- The data were collected in response to efforts for process improvement in software testing by code inspection.
- The variables are normalized by the size of the inspection (the number of pages or SLOC (single lines of code)):
  - the preparation time in minutes (**`prepage, prepsloc`**),
  - the total work hours in minutes for the meeting (**`mtgsloc`**),
  - and the number of defects found (**`defpage`**, **`defsloc`**).

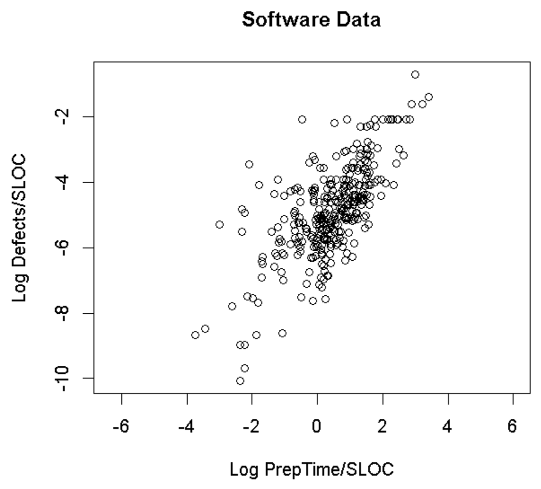
```
> library('R.matlab')
> data <- readMat("software.mat")
> print(data)
...
> str(data)
List of 5
 $ prepsloc: num [1:426, 1] 0.485 0.54 0.54 0.311 0.438 ...
 $ defsloc : num [1:426, 1] 0.005 0.002 0.002 0.00328 0.00278 ...
 $ mtgsloc : num [1:426, 1] 0.075 0.06 0.06 0.2787 0.0417 ...
 $ prepage : num [1:491, 1] 6.15 1.47 1.47 5.06 5.06 ...
 $ defpage : num [1:491, 1] 0.0385 0.0267 0.0133 0.0128 0.0385 ...
```

- **Interested in**: understanding the relationship between the inspection time and the number of defects found.

對數轉換 (Log Transformation)





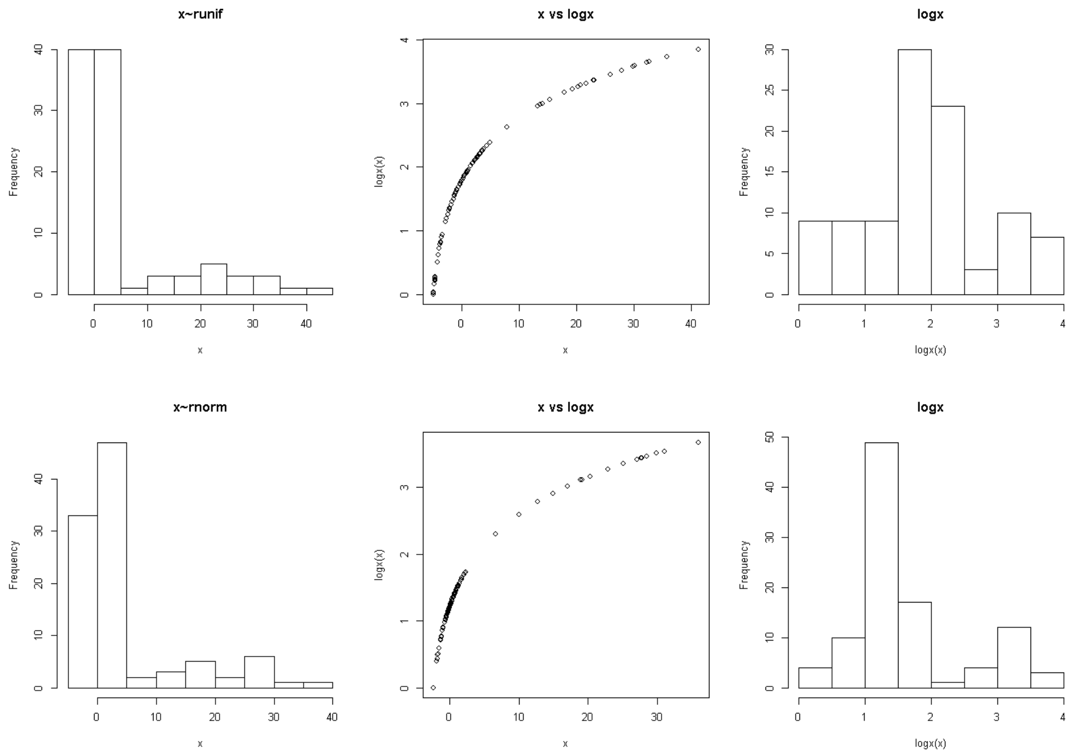


```
plot(data$prepsloc, data$defsloc, xlab="PrepTime(min)/SLOC", ylab="Defects/SLOC",
main="Software Data")

plot(log(data$prepsloc), log(data$defsloc), xlab="Log PrepTime/SLOC",
ylab="Log Defects/SLOC", main="Software Data")

plot(log(data$prepsloc), log(data$defsloc), xlab="Log PrepTime/SLOC",
ylab="Log Defects/SLOC", main="Software Data", asp=1)
```

對數轉換: How to Handle Negative Data Values?

- Solution 1: Translate, then Transform
  - log(x + 1 - min(x))

```
logx <- function(x){
   log(x + 1 - min(x))
}

x <- runif(80, min = -5, max = 5)
# x <- rnorm(80)
x <- c(x, rnorm(20, mean=20, sd=10))
par(mfrow=c(1, 3))
hist(x, main="x~runif")
plot(x, logx(x), main="x vs logx")
hist(logx(x), main="logx")
```



- Solution 2: Use Missing Values
  - A criticism of the previous method is that some practicing statisticians don't like to add an arbitrary constant to the data.
  - They argue that a better way to handle negative values is to use missing values for the logarithm of a nonpositive number.

Box-Cox Transformations

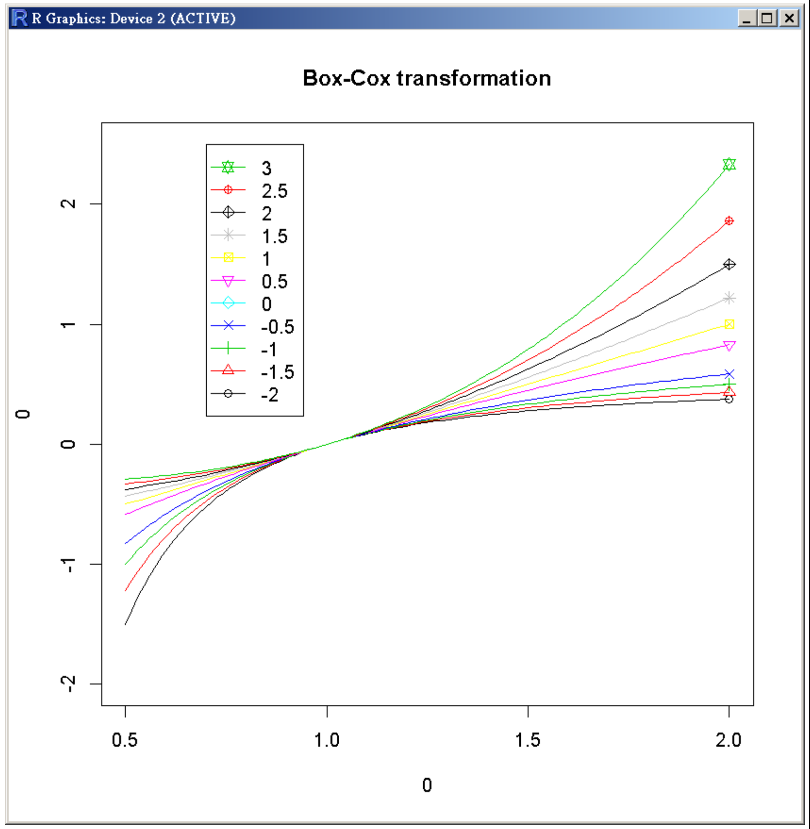
$$y(\lambda) = \begin{cases} \frac{y^\lambda - 1}{\lambda}, & \text{if } \lambda \neq 0; \\ \log y, & \text{if } \lambda = 0. \end{cases}$$

Box and Cox(1964)

- The aim of the Box-Cox transformations is to ensure the usual assumptions for Linear Model hold.

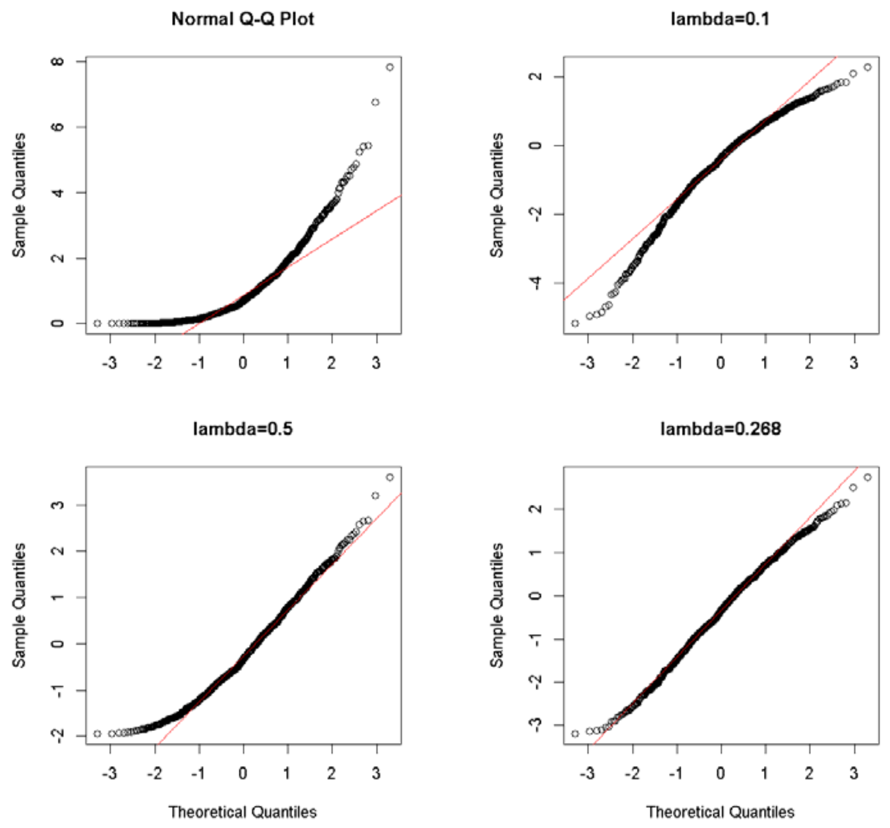
$$\boldsymbol{y} \sim \mathrm{N}(\mathbf{X}\boldsymbol{\beta}, \sigma^2 \mathbf{I}_n)$$

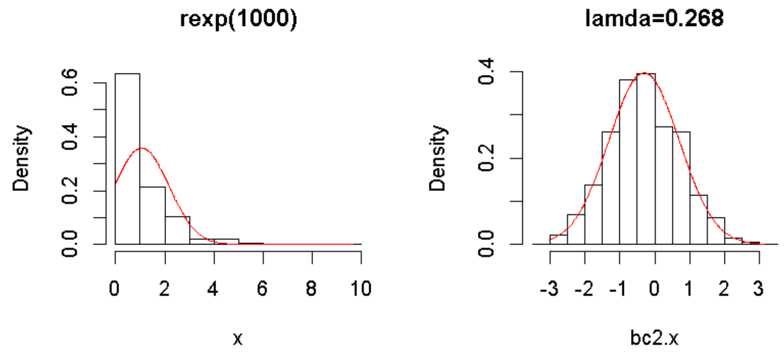
```
x <- seq(0.5, 2, length.out=100)
bc <- function(y, lambda){
    (y^lambda -1)/lambda
}
lambda <- seq(-2, 3, 0.5)
plot(0, 0, type="n", xlim=c(0.5, 2),
     ylim=c(-2, 2.5), main="Box-Cox transformation")
for(i in 1:length(lambda)){
   points(x, bc(x, lambda[i]), type="l", col=i)
   points(2, bc(2, lambda[i]), col=i, pch=i)
}
legend(0.7, 2.5, legend=as.character(rev(lambda)),
       lty=1, pch=length(lambda):1,
       col=length(lambda):1)
```



Box-Cox Transformations

```
x <- rexp(1000)
bc <- function(y, lambda){
    (y^lambda -1)/lambda
}
qqnorm(x); qqline(x, col="red")

bc1.x <- bc(x, 0.1)
qqnorm(bc1.x, main="lambda=0.1")
qqline(bc1.x, col="red")
bc3.x <- bc(x, 0.5)
qqnorm(bc3.x, main="lambda=0.5")
qqline(bc3.x, col="red")

bc2.x <- bc(x, 0.268)
qqnorm(bc2.x, main="lambda=0.268")
qqline(bc2.x, col="red")

hist(x, main="rexp(1000)")
hist(bc2.x, main="lambda=0.268")
```





$$\left(\Phi^{-1}\left(\frac{i-0.5}{n}\right), x_{(i)}\right), \qquad \text{for } i = 1, 2, \ldots, n,$$

Source: Box-Cox Transformations: An Overview, Pengfei Li, Department of Statistics, University of Connecticut, Apr 11, 2005

Modified Box-Cox Transformations

Manly(1971)

$$y(\lambda) = \begin{cases} \frac{e^{\lambda y}-1}{\lambda}, & \text{if } \lambda \neq 0; \\ y, & \text{if } \lambda = 0. \end{cases}$$

Negative y's could be allowed. The transformation was reported to be successful in transform unimodal skewed distribution into normal distribution, but is not quite useful for **bimodal** or **U-shaped distribution**.

John and Draper(1980) "Modulus Transformation"

$$y(\lambda) = \begin{cases} \text{Sign}(y)\frac{(|y|+1)^{\lambda}-1}{\lambda}, & \text{if } \lambda \neq 0; \\ \text{Sign}(y)\log(|y|+1), & \text{if } \lambda = 0, \end{cases}$$

$$\text{Sign}(y) = \begin{cases} 1, & \text{if } y \geq 0; \\ -1, & \text{if } y < 0. \end{cases}$$

Bickel and Doksum(1981)

$$y(\lambda) = \frac{|y|^{\lambda}\text{Sign}(y) - 1}{\lambda}, \qquad \text{for } \lambda > 0,$$

Yeo and Johnson(2000)

$$y(\lambda) = \begin{cases} \frac{(y+1)^{\lambda}-1}{\lambda}, & \text{if } \lambda \neq 0,\ y \geq 0; \\ \log(y+1), & \text{if } \lambda = 0,\ y \geq 0; \\ \frac{(1-y)^{2-\lambda}-1}{\lambda-2}, & \text{if } \lambda \neq 2,\ y < 0; \\ -\log(1-y), & \text{if } \lambda = 2,\ y < 0. \end{cases}$$

Source: Box-Cox Transformations: An Overview, Pengfei Li, Department of Statistics, University of Connecticut, Apr 11, 2005

標準化 (Standardization)

- Standardization: (called z-score): the new variate Z will have a mean of zero and a variance of one. (also called centering and scaling.)

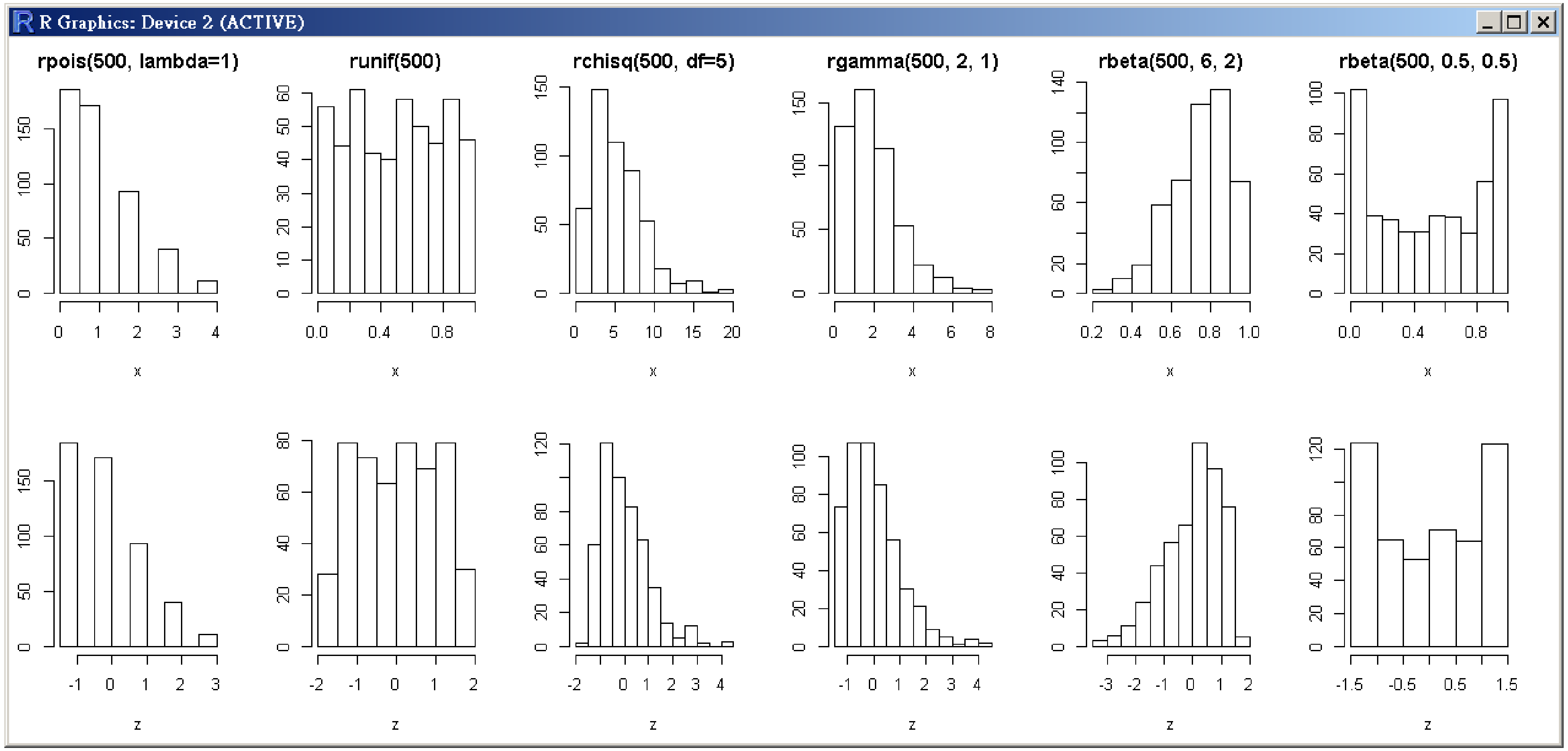
$$z_i = \frac{x_i - \bar{x}}{s}$$

- If the variables are measurements along a **different scale** or if the standard deviations for the variables are different from one another, then one variable might **dominate** the distance (or some other similar calculation) used in the analysis.

- Standardization is useful for comparing variables expressed in different units.

標準化 (Standardization)

Standardization makes no difference to the shape of a distribution.



```
x <- rpois(500, lambda=1)
hist(x, main="rpois(500, lambda=1)"); z <- scale(x); hist(z, main="")
```

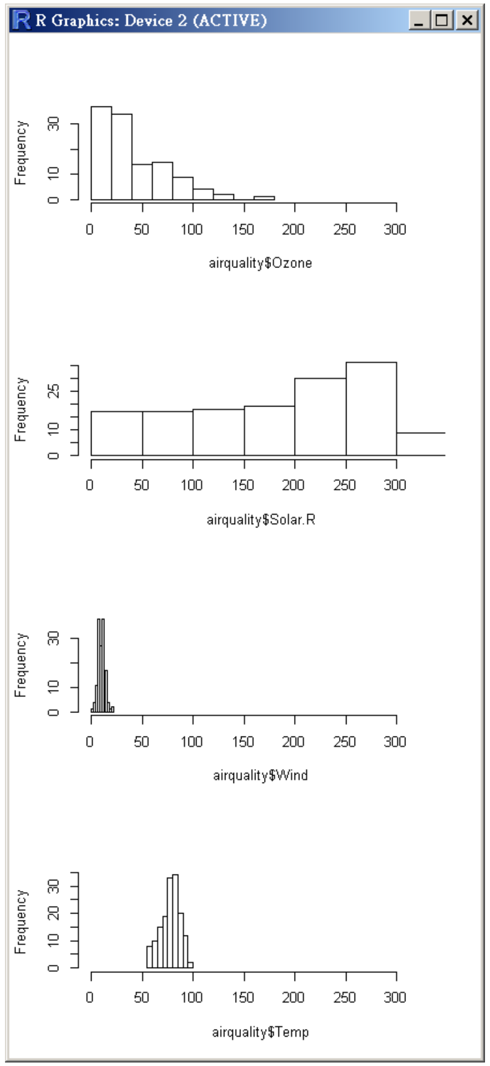
範例: Standardization

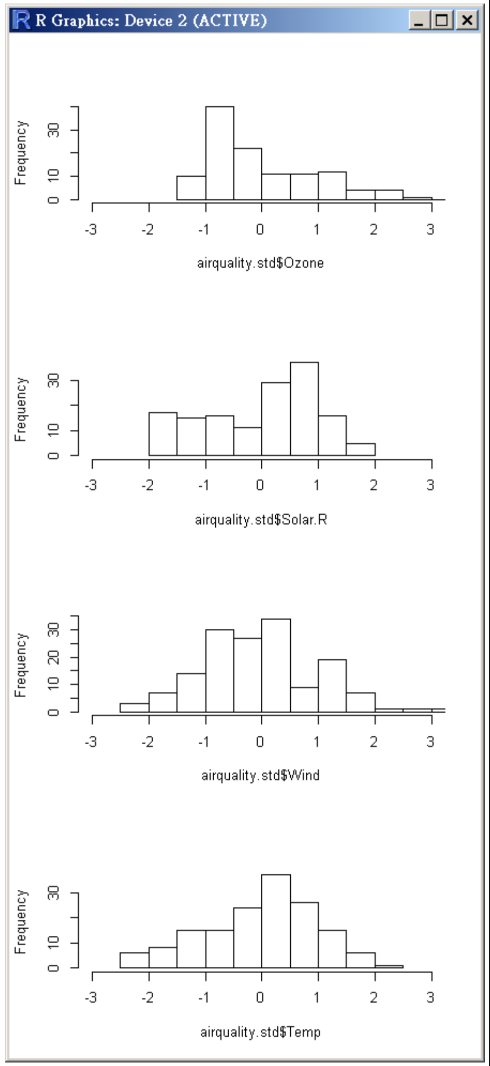
airquality {datasets}

New York Air Quality Measurements: Daily air quality measurements in New York, May to September 1973.
A data frame with 154 observations on 6 variables.

[1] Ozone: Ozone (ppb)
[2] Solar.R: Solar R (lang)
[3] Wind: Wind (mph)
[4] Temp: Temperature (degrees F)
[5] Month: Month (1--12)
[6] Day: Day of month (1--31)

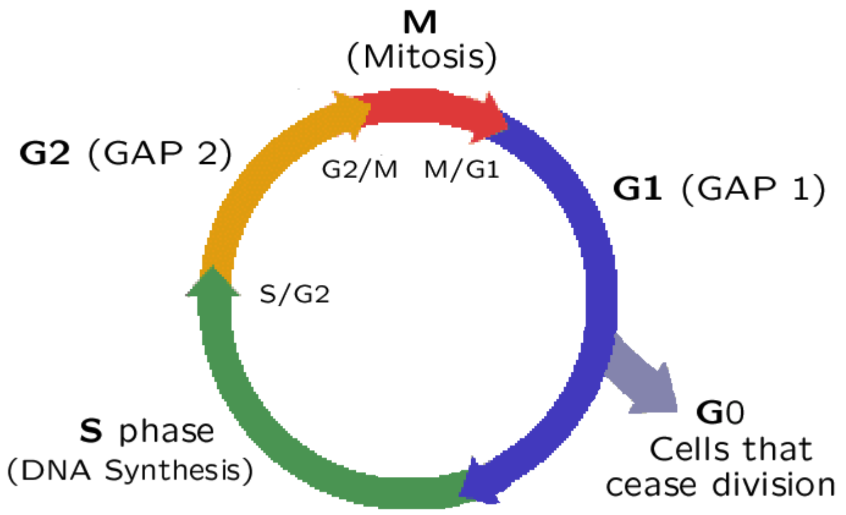
```
> head(airquality )
  Ozone Solar.R Wind Temp Month Day
1    41     190  7.4   67     5   1
2    36     118  8.0   72     5   2
3    12     149 12.6   74     5   3
4    18     313 11.5   62     5   4
5    NA      NA 14.3   56     5   5
6    28      NA 14.9   66     5   6
> r <- range(airquality[,1:4], na.rm = T)
> hist(airquality$Ozone , xlim = r)
> hist(airquality$Solar.R, xlim = r)
> hist(airquality$Wind, xlim = r)
> hist(airquality$Temp, xlim = r)
>
> airquality.std <- as.data.frame(
apply(airquality, 2, scale))
> r.std <- c(-3, 3)
> hist(airquality.std$Ozone, xlim = r.std)
> hist(airquality.std$Solar.R, xlim = r.std)
> hist(airquality.std$Wind, xlim = r.std)
> hist(airquality.std$Temp, xlim = r.std)
```

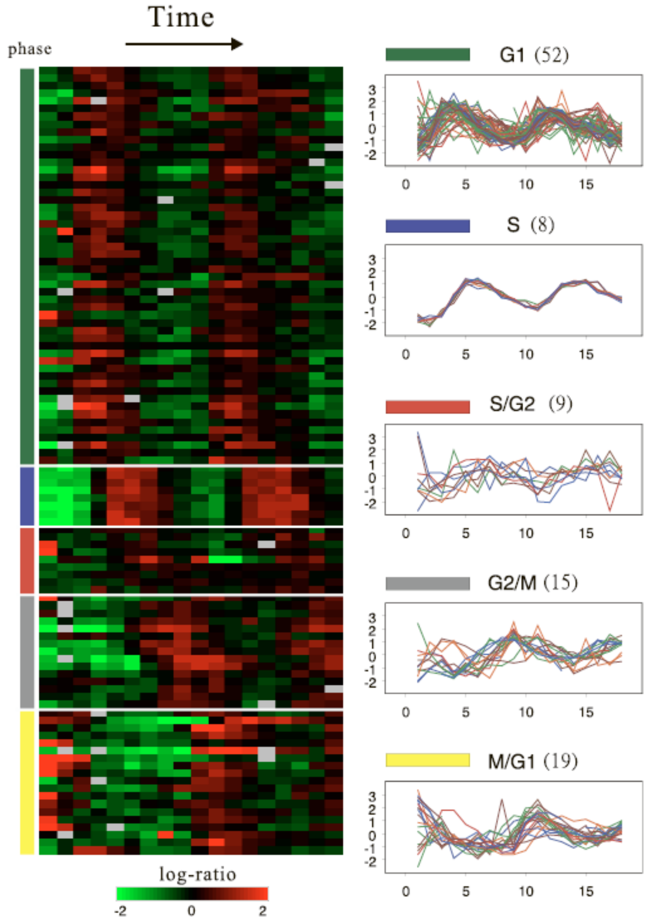




範例: Microarray Data of Yeast Cell Cycle

- Lu and Wu (2010)
  - Time course data: every 7 minutes and totally 18 time points.
  - Known genes: there are 103 cell cycle-regulated genes by traditional method in G1, S, S/G2, G2/M, or M/G1. (Remove NA’ s: 79.)

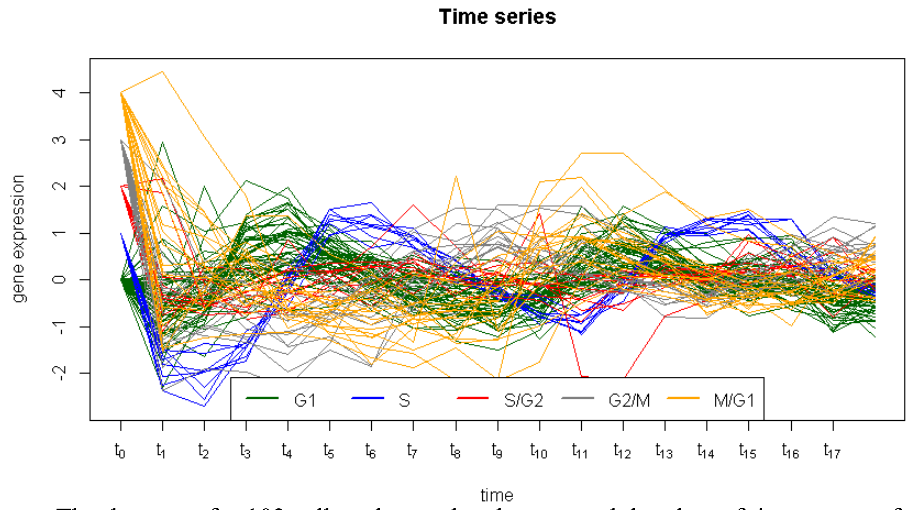


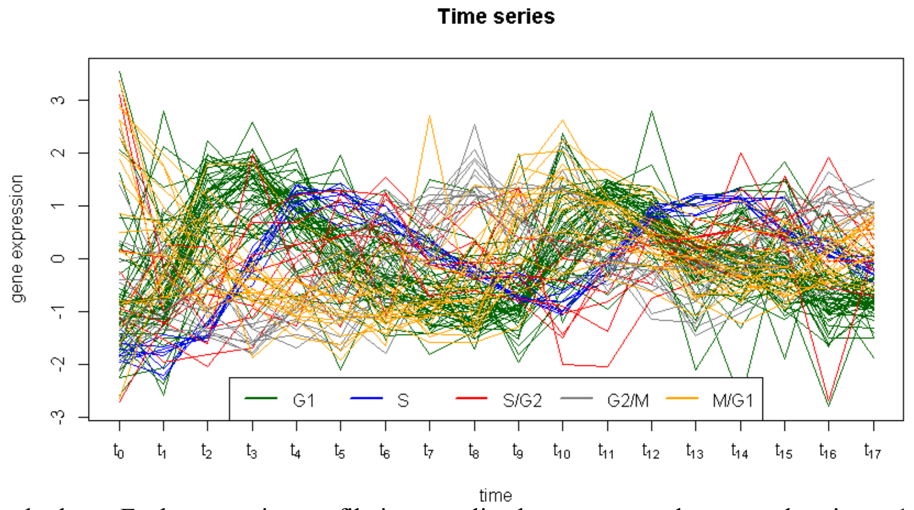


See also: Using R to draw a Heatmap from Microarray Data
http://www2.warwick.ac.uk/fac/sci/moac/people/students/peter_cock/r/heatmap/

Standardization in Time Series Microarray Gene Expression Experiments

```
cell.raw <- read.table("trad_alpha103.txt", row.names=1, header=T)
head(cell.raw)
cell.xdata <- t(scale(t(cell.raw[,2:19]), center=T, scale=T))
y.C <-  as.integer(cell.raw[,1])
table(y.C)
no.cluster <- length(unique(y.C))
cellcycle.color <- c("darkgreen", "blue", "red", "gray50", "orange")
p <- ncol(cell.raw) -1
ycolors <- cellcycle.color[y.C+1]
my.pch <- c(1:no.cluster)[y.C+1]
phase <- c("G1", "S", "S/G2", "G2/M", "M/G1")
matplot(t(cell.xdata), pch = 1:p, lty=1, type = "l", ylab="gene expression",
            col=ycolors, xlab="time", main="Time series", xaxt="n")
time.label <- parse(text=paste("t[",0:p,"]",sep=""))
axis(1, 1:(p+1), time.label)
legend("bottom", legend=phase, col=cellcycle.color, lty=1, horiz = T, lwd=2)
```





The data map for 103 cell cycle-regulated genes and the plots of time courses for each phase. Each expression profile is normalized as mean equals zero and variance 1.

範例: Crab Data

crabs {MASS}

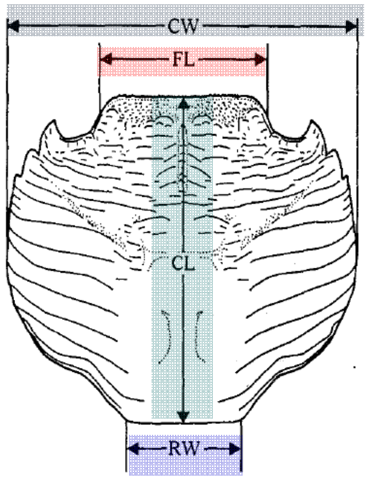
Morphological Measurements on Leptograpsus Crabs

Description: The crabs data frame has 200 rows and 8 columns, describing 5 morphological measurements on **50 crabs each of two colour forms and both sexes**, of the species Leptograpsus variegatus (紫岩蟹) collected at Fremantle, W. Australia.

This data frame contains the following columns:

sp: species - "B" or "O" for blue or orange.
sex: “M” or “F” for male or female
index: 1:50 within each of the four groups.
FL: carapace frontal lobe (lip) size (mm).
RW: carapace rear width (mm).
CL: carapace length (mm).
CW: carapace width (mm).
BD: body depth (mm).

```
> library(MASS)
> data(crabs)
```

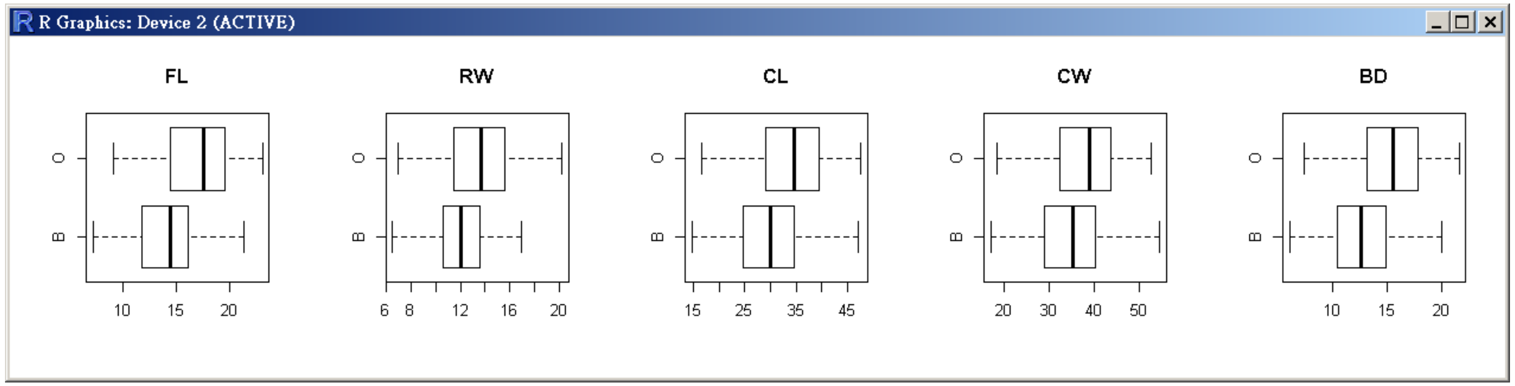


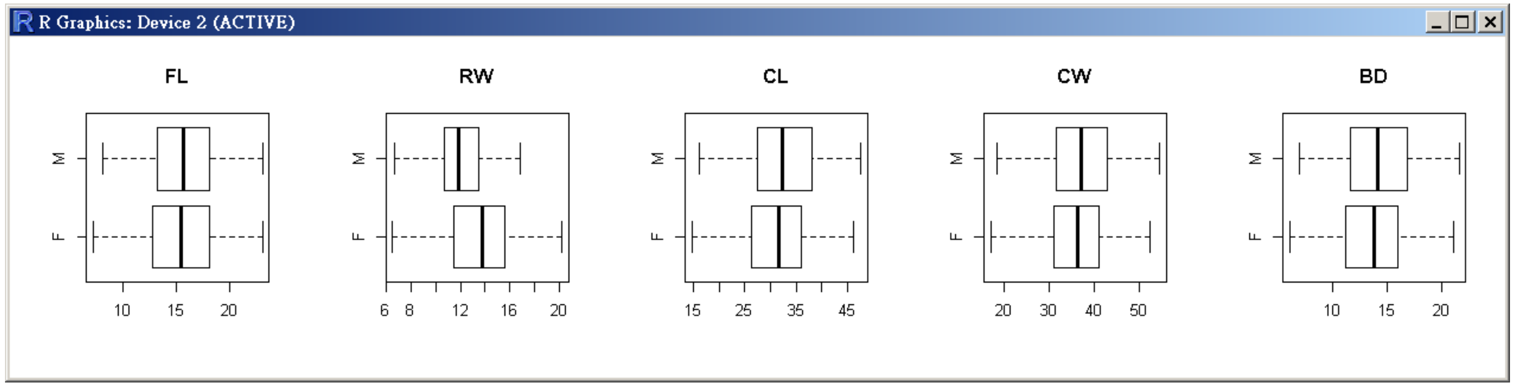
Aust. J. Zool. 1974, 22, 417-25

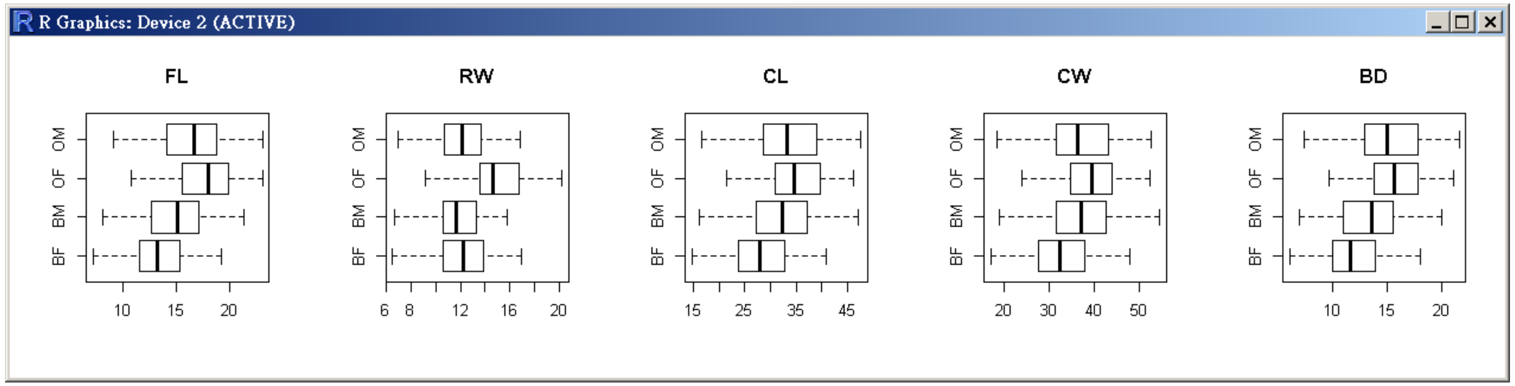
http://www.qm.qld.gov.au/Find+out+about/Animals+of+Queensland/Crustaceans/Common+marine+crustaceans/Crabs/Purple+Swift-footed+Shore+Crab#.VhPWYiurFhs

範例: Crab Data

```
boxplot(crabs$FL~crabs$sp, main="FL", horizontal=T)
```





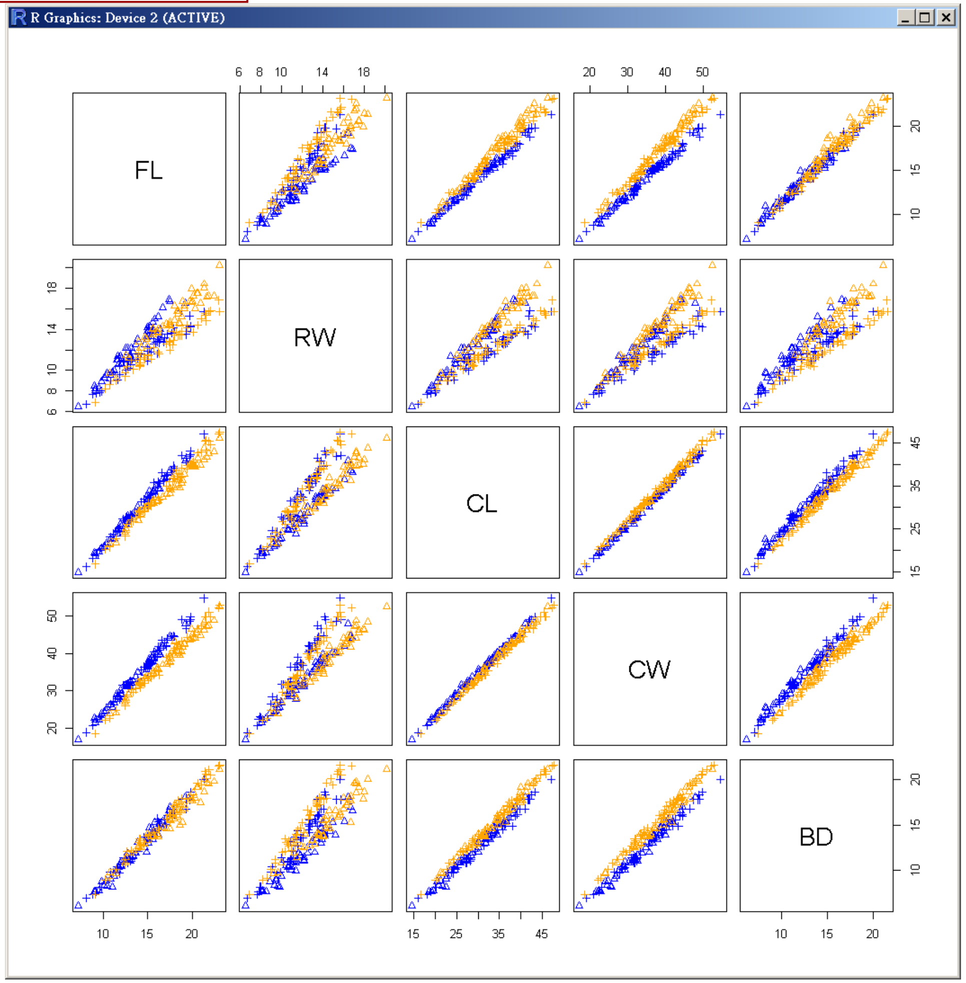


範例: Crab Data

```
# tri: F,  cross: M
pairs(crabs[,4:8],
pch=as.integer(crabs$sex)+1,
col=c("blue","orange")[as.integer(crabs$sp)])
```

- The analysis of ratios of body measurements is deeply ingrained in the taxonomic literature.
- Whether for plants or animals, certain ratios are commonly indicated in identification keys, diagnoses, and descriptions.

(Hannes Baur and Christoph Leuenberger, Analysis of Ratios in Multivariate Morphometry, Systematic Biology 60(6), 813-825.)

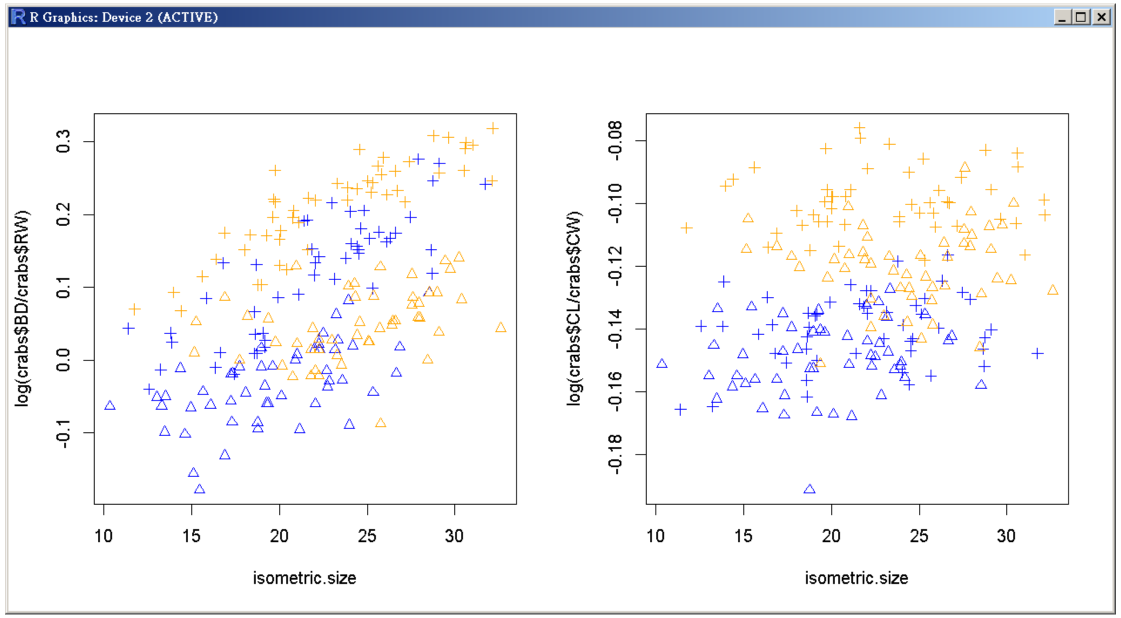


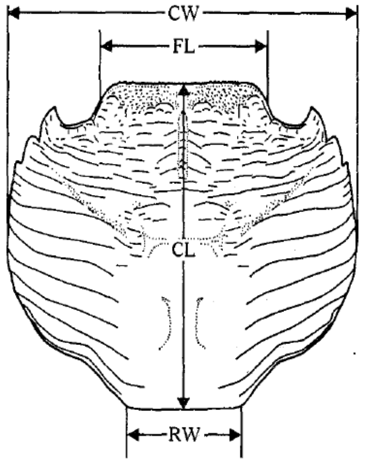
範例: Crab Data

- The use of **ratios of measurements** (i.e., of body proportions), has a long tradition and is deeply ingrained in morphometric taxonomy.

Three size vectors have been commonly proposed in the literature:

(a) isometric size
(the arithmetic mean of x),
(b) allometric size,
(c) shape-uncorrelated size.

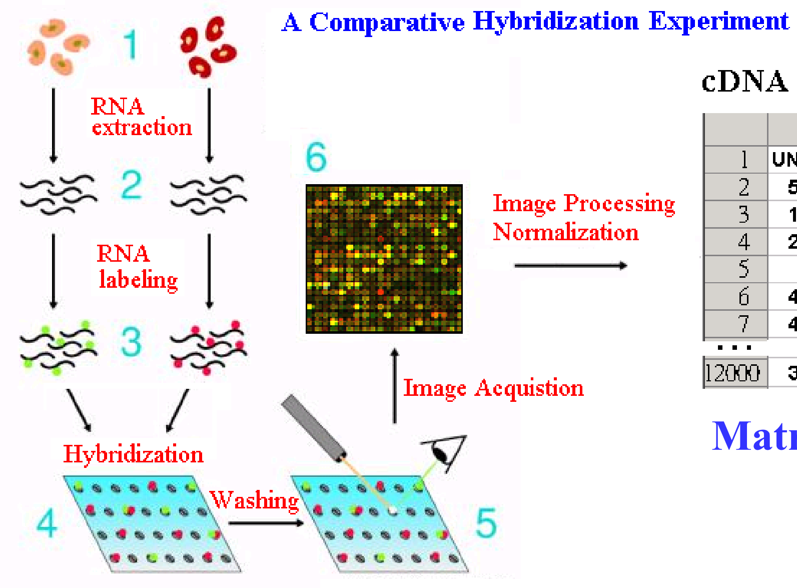




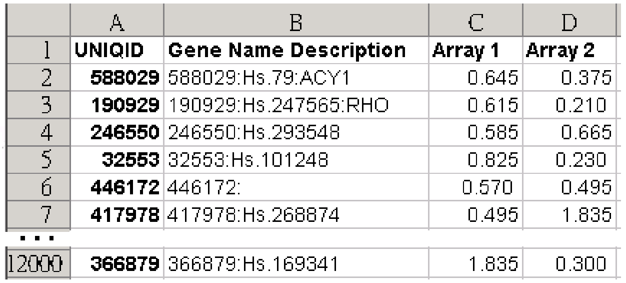
```
par(mfrow=c(1,2))
mp <- as.integer(crabs$sex)+1
mc <- c("blue","orange")[as.integer(crabs$sp)]
isometric.size <- apply(crabs[,4:8], 1, mean)
plot(isometric.size,  log(crabs$BD/crabs$RW), pch=mp, col=mc)
plot(isometric.size, log(crabs$CL/crabs$CW), pch=mp, col=mc)
```

範例: cDNA Microarray Gene Expression Data

微陣列資料統計分析 Statistical Microarray Data Analysis
http://www.hmwu.idv.tw/index.php/mada


A Comparative Hybridization Experiment
1
RNA extraction
2
RNA labeling
3
Hybridization
4
Washing
5
Image Acquistion
6
Image Processing Normalization

cDNA Microarray Data



	A	B	C	D
1	UNIQID	Gene Name Description	Array 1	Array 2
2	588029	588029:Hs.79:ACY1	0.645	0.375
3	190929	190929:Hs.247565:RHO	0.615	0.210
4	246550	246550:Hs.293548	0.585	0.665
5	32553	32553:Hs.101248	0.825	0.230
6	446172	446172:	0.570	0.495
7	417978	417978:Hs.268874	0.495	1.835
...				
12000	366879	366879:Hs.169341	1.835	0.300

Matrix of genes (rows) and samples (columns)

Why Normalization?

Non-biological factor can contribute to the variability of data, in order to reliably compare data from multiple probe arrays, differences of non-biological origin must be minimized.
(Remove the systematic bias in the data).

- Within-Array Normalization
- Between-Array Normalization
- Paired-slides Normalization
- ...

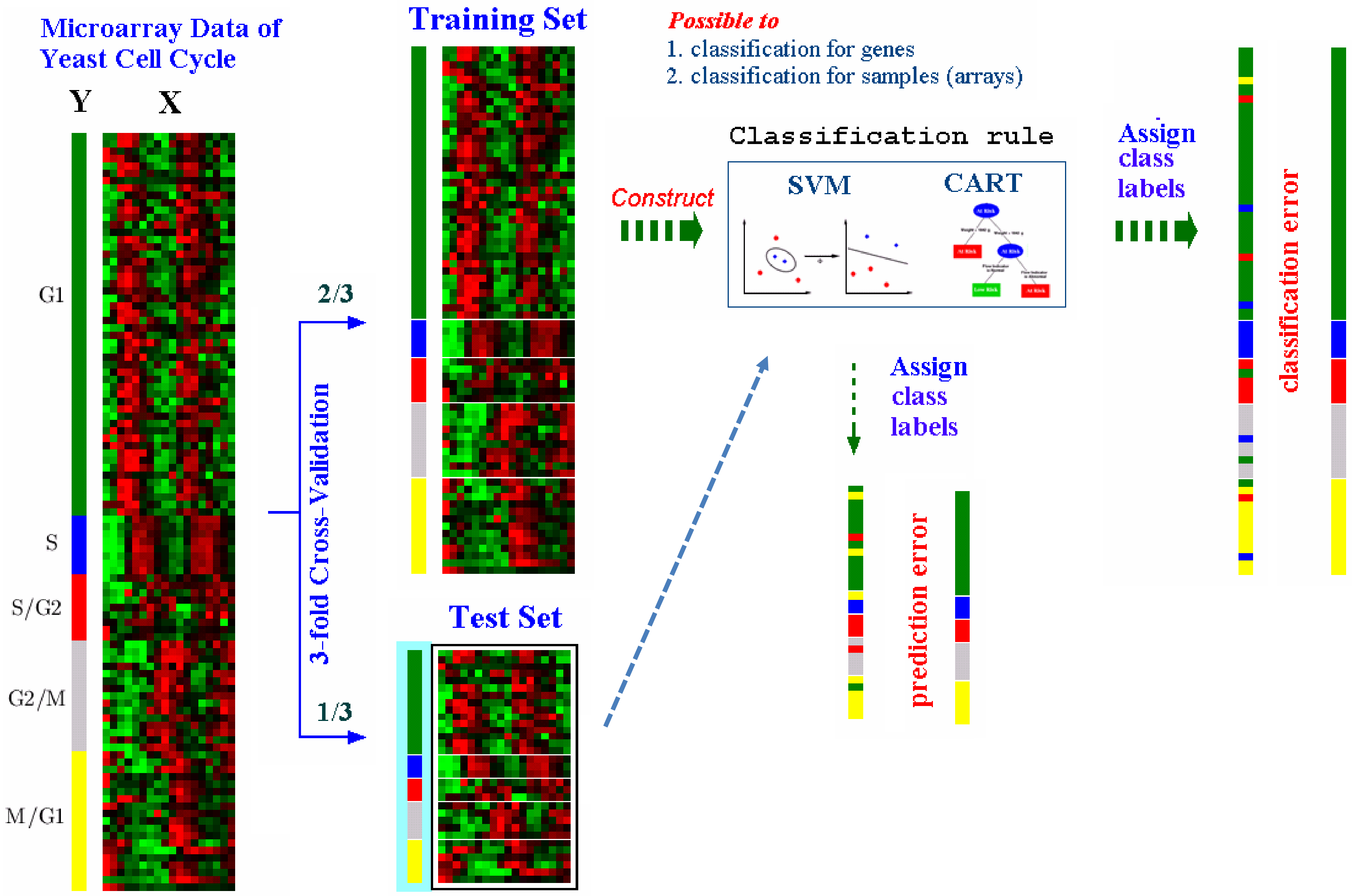
要使用哪一種資料轉換方式?

- Use a transformation that other researchers **commonly use in your field**.
- Guidance for how data should be transformed, or whether a transformation should be applied at all, should come from the particular statistical analysis to be performed.
- The main criterions in choosing a transformation:
  - what works with the data?
  - what makes physical (biological, economic, whatever) sense.
- If you have a **large** number of observations, compare the effects of different transformations on the normality and the homoscedasticity of the variable.

http://www.biostathandbook.com/transformation.html

Classification: *k*-fold Cross-Validation Error Rates

Possible to
1. classification for genes
2. classification for samples (arrays)



Split Data into Test and Train Set According to Group Labels

```
library(caTools) # Tools: moving window statistics, GIF, Base64, ROC AUC, etc
set.seed(12345)
id <- sample.split(1:nrow(iris), SplitRatio = 0.90)
iris.train <- subset(iris, id == TRUE)
iris.test <- subset(iris, id == FALSE)
```

```
> require(caTools)
> Y <- iris[,5] # extract labels from the data
> msk <- sample.split(Y, SplitRatio=4/5)
> msk
  [1]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE
...
[144]  TRUE  TRUE  TRUE FALSE  TRUE  TRUE FALSE
> table(Y, msk)
            msk
Y             FALSE TRUE
  setosa         10   40
  versicolor     10   40
  virginica      10   40
> iris.train <- iris[msk, ]
> iris.test <- iris[!msk, ]
> dim(iris.train)
[1] 120   5
> dim(iris.test)
[1] 30  5
```

```
> library(caret)
> createFolds(iris$Species, k=3)
$Fold1
 [1]    2    8   15   22   25   27   30   ...

$Fold2
 [1]    5    6    9   10   11   12   17 ...

$Fold3
 [1]    1    3    4    7   13   14   16   20...
```

```
library(caret)
id <- createDataPartition(y=iris$Species, p=0.9, list=FALSE)
iris.train <- iris[id, ]
iris.test <- iris[-id, ]
```

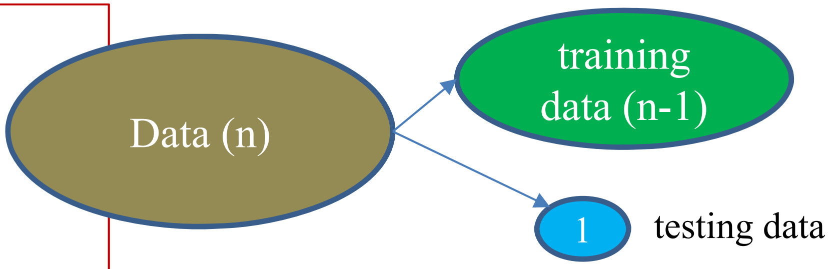
Jackknife Resampling: Leave-one-out

$\hat{\theta}$ the calculated estimator of the parameter based on all n observations

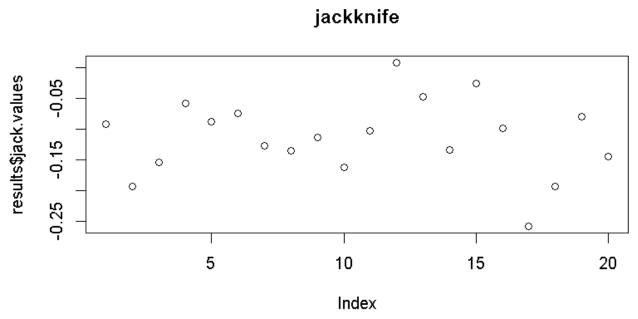
$$\hat{\theta}_{(.)} = \frac{1}{n}\sum_{i=1}^{n}\hat{\theta}_{(\mathrm{i})}$$ the average of these "leave-one-out" estimates

$$\hat{\theta}_{\text{Jack}} = n\hat{\theta} - (n-1)\hat{\theta}_{(.)}$$ the resulting bias-corrected jackknife estimate

```
> # install.packages("bootstrap")
> library(bootstrap)
> x <- rnorm(20)
> theta <- function(x){mean(x)}
> (theta.hat <- theta(x))
[1] -0.1135763
> results <- jackknife(x,theta)
> results
$jack.se
[1] 0.264117

$jack.bias
[1] 2.63678e-16

$jack.values
 [1] -0.091950484 -0.193139320 -0.153668397 ...
...
$call
jackknife(x = x, theta = theta)
```

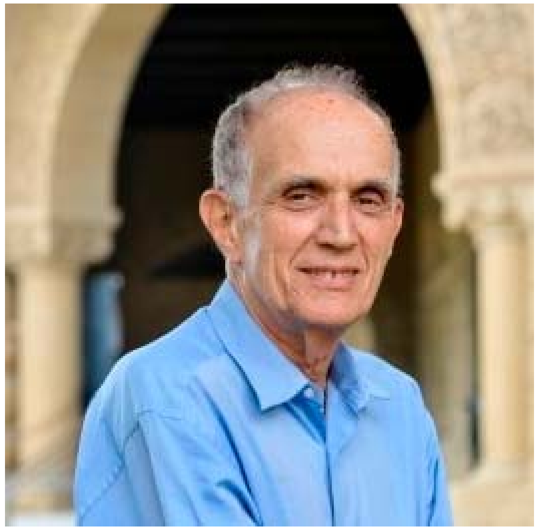


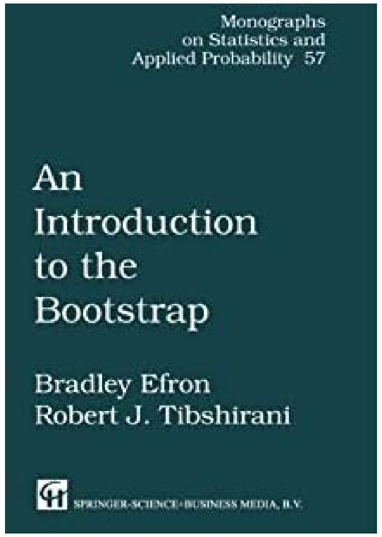
```
> theta.hat.loo <- mean(results$jack.values)
> (theta.hat.jack <- n * theta.hat - (n-1) * theta.hat.loo)
[1] -0.1135763
> plot(results$jack.values, main="jackknife")
```

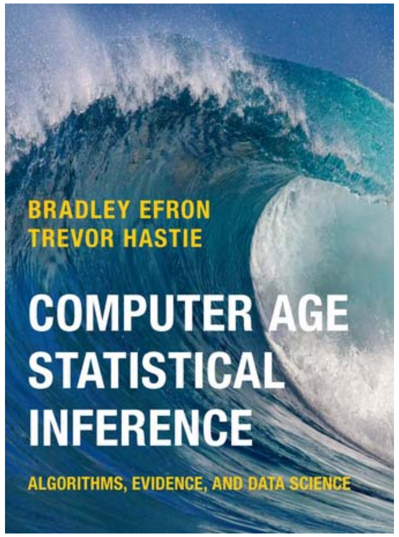


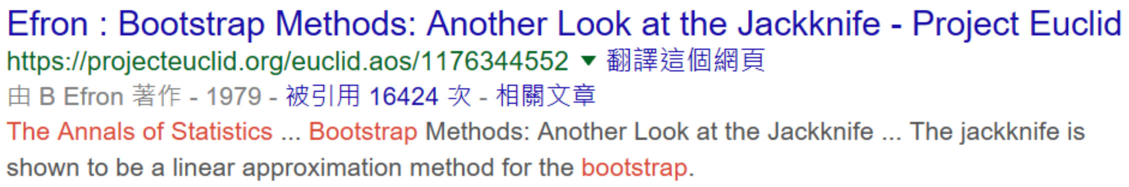
自助法、拔靴法
Bootstrap Methods

- Bootstrapping is a statistical method for estimating the sampling distribution of an estimator by **sampling with replacement** from the original sample, of the same size as the original sample.

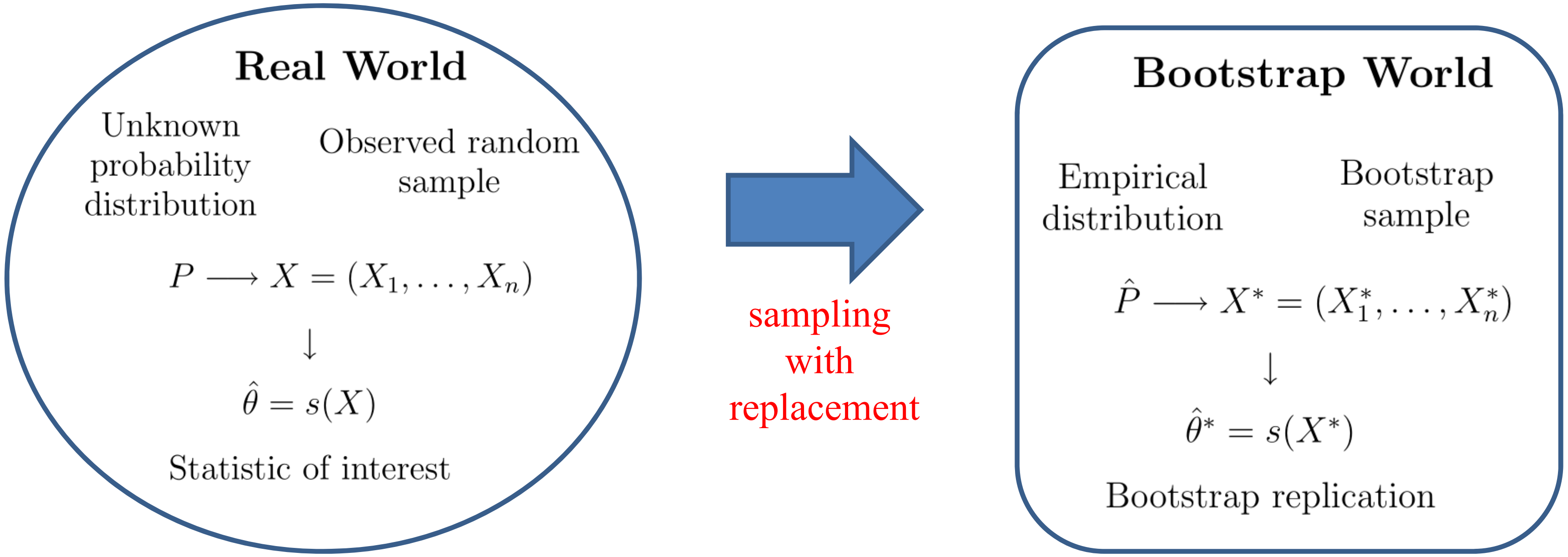

Bradley Efron 1938~
Department of Statistics, Stanford University

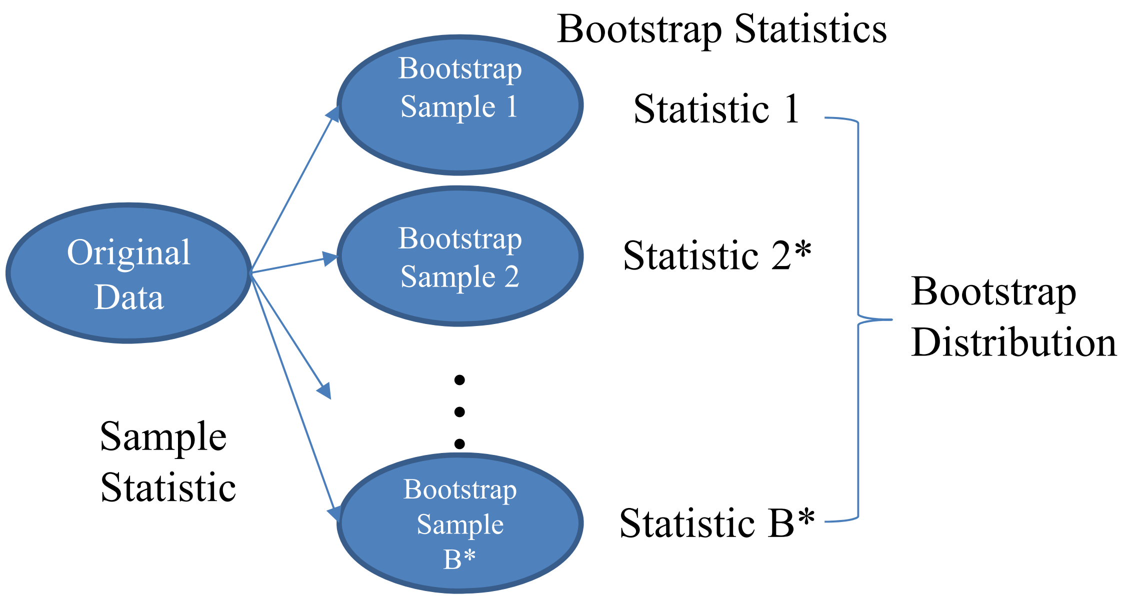

Monographs on Statistics and Applied Probability 57
An Introduction to the Bootstrap
Bradley Efron
Robert J. Tibshirani
SPRINGER-SCIENCE+BUSINESS MEDIA, B.V.
1993


BRADLEY EFRON
TREVOR HASTIE
COMPUTER AGE STATISTICAL INFERENCE
ALGORITHMS, EVIDENCE, AND DATA SCIENCE
2016


Efron : Bootstrap Methods: Another Look at the Jackknife - Project Euclid
https://projecteuclid.org/euclid.aos/1176344552 ▾ 翻譯這個網頁
由 B Efron 著作 - 1979 - 被引用 16424 次 - 相關文章
The Annals of Statistics ... Bootstrap Methods: Another Look at the Jackknife ... The jackknife is shown to be a linear approximation method for the bootstrap.

Bootstrapping


Real World
Unknown probability distribution
Observed random sample
$P \longrightarrow X = (X_1, \ldots, X_n)$
$\downarrow$
$\hat{\theta} = s(X)$
Statistic of interest
sampling with replacement
Bootstrap World
Empirical distribution
Bootstrap sample
$\hat{P} \longrightarrow X^* = (X_1^*, \ldots, X_n^*)$
$\downarrow$
$\hat{\theta}^* = s(X^*)$
Bootstrap replication



bootstrap Package

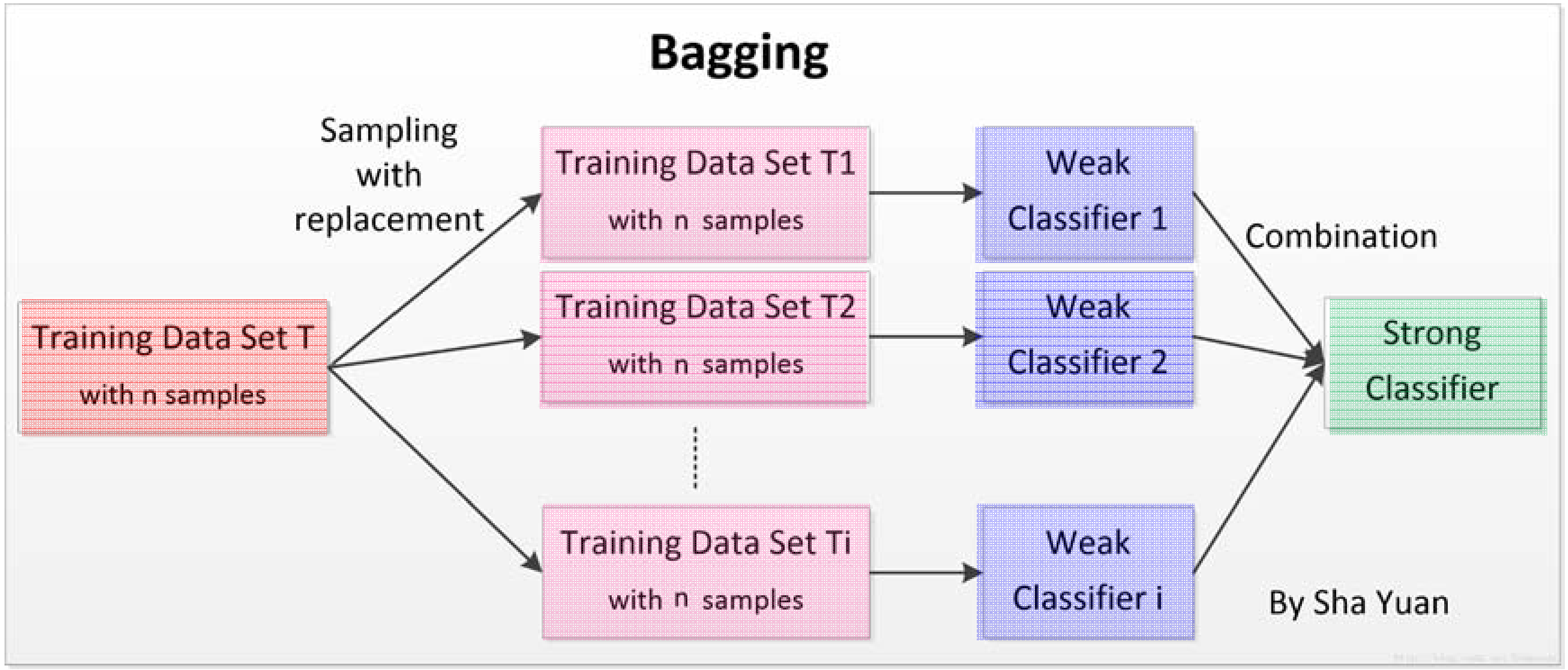
Bootstrap Estimation of
the Sample Mean

語法:
bootstrap(x, nboot, theta, ..., func=NULL)

x:	a vector containing the data.
nboot:	the number of bootstrap samples.
theta:	function to be bootstrapped.

```
> # install.packages("bootstrap")
> library(bootstrap)
> set.seed(12345)
> x <- rnorm(20)
> mean(x)
[1] 0.07651681
> (x.bootstrap.mean <- bootstrap(x, 50, theta=mean))
$thetastar
 [1]  0.486197466 -0.160488357  0.274920990  0.398499864 -0.399967845  0.116086370
...
[43] -0.348643786  0.185330636 -0.070823890  0.057609481  0.062067504  0.043716794
[49] -0.279597885  0.243843620

$func.thetastar
NULL

$jack.boot.val
NULL

$jack.boot.se
NULL

$call
bootstrap(x = x, nboot = 50, theta = mean)
> mean(x.bootstrap.mean$thetastar)
[1] 0.08647268
```

Bagging: Bootstrap Aggregating


Bagging
Sampling with replacement
Training Data Set T with n samples
Training Data Set T1 with n samples
Training Data Set T2 with n samples
Training Data Set Ti with n samples
Weak Classifier 1
Weak Classifier 2
Weak Classifier i
Combination
Strong Classifier
By Sha Yuan

http://blog.csdn.net/bymaymay/article/details/77824574

- Breiman, L. (1996). Bagging predictors, Machine Learning, Vol. 26, pp. 123-140.
- Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm, Proceedings of the Thirteenth International Conference, Machine Learning.

Boosting

$$\boxed{W_{x_i}^{(1)} = N^{-1} \text{ for all } x_i.}$$

↓

a bootstrap sample $\mathcal{L}_t^{(B)}$

error ϵ_t of classifier $\varphi_t(\mathbf{x})$

$$\epsilon_t = \sum_{\{i:\varphi_t(x_i)\neq y_i\}} W_{x_i}^{(t)}.$$

↓

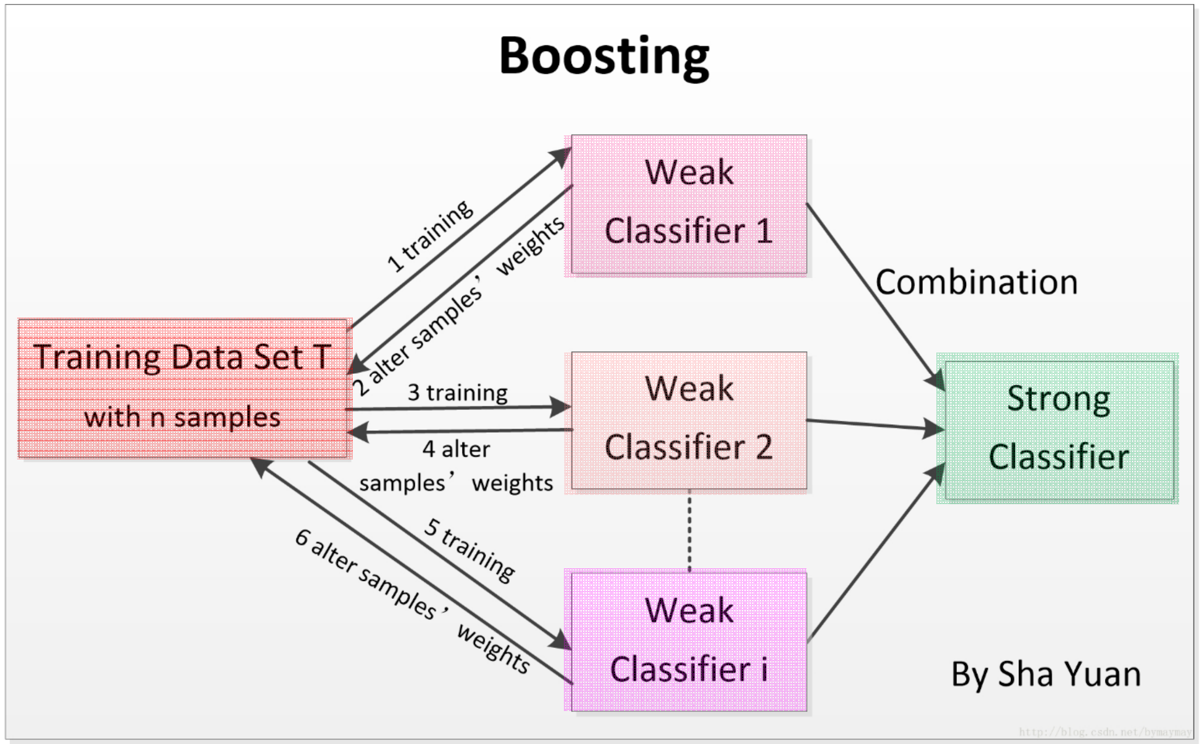
$$\boxed{\beta_t = (1-\epsilon_t)\epsilon_t^{-1}}$$

↓

$$W_{x_i}^{(t+1)} = \frac{W_{x_i}^{(t)}\beta_t^{d(i)}}{\sum_i W_{x_i}^{(t)}\beta_t^{d(i)}},$$

↓

boosted classifier



http://blog.csdn.net/bymaymay/article/details/77824574

$d(i) = 1$ if ith case is classified incorrectly,
$d(i) = 0$, otherwise

$$\varphi_B(x_i) = arg\ max_j \sum_{t=1}^{T} \log \beta_t I[\varphi_t(x_i) = j]$$

Ad-Boost.M1 (Freund and Schapire, 1996)

Example: Apply rpart to Vehicle Data

```
> library(rpart); library(mlbench); library(adabag)
> data(Vehicle)
> dim(Vehicle)
[1] 846  19
> head(Vehicle)
 Comp Circ D.Circ Rad.Ra Pr.Axis.Ra Max.L.Ra Scat.Ra Elong Pr.Axis.Rect Max.L.Rect Sc.Var.Maxis
1   95   48     83    178         72        10     162    42            20        159          176
...
  Sc.Var.maxis Ra.Gyr Skew.Maxis Skew.maxis Kurt.maxis Kurt.Maxis Holl.Ra Class
1          379    184         70          6         16        187     197   van
...
6          957    264         85          5          9        181     183   bus
> table(Vehicle$Class)
 bus opel saab  van
 218  212  217  199
```

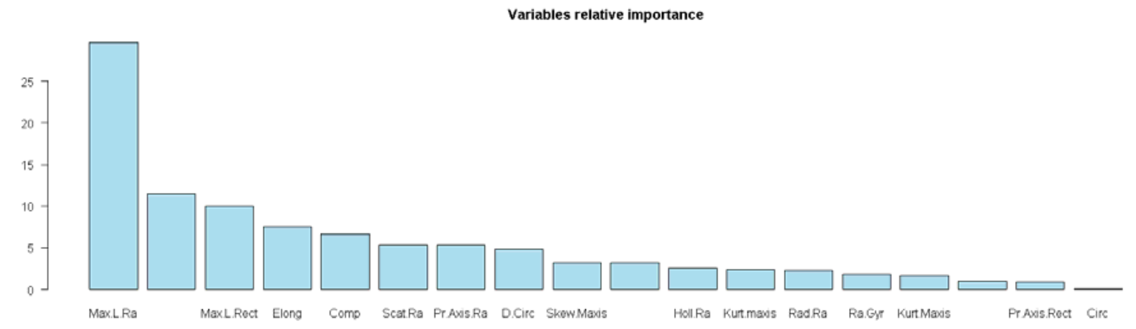
```
> n <- nrow(Vehicle)
> sub <- sample(1:n, 2*n/3)
> Vehicle.train <- Vehicle[sub, ]
> Vehicle.test <- Vehicle[-sub, ]
```

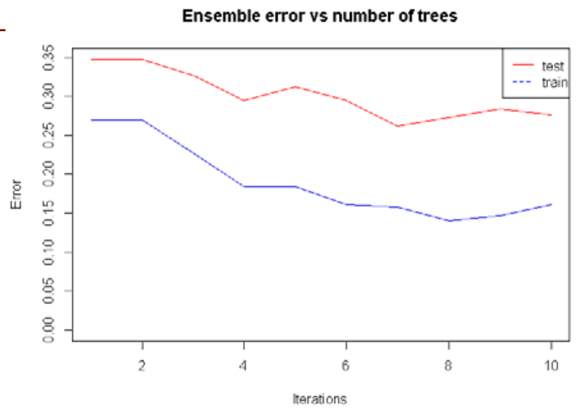
```
> mfinal <- 10 # Defaults to mfinal=100 iterations
> maxdepth <- 5
> Vehicle.rpart <- rpart(Class ~ ., data = Vehicle.train, maxdepth = maxdepth)
> Vehicle.rpart.pred <- predict(Vehicle.rpart, newdata = Vehicle.test, type = "class")
> (tb <- table(Vehicle.rpart.pred, Observed.Class=Vehicle.test$Class))
                  Observed.Class
Vehicle.rpart.pred bus opel saab van
              bus   69   10    6   2
              opel   1   25   13   3
              saab   1   34   37   8
              van    2    7    5  59
> (error.rpart <- 1 - (sum(diag(tb)) / sum(tb)))
[1] 0.3262411
```

adabag: An R Package for Classification with Boosting and Bagging

```
> library(adabag)
> Vehicle.adaboost <- boosting(Class ~., data = Vehicle.train, mfinal = mfinal,
+                              control = rpart.control(maxdepth=maxdepth))
> Vehicle.adaboost.pred <- predict.boosting(Vehicle.adaboost, newdata = Vehicle.test)
> Vehicle.adaboost.pred$confusion
               Observed Class
Predicted Class bus opel saab van
           bus   69    2    3   1
           opel   1   30   16   4
           saab   1   38   39   1
           van    2    6    3  66
> Vehicle.adaboost.pred$error
[1] 0.2765957
> importanceplot(Vehicle.adaboost)
>
> # comparing error evolution in training and test set
> evol.train <- errorevol(Vehicle.adaboost, newdata = Vehicle.train)
> evol.test <- errorevol(Vehicle.adaboost, newdata = Vehicle.test)
> plot.errorevol(evol.test, evol.train)
```

```
> sort(Vehicle.adaboost$importance, dec=T)[1:5]
   Max.L.Ra Sc.Var.maxis   Max.L.Rect
  29.623783    11.473254     9.956137
      Elong         Comp
   7.570798     6.656360
```





Alfaro, E., Gamez, M. and Garcia, N. (2013): "adabag: An R Package for Classification with Boosting and Bagging". Journal of Statistical Software, 54(2), 1–35.

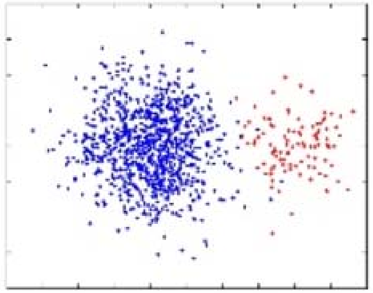
Example: 10-fold CV adaboost.M1

```
> # 10-fold CV adaboost.M1
> Vehicle.boost.cv <- boosting.cv(Class ~., data = Vehicle, v = 10, mfinal = 5,
+                                 control = rpart.control(maxdepth = maxdepth))
> Vehicle.boost.cv$confusion
               Observed Class
Predicted Class bus opel saab van
           bus  209    9   11   3
           opel   1  101   72   2
           saab   0   88  117   6
           van    8   14   17 188
> Vehicle.boost.cv$error
[1] 0.2730496
```

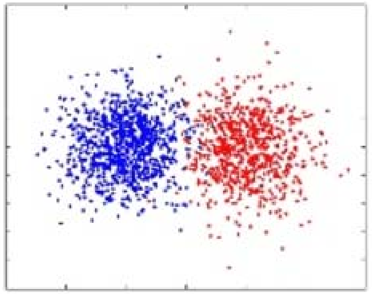
```
> Vehicle.bag.cv <- bagging.cv(Class ~., data = Vehicle, v = 10, mfinal = 5,
+                               control = rpart.control(maxdepth = maxdepth))
> Vehicle.bag.cv$confusion
               Observed Class
Predicted Class bus opel saab van
           bus  208   20   24   2
           opel   1   91   53   1
           saab   1   81  116   5
           van    8   20   24 191
> Vehicle.bag.cv$error
[1] 0.2836879
```

不平衡資料問題
The Imbalanced Data Problem

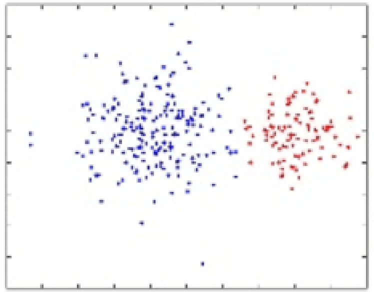
- A dataset is said to be unbalanced when the class of interest (minority class) is much rarer than normal behaviour (majority class).
- **Example**: 5% of the target class represents fraudulent transactions, 95% of the target class represents legitimate transactions.
- Most learning systems are not prepared to cope with unbalanced data and several techniques have been proposed.


Imbalanced Data

Over-sampling



Under-sampling



http://www.srutisj.in/blog/research/statisticalmodeling/balancing-techniques-for-unbalanced-datasets-in-python-r/

unbalanced

Racing for Unbalanced Methods Selection

Re-balance or remove noisy instances in unbalanced datasets.

`ubBalance {unbalanced}`

Usage

```
ubBalance(X, Y, type="ubSMOTE", positive=1,
          percOver=200, percUnder=200,
          k=5, perc=50, method="percPos", w=NULL, verbose=FALSE)
```

Arguments

- `X`: the input variables of the unbalanced dataset.
- `Y`: the response variable of the unbalanced dataset.
- `type`: the balancing technique to use (`ubOver, ubUnder, ubSMOTE, ubOSS, ubCNN, ubENN, ubNCL, ubTomek`).
- `positive`: the majority class of the response variable.
- `percOver`: parameter used in `ubSMOTE`
- `percUnder`: parameter used in `ubSMOTE`
- `k`: parameter used in `ubOver, ubSMOTE, ubCNN, ubENN, ubNCL`
- `perc`: parameter used in `ubUnder`
- `method`: parameter used in `ubUnder`
- `w`: parameter used in `ubUnder`
- `verbose`: print extra information (TRUE/FALSE)

`ubSMOTE {unbalanced}`: synthetic minority over-sampling technique

Usage

```
ubSMOTE(X, Y, perc.over = 200, k = 5, perc.under = 200, verbose = TRUE)
```

Other R packages: `imbalance`: Preprocessing Algorithms for Imbalanced Datasets, Imbalanced Classification in R: `ROSE` (Random Over Sampling Examples) and `DMwR` (Data Mining with R).

The Balancing Technique

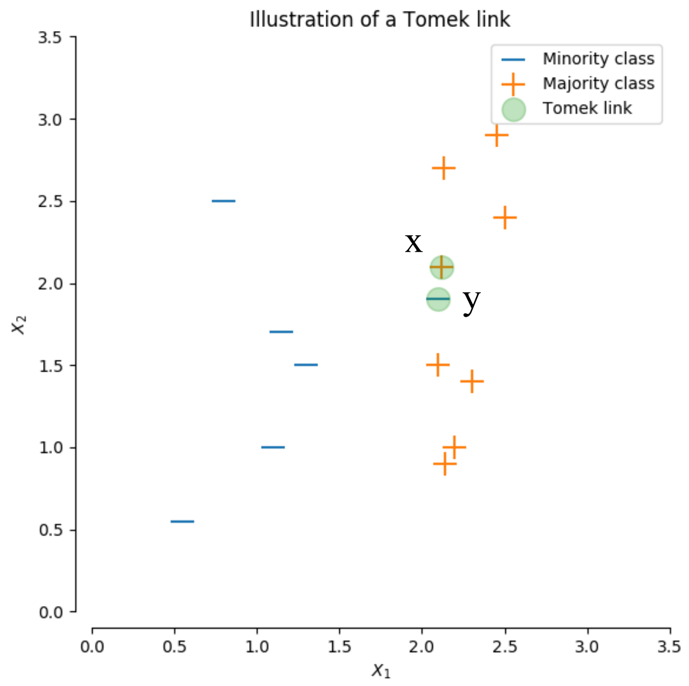
- **`ubOver`**: replicates randomly some instances from the minority class in order to obtain a final dataset with the same number of instances from the two classes.
- **`ubUnder`**: removes randomly some instances from the majority (negative) class and keeps all instances in the minority (positive) class in order to obtain a more balanced dataset.

- **`ubCNN`**: Condensed Nearest Neighbor selects the subset of instances that are able to correctly classifying the original datasets using a one-nearest neighbor rule.
- **`ubENN`**: Edited Nearest Neighbor removes any example whose class label differs from the class of at least two of its three nearest neighbors.
- **`ubNCL`**: Neighborhood Cleaning Rule modifies the Edited Nearest Neighbor method by increasing the role of data cleaning.
  - Firstly, NCR removes negatives examples which are misclassified by their 3-nearest neighbors.
  - Secondly, the neighbors of each positive examples are found and the ones belonging to the majority class are removed.

The Balancing Technique

- **ubTomek**: finds the points in the dataset that are tomek link using 1-NN and then removes only majority class instances that are tomek links.

x's nearest neighbor is y
y's nearest neighbor is x
x and y are different classes

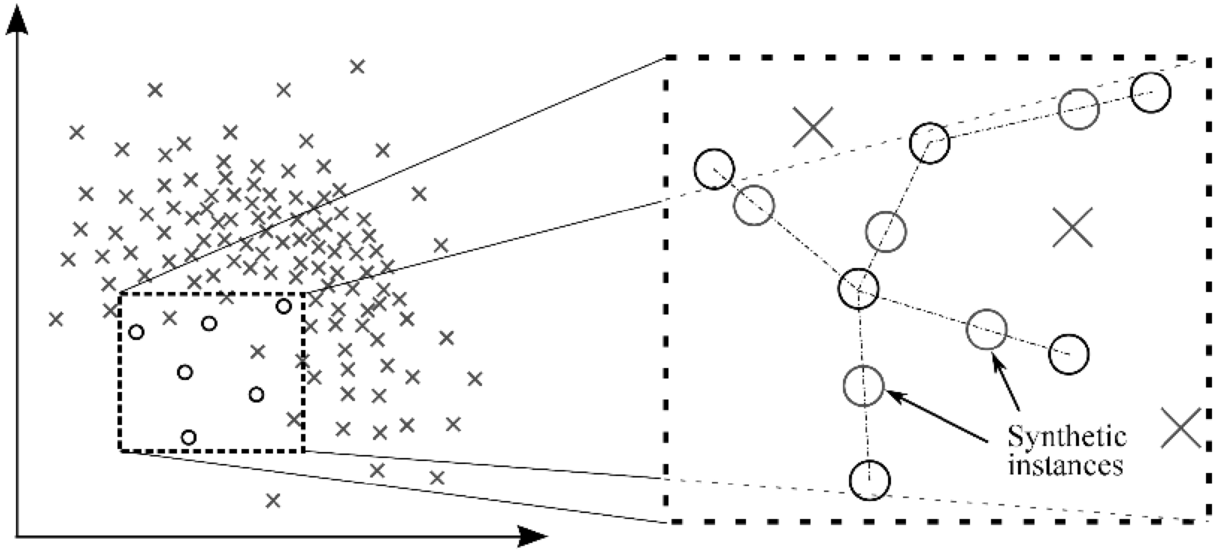


http://contrib.scikit-learn.org/imbalanced-learn/stable/auto_examples/under-sampling/plot_illustration_tomek_links.html

- **ubOSS**: One Side Selection is an undersampling method resulting from the application of Tomek links followed by the application of Condensed Nearest Neighbor.

The Balancing Technique

- **ubSMOTE**: synthetic minority over-sampling technique generates new examples by filling empty areas among the positive instances.



SMOTE: Synthetic Minority Over-sampling Technique | Journal of ...
https://jair.org/papers/paper953.html ▾ 翻譯這個網頁
由 NV Chawla 著作 - 2002 - 被引用 5757 次 - 相關文章
An approach to the construction of classifiers from imbalanced datasets is described. A dataset is imbalanced if the classification categories are not ...

Ionosphere (電離層) dataset ubIonosphere {unbalanced}

- The datasets is a modification of Ionosphere dataset contained in "`mlbench`" package.

```
> data(ubIonosphere)
> dim(ubIonosphere)
[1] 351  33
> head(ubIonosphere)
       V3       V4          V34 Class
1 0.99539 -0.05889 ... -0.45300     0
...
6 0.02337 -0.00592 ...  0.12011     1
> table(ubIonosphere$Class)
  0   1
225 126
```

```
> # install.packages("unbalanced")
> library(unbalanced)
> p <- ncol(ubIonosphere)
> y <- ubIonosphere$Class
> x <- ubIonosphere[ ,-p]
> data <- ubBalance(X=x, Y=y, type="ubOver", k=0)
> overData <- data.frame(data$X, Class=data$Y)
> table(overData$Class)
  0   1
225 225
> data <- ubBalance(X=x, Y=y, type="ubUnder", perc=50, method="percPos")
> underData <- data.frame(data$X, Class=data$Y)
> table(underData$Class)
  0   1
126 126
> bdata <- ubBalance(X=x, Y=y, type="ubSMOTE", percOver=300, percUnder=150, verbose=TRUE)
Proportion of positives after ubSMOTE : 47.06 % of 1071 observations
> str(bdata)
List of 3
 $ X    :'data.frame':  1071 obs. of  32 variables:
  ..$ V3 : num [1:1071] -0.787 1 1 0.5 1 ...
...
..$ V34: num [1:1071] -0.576 0.714 -0.243 0.174 -0.892 ...
 $ Y    : Factor w/ 2 levels "0","1": 2 1 1 1 1 2 1 2 1 2 ...
 $ id.rm: logi NA
> table(bdata$Y)
  0   1
567 504
```

perc: percentage of sampling

K=0: sample with replacement from the minority class until we have the same number of instances in each class.
If K>0: sample with replacement from the minority class until we have k-times the orginal number of minority instances

`per.over/100` : number of new instances generated for each rare instance
`perc.under/100`: number of "normal" (majority class) instances that are randomly selected for each smoted observation.

Compare the Performances using SVM

```
> set.seed(12345)
> n <- nrow(ubIonosphere) # 351
> no.train <- floor(0.5*n) # 175, keep half for training and half for testing
> id <- sample(1:n, no.train)
> x.train  <- x[id, ]  # 175 x 32
> y.train <- y[id]
> x.test <- x[-id, ] # 176  32
> y.test <- y[-id]
>
> library(e1071)
> model1 <- svm(x.train, y.train)
> y.pred1 <- predict(model1, x.test)
> table(y.pred1, y.test)
       y.test
y.pred1   0   1
      0 113  10
      1   4  49
>
> # rebalance the training set before building a model
> balancedData <- ubBalance(X=x.train, Y=y.train, type="ubSMOTE",
                            percOver=200, percUnder=150)
> table(balancedData$Y)
  0   1
201 201
```

```
> model2 <- svm(balancedData$X, balancedData$Y)
> y.pred2 <- predict(model2, x.test)
> table(y.pred2, y.test)
       y.test
y.pred2   0   1
      0 112   8
      1   5  51
```

ubRacing {unbalanced}
Racing for Strategy Selection

Selecting the best technique to re-balance or remove noisy instances in unbalanced datasets.

```
ubRacing(formula, data, algo, positive=1, ncore=1, nFold=10, maxFold=10, maxExp=100,
          stat.test="friedman", metric="f1", ubConf, verbose=FALSE, ...)
```

Arguments

- `algo`: the classification algorithm to use with the mlr package.
- `positive`: label of the positive (minority) class.
- `ncore`: the number of core (parallel execution) to use in the Race.

```
> set.seed(1234)
> # load(url("http://www.ulb.ac.be/di/map/adalpozz/data/creditcard.Rdata"))
> load("creditcard.Rdata")
> str(creditcard)
'data.frame':   284807 obs. of  31 variables:
 $ Time  : num  0 0 1 1 2 2 4 7 7 9 ...
 $ V1    : num  -1.36 1.192 -1.358 -0.966 -1.158 ...
...
 $ V28   : num  -0.0211 0.0147 -0.0598 0.0615 0.2152 ...
 $ Amount: num  149.62 2.69 378.66 123.5 69.99 ...
 $ Class : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
> table(creditcard$Class)

     0      1 
284315    492 
> # configuration of the sampling method used in the race
> ubConf <- list(percOver=200, percUnder=200, k=2, perc=50, method="percPos", w=NULL)
> # Race with 5 trees in the Random Forest
> results <- ubRacing(Class ~., creditcard, "randomForest",
+                      positive=1, metric="auc", ubConf=ubConf, ntree=5)
```

The function `ubRacing` compares the 8 unbalanced methods (ubUnder, ubOver, ubSMOTE, ubOSS, ubCNN, ubENN, ubNCL, ubTomek) against the unbalanced distribution.

Racing for Strategy Selection

```
Racing for unbalanced methods selection in 10 fold CV
Number of candidates..........................................9
Max number of folds in the CV................................10
Max number of experiments...................................100
Statistical test.................................Friedman test

                           Markers:
                              x No test is performed.
                              - The test is performed and
                                some candidates are discarded.
                              = The test is performed but
                                no candidate is discarded.

+-+-----------+-----------+-----------+-----------+-----------+
| |       Fold|      Alive|       Best|  Mean best| Exp so far|
+-+-----------+-----------+-----------+-----------+-----------+
|x|          1|          9|          4|     0.9543|          9|
|=|          2|          9|          3|     0.9433|         18|
|-|          3|          3|          4|     0.9567|         27|
|-|          4|          2|          4|     0.9566|         30|
|=|          5|          2|          4|     0.9582|         32|
|=|          6|          2|          4|     0.9546|         34|
|=|          7|          2|          4|     0.9531|         36|
|=|          8|          2|          4|     0.9539|         38|
|=|          9|          2|          4|     0.9531|         40|
|=|         10|          2|          4|     0.9529|         42|
+-+-----------+-----------+-----------+-----------+-----------+
Selected candidate: ubSMOTE      metric: auc     mean value: 0.9529
```

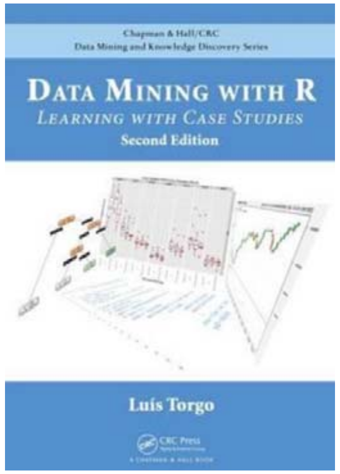
Racing for Strategy Selection

```
> results
$best
[1] "ubSMOTE"

$avg
[1] 0.9529177

$sd
[1] 0.009049014

$N.test
[1] 42

$Gain
[1] 53

$Race
         unbal    ubOver   ubUnder   ubSMOTE     ubOSS     ubCNN     ubENN     ubNCL   ubTomek
 [1,] 0.8844582 0.9138946 0.9354739 0.9543104 0.8957273 0.9139340 0.9024656 0.9014143 0.9048642
 [2,] 0.9116642 0.9104928 0.9511485 0.9507221 0.9037491 0.9104840 0.9139047 0.9094542 0.9105558
 [3,] 0.8979478 0.9013642 0.9502417 0.9649361 0.9092505 0.9081796 0.9103668 0.9036617 0.9058917
 [4,]        NA        NA 0.9503782 0.9564226        NA        NA 0.8999928        NA        NA
 [5,]        NA        NA 0.9537802 0.9647722        NA        NA        NA        NA        NA
 [6,]        NA        NA 0.9494913 0.9362763        NA        NA        NA        NA        NA
 [7,]        NA        NA 0.9411979 0.9440379        NA        NA        NA        NA        NA
 [8,]        NA        NA 0.9576971 0.9594249        NA        NA        NA        NA        NA
 [9,]        NA        NA 0.9530119 0.9473722        NA        NA        NA        NA        NA
[10,]        NA        NA 0.9633438 0.9509024        NA        NA        NA        NA        NA
```

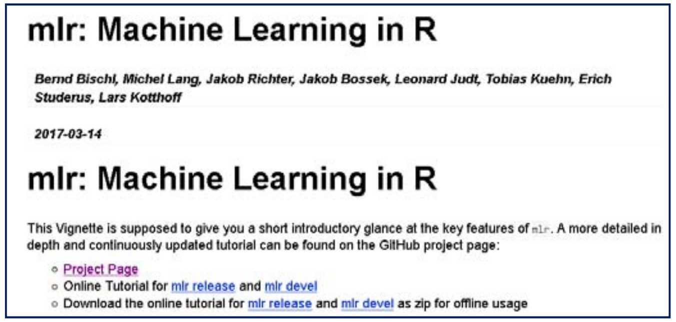
```
> # Race using 4 cores and 500 trees (default)
> results <- ubRacing(Class ~., creditcard, "randomForest",
                      positive=1, metric="auc", ubConf=ubConf, ncore=4)
> library(e1071)
> results <- ubRacing(Class ~., creditcard, "svm",
                      positive=1, ubConf=ubConf)
> library(rpart)
> results <- ubRacing(Class ~., creditcard, "rpart",
                      positive=1, ubConf=ubConf)
```

Useful R Packages

imbalance: Preprocessing Algorithms for Imbalanced Datasets
https://cran.r-project.org/web/packages/imbalance/index.html
Working with imbalanced datasets
https://cran.r-project.org/web/packages/imbalance/vignettes/imbalance.pdf



mlr: Machine Learning in R
https://cran.r-project.org/web/packages/mlr/vignettes/mlr.html



mlr: Machine Learning in R

Bernd Bischl, Michel Lang, Jakob Richter, Jakob Bossek, Leonard Judt, Tobias Kuehn, Erich Studerus, Lars Kotthoff

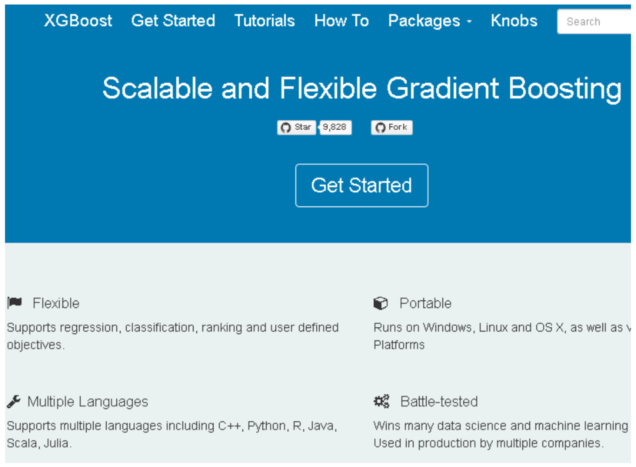
2017-03-14

mlr: Machine Learning in R

This Vignette is supposed to give you a short introductory glance at the key features of `mlr`. A more detailed in depth and continuously updated tutorial can be found on the GitHub project page:

- Project Page
- Online Tutorial for mlr release and mlr devel
- Download the online tutorial for mlr release and mlr devel as zip for offline usage

DMwR: Functions and data for "Data Mining with R"
https://cran.r-project.org/web/packages/DMwR/index.html



Scalable and Flexible Gradient Boosting

Flexible
Supports regression, classification, ranking and user defined objectives.

Portable
Runs on Windows, Linux and OS X, as well as v
Platforms

Multiple Languages
Supports multiple languages including C++, Python, R, Java, Scala, Julia.

Battle-tested
Wins many data science and machine learning
Used in production by multiple companies.

XGBoost: eXtreme Gradient Boosting
(used for supervised learning tasks such as Regression, Classification, and Ranking)
https://github.com/dmlc/xgboost
http://xgboost.readthedocs.io/en/latest/
How to use XGBoost algorithm in R in easy steps
https://www.analyticsvidhya.com/blog/2016/01/xgboost-algorithm-easy-steps/
Kaggle 神器 XGBoost 入門：為什麼要用它?怎麼用?
https://weiwenku.net/d/100778240

進階選讀

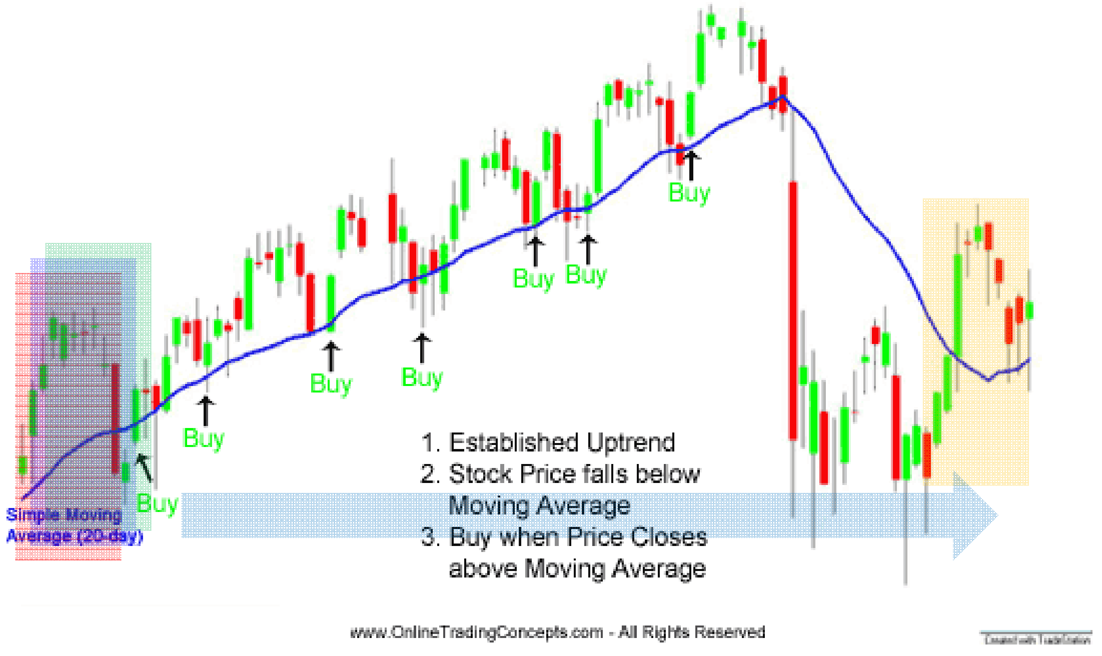
Simple Moving Average

- A moving average (移動平均) (簡稱均線) is a calculation to analyze data points by creating series of averages of different subsets of the full data set.

$$SMA = \frac{p_1 + p_2 + \cdots + p_n}{n}$$

When price is in an uptrend and subsequently, the moving average is in an uptrend, and the moving average has been tested by price and price has bounced off the moving average a few times (i.e. the moving average is serving as a support line), then a trader might buy on the **next pullbacks back** to the Simple Moving Average.

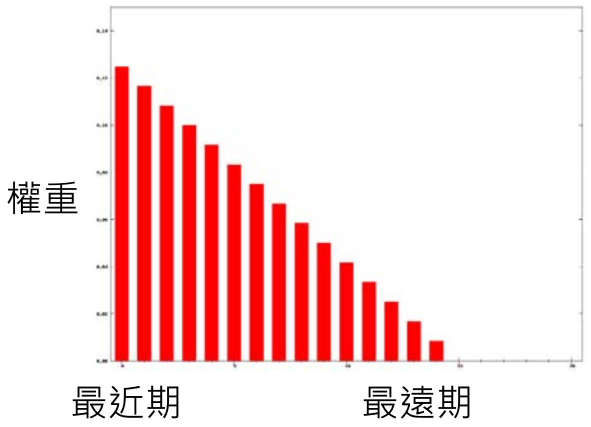
Moving Average Acting as Support - Potential Buy Signal

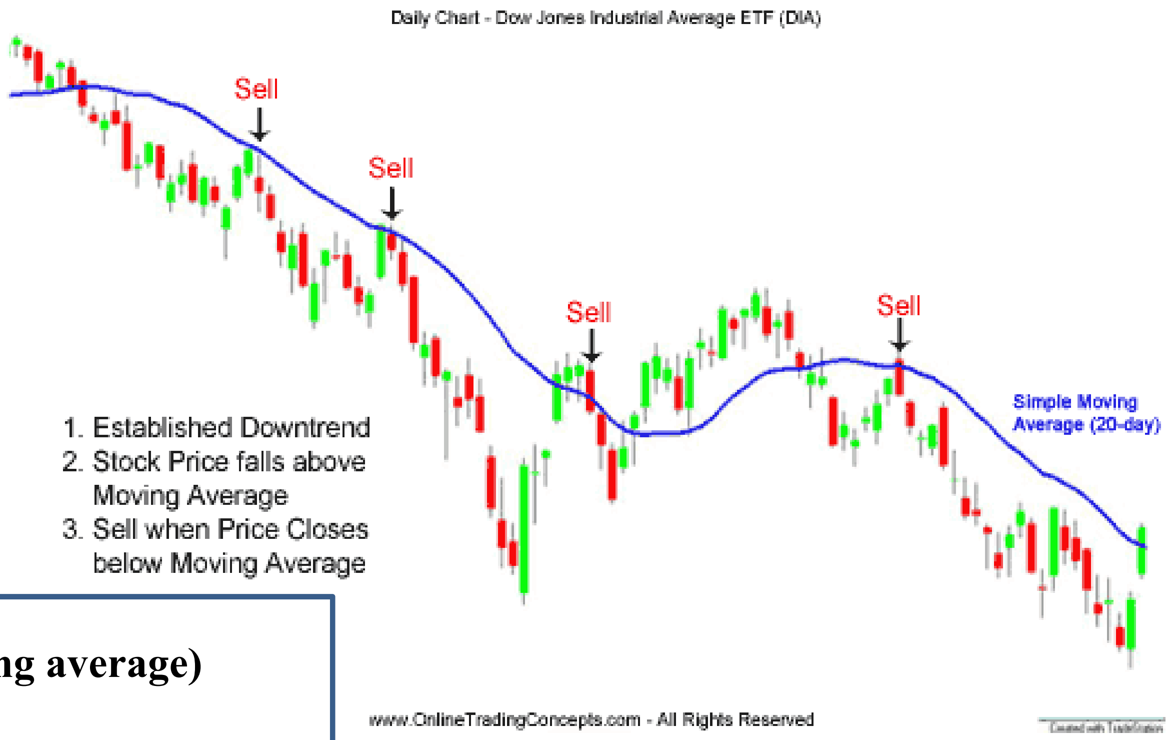


http://www.onlinetradingconcepts.com/TechnicalAnalysis/MASimple.html

Moving Average Acting as Resistance - Potential Sell Signal

- At times when price is in a downtrend and the moving average is in a downtrend as well, and price tests the SMA above and is rejected a few consecutive times (i.e. the moving average is serving as a resistance line), then a trader might sell on the next rally up to the Simple Moving Average.





An n-day WMA (Weighted moving average)

$$\mathrm{WMA}_M = \frac{np_M + (n-1)p_{M-1} + \cdots + 2p_{(M-n+2)} + p_{(M-n+1)}}{n + (n-1) + \cdots + 2 + 1}$$

http://www.onlinetradingconcepts.com/TechnicalAnalysis/MASimple.html

Smoothing in R

smooth: Forecasting Using Smoothing Functions

https://cran.r-project.org/web/packages/smooth/index.html

- `es()` - Exponential Smoothing;
- `ssarima()` - State-Space ARIMA, also known as Several Seasonalities ARIMA;
- `ces()` - Complex Exponential Smoothing;
- `ges()` - Generalised Exponential Smoothing;
- `ves()` - Vector Exponential Smoothing;
- `sma()` - Simple Moving Average in state-space form;

TTR: Technical Trading Rules

https://cran.r-project.org/web/packages/TTR/index.html

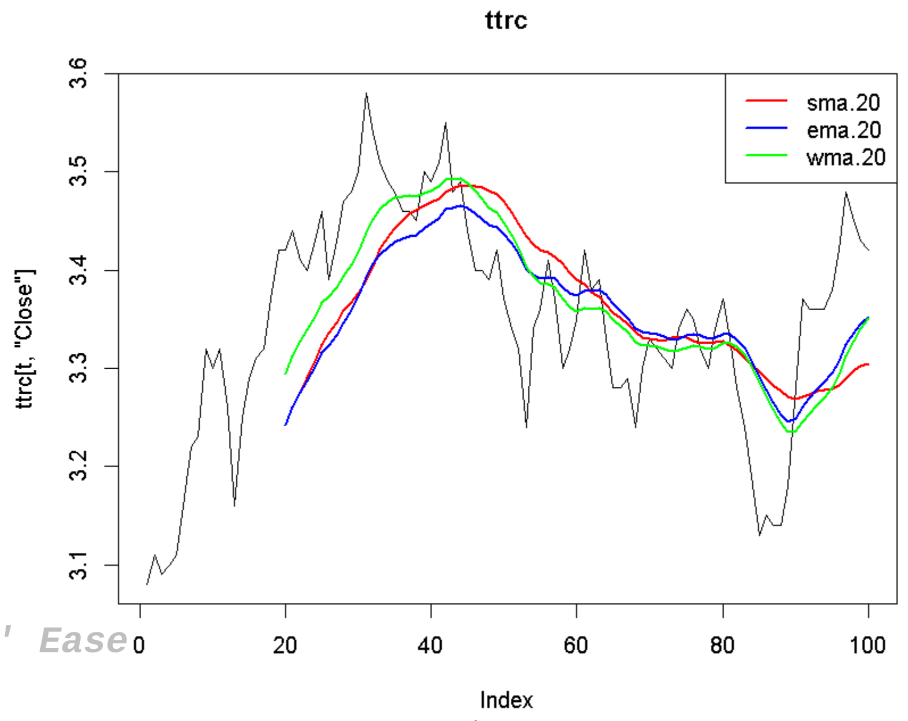
```
SMA(x, n = 10, ...)
EMA(x, n = 10, wilder = FALSE, ratio = NULL, ...)
DEMA(x, n = 10, v = 1, wilder = FALSE, ratio = NULL)
WMA(x, n = 10, wts = 1:n, ...)
EVWMA(price, volume, n = 10, ...)
ZLEMA(x, n = 10, ratio = NULL, ...)
VWAP(price, volume, n = 10, ...)
VMA(x, w, ratio = 1, ...)
HMA(x, n = 20, ...)
ALMA(x, n = 9, offset = 0.85, sigma = 6, ...)
```

Example

ttrc {TTR}: Technical Trading Rule Composite data

Historical Open, High, Low, Close, and Volume data for the periods January 2, 1985 to December 31, 2006. Randomly generated.

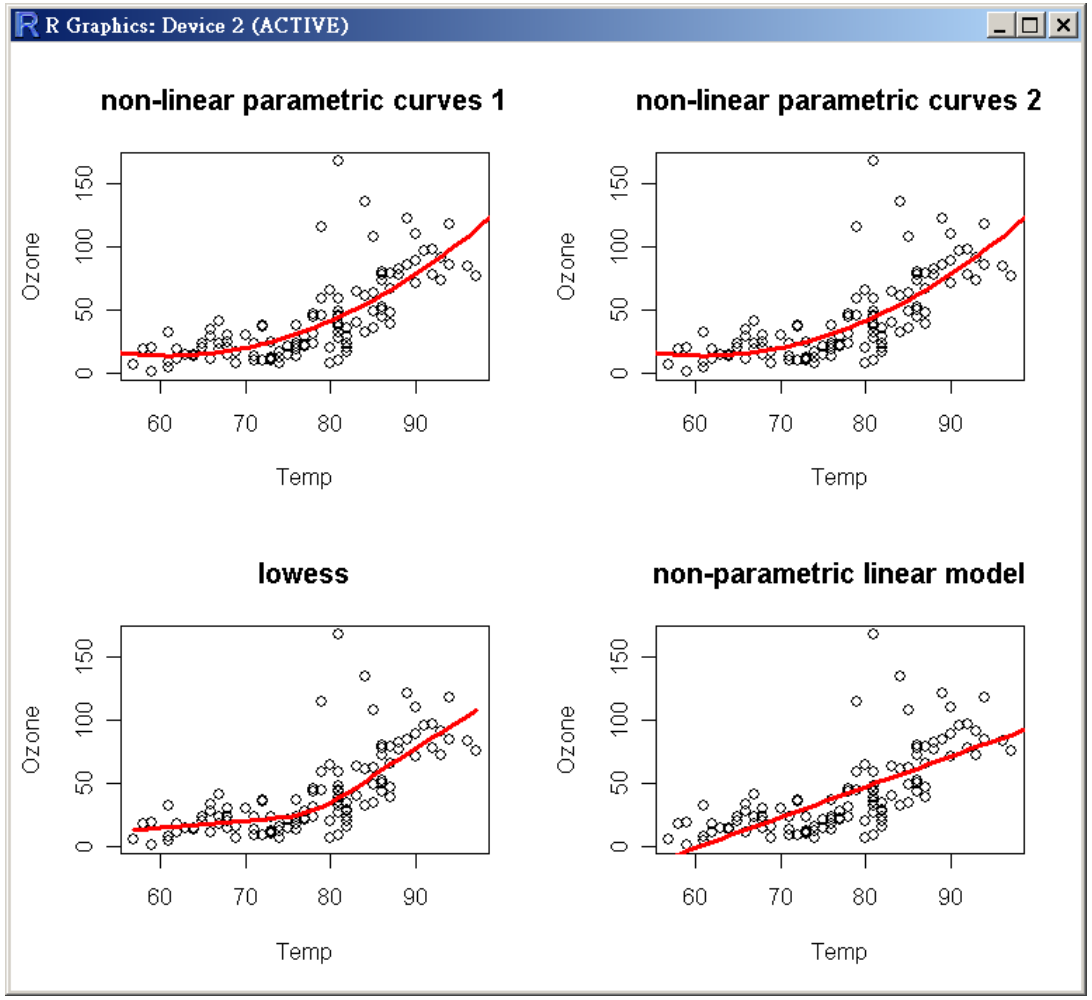
```
> # install.packages("TTR")
> library(TTR)
> data(ttrc)
> dim(ttrc)
[1] 5550    6
> head(ttrc)
        Date Open High  Low Close  Volume
1 1985-01-02 3.18 3.18 3.08  3.08 1870906
2 1985-01-03 3.09 3.15 3.09  3.11 3099506
3 1985-01-04 3.11 3.12 3.08  3.09 2274157
4 1985-01-07 3.09 3.12 3.07  3.10 2086758
5 1985-01-08 3.10 3.12 3.08  3.11 2166348
6 1985-01-09 3.12 3.17 3.10  3.16 3441798

> t <- 1:100
> sma.20 <- SMA(ttrc[t, "Close"], 20)
> ema.20 <- EMA(ttrc[t, "Close"], 20) # Arms' Ease
> wma.20 <- WMA(ttrc[t, "Close"], 20)
>
> plot(ttrc[t,"Close"], type="l", main="ttrc")
> lines(sma.20, col="red", lwd=2)
> lines(ema.20, col="blue", lwd=2)
> lines(wma.20, col="green", lwd=2)
> legend("topright", legend=c("sma.20", "ema.20", "wma.20"),
+         col=c("red", "blue", "green"), lty=1, lwd=2)
```



曲線配適 (Fitting Curves)

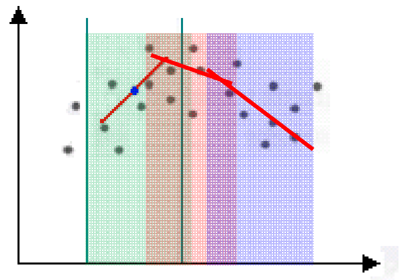
Example Methods

- non-linear parametric curves
- lowess
  (a non-parametric curve fitter)
- loess (a modelling tool)
- gam (fits generalized additive models)
- lm (linear model)

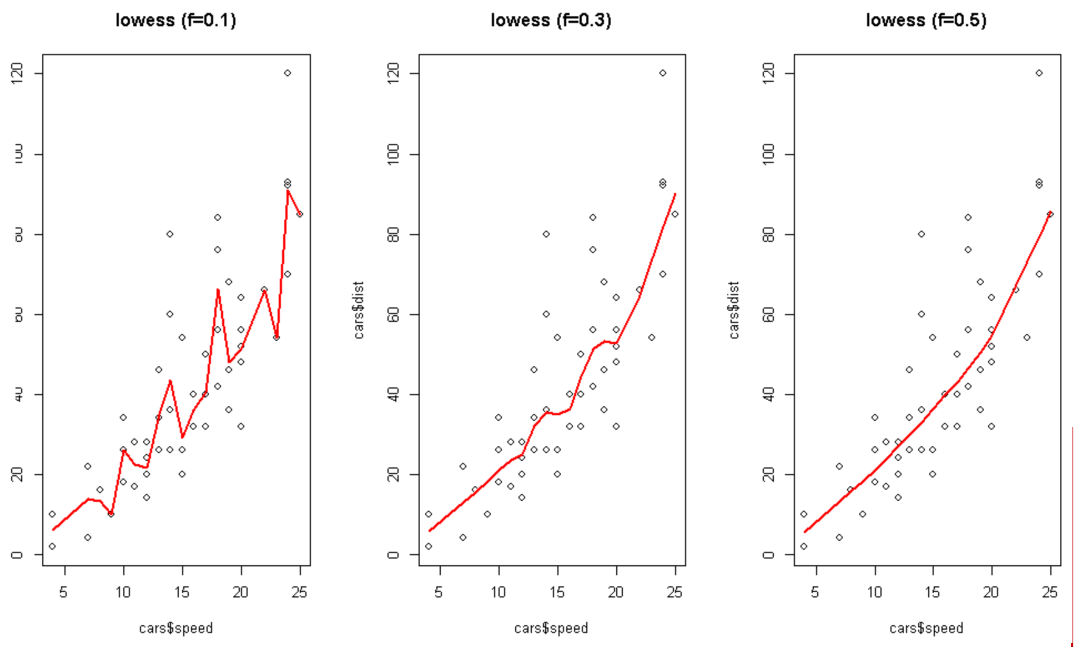


lowess {stats}
locally-weighted polynomial regression

Loess regression
(locally weighted polynomial regression)



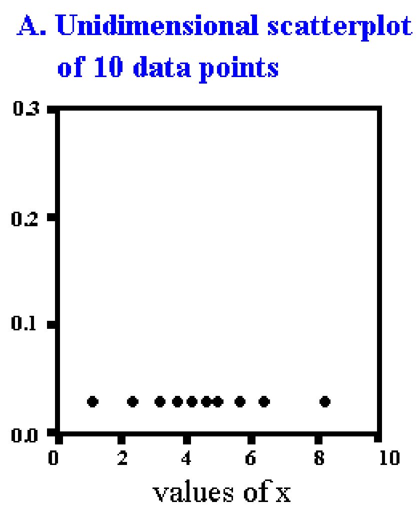
```
> data(cars)
> dim(cars)
[1] 50  2
> head(cars)
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
> par(mfrow=c(1, 3))
> for(i in c(0.1, 0.3, 0.5)){
+   plot(cars$dist ~ cars$speed, main=paste0("lowess (f=", i,")"))
+   lines(lowess(cars$dist ~ cars$speed, f = i), col="red", lwd=2)
+ }
```

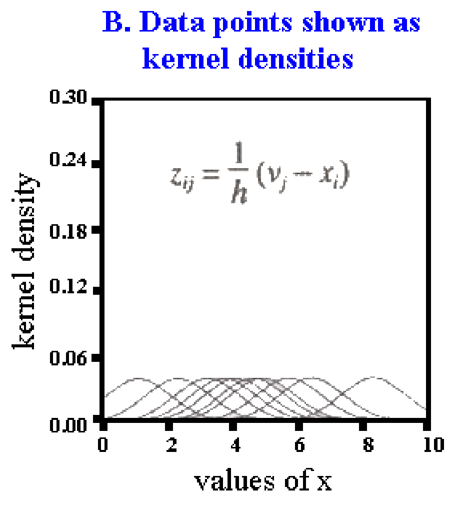


cars {datasets}:
The data give the speed of cars and the distances taken to stop. Note that the data were recorded in the 1920s.

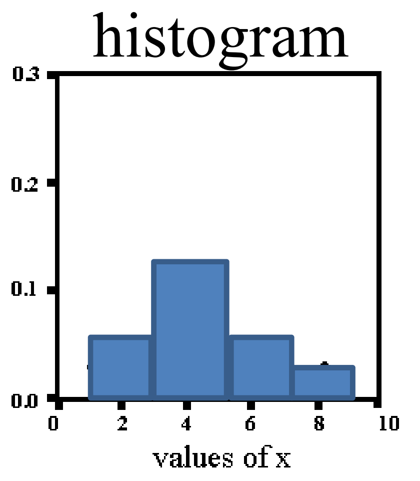
Density Plots (Smoothed Histograms) (1/3)

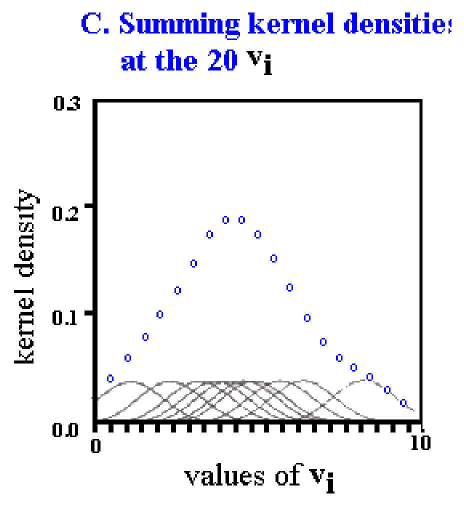
Constructing a Smoothed Histogram (Jacoby, 1997)

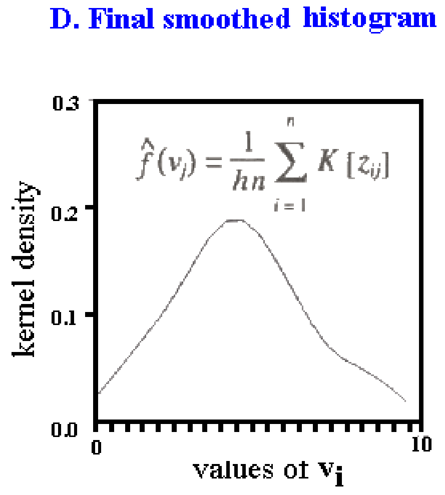




$$z_{ij} = \frac{1}{h}(v_j - x_i)$$





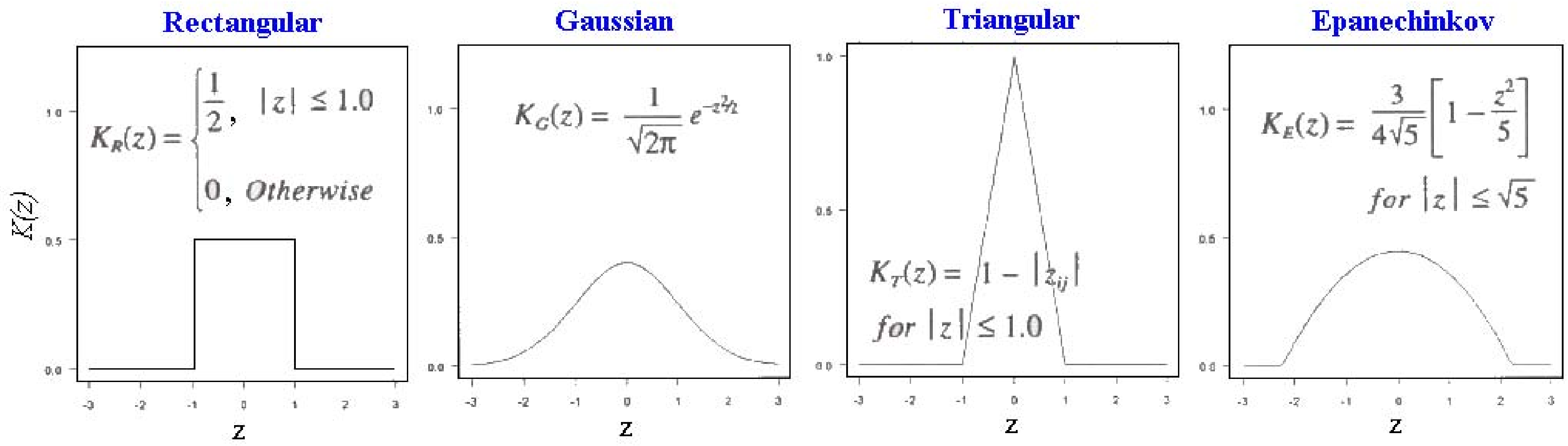


$$\hat{f}(v_j) = \frac{1}{hn}\sum_{i=1}^{n} K[z_{ij}]$$

Kernel Density Estimation

- Selection of **kernels (K)**
- Selection of **bandwidth (h)**

Figures modified from Jacoby (1997)



nonparametric regression

$$y_i = f_0(x_i) + \epsilon_i, \quad i = 1, \dots n,$$

$\epsilon_1, \dots \epsilon$ are still i.i.d. random errors with $\mathbb{E}(\epsilon_i) = 0$

$$\hat{f}(v_i) = \frac{1}{hn} \sum_{i=1}^{n} K[z_{ij}]$$

$$z_{ij} = \frac{1}{h}(v_j - x_i)$$

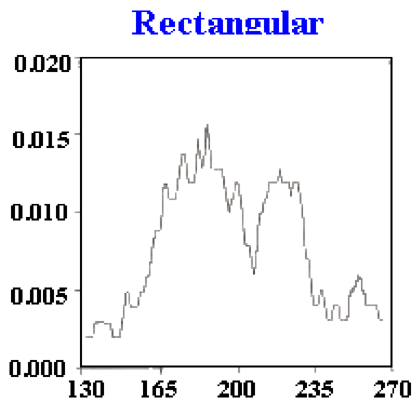
kernel regression

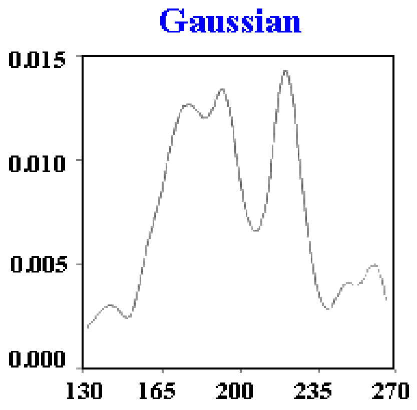
$$\hat{f}(x) = \frac{\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right) y_i}{\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)}$$

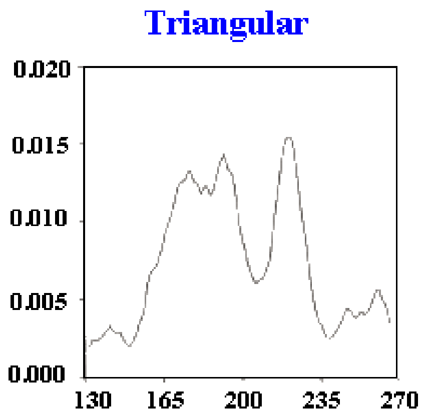
Kernel Density Estimation

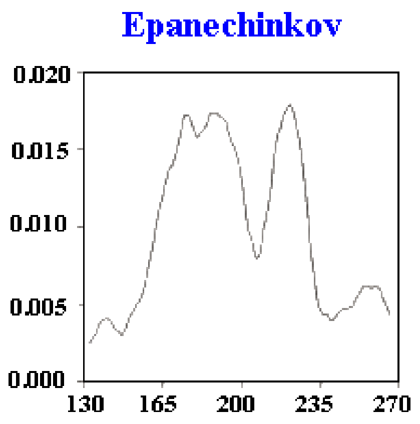
進階選讀

Different kernels

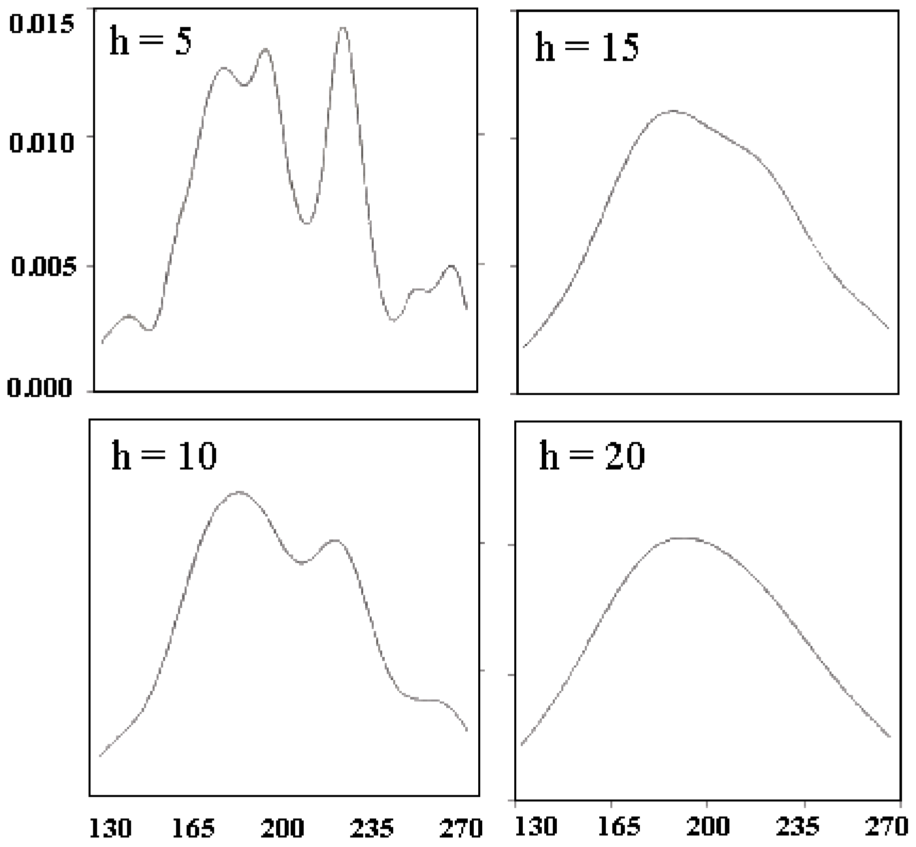








Different bandwidth

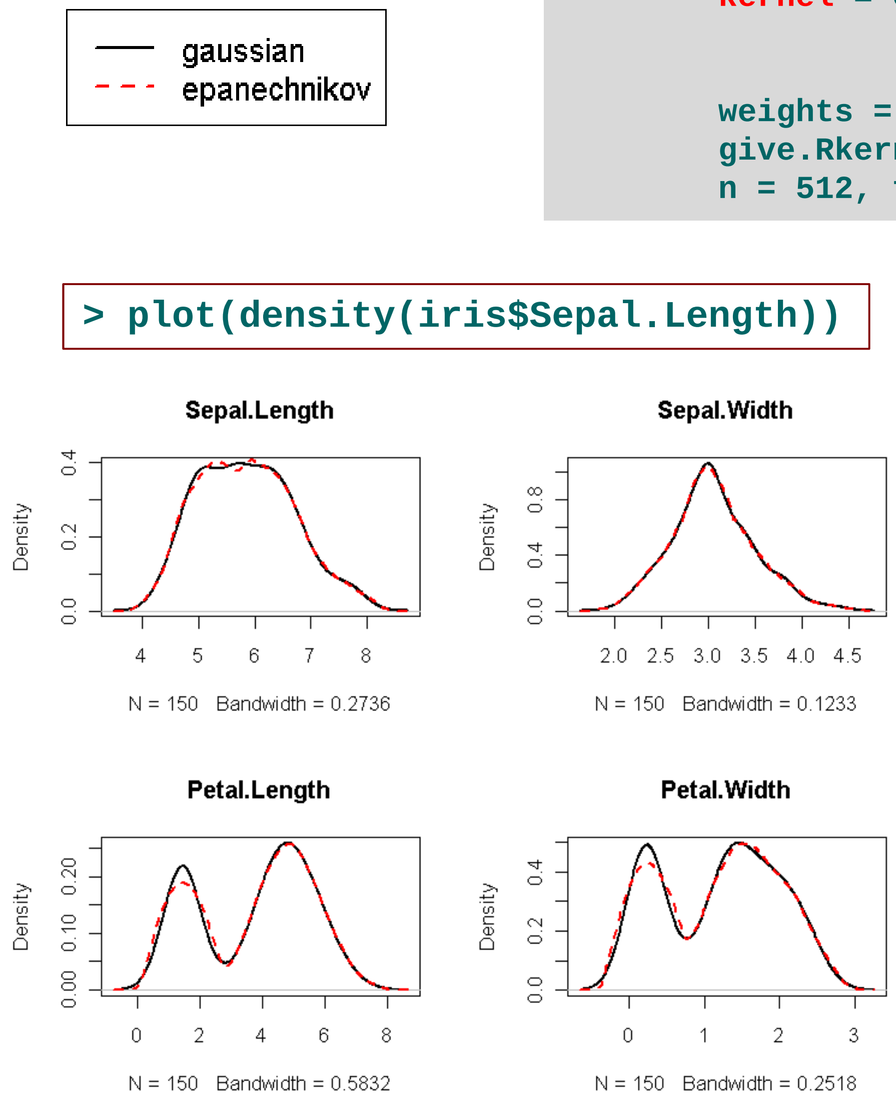


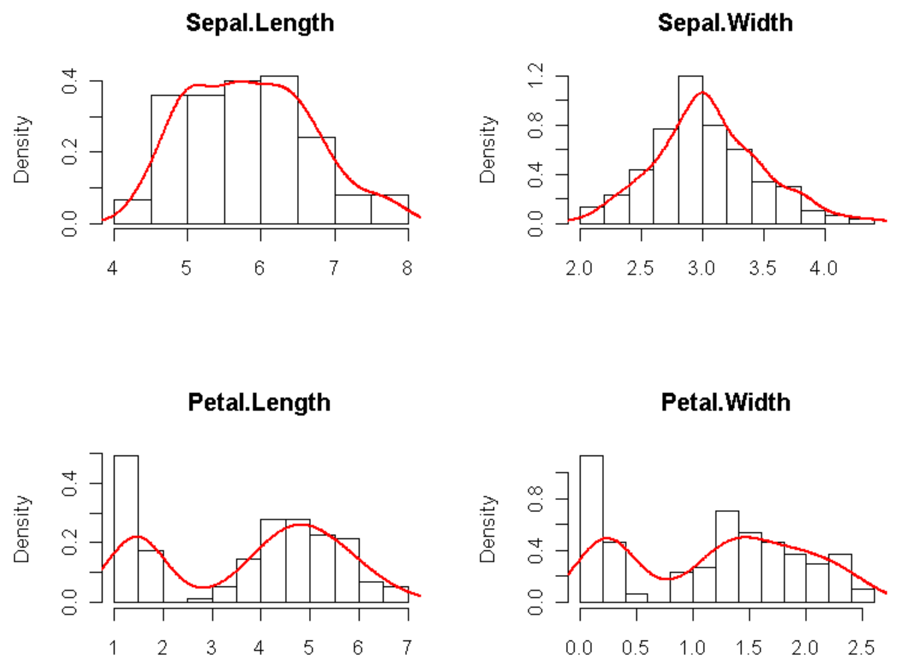
Kernel Density Estimation in R

```
density(x, bw = "nrd0", adjust = 1,
        kernel = c("gaussian", "epanechnikov", "rectangular",
                   "triangular", "biweight",
                   "cosine", "optcosine"),
        weights = NULL, window = kernel, width,
        give.Rkern = FALSE,
        n = 512, from, to, cut = 3, na.rm = FALSE, ...)
```

```
> plot(density(iris$Sepal.Length))
```

```
> hist(iris$Sepal.Length, prob=T)
> lines(density(iris$Sepal.Length), col="red")
```

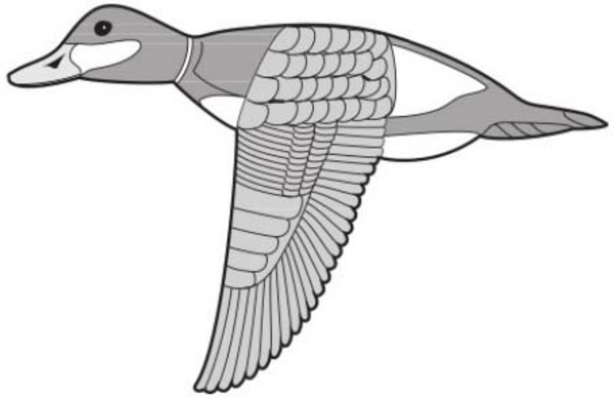


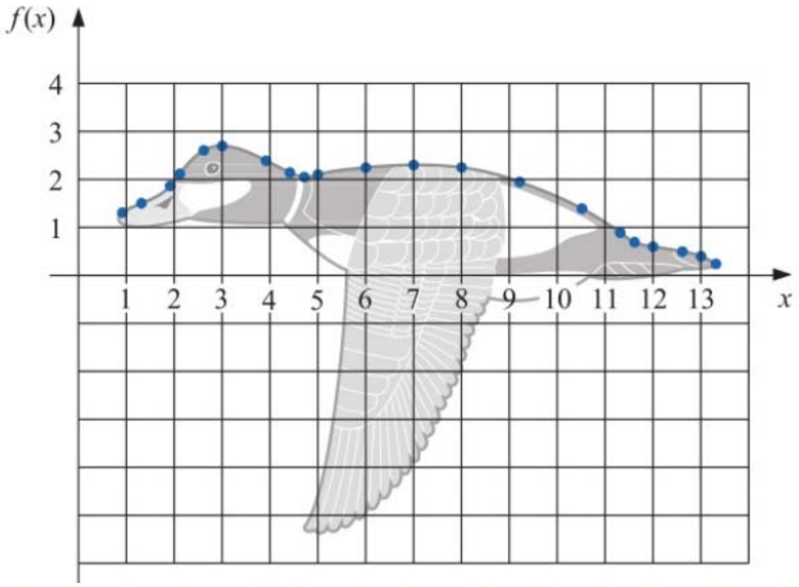


Spline approximate to the top profile of the ruddy duck

Ruddy duck (棕硬尾鴨) (雄)

"棕硬尾鴨棲息在北美洲的沼澤湖及池中，在南美洲的安地斯山脈也有分布。" https://zh.wikipedia.org/wiki/棕硬尾鴨





x	0.9	1.3	1.9	2.1	2.6	3.0	3.9	4.4	4.7	5.0	6.0	7.0	8.0	9.2	10.5	11.3	11.6	12.0	12.6	13.0	13.3
$f(x)$	1.3	1.5	1.85	2.1	2.6	2.7	2.4	2.15	2.05	2.1	2.25	2.3	2.25	1.95	1.4	0.9	0.7	0.6	0.5	0.4	0.25

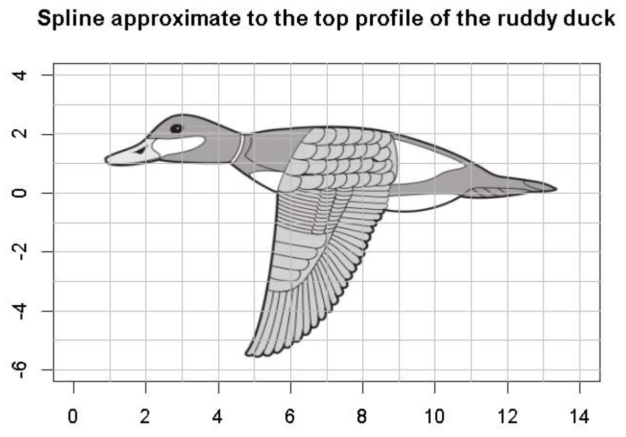
smooth.spline {stats}: Fit a Smoothing Spline

三次樣條插值法

Usage

```
smooth.spline(x, y = NULL, w = NULL, df, spar = NULL, lambda = NULL, cv = FALSE,
              all.knots = FALSE, nknots = .nknots.smspl,
              keep.data = TRUE, df.offset = 0, penalty = 1,
              control.spar = list(), tol = 1e-6 * IQR(x), keep.stuff = FALSE)
```

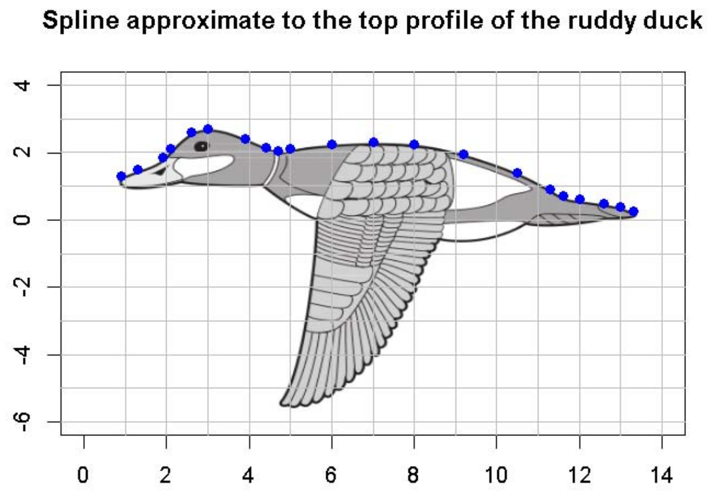
```
> #install.packages("jpeg")
> library(jpeg)
> ruddyduck.img <- readJPEG("ruddyduck.jpg")
> plot(0, xlim=c(0, 14), ylim=c(-6, 4), type='n', xlab="", ylab="",
+      main="Spline approximate to the top profile of the ruddy duck")
> rasterImage(ruddyduck.img, 0.6, -6, 13.8, 3.3)
> abline(v=1:14, h=-6:4, col=gray)
```

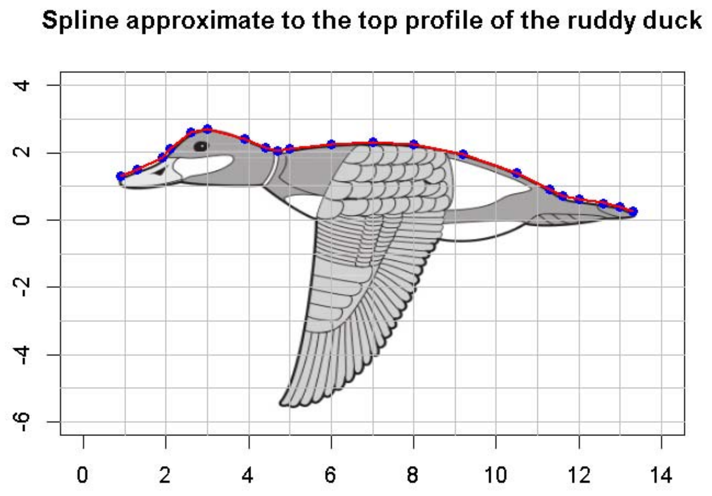


smooth.spline {stats}: Fit a Smoothing Spline

進階選讀

```
> ruddyduck.dat <- read.table("ruddyduck.txt", header=T, sep="\t")
> head(ruddyduck.dat)
    x   fx
1 0.9 1.30
2 1.3 1.50
3 1.9 1.85
4 2.1 2.10
5 2.6 2.60
6 3.0 2.70
> points(ruddyduck.dat, col="blue", pch=16)
>
> duck.spl <- smooth.spline(ruddyduck.dat$fx ~ ruddyduck.dat$x)
> lines(duck.spl, col = "red", lwd=2)
```





Cubic Spline Interpolation

Cubic Splines Interpolant

Definition 3.10

Given a function f defined on $[a, b]$ and a set of nodes $a = x_0 < x_1 < \cdots < x_n = b$, a cubic spline interpolant S for f is a function that satisfies the following conditions:

(a) $S(x)$ is a cubic polynomial ($S_j(x)$) on $[x_j, x_{j+1}]$.

(b) $S_j(x_j) =$ $f(x_j)$ and $S_j(x_{j+1}) =$ $f(x_{j+1})$, $j = 0, 1, \cdots, n-1$;

(c) $S_{j+1}(x_{j+1}) =$ $S_j(x_{j+1})$; (d) $S'_{j+1}(x_{j+1}) =$ $S'_j(x_{j+1})$;
(e) $S''_{j+1}(x_{j+1}) =$ $S''_j(x_{j+1})$ for each $j = 0, 1, \cdots, n-2$;

(f) One of the following sets of boundary conditions is satisfied:

(i) $S''(x_0) = S''(x_n) = 0$ (natural or free boundary);
(ii) $S'(x_0) = f'(x_0)$ and $S'(x_n) = f'(x_n)$ (clamped boundary).

Construction of a Cubic Spline (conti.)

(12) This system involves only the $\{c_j\}_{j=0}^{n}$ as unknowns.

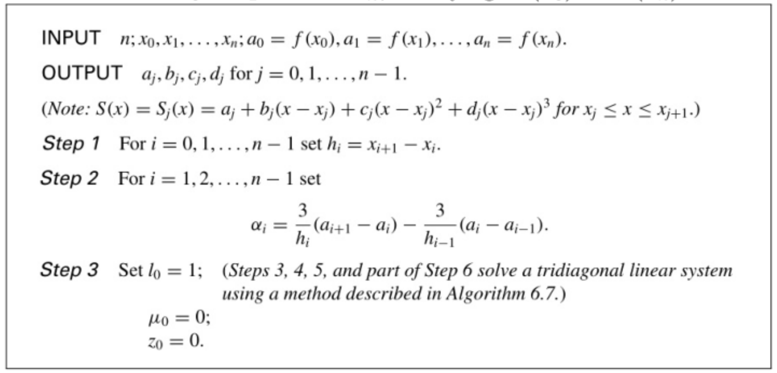
(13) The values of $\{h_j\}_{j=0}^{n-1}$ and $\{a_j\}_{j=0}^{n}$ are given, respectively, by the spacing of the nodes $\{x_j\}_{j=0}^{n}$ and the values of f at the nodes.

(14) So once the values of $\{c_j\}_{j=0}^{n}$ are determined, it is a simple matter to find the remainder of the constants $\{b_j\}_{j=0}^{n-1}$ from Eq. (3.20) and $\{d_j\}_{j=0}^{n-1}$ from Eq. (3.17)

(15) The major question that arises in connection with this construction is whether the values of $\{c_j\}_{j=0}^{n}$ can be found using the system of equations given in (3.21) and, if so, whether these values are unique .

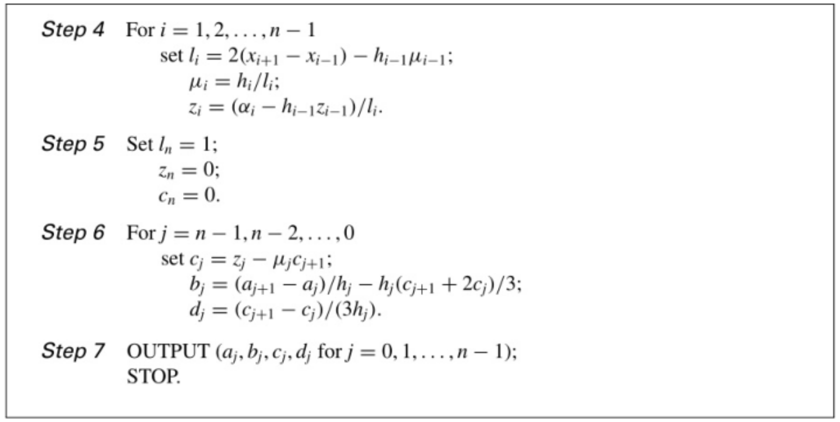
ALGORITHM 034: Natural Cubic Spline

To construct the cubic spline interpolant S for the function f, defined at the numbers $x_0 < x_1 < \cdots < x_n$, satisfying $S''(x_0) = S''(x_n) = 0$:



```
INPUT   $n; x_0, x_1, \ldots, x_n; a_0 = f(x_0), a_1 = f(x_1), \ldots, a_n = f(x_n)$.
OUTPUT  $a_j, b_j, c_j, d_j$ for $j = 0, 1, \ldots, n-1$.
(Note: $S(x) = S_j(x) = a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3$ for $x_j \le x \le x_{j+1}$.)
Step 1  For $i = 0, 1, \ldots, n-1$ set $h_i = x_{i+1} - x_i$.
Step 2  For $i = 1, 2, \ldots, n-1$ set
            $\alpha_i = \frac{3}{h_i}(a_{i+1} - a_i) - \frac{3}{h_{i-1}}(a_i - a_{i-1})$.
Step 3  Set $l_0 = 1$;   (Steps 3, 4, 5, and part of Step 6 solve a tridiagonal linear system
                          using a method described in Algorithm 6.7.)
            $\mu_0 = 0$;
            $z_0 = 0$.
```

ALGORITHM 034: Natural Cubic Spline (conti.)



```
Step 4  For $i = 1, 2, \ldots, n-1$
            set $l_i = 2(x_{i+1} - x_{i-1}) - h_{i-1}\mu_{i-1}$;
                $\mu_i = h_i / l_i$;
                $z_i = (\alpha_i - h_{i-1} z_{i-1}) / l_i$.
Step 5  Set $l_n = 1$;
            $z_n = 0$;
            $c_n = 0$.
Step 6  For $j = n-1, n-2, \ldots, 0$
            set $c_j = z_j - \mu_j c_{j+1}$;
                $b_j = (a_{j+1} - a_j)/h_j - h_j(c_{j+1} + 2c_j)/3$;
                $d_j = (c_{j+1} - c_j)/(3h_j)$.
Step 7  OUTPUT ($a_j, b_j, c_j, d_j$ for $j = 0, 1, \ldots, n-1$);
        STOP.
```