

# R Programming as a Part of Bigdata Course

## Clustering

John Aa. Sørensen, lektor  
Section for Information Technology, DTU Diplom

23 February 2017

# Examples on Clustering Preparation Steps

- # Step 1: Choose appropriate attributes.
- # Step 2: Scale the data.
- # Step 3: Screen for outliers.
- # Step 4: Calculate distances.
- # Step 5: Select a clustering algorithm.
- # Hierarchical and/or partitioning cluster analysis.
- # Step 5: Repeat using outlier substitution with median.
- # Step 5: Repeat using "single" link clustering.
- # Step 6: Obtain one or more cluster solutions.
- # Step 7: Determine the number of clusters present.
- # Step 8: Obtain final clustering solution.
- # Step 9: Visualize the results.

# Examples on References of Relevance to Clustering

- ▶ The main reference is [Kabacoff, 2015] Chap. 16 "Clustering Analysis"
- ▶ and a selection of R packages and their corresponding background papers typical in Journal of Statistical Software,
- ▶ e.g. [Charrad, 2014] NbClust: An R Package for Determining the Relevant Number of Clusters in a Data Set
- ▶ and [Pison, 1999] Displaying a Clustering with CLUSPLOT, from Elsevier.

## Choose Attributes

```
> # 5.1: Step 1: Choose appropriate attributes.
> # Example p. 373 [Kabacoff, 2015]
> data(nutrient, package="flexclust")
> str(nutrient)
'data.frame': 27 obs. of 5 variables:
 $ energy : int  340 245 420 375 180 115 170 160 265 300 ...
 $ protein: int  20 21 15 19 22 20 25 26 20 18 ...
 $ fat     : int  28 17 39 32 10 3 7 5 20 25 ...
 $ calcium: int   9 9 7 9 17 8 12 14 9 9 ...
 $ iron    : num  2.6 2.7 2 2.6 3.7 1.4 1.5 5.9 2.6 2.3 ...
> head(nutrient, 3) # Display upper 3 rows from 5 dim. DS
```

	energy	protein	fat	calcium	iron
BEEF BRAISED	340	20	28	9	2.6
HAMBURGER	245	21	17	9	2.7
BEEF ROAST	420	15	39	7	2.0

# Scale Attributes

```
#####  
# 5.2: Step 2: Scale the data.  
#  
# Scaling: For each attribute (variable):  
#   Compute mean value, subtract mean value => mean = 0  
#   Compute spread, normalize with spread => variance = 1  
#       The data scaling is included in Step. 5.  
#
```

# Scale Attributes

```
#####  
# 5.3: Step 3: Screen for outliers.  
#       This step is exemplified using ref. [Filzmoser, 2005].  
#  
# Ref. [Kabacoff, 2015] p. 371  
# screen and remove univariate outliers using package: outliers  
# install.packages("outliers") # Moved to top of this script.  
# install.packages("mvoutlier") # Moved to top of this script.  
# library(outliers)  
# library(mvoutlier)  
# help(package="outliers")      # Single variable outliers.  
# help(package="mvoutliers")    # Multivariable outliers.  
#
```

# Scale Attributes

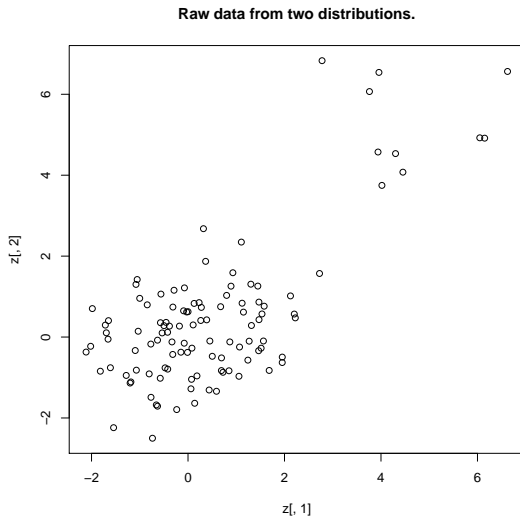
```
# ref. [Filzmoser, 2005], package "mvoutliers"  
# Demonstrate the function symbol.plot()  
# Objective:  
# A method for multivariate outlier detection able  
# to distinguish between extreme values of a normal  
# distribution and values originating from a different  
# distribution (outliers).  
# Method: Find subset of observations, h, with  
# MCD (Minimum Covariance Determinant).  
#
```

## Scale Attributes

```
# Create data matrices with two different distributions:
# Matrix x: Two column vectors with 100 rows of std. normal.
x <- cbind(rnorm(100), rnorm(100))
# Matrix y: Two column vectors 10 rows of norm. mean=5, std=1
y <- cbind(rnorm(10, 5, 1), rnorm(10, 5, 1))
# Matrix z: Two column vectors, each with 110 rows.
z <- rbind(x,y)
str(z)          # Check structure of resulting z matrix
head(z, 10)    # Check the top 20 rows in z.
par(opar.org)
plot(z[,1],z[,2])
title("Raw data from two distributions.")
```



# Raw data for outlier detection experiment.



## Boolean with outlier True, False

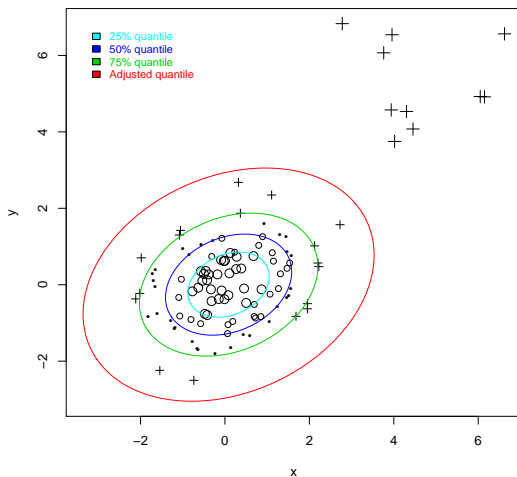
```
> symbol.plot(z, quan=0.75)
$outliers
[1] FALSE FALSE FALSE FALSE FALSE FALSE ...
[20] FALSE FALSE FALSE FALSE FALSE FALSE ...
[39] FALSE FALSE FALSE FALSE FALSE FALSE ...
[58] FALSE FALSE FALSE FALSE FALSE FALSE ...
[77] FALSE FALSE FALSE FALSE FALSE FALSE ...
[96] FALSE FALSE FALSE FALSE FALSE  TRUE
[102] TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

# Mahalanobis distance

```
$md
```

```
[1] 1.5488775 0.6202039 0.5959240 2.2306050 ...  
[12] 0.9678372 0.3507535 0.6464982 0.2694994 ...  
.  
.  
.  
[78] 0.3398362 0.9764323 0.5955232 1.5111894 ...  
[89] 2.1523832 1.6060837 1.1749813 1.3489594 ...  
[100] 0.3659154 4.0687445 5.7896500 6.9665733 ...  
[105] 5.6263406 6.1093877 4.6828475 6.0451329 ...
```

`symbol.plot()` for outlier removal.



# Distance Calculation I

```
#####  
# 5.4: Step 4: Calculate distances.  
#  
# Compute dist. matrix btw. first 4 rows of nutrient,  
# using the dist() function in R base installation.  
# dist() contains the following distances using ?dist()  
#      "euclidean", "maximum", "manhattan",  
#      "canberra", "binary", "minkowski"  
#  
?dist()          # Check manual.  
d <- dist(nutrient) #  
as.matrix(d)[1:4,1:4] # Symm. dist. matrix, 0 diagonal.  
#      Notice the very different numerical ranges  
#      of the variables => need for scaling.
```

## Distance Calculation II

```
> d <- dist(nutrient)
> as.matrix(d)[1:4,1:4] # Symm. dist. mtx, 0 diagonal.
```

	BEEF BRAISED	HAMBURGER	BEEF ROAST	BEEF STEAK
BEEF BRAISED	0.00000	95.6400	80.93429	35.24202
HAMBURGER	95.64000	0.0000	176.49218	130.87784
BEEF ROAST	80.93429	176.4922	0.00000	45.76418
BEEF STEAK	35.24202	130.8778	45.76418	0.00000

```
>
```

## Distance Calculation III

```
# If mixed data types:  
# binary, nominal (category), ordinal (eg. Likert)  
# or continuous (e.g. temperature), use e.g. the daisy()  
# function in the cluster package.  
# Example on packages for clustering on mixed types:  
# Functions for agglomerative clustering: agnes()  
# Functions for partitioning around medoids: pam().
```

## Select a Clustering Algorithm

```
#####  
# 5.5: Step 5: Select a clustering algorithm.  
# Average-linkage clustering of nutrient data.  
# Listing 16.1 p. 375 [Kabacoff, 2015].  
#  
data(nutrient, package="flexclust")  
?tolower()          # Check manual.  
row.names(nutrient) <- tolower(row.names(nutrient))  
?scale()            # Check manual for the scaling function.  
nutrient.scaled <- scale(nutrient) # Scale all variables to  
                                # mean = 0 and spread = 1.
```

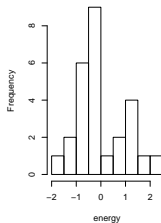


## Verify Scaling of the Attributes

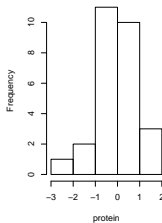
```
# Check the scaled variables properties.  
#   Mean values = 0  
#   Standard deviations = 1.  
#  
attributes(nutrient.scaled)  
# Verify that mean values are = 0.  
summary(nutrient.scaled)
```

# Scaled Histograms of Attributes for Clustering

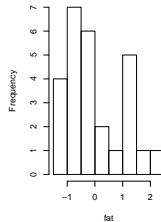
**Scaled Energy**



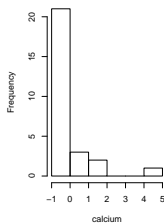
**Scaled Protein**



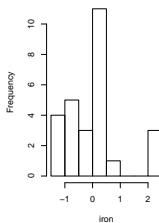
**Scaled Fat**



**Scaled Calcium**



**Scaled Iron**



# Euclidean Distance Matrix for scaled nutrient

```
#  
# Display the euclidean distance matrix of  
#  nutrient.scaled  
#  
?dist()      # Manual for dist()  
d_eucli <- dist(nutrient.scaled,"euclidean")  
d_eucli  
#
```

# Check Attributes of Dataset in d\_eucli

```
> attributes(d_eucli)
```

```
$Size
```

```
[1] 27
```

```
$Labels
```

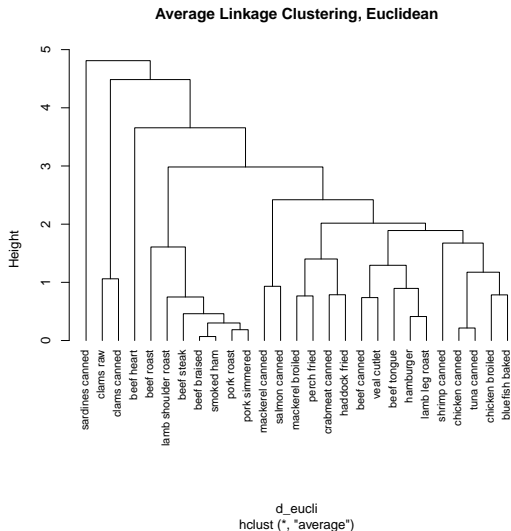
```
[1] "beef braised"      "hamburger"      ...  
[5] "beef canned"      "chicken broiled" ...  
[9] "lamb leg roast"   "lamb shoulder roast" ...  
[13] "pork simmered"    "beef tongue"    ...  
[17] "clams raw"        "clams canned"   ...  
[21] "mackerel broiled" "mackerel canned" ...  
[25] "sardines canned"  "tuna canned"    ...
```

```
#
```

# Average Link Clustering, Euclidean Distance

```
par(opar)
fit.average_eucli <- hclust(d_eucli, method="average")
plot(fit.average_eucli, hang=-1, cex=0.8,
     main="Average Linkage Clustering, Euclidean")
# hang=-1, the labels are below the plot, rotated 90 Degree.
# cex=.8, scale text cx times, [Kabacoff, 2015] p. 53.
#
pdf("fig_5_2_AV_Link_Clust.pdf")
par(opar)
plot(fit.average_eucli, hang=-1, cex=0.8,
     main="Average Linkage Clustering, Euclidean")
par(opar)
dev.off()
```

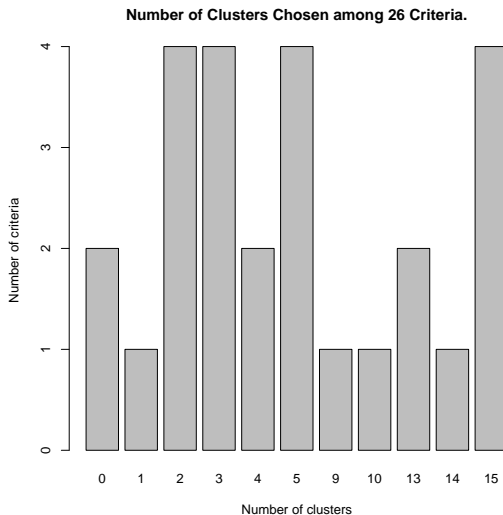
# Average Link Clustering, Euclidean Distance



# Estimate Number of Clusters

```
par(opar)
nc <- NbClust(nutrient.scaled, distance="euclidean",
              min.nc=2, max.nc=15, method="average")
table(nc$Best.n[1,])
barplot(table(nc$Best.n[1,]),
        xlab="Number of clusters", ylab="Number of criteria",
        main="Number of Clusters Chosen among 26 Criteria.")
par(opar)
#
```

# Estimation of the Number of Clusters





# Estimation of the Number of Clusters, [Charrad, 2014]

	Name of the index in <b>NbClust</b>	Optimal number of clusters
1.	"ch" (Calinski and Harabasz 1974)	Maximum value of the index
2.	"duda" (Duda and Hart 1973)	Smallest number of clusters such that index > criticalValue
3.	"pseudot2" (Duda and Hart 1973)	Smallest number of clusters such that index < criticalValue
4.	"cindex" (Hubert and Levin 1976)	Minimum value of the index
5.	"gamma" (Baker and Hubert 1975)	Maximum value of the index
6.	"beale" (Beale 1969)	Number of clusters such that critical value $\geq$ alpha
7.	"ccc" (Sarle 1983)	Maximum value of the index
8.	"ptbserial" (Milligan 1980, 1981)	Maximum value of the index
9.	"gplus" (Rohlf 1974; Milligan 1981)	Minimum value of the index
10.	"db" (Davies and Bouldin 1979)	Minimum value of the index
11.	"frey" (Frey and Van Groenewoud 1972)	Cluster level before index value < 1.00
12.	"hartigan" (Hartigan 1975)	Maximum difference between hierarchy levels of the index
13.	"tau" (Rohlf 1974; Milligan 1981)	Maximum value of the index
14.	"ratkowsky" (Ratkowsky and Lance 1978)	Maximum value of the index
15.	"scott" (Scott and Symons 1971)	Maximum difference between hierarchy levels of the index
16.	"marriot" (Marriot 1971)	Max. value of second differences between levels of the index
17.	"ball" (Ball and Hall 1965)	Maximum difference between hierarchy levels of the index
18.	"trcovw" (Milligan and Cooper 1985)	Maximum difference between hierarchy levels of the index
19.	"tracew" (Milligan and Cooper 1985)	Max. value of second differences between levels
20.	"friedman" (Friedman and Rubin 1967)	Maximum difference between hierarchy levels of the index
21.	"mcclain" (McClain and Rao 1975)	Minimum value of the index
22.	"rubin" (Friedman and Rubin 1967)	Minimum value of second differences between levels
23.	"kl" (Krzanowski and Lai 1988)	Maximum value of the index
24.	"silhouette" (Rousseeuw 1987)	Maximum value of the index
25.	"gap" (Tibshirani <i>et al.</i> 2001)	Smallest number of clusters such that criticalValue $\geq$ 0
26.	"dindex" (Lebart <i>et al.</i> 2000)	Graphical method
27.	"dunn" (Dunn 1974)	Maximum value of the index
28.	"hubert" (Hubert and Arabie 1985)	Graphical method
29.	"sdindex" (Halkidi <i>et al.</i> 2000)	Minimum value of the index
30.	"sdbw" (Halkidi and Vazirgiannis 2001)	Minimum value of the index

Table 2: Overview of the indices implemented in the **NbClust** package.

# Final Clustering Solution I

```
par(opar)
> # 5.8: Step 8: Obtain final clustering solution.
> #
> # Listing 16.3 page 377 [Kabacoff, 2015]
> #
> ?cutree()      # Manual for tree cutting.
# Use k=5 groups as desired.
> clusters <- cutree(fit.average_eucli, k=5)
> table(clusters)
clusters
 1  2  3  4  5
7 16  1  2  1
```

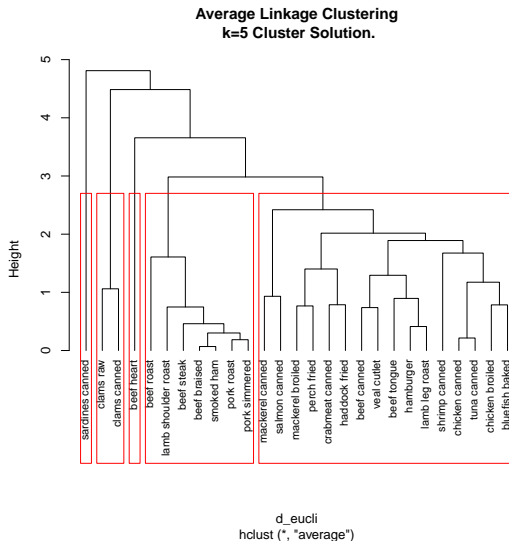
## Final Clustering Solution II

```
> aggregate(nutrient, by=list(cluster=clusters), median)
clust energy prote fat calc iron
  1    340.0   19    29   9  2.50
  2    170.0   20     8  13  1.45
  3    160.0   26     5  14  5.90
  4     57.5    9     1  78  5.70
  5    180.0   22     9 367  2.50
> #
> aggregate(as.data.frame(nutrient.scaled),
+ by=list(cluster=clusters), median)
clust energy prote   fat   calc   iron
  1  1.310  0.000  1.378 -0.4480  0.0811
  2 -0.369  0.235 -0.486 -0.3967 -0.6374
  3 -0.468  1.646 -0.753 -0.3839  2.4077
  4 -1.481 -2.352 -1.108  0.4361  2.2709
  5 -0.270  0.705 -0.398  4.1396  0.0811
> #
```

# Visualize Final Solution I

```
#  
# Listing 16.3 page 377 [Kabacoff, 2015]  
#  
par(opar)  
plot(fit.average_eucli, hang=-1, cex=0.8,  
     main="Average Linkage Clustering\n k=5 Cluster Solution.")  
# Display the k=5 cluster solution.  
rect.hclust(fit.average_eucli, k=5)  
#
```

# Visualize Final Solution II



# Partitioning Clustering k-means

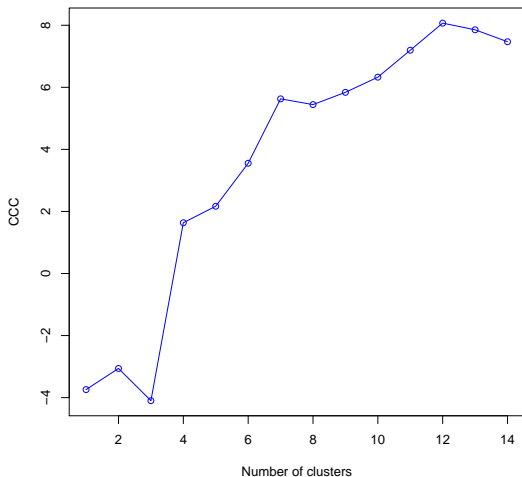
```
#####  
# 5.20: Examples using the Partitioning Clustering k-means  
#  
# Ref. page 379 [Kabacoff, 2015] "Plotting function of  
#   k-means clusters".  
#  
# Installed on top of script.  
# install.packages("rattle",lib="C:/R_packages/")  
# library("rattle",lib="C:/R_packages/")  
data(wine, package="rattle")
```

# Partitioning Clustering k-means, Dataset: Wine

```
str(wine)
'data.frame': 178 obs. of  14 variables:
 $ Type          : Factor w/ 3 levels "1","2","3": 1 1 1 1 .
 $ Alcohol       : num  14.2 13.2 13.2 14.4 13.2 ...
 $ Malic        : num  1.71 1.78 2.36 1.95 2.59 1.76 1.87 ...
 $ Ash          : num  2.43 2.14 2.67 2.5 2.87 2.45 2.45 ...
 $ Alcalinity    : num  15.6 11.2 18.6 16.8 21 15.2 14.6...
 $ Magnesium     : int  127 100 101 113 118 112 96 121 97 98 ..
 $ Phenols       : num  2.8 2.65 2.8 3.85 2.8 3.27 2.5 2 ...
 $ Flavanoids    : num  3.06 2.76 3.24 3.49 2.69 3. 3.15 ...
 $ Nonflavanoids : num  0.28 0.26 0.3 0.24 0.39 0.34 0.3 ...
 $ Proanthocyanins: num  2.29 1.28 2.81 2.18 1.82 1.97 1 ...
 $ Color         : num  5.64 4.38 5.68 7.8 4.32 6.75 5. ...
 $ Hue           : num  1.04 1.05 1.03 0.86 1.04 1.0 ...
 $ Dilution     : num  3.92 3.4 3.17 3.45 2.93 2.85 ...
 $ Proline       : int  1065 1050 1185 1480 735 1450 ...
```

# The CCC index for the Number of Clusters

The CCC (Cubic Cluster Criteria) from NbClust() for assessing the quality of the clustering, cf. [Kabacoff, 2015] page 387 and [Charrad, 2014]. If CCC is negative and decreasing for increasing number of clusters, then the dataset has a tendency to unimodality.

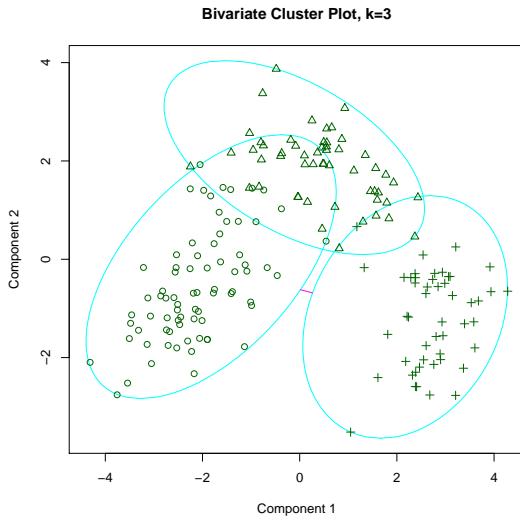




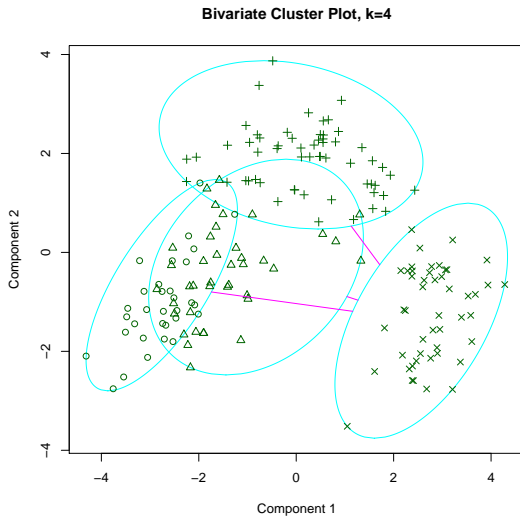
## Partitioning around Medoids

```
#####  
# 5.30: Examples using the Partitioning around medoids  
# Use dataset wine from package rattle  
# loaded in section 5.20: data(wine, package="rattle")  
#  
# library(cluster) moved to top of script  
set.seed(1234)  
fit.pam_3 <- pam(wine[-1], k=3, stand=TRUE)  
fit.pam_3$medoids  
clusplot(fit.pam_3, main="Bivariate Cluster Plot, k=3")  
#
```

# Partitioning around Medoids, I



# Partitioning around Medoids, II



These two components explain 55.41 % of the point variability.

# References I



Joseph Adler (2012)

R in a Nutshell

*O'Reilly*



Robert I. Kabacoff (2015)

R in Action

*Manning Publications 2'Ed.*



Malika Charrad, Nadia Ghazzali, Vronique Boiteau, Azam Niknafs (2014)

NbClust: An R Package for Determining Relevant Number of Clusters in a Data Set

*October 2014, Volume 61, Issue 6, Journal of Statistical Software*



Peter Filzmoser, Robert G. Garrett, Clemens Reimann

Multivariate outlier detection in exploration geochemistry

*Computers & Geosciences 31 (2005) 579-587.*



Greet Pison, Anja Struyf, Peter J. Rousseeuw (1999)

Displaying a Clustering with CLUSPLOT.

*Computational Statistics & Data Analysis 30 (1999) 381-392, Elsevier*

# References II



R Core Team and contributors worldwide (2015)

The R Language Manual System

*CRAN* e.g. via RStudio



Tom Short, (2004)

Short Reference Card

*CRAN* [cran.r-project.org/doc/contrib/Short-refcard.pdf](http://cran.r-project.org/doc/contrib/Short-refcard.pdf)



Paul Teetor

R Cookbook

*O'Reilly*



Paul Torfs, Caludia Brauer

A (very) Short Introduction to R.

*CRAN* [cran.r-project.org/doc/contrib/Torfs+Brauer-Short-R-Intro.pdf](http://cran.r-project.org/doc/contrib/Torfs+Brauer-Short-R-Intro.pdf)



Yanchang Zhao

R and Data Mining.

*Elsevier* 2013.

# References III



Yanchang Zhao

R Reference Card for Data Mining.

[www.rdatamining.com](http://www.rdatamining.com)

[www.rdatamining.com/docs/r-reference-card-for-data-mining.pdf](http://www.rdatamining.com/docs/r-reference-card-for-data-mining.pdf)