Series 2

1. Have a look at the data set `USairpollutionNA.csv`.

- **a)** How many missing values are in the data set? (Hint: Use `md.pattern` from package `mice`)? What percentage of all values in the data matrix is that?
- **b)** How many colums are without any missing value?
- **c)** How many rows are there without any missing value (=complete cases)?
- **d)** Do you think that complete-case analysis is suitable for further analysis?
- **e)** Use `missForest` to impute the dataset (use `set.seed(123)` to compare your result with the solution).
- **f)** How accurate do you think the `missForest` imputation was?
- **g)** Draw a stars plot of the imputed data set.
- **h)** Compare with a stars plot on the original data `USairpollution.csv`. Do you see striking differences indicating that the missing values were not imputed properly? Note that every city had exaclty one missing value. Can you find it by comparing the stars plots? Which cities were imputed well or badly? There was a substantial amount of missing data; discuss with a collegue whether you think that imputation was a useful way to recover the information in the available fields of the data matrix.

2. Investigate the behavior of case-wise deletion in a small simulation study. The setting is exactly as in Table 2 of the paper "*Missing Data: Our View of the State of the Art*" by J.L. Schafer *et al.* mentioned in the lecture and we try to reproduce some of the results shown there. We simulate (systolic) blood pressure of 50 people. Each person is measured twice (one month apart). Measurements are pos. correlated. We simulate three possible causes for missing values:

MCAR 73% of the second measurements are missing completely at random (e.g. the hard disk of the computer crashed and only part of the data could be reconstructed). Thus, the probability that a value is missing depends neither on the observed (first) measurement nor on the missing (second) measurement. This setting is called "missing completely at random (MCAR)".

MAR Only people with measurement of >140 in the first test are invited for the second test. Thus, the probability that a value is missing depends on the observed (first) measurement. This setting is called "missing at random (MAR)".

MNAR Everybody was tested twice, but the second result was only recorded if the value was >140. Thus, the probability that a value is missing depends on the missing (second) measurement. This setting is called "missing not at random (MAR)".

Since the data was simulated, we know the true values for the second measurement. The true expected value is 125, the true standard deviation is 25.

Use the same settings as in the paper to simulate 100×50 (`nrep`$\times$ `n`) observations:

```
> n <- 50
> nreps <- 100
> mu <- c(125, 125)
> sigma <- matrix( c(625, 375, 375, 625), 2,2)
```

R-hint: Use `mvrnorm` from the `MASS`-package to create multivariate normallly distributed data. If you want to reproduce the random numbers, use `set.seed()`
Overwrite some data by NAs according to the three methods. Then estimate the mean and the standard deviation for every method and compare with the real mean and real standard deviation.

3. We are investigating a data set comparing political world leaders during World War II. One person was asked to judge the similarity of 12 world leaders. Similarity was scaled from 1 to 9, where "1 = *very similar*" up to "9 = *very dissimilar*".
We will need the package `MVA`.

a) Load the data `WWIIleaders.rda` and have a look at it.
R-hint: Use the function `load` for this type of data (i.e. `.rda`).

b) Visualize the similarities according to this person using non-metrical MDS. What is the stress? Is it acceptable?
R-hint: Use the function `isoMDS`.

c) Can you find meaningful groups in the display?

d) Use a Shepard plot to see if the original dissimilarities have been distorted by the monotonic transformation.
R-hint: Use the function `Shepard`.

4. The data `ch-dist.dat` consists of distances between 12 Swiss cities. We would like to determine the location of these cities. (The distances are given as fastest routes, not shortest routes nor air-distances).

a) Determine the "coordinates" of these cities using the metric MDS `cmdscale()` and plot the resulting points. Are the locations of the cities comparably equal to their locations on the map of Switzerland? For comparison you may have to reflect or transform the coordinates.

- `cmdscale()`: The metric MDS is based on the idea of principal components - but we do not go into details here. Let d_{ij} be the Euclidean distance matrix and $d^*_{ij} = ||\underline{z}_h - \underline{z}_i||$ the Euclidean distance matrix in a k dimensional space, then the metric MDS minimizes the measure

$$\sum_{i<j} \left(d_{ij}^2 - d_{ij}^{*2}\right) .$$

For non-Euclidean distance matrices the metric MDS also minimizes a similar measure which - however - is much more complicated and therefore will not be stated here.

R-Hint: Use the following code:

```
t.url <- "http://stat.ethz.ch/Teaching/Datasets/WBL/ch-dist.dat"
chdist <- read.table(t.url, header=T)
library(cluster); library(MASS)
koord <- cmdscale(chdist)
plot(...)
text(...)
```

Note: `cmdscale()` gives you the coordinates of the point.

b) Plot the points using the distances of the MDS as x-axis and the dissimilarities (distances) as the y-axis. Which range of points of the dissimilarities are well approximated?
R-Hint: Since the distance function is a matrix, we need to convert it to a vector first:

```
f.dist1 <- c(as.matrix(...))
f.dist2 <- c(as.matrix(dist(...)))
p.p1 <- plot(t.dist2~t.dist1)
```

c) We consider a non-metric MDS method - you may find it in the package `MASS`. Determine the "coordinates" of the cities using the non-metric MDS and plot the points. Make the same analysis as in **b)**.
`isoMDS()`: This is the implementation of the Kruskal-Shepard-algorithm, which has been discussed the class. The corresponding stress-function is:

$$S(\hat{x}) = \min \frac{\sum_{i<j} (\hat{d_{ij}} - d_{ij})^2}{\sum_{i<j} d_{ij}^2}$$

R-Hint: The input of the distance-matrix for this function must be converted to matrix first, `as.matrix(chdist)`. The coordinates of the MDS are stored in `...$points`.

Preliminary discussion: 12.03.12.

Deadline: No hand-in.