Chapter 3

Resampling Methods

3.1 Bias and variance for point estimators

Let $X_1, \ldots, X_n$ be a sample from an unknown distribution function $F(x) = \mathbf{P}\{X \leq x\}$. We estimate a parameter $\theta = \theta(F)$, for that distribution, with an estimator $\hat{\theta} = \hat{\theta}(X_1, \ldots, X_n)$, that is a function of the available information $X_1, \ldots, X_n$.

The functional relation $\theta(F)$, describing how θ depends on F, can be known or unknown. In any case, the value of θ is unknown, since F is not known.

It is important to know if the estimator $\hat{\theta}$ is *unbiased*, i.e., if

$$\mathbf{E}\{\hat{\theta}\} = \mathbf{E}\{\hat{\theta}(X_1, \ldots, X_n)\} = \theta.$$

If that is not the case, it is of interest to know the magnitud of the *bias*

$$\mathbf{Bias}(\hat{\theta}) = \mathbf{E}\{\hat{\theta} - \theta\} = \mathbf{E}\{\hat{\theta}(X_1, \ldots, X_n)\} - \theta.$$

If the bias is small, compared to the variance of the estimator, then that variance

$$\mathbf{Var}\{\hat{\theta}(X_1, \ldots, X_n)\} = \mathbf{E}\left\{(\hat{\theta} - \mathbf{E}\{\hat{\theta}\})^2\right\} \approx \mathbf{E}\{(\hat{\theta} - \theta)^2\},$$

measures the mean-square deviation of the estimator $\hat{\theta}$ from the real value of θ.

Example 3.1. For estimation of the expected value $\mu = \mathbf{E}\{X\}$ of a random variable X, the sample mean of n independent observations $X_1, \ldots, X_n$ of X,

$$\hat{\mu} = \hat{\mu}(X_1, \ldots, X_n) = \bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i,$$

is unbiased. Further, the variance of the estimator is

$$\mathbf{Var}\{\hat{\mu}\} = \mathbf{Var}\{\hat{\mu}(X_1, \ldots, X_n)\} = \frac{\mathbf{Var}\{X\}}{n}.$$

That variance, in turn, can be estimated using the sample variance, as

$$\widehat{\mathbf{Var}\{\hat{\mu}\}} = \frac{\widehat{\mathbf{Var}\{X\}}}{n} = \frac{s_X^2}{n} = \frac{1}{n}\left(\frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2\right).$$

Based on the intuition from Example 3.1, basic courses in statistics tend to communicate the impression that bias and variance for estimators are quantities that can be studied analytically, and calculated or estimated numerically, in some generality.

However, the situation is in fact much the other way around: For most estimators, except the one in Example 3.1, bias and variance cannot be studied analytically, at least not for moderate sample sizes. (Often there is asymptotic theory that is useful for large sample sizes.)

For very large samples, the method described in the following example may be useful:

Example 3.2. The variance of $\hat{\theta}(X_1, \dots, X_n)$ can be estimated as the sample variance of $\{\hat{\theta}(X_1^{(i)}, \dots, X_n^{(i)})\}_{i=1}^N$, where $\{(X_1^{(i)}, \dots, X_n^{(i)})\}_{i=1}^N$ are N independent samples of X. Notice that this requires $n \times N$ observations of X, rather than n only.

The great number of observations required for the method in Example 3.2 often makes it impossible to use in practice.

With *resampling* methods one can, from one single sample $X_1, \dots, X_n$ of X, estimate bias and variance for an estimator $\hat{\theta}(X_1, \dots, X_n)$. The idea is to create new samples $\{(\tilde{X}_1^{(i)}, \dots, \tilde{X}_n^{(i)})\}_{i=1}^N$, using the original sample $X_1, \dots, X_n$. The following sections explain two resampling schemes - the *bootstrap* method and the *jack-knife* method.

3.2 The empirical distribution

3.2.1 Change of distribution

If $\varphi : \mathbb{R}^n \to \mathbb{R}$ is a function, and Y an $\mathbb{R}^n$-valued random variable with distribution function

$$F_Y(y) = F_Y(y_1, \dots, y_n) = \mathbf{P}\{Y_1 \leq y_1, \dots, Y_n \leq y_n\} \quad \text{for} \quad y = (y_1, \dots, y_n) \in \mathbb{R}^n,$$

then $\varphi(Y)$ has expected value

$$\mathbf{E}\{\varphi(Y)\} = \int_{y \in \mathbb{R}^n} \varphi(y)\, dF_Y(y) = \int_{y \in \mathbb{R}^n} \varphi(y) \partial_1 \dots \partial_n F_Y(y_1, \dots, y_n)\, dy_1 \dots dy_n = \mathbf{E}_{F_Y}\{\varphi\}.$$

The notation $\mathbf{E}_G\{\varphi\}$ makes sense for any distribution function G on $\mathbb{R}^n$, in the following way:

$$\mathbf{E}_G\{\varphi\} = \int_{\mathbb{R}^n} \varphi dG = \mathbf{E}\{\varphi(Z)\} \quad \text{for } Z \text{ a random variable with distribution function } G.$$

It is simple to generalize this. For example,

$$\mathbf{Var}_G\{\varphi\} = \mathbf{E}_G\{\varphi^2\} - (\mathbf{E}_G\{\varphi\})^2 = \int_{\mathbb{R}^n} \varphi^2 dG - \left(\int_{\mathbb{R}^n} \varphi dG \right)^2.$$

If $Y = (Y_1, \dots, Y_n)$ is a sample, that is, if $Y_1, \dots, Y_n$ are independent with common distribution function $F(y) = \mathbf{P}\{Y_i \leq y\}$, then

$$F_Y(y_1, \dots, y_n) = \mathbf{P}\{Y_1 \leq y_1, \dots, Y_n \leq y_n\} = F(y_1) \cdot \dots \cdot F(y_n) \equiv F^n(y).$$

Example 3.3. If $X_1, \ldots, X_n$ is a sample of an $\mathbb{R}$-valued random variable X with distribution F, then

$$\mathbf{E}_{F^n}\{\varphi\} = \mathbf{E}\{\varphi(X_1, \ldots, X_n)\} \equiv \mathbf{E}_F\{\varphi\}.$$

Notice that the n is usually left out of the notation, which thus simplifies to $\mathbf{E}_F\{\varphi\}$.

3.2.2 The empirical distribution

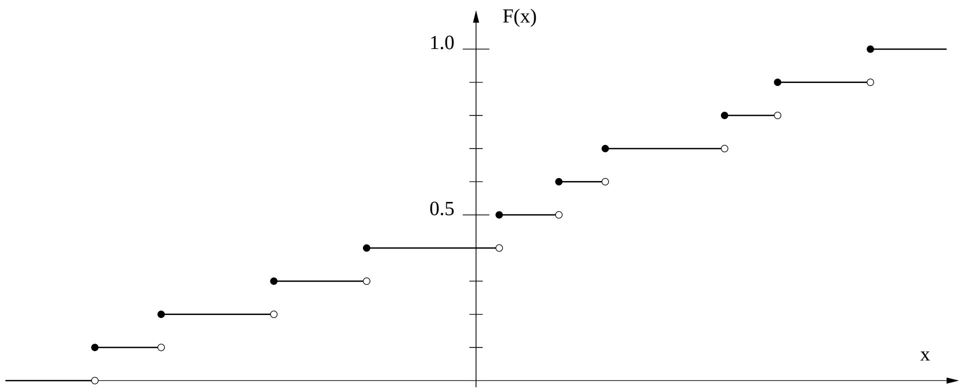
Let $X_1, \ldots, X_n$ be a sample of a random variable X with distribution function $F(x) = \mathbf{P}\{X \le x\}$, and let $X_{(1)} \le \ldots \le X_{(n)}$ be the corresponding ordered sample.

Definition 3.1. *With # denoting the number of members of a set, the* empirical distribution function $\tilde{F}$ *is given by*

$$\tilde{F}(x) = \frac{\#\{i : X_i \le x\}}{n} = \begin{cases} 0 & \text{if } \; x \in (-\infty, X_{(1)}), \\ i/n & \text{if } \; x \in [X_{(i)}, X_{(i+1)}) \quad \text{for} \quad i \in \{1, \ldots, n-1\}, \\ 1 & \text{if } \; x \in [X_{(n)}, \infty). \end{cases}$$

Figure 3.1 below displays what an empirical distribution function might look like.

Figure 3.1. *An empirical distribution function for $n = 10$ data.*



The following simple theorem indicates how empirical distributions can be simulated:

Theorem 3.1. *If η is a discrete random variable with possible values $\{1, \ldots, n\}$, with equal probabilities $1/n$, then the random variable X_η has the empirical distribution function $\tilde{F}$.*

Moreover, we have the following basic result, on approximating F with $\tilde{F}$:

Theorem 3.2. $\mathbf{P}\left\{\lim_{n\to\infty} |\tilde{F}(x) - F(x)| = 0\right\} = 1$ *for $x \in \mathbb{R}$.*

Proof. Let $Y_i = 1$ if $X_i \leq x$ and $Y_i = 0$ if $X_i > x$. Then $Y_1, Y_2, \ldots$ are independent identically distributed random varibles, with expected value $\mathbf{E}\{Y_i\} = \mathbf{P}\{X_i \leq x\} = F(x)$. Using that $\sum_{i=1}^n Y_i/n = \tilde{F}(x)$, the law of large numbers therefore shows that

$$\mathbf{P}\left\{\lim_{n\to\infty} |\tilde{F}(x) - F(x)| = 0\right\} = \mathbf{P}\left\{\lim_{n\to\infty}\left|\frac{1}{n}\sum_{i=1}^n Y_i - \mathbf{E}\{Y_1\}\right| = 0\right\} = 1. \quad \square$$

The following quite famous result gives a stronger version of Theorem 3.2:

Theorem 3.3 (Glivenko-Cantelli). $\mathbf{P}\left\{\lim_{n\to\infty} \max_{x\in\mathbb{R}} |\tilde{F}(x) - F(x)| = 0\right\} = 1.$

The following rather important formula is quite easy to prove, using that F is increasing and $\tilde{F}$ a pure-jump function:

Theorem 3.4. $\max_{x\in\mathbb{R}} |\tilde{F}(x) - F(x)| = \max_{1\leq i\leq n} \max\left\{\left|\frac{i-1}{n} - F(X_{(i)})\right|, \left|\frac{i}{n} - F(X_{(i)})\right|\right\}.$

3.3 The bootstrap method

3.3.1 Estimating variance and bias with the bootstrap method

The basic idea in the *bootstrap method*, is to estimate

$$\mathbf{Var}_F\{\hat{\theta}\} = \mathbf{Var}_{F^n}\{\hat{\theta}\} = \mathbf{Var}\{\hat{\theta}(X_1, \ldots, X_n)\}$$

with

$$\widehat{\mathbf{Var}_F}\{\hat{\theta}\} = \mathbf{Var}_{\tilde{F}}\{\hat{\theta}\} = \mathbf{Var}\{\hat{\theta}(\tilde{X}_1, \ldots, \tilde{X}_n)\} \equiv \mathbf{Var}_{\mathbf{BOOT}}\{\hat{\theta}\},$$

where $\tilde{X}_1, \ldots, \tilde{X}_n$ denotes a sample of the empirical distribution $\tilde{F}$ calculated from $X_1, \ldots, X_n$.

Since $\tilde{F}$ is known, one can in principle compute the variance

$$\mathbf{Var}_{\mathbf{BOOT}}\{\hat{\theta}\} = \int_{\mathbb{R}^n} \hat{\theta}^2 d(\tilde{F}^n) - \left(\int_{\mathbb{R}^n} \hat{\theta} d(\tilde{F}^n)\right)^2.$$

However, usually this variance is estimated by the *Monte-Carlo method:* If $\{(\tilde{X}_1^{(i)}, \ldots, \tilde{X}_n^{(i)})\}_{i=1}^N$ are N independent samples of the empirical distribution $\tilde{F}$, one estimates

$$\widehat{\mathbf{Var}_{\mathbf{BOOT}}}\{\hat{\theta}\} = \text{the sample varians of } \{\hat{\theta}(\tilde{X}_1^{(i)}, \ldots, \tilde{X}_n^{(i)})\}_{i=1}^N.$$

Recall that it is simple to simulate the sample $\{(\tilde{X}_1^{(i)}, \ldots, \tilde{X}_n^{(i)})\}_{i=1}^N$, using Theorem 3.1. Also, notice that, regardless if $\mathbf{Var}_{\mathbf{BOOT}}\{\hat{\theta}\}$ is calculated exactly or by Monte Carlo simulation, bootstrap is a question of assigning a uniform distribution on the sample $X_1, \ldots, X_n$, so that the observations are picked with equal probabilities when sampling the empirical distribution, and then making use of that distribution: This is *resampling*!

In the same manner, one can use bootstrap to estimate the bias

$$\mathbf{Bias}(\hat{\theta}) = \mathbf{E}\{\hat{\theta}(X_1,\ldots,X_n)\} - \theta(F) = \mathbf{E}_{F^n}\{\hat{\theta}\} - \theta(F) = \mathbf{E}_F\{\hat{\theta}\} - \theta(F)$$

with

$$\widehat{\mathbf{Bias}(\hat{\theta})} = \mathbf{E}_{\tilde{F}}\{\hat{\theta}\} - \theta(\tilde{F}) \equiv \mathbf{Bias}_{\mathbf{BOOT}}(\hat{\theta}).$$

Here $\mathbf{E}_{\tilde{F}}\{\hat{\theta}\}$ can in principle be calculated, as

$$\mathbf{E}_{\tilde{F}}\{\hat{\theta}\} = \int_{\mathbb{R}^n} \hat{\theta} d(\tilde{F}^n).$$

Again, it is usually instead estimated with the Monte-Carlo method, as

$$\widehat{\mathbf{E}_{\tilde{F}}\{\hat{\theta}\}} = \text{the sample mean of } \{\hat{\theta}(\tilde{X}_1^{(i)},\ldots,\tilde{X}_n^{(i)})\}_{i=1}^N,$$

where $\{(\tilde{X}_1^{(i)},\ldots,\tilde{X}_n^{(i)})\}_{i=1}^N$ are N independent samples of the empirical distribution $\tilde{F}$.

Notice that computation of $\theta(\tilde{F})$ requires knowledge of the functional relation $\theta = \theta(F)$. This makes the bootstrap method less generally useful to estimate bias than to estimate variance.

If $\mathbf{Bias}(\hat{\theta})$ has been estimated by $\widehat{\mathbf{Bias}(\hat{\theta})}$, that in turn was found to be significantly different from zero, compared with the standard deviation of the estimator, then the estimator $\hat{\theta}$ can be bias corrected, by modifying it to $\hat{\theta} - \widehat{\mathbf{Bias}(\hat{\theta})}$.

3.3.2 Confidence intervals with the bootstrap method

The standard method to make *confidence intervals*, is to use the following normal distribution approximation:

$$\mathbf{P}\left\{\hat{\theta} - \lambda_{p/2}\sqrt{\widehat{\mathbf{Var}\{\hat{\theta}\}}/n} \le \theta \le \hat{\theta} + \lambda_{p/2}\sqrt{\widehat{\mathbf{Var}\{\hat{\theta}\}}/n}\right\} \approx 1 - p. \tag{3.1}$$

Here $\widehat{\mathbf{Var}\{\hat{\theta}\}}$ is some estimate of $\mathbf{Var}\{\hat{\theta}\}$, for example by bootstrap, while $\lambda_{p/2}$ is a normal distribution *quantile*, given by

$$\mathbf{P}\left\{|\mathrm{N}(0,1)| > \lambda_{p/2}\right\} = 2\left(1 - \Phi(\lambda_{p/2})\right) = p.$$

There are several ways to justify the formula (3.1), of which the most common is asymptotic maximum likelihood theory: You have to consult your favorite course in statistics for more on this.

However, there are also many situations when (3.1) does not hold, or at least cannot be proven to hold.

Example 3.4. Let $X_1, \ldots, X_n$ be a sample of a random variable X with density function

$$f_X(x) = \frac{1}{2\,(1+(x-\mu)^2)^{3/2}} \quad \text{for} \quad x \in \mathbb{R}.$$

In other words, X has a Student $t(2)$ distribution, that has been relocated to be centered at $\mu \in \mathbb{R}$, with expected value $\mathbf{E}\{X\} = \mu$.

As usual, the sample mean $\hat{\mu} = \overline{X}$ is an unbiased estimator of μ. However, the variance of that estimator is infinite, because the variance of X is infinite. Thus the formula (3.1) is not applicable. Still, $\hat{\mu}$ is a sensible estimator of μ, since it is consistent, by the law of large numbers. Thus some other method than (3.1), to find confidence intervals, is required.

One attractive and simple, as well as generally applicable method, to make confidence intervals, is to use resampling in the following way: The unknown parameter $\theta(F)$ is approximated by the estimator $\hat{\theta}(X_1, \ldots, X_n)$, which in turn is approximated by bootstrap resampling, by $\hat{\theta}(\tilde{X}_1, \ldots, \tilde{X}_n)$. Therefore a confidence interval for $\hat{\theta}(\tilde{X}_1, \ldots, \tilde{X}_n)$, with confidence level $1-p$, say, should be a good approximate confidence interval for θ, on the level $1-p$.

In practice, making use of the Monte Carlo method as before, we get a bootstrap confidence interval $[a,b]$ for θ, by simulating observations $\{\hat{\theta}(\tilde{X}_1^{(i)}, \ldots, \tilde{X}_n^{(i)})\}_{i=1}^N$ of $\hat{\theta}(\tilde{X}_1, \ldots, \tilde{X}_n)$, and then chosing the interval limits a and b so that a fraction p of the observations $\{\hat{\theta}(\tilde{X}_1^{(i)}, \ldots, \tilde{X}_n^{(i)})\}_{i=1}^N$ fall outside $[a,b]$.

3.4 The jack-knife method

In the bootstrap mathod, F is approximated by the empirical distribution $\tilde{F}$. But there are other distributions than $\tilde{F}$, built on $X_1, \ldots, X_n$, which can be used to approximate F. In such a way, the *jack-knife method* is obtained. We will here only describe the method, but not offer any motivation for it.

Let $\hat{\theta}^{(i)}$ be the estimator of θ based on the sample $X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n$,

$$\hat{\theta}^{(i)}(X_1, \ldots, X_n) = \hat{\theta}(X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n)$$

for $i = 1, \ldots, n$, and $\hat{\theta}^{(\cdot)}$ the sample mean of $\hat{\theta}^{(1)}, \ldots, \hat{\theta}^{(n)}$,

$$\hat{\theta}^{(\cdot)} = \frac{1}{n}\sum_{i=1}^{n} \hat{\theta}^{(i)}.$$

In the *jack-knife* method, $\mathbf{Var}_F\{\hat{\theta}\}$ is estimated by

$$\mathbf{Var}_{\mathbf{JACK}}\{\hat{\theta}\} = \frac{n-1}{n}\sum_{i=1}^{n}(\hat{\theta}^{(i)} - \hat{\theta}^{(\cdot)})^2,$$

while $\mathbf{Bias}(\hat{\theta})$ is estimated by

$$\mathbf{Bias}_{\mathbf{JACK}}(\hat{\theta}) = (n-1)(\hat{\theta}^{(\cdot)} - \hat{\theta}).$$

Notice that jack-knife estimation of bias does not require knowledge of the functional relation $\theta = \theta(F)$.

3.5 S^+ and R

S and R are program languages for general statistic data treatment, including simulation. S^+ is a development of S with macro commands written in S. Usually, it is S^+ that is implemented on computer systems. S^+ is one of the most versatile statistical program packages available. It combines powerful traditional statistics, that challenge the "colosses" of the market, such as, for example, *SAS* and *SPSS*, with a general programing function, for example, for stochastic simulations, that is superior to those of SAS and SPSS.

S^+ has a strong position at universites, but a relatively small commercial usage. The main reason for this is that S^+ is a very slow program, in terms of usage of computer time. (However, it often saves a lot of time for the computer programmer, when compared with faster programs!)

R is a freeware software which is close to identical to S^+.

Whenever we encouter important differences between R and S^+, we will comment on them.

S^+ is started with

```
dali> Splus
s-PLUS : Copyright (c) 1988, 2001 Insightful Corp.
S : Copyright Lucent Technologies, Inc.
Version 6.0.4 Release 1 for Sun SPARC, SunOS 5.6 :  2001
Working data are in /users/mdstud/stada/lab_grupper/stada-??/MySwork
>
```

R is started with

```
dali> R
```

S^+ has two kinds of basic commands; expressions which are evaluated directly

```
> sqrt(2)
[1] 1.414214
```

and the operators `<-` and `->` to assign values

```
> x <- 3**(0.5); x
[1] 1.732051
```

Notice that several commands can be given on the same row, if separated by `;`

```
> x <- log(2); exp(x) -> y; x; y; as.integer(x); as.integer(y)
[1] 0.6931472
[1] 2
[1] 0
[1] 2
```

Incomplete commands result in the prompt + in stead of >

```
> sin(pi/4) *
+ 2**(-1/2)
[1] 0.5
```

A compiled assistance for a function is obtained via `args(function)`

```
> args(logb)
```

```
function(x, base)
NULL
> logb(3, 3)
[1] 1
```

More thorough assistance is obtained with `help(function)` or `? function`

```
> help(gamma)
Gamma Function (and its Natural Logarithm)
DESCRIPTION:
Returns the gamma function or the log of the gamma function.
...
```

To leave `help`, use `q` (for "quit").

S^+ is a vector language. Functions of S^+ operate on vectors, which, for example, can be created according to

```
> c(pi/2, pi/4) -> x; y <- c(1, 1/sqrt(2)); z <-1:15*2
> sin(x); y; sin(x)+y; cos(x)*y; z
[1] 1.0000000  0.7071068
[1] 1.0000000  0.7071068
[1] 2.0000000  0.7071068
[1] 6.123234e-17  5.000000e-01
[1] 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30
```

(Note the priorities between : and $*$!) More examples of vector operations are

```
> x <- c(rep(1, times=5), rep(2, times=5), rep(3, times=5); x
[1] 1 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3
> y <- x[1:5]; z <- c(x[1:4],x[10:11]); y; z
[1] 1 1 1 1 1
[1] 1 1 1 1 2 3
> x[-7]
[1] 1 1 1 1 1 2 2 2 2 3 3 3 3 3
```

A vector x can be reinterpreted as a matrix y, and S^+ functions also works on matrices, componentvise (!!), as the following example on multiplications of S^+ matrices illustrates:

```
> x <- c(1, 2, 3, 4, 5, 6); matrix(x, 2, 3) -> y; y**2 -> z; y; z
     [,1] [,2] [,3]
[1,]   1    3    5
[2,]   2    4    6
     [,1] [,2] [,3]
[1,]   1    9   25
[2,]   4   16   36
```

Notice that the order in which the elements from the vector x appear in the matrix y is not what one typically would expected: This is a common source of programming errors in S^+!

It is easy to refer to single elemenst in vectors and matrices:

```
> x[4]; y[2,2]; z[2,1]
[1] 4
```

```
[1] 4
[1] 4
```

An element in an *atomic vector* is a number, character string or a logical value. *Recursive vectors*, more oftenly called *lists*, have some element or elements which in turn are vectors or matrices themselves. Lists can be created with `list`, and one can test whether a vector is recursive or atomic:

```
> vekt1 <- c(10,TRUE); vekt2 <- c(5,FALSE)
> lista <- list(vekt1,vekt2)
> is.atomic(vekt1); is.atomic(lista); is.recursive(lista)
[1] T
[1] F
[1] T
```

The logical functions and and or are written in the following way:

```
> (1<2) & (2<1); (1<2) | (2<1)
[1] F
[1] T
```

There are also logical equivalent versions of these logical operator, as follows:

```
> (1<2) && (2<1); (1<2) || (2<1)
[1] F
[1] T
```

When `&` is used, the logical value of both of the logical expressions which `&` are applied to are computed. Then the logical value under `&` is determined. When `&&` is used, the logical value of the expression to the left of `&&` is computed. If that value is false, the value of `&&` is set to false directly, without checking the values of the expression to the right of `&&`, which thus need not even be well-defined. Only if the expression on the left of `&&` is true, the expression on the right is called for.

The difference between `|` and `||` is analogous to that between `&` and `&&`.

Usually, `&` and `|` are used for common logical computations, while `&&` and `||` are used, for example, in connection with conditions to stop while sloops (see below), where occasionally only the expressions on the left of `&&` or `||` is well-defined.

Elements of lists can be referred to with double bracket paranthesises:

```
> lista[[1]]; lista[[1]][1]
[1] 10 1
[1] 10
```

Notice that `TRUE` has been written as 1.

Elements of lists can be given names in the following manner:

```
> ny <- list(no1=vekt1,no2=vekt2); vekt <- ny$no2; vekt[1]; vekt[2]
[1] 5
[1] 0
```

Many S^+ functions, for example, linear regression `lsfit`, have a list as their output values. Using `$`, one may refer to different list elements, for example, `lsfit$coef` gives the regression coefficients.

Files can be read to S^+ with `scan`. For example, a file *D4-fil1.dat*, with contents

```
1 4
9 16 25
```

can be read in the following way:

```
> x <- scan("D4-fil1.dat"); sqrt(x)
[1] 1 2 3 4 5
```

Observe that " " is used around the file name!

Of course, for files which are not placed in the directory from which S^+ was started, a path to the correct directory has to be specified: For example, a file *D4-fil2.dat*, with contents

```
1 4
9 16
```

placed in the directory */users/mdstud/stada/lab_grupper/stada-??*, can be read with

```
> y <- scan("/users/mdstud/stada/lab_grupper/stada-??/D4-fil2.dat")
> sqrt(y)
[1] 1 2 3 4
```

The function `punif(x,min,max)` gives the distribution function for a uniform distribution over the interval `[min,max]`, and `qunif(x,min,max)` its invers. Further, `runif(n,min,max)` generates n independent observations of such a random variable.

```
> punif(0.7,0,1); qunif(0.7,0,2); runif(5,1,2)
[1] 0.7
[1] 1-4
[1] 1-76225 1.927495 1.659281 1.727141 1.077847
```

In the same way, `pnorm(x,mean,sd)`, `qnorm(x,mean,sd)` and `rnorm(n,mean,sd)` give a normal distribution function, its inverse, and a normal distributed random variable, respectively.

See the S+-manual on other examples of distributions that can be obtained with the commands `p.(.)`, `q.(.)` and `r.(.)`.

One can define ones own functions in S^+, in the following way:

```
my.func <- function(x,y,z) sin(x)*(y**2)*exp(-z); my.func(pi/2,1,0)
[1] 1
```

S^+ offers an array of common statistical routines (disregard eventual warnings they tend to give you!):

```
> N <- rnorm(25,5,2); N
[1] 5.878507 5.722384 4.906421 6.930634 5.791734 3.180767 2.026125
[8] 7.274999 4.111786 5.673488 5.396242 2.414764 1.793812 6.724480
[15] 4.036401 4.533922 0.690289 5.12308 9.059758 8.625384 6.650143
[22] 3.799066 6.201977 2.6770640 8.795774
> mean(N); var(N); median(N)
[1] 5.120513
[1] 4.948436
[1] 5.396242
> min(N); quantile(N,0.25); quantile(N,0.5); quantile(N,0.75); max(N)
[1] 0.690289
```

```
25% 3.799066
50% 5.396242
75% 6.650144
[1] 9.059758
```

It is easy to write your own programs in S^+:

```
> x<-NULL; for (i in 1:10) {n<-1; while (n<i) {
+    n<-n+1; fac<-fac*n}; x<-c(x,fac)}; x
[1] 1 2 6 24 120 720 5040 40320 362880 3628800
```

Two of the most common programing errors in S^+ are due to the fact that : does not function as typically is expected:

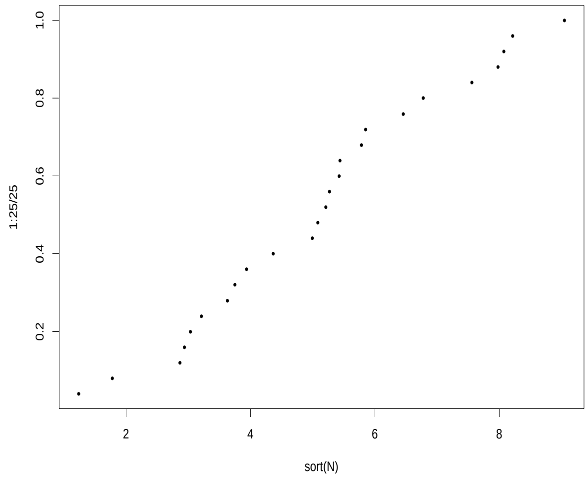
```
> 1:2*5; 1:(-1)
[1] 5 10
[1] 1 0 -1
```

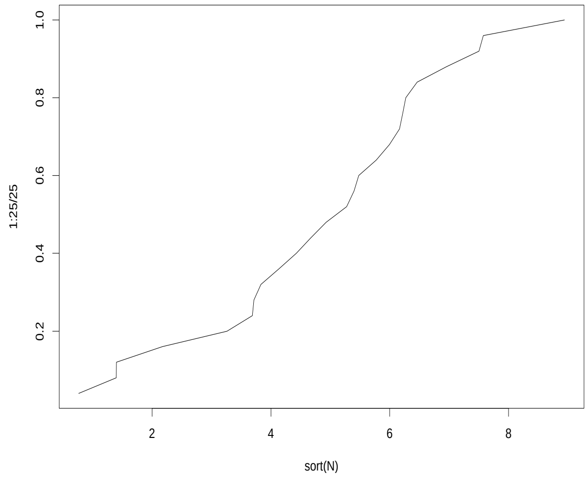
Notice that the operation : has the highest priority! And that the list `m:n` is not empty when $m > n$.

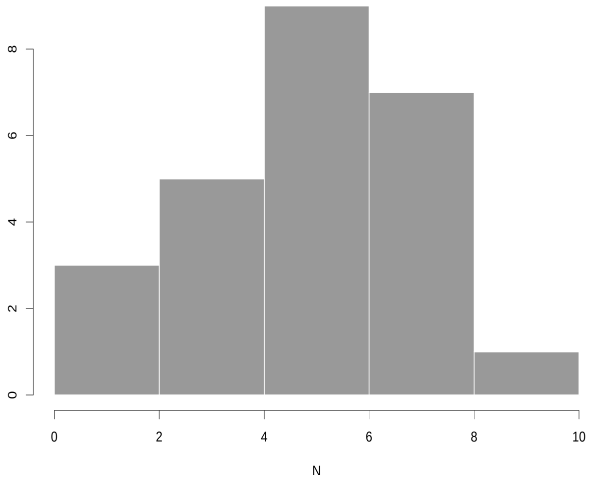
S^+ graphics are displayed in a special graphic window. For example, two vectors x and y, of the same length, are ploted against each other with the command `plot(x,y)`.

The empirical distribution function and histogram for N can be plotted with

```
plot(sort(N),1:25/25); plot(sort(N),1:25/25,type=1''); hist(N)
```







Graphics are guided to the deafult printer with the command `postscript()`. This gives the result that the plots are saved in the file *ps.out.0001.ps*, and then gradually printed out. Graphics are guided back to the screen with `dev.off()`. First then, or when the S^+ session is ended, the content of *ps.out.0001.ps* is printed to the printer.

When `postscript()` is called a second time, during a S^+ session, graphics are saved to *ps.out.0002.ps*, etc.

In R one can use the command `postscript(''filename.eps'')` to save to a file *filename.eps*. The syntax is as follows:

```
> postscript("filename.eps") ; hist(N); dev.off()
```

One can make comments in an S^+ program, in the following way:

```
> command 1; command 2; ...; # comment
```

Everything on a command line that follows after # is neglected when running S^+.

An S^+ session is ended with the command `q()`.

The primitive, so called *interface*, we have seen to run S^+ so far, is unsatisfactory for more extensive use of the program. There are a few alternative, more sophisticated ways to run S^+:

1. Start S^+ with the command `Splus -e`, instead of with `Splus`. Then one gets access to a simple command editor, so that, for example, old commands can be retrieved and edited. Unfortunately, the commands of this editor are rather unnatural, so that the editor is less pleasant to use than it could have been.

   R does not have the mentioned command editor.

2. Write the S^+-program code in a file *clever*, say, with some editor, for example, emacs. The run the program, in an S^+ session, in the following way:

   ```
   > source("/users/math/mdstud/stada/lab_grupper/stada-??/clever")
   ```

   Shorter programs can be copied and pasted from the editor to the S^+ window.

   To copy and paste is of course useful in many other ways, for example, to reuse commands that were fed earlier, completely or in part, or to save some output results for reportwriting.

3. Start S^+ in the following way: First start the editor emacs with the UNIX command `emacs`. In the emacs window so obtained, execute `Meta-x`, i.e., press the `Meta` tangent (the tangent marked ♦ down to the left), and then (with `Meta` tangent pressed), press also the `x` tangent. Then write `load-library`, at the bottom command row of the emacs window, followed by `[Carreage Return]`, followed by `S-mode [Carriage Return]`. Finally, execute `Meta-x` again, and write `S [Carreage Return]`, at the command row mentioned.

   A more modern way to run S^+ now becomes available, where the functioning of emacs and S^+ are integrated. This is possible because both programs are written in the program language *LISP*. Notice that one uses emacs command `Ctrl-x` followed by the `o` (for other), to move between the two windows that the emacs window is splitted to. (Or simply use the mouse!)

   The described way to run S^+, from emacs, does nor work for R, as of current on the Chalmers computer system.

Example 3.4. (Continued) The central limit theorem does not apply to calculate confidence intervals, when estimating the location parameter of a relocated Student $t(2)$ distribution with the sample mean, because the variance of the sampled distribution is infinite. However, if using the usual formula, based on the central limit theorem, anyway, we can check the correctnes of the interval obtained, by means of comparing it with a bootstrap confidence interval.

Enclosed below is an S^+ program, that simulates a sample of size $n = 1000$ from a Student $t(2)$ distribution, and then calculates confidence intervals on the 95% level, for the location parameter $\mu = 0$, based on estimating μ with the sample mean, first using the usual central limit theorem formula, giving the interval $(-0.131, 0.066)$, which lacks theoretical support, and then using the bootstrap method, giving the close to indentical interval $(-0.135, 0.070)$, that is supported by bootstrap theory.

```
> x = rt(1000,3); c(mean(x)+qnorm(0.025)*sqrt(var(x)/1000),
+ mean(x)+qnorm(0.975)*sqrt(var(x)/1000))
[1] -0.13056517  0.06652838
> y = NULL; for (i in 1:1000) {
+ z = x[as.integer(runif(1000,1,1001))];
+ y = c(y,mean(z))}
> c(quantile(y,0.025),quantile(y,0.975))
        2.5%      97.5%
  -0.1345719 0.06958057
```

3.6 Laboration

3.6.1 Bootstrap and jack-knife

Throughout the laboration, $X_1, \ldots, X_n$ are observations of a random variable X with unknown distribution function F.

1. The file `boot1.dat` contains the data $F(X_1), \ldots, F(X_{1600})$. *Illustrate the convergence in Theorem 3.3 by means of plotting*

$$\sup_{-\infty<x<\infty} |\tilde{F}^{(n)}(x) - F(x)| \quad \textit{for} \quad n = 1, 4, 9, 16, 25, \ldots, 1600,$$

where $\tilde{F}^{(n)}$ is the empirical distribution based on the observations $X_1, \ldots, X_n$.

For this problem, it is suitable to make use of Theorem 3.4. Also notice that smaller data materials should be picked from $F(X_1), \ldots, F(X_{1600})$ before sorting, rather than afterwards.

2. The theoretical standard deviation

$$\sigma = \sigma(F) = \sqrt{\mathbf{Var}\{X\}}$$

is estimated with the sample standard deviation

$$\hat{\sigma} = \hat{\sigma}(X_1, \ldots, X_n) = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (X_i - \overline{X})^2}.$$

The task is to find the variance for this estimator, i.e.,

$$\mathbf{Var}\{\hat{\sigma}(X_1, \ldots, X_n)\} = \mathbf{Var}_F\{\hat{\sigma}\} = \mathbf{Var}\{\hat{\sigma}(X_1, \ldots, X_n)\}.$$

This variance cannot be calculated analytically, because we do not know the distribution function F. For that reason, F is replaced with the empirical distribution $\tilde{F}$. Then the variance is approximated by

$$\mathbf{Var_{BOOT}}\{\hat{\sigma}\} = \mathbf{Var}_{\tilde{F}}\{\hat{\sigma}\} = \mathbf{Var}\{\hat{\sigma}(\tilde{X}_1, \ldots, \tilde{X}_n)\},$$

where $\tilde{X}_1, \ldots, \tilde{X}_n$ is a sample of the empirical distribution function $\tilde{F}$. Here $\mathbf{Var}\{\hat{\sigma}(\tilde{X}_1, \ldots, \tilde{X}_n)\}$ is not calculated as it is either, but is estimated by the sample variance for N observations $\{\hat{\sigma}(\tilde{X}_1^{(i)}, \ldots, \tilde{X}_n^{(i)})\}_{i=1}^N$ of $\hat{\sigma}(\tilde{X}_1, \ldots, \tilde{X}_n)$.

In other words, N independent samples $\{(\tilde{X}_1^{(i)}, \ldots, \tilde{X}_n^{(i)})\}_{i=1}^N$ should be generated in the computer, to yield N independent observations $\{\hat{\sigma}(\tilde{X}_1^{(i)}, \ldots, \tilde{X}_n^{(i)})\}_{i=1}^N$ of $\hat{\sigma}(\tilde{X}_1, \ldots, \tilde{X}_n)$, in order to estimate

$$\widehat{\mathbf{Var_{BOOT}}}\{\hat{\sigma}\} = \frac{1}{N-1} \sum_{i=1}^{N} \left(\hat{\sigma}(\tilde{X}_1^{(i)}, \ldots, \tilde{X}_n^{(i)}) - \frac{1}{N} \sum_{j=1}^{N} \hat{\sigma}(\tilde{X}_1^{(j)}, \ldots, \tilde{X}_n^{(j)}) \right)^2.$$

(The quantities that feature here can be calculated using the variance command in S^+.)

The file `boot2.dat` contains the data $X_1, \ldots, X_{100}$. *Estimate*

$$\mathbf{Var_{BOOT}}\{\hat{\sigma}\} = \mathbf{Var}\{\hat{\sigma}(\tilde{X}_1, \ldots, \tilde{X}_n)\} \quad \textit{for} \quad n = 20, 100,$$

by means of $N = 200$ simulated observations $\{\hat{\sigma}(\tilde{X}_1^{(i)}, \ldots, \tilde{X}_n^{(i)})\}_{i=1}^{200}$ of $\hat{\sigma}(\tilde{X}_1, \ldots, \tilde{X}_n)$. Further, calculate $\mathbf{Var_{JACK}}\{\hat{\sigma}\}$ for $n = 20, 100$.

The S^+-command `eta <- as.integer(runif(1,1,n+1))` gives an observation `eta` of a discrete random variable with uniform distribution over $\{1, \ldots, n\}$. Notice that the variances for $n = 20$ and for $n = 100$ should differ roughly by a factor 5. Further, the bootstrap and jack-knife methods should give roughly the same results.

3. The task is to estimate the bias $\mathbf{Bias}(\hat{\sigma}) = \mathbf{E}\{\hat{\sigma}(X_1, ..., X_n)\} - \sigma(X)$, which cannot be calculated analytically, because we do not know F. Therefore we use the approximation

$$\begin{aligned}\mathbf{Bias}(\hat{\sigma}) &= \mathbf{E}_F\{\hat{\sigma}\} - \sigma(F)\\ &\approx \mathbf{Bias_{BOOT}}(\hat{\sigma})\\ &= \mathbf{E}_{\tilde{F}}\{\hat{\sigma}\} - \sigma(\tilde{F})\\ &= \mathbf{E}\{\hat{\sigma}(\tilde{X}_1, \ldots, \tilde{X}_n)\} - \sigma(\tilde{F}),\end{aligned}$$

where $\tilde{X}_1, \ldots, \tilde{X}_n$ is a sample of the empirical distribution $\tilde{F}$. Now, $\mathbf{E}\{\hat{\sigma}(\tilde{X}_1, \ldots, \tilde{X}_n)\}$ is not calculated analytically either, but instead estimated by the sample mean of N observations $\{\hat{\sigma}(\tilde{X}_1^{(i)}, \ldots, \tilde{X}_n^{(i)})\}_{i=1}^N$ of $\hat{\sigma}(\tilde{X}_1, \ldots, \tilde{X}_n)$.

In other words, N independent observations $\{\hat{\sigma}(\tilde{X}_1^{(i)}, \ldots, \tilde{X}_n^{(i)})\}_{i=1}^N$ of $\hat{\sigma}(\tilde{X}_1, \ldots, \tilde{X}_n)$ should be created, in order to estimate

$$\mathbf{E}\{\hat{\sigma}(\widehat{\tilde{X}_1, \ldots}, \tilde{X}_n)\} = \frac{1}{N}\sum_{i=1}^{N} \hat{\sigma}(\tilde{X}_1^{(i)}, \ldots, \tilde{X}_n^{(i)}).$$

The term $\sigma(\tilde{F})$ in the expression for $\mathbf{Bias_{BOOT}}(\hat{\sigma})$ is the (theoretical) standard deviation of $\tilde{F}$. Since this distribution is uniformly distributed over the observed values $X_1, \ldots, X_n$, it follows that

$$\sigma(\tilde{F}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(X_i - \mathbf{E}\{\tilde{X}\})^2} \qquad \text{where} \qquad \mathbf{E}\{\tilde{X}\} = \frac{1}{n}\sum_{i=1}^{n} X_i = \overline{X}.$$

This means that $\sigma(\tilde{F})$ is the sample standard deviation of the sample $X_1, \ldots, X_n$, multiplied by the factor $\sqrt{(n-1)/n}$.

When estimating a very small or zero bias, one method can indicate a small positive bias, and another method a small negative bias. Consequently, one does bias corrections only when the estimated bias is reasonably big, in comparison with the standard deviation of the estimator. Otherwise, it is satisfactory to say that the bias is small, compared with other errors, so that a bias correction is not motivated.

A small bias is what we expect from a good estimator, as the sample standard deviation certainly should be of the standard deviation. However, typically it is hard to prove this analytically. This makes the approach that we have used an important tool for bias estimation. Also, even if analytical methods do apply, why should such theoretical efforts be made when a quite simple computer program does the job? Notice that even if the sample variance is an unbiased estimator of the variance, the sample standard deviation is typically not an unbiased estimate of the standard deviation!

Estimate $\mathbf{Bias_{BOOT}}\{\hat{\sigma}\}$ *and calculate* $\mathbf{Bias_{JACK}}\{\hat{\sigma}\}$ *for* $n = 20, 100$.

4. *Make a bootstrap confidence interval for* σ*, based on* $N = 200$ *observations* $\{\hat{\sigma}(\tilde{X}_1^{(i)}, \ldots, \tilde{X}_n^{(i)})\}_{i=1}^N$ *of* $\tilde{\sigma} = \hat{\sigma}(\tilde{X}_1, \ldots, \tilde{X}_n)$ (see Section 3.3.2).