Solution to Series 1

1.

Function	Transformation	Linear form
$y = \alpha x^\beta$	$y' = \log(y),\ x' = \log(x)$	$y' = \log(\alpha) + \beta \cdot x'$
$y = \alpha e^{\beta \cdot x}$	$y' = \log(y)$	$y' = \log(\alpha) + \beta \cdot x$
$y = \alpha + \beta \cdot \log(x)$	$x' = \log(x)$	$y = \alpha + \beta \cdot x'$
$y = x/(\alpha \cdot x - \beta)$	$y' = \frac{1}{y},\ x' = -\frac{1}{x}$	$y' = \alpha + \beta \cdot x'$
$y = \frac{e^{\alpha+\beta x}}{1+e^{\alpha+\beta \cdot x}}$	$y' = \frac{1}{y},\ y'' = y' - 1,\ y''' = -\log(y'')$	$y''' = \alpha + \beta \cdot x$
$y = \alpha e^{\beta/x}$	$y' = \log(y),\ x' = \frac{1}{x}$	$y' = \log(\alpha) + \beta \cdot x'$
$y = 1/(\alpha + \beta e^{-x})$	$y' = \frac{1}{y},\ x' = e^{-x}$	$y' = \alpha + \beta \cdot x'$

2. The command `set.seed` initializes the random generator, so that the results are repeatable.

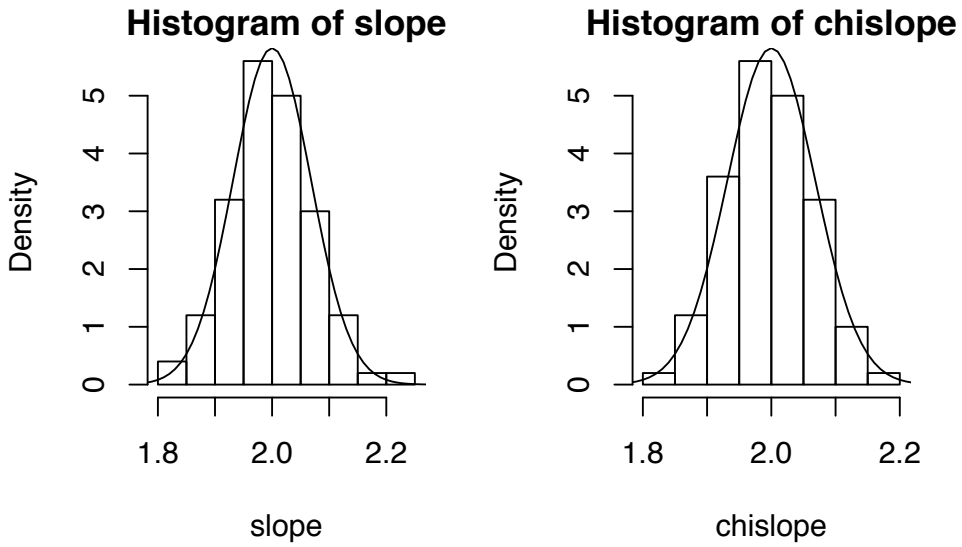
a)
```
> set.seed(21)                                  ## initializes the rng
> nrep <- 100                                   ## number of repetitions
> slope <- numeric(nrep)                        ## initialization of vector
> x <- seq(1,40,1)                              ## equidistant x-values
> for (i in 1:nrep){
     y <- 2*x+1+5*rnorm(length(x))              ## simulation of y-values
     reg <- lm(y~x)                             ## least squares regression
     slope[i] <- coefficients(reg)[2]           ## saves the slope
  }
```

b)
```
> par(mfrow=c(1,2))
> hist(slope, freq = FALSE)                     ## histogram
> X <- cbind(rep(1,40),x)                       ## design matrix
> XtXinv <- solve(crossprod(X))                 ## theoretical s.d. of the slope,
> tsd <- sqrt(5^2*XtXinv[2,2])                  ## according to Sec. 1.4.2(script)
>                                               ## crossprod(X)<=>t(X)%*%X
> lines(seq(1.7,2.3,by=0.01),dnorm(seq(1.7,2.3,by=0.01),mean=2,sd=tsd))
```

c)
```
> summary(slope)                                ## mean and other information
> sd(slope)                                     ## empirical standard deviation
```

d)
```
> chislope <- numeric(nrep)                     ## slopes vector
> for (i in 1:nrep){
     y <- 2*x+1+5*(1-rchisq(length(x), df=1))/sqrt(2)
     reg <- lm(y~x)
     chislope[i] <- coefficients(reg)[2]
  }
> hist(chislope, freq=FALSE)
> summary(chislope)
> sd(chislope)
> lines(seq(1.7,2.3,by=0.01),dnorm(seq(1.7,2.3,by=0.01),mean=2,sd=tsd))
```

The mean of the slopes for the normal distributed errors (part c) is 2.001, and 1.999 for the skewly distributed errors (part d). Standard deviations are 0.0701 (part c) and 0.0662 (part d). The theoretical mean is 2, the standard deviation `tsd` is 0.0684. The normal density curve fits both histograms almost perfectly. Though all these results depend on the value given to `set.seed`, the fit is usually very good: The estimated slope is a linear combination of the response variates, and hence, thanks to the central limit theorem, approximately follows a normal distribution. The variance of this normal distribution is $\left[\sigma^2 (X^T X)^{-1}\right]_{22}$. Since in both cases the predictors and the variances of the errors ($\sigma^2 = 5^2$) are equal, the standard deviations of the estimated slopes in part c and part d are equal.



3. **a)** The original data show an increasing trend, which might be linear. Moreover, there are monthly fluctuations, which get stronger with time. If a linear model would be fitted to these data, we could observe the residual variance increasing with time.

b) With logarithmized data, the global trend remains more or less linear, while the monthly fluctuations get stable. The fit of a linear model is much more reasonable here. Taking logarithms or other transformations of the target variable is often a good method to remove monotone trends in the variation. In terms of the original variables, this means that a multiplicative model is fitted instead of an additive one (compare the second line of the solution for Exercise 1).

c) Preparation for **d)**.
About the remark: An extra intercept would lead to a singular matrix X and thus the solution to the linear regression problem would not be unique (see section **1.3.1** in the lecture script). More exactly, the added column of 1's to X can be written as a linear combination of the other columns and so X is a singular matrix.

d) Apart from defining all month effects as 12 different vectors, there is a more elegant way: there is a predefined vector `month.name` in R. The linear model can be fitted with so-called "factors", which assign every observation to one of the factor levels (effect sources), so that all twelve month effects can be included in a single vector.

```
> airline <- scan("http://stat.ethz.ch/Teaching/Datasets/airline.dat")
> trend <- 1:144   # time
jan <- rep(c(1,rep(0,11)),12)                # effect January
feb  <- rep(c(0,1,rep(0,10)),12)             # effect February
#etc.
reg <- lm(log(airline)~trend+jan+feb+...-1)
```

Hint: The following R-code gives plots with "correctly" labeled x-axis.

```
>      t<-ts(airline,start=c(1949,1),frequency=12)
>      plot(c(time(t)), airline, xlab="Time",type="l")
```

Departures from normality, homogeneity of variances or linearity are not clearly visible, but the residuals seem to be correlated. Some experience is needed to assess such plots. One possibility to acquire such experience is to take a look at artificial data generated according to the model which we want to check (i.i.d. normally distributed residuals). You may compare the actual residuals with the two plots from artificial data.

```
> ## more elegantly
> trend <- 1:144
> month <- as.factor(rep(month.name,12))       # month effects as single factor
> reg <- lm(log(airline)~trend+month-1)
> par(mfrow=c(3,2))
> plot(trend, airline, type="l", main="Original data")
> plot(trend, log(airline), type="l", main="Logarithmized data")
> ## plots of fitted values and residuals
> plot(trend, fitted(reg), type="l", main="Fitted data")
> plot(trend, residuals(reg), type="l", main="Residuals")
> ## two artificial normal datasets to compare
> plot(trend, rnorm(144), type="l", main="Simulated normal data")
> plot(trend, rnorm(144), type="l", main="More simulated normal data")
```