A **time series** is a series of observations on a variable at successive times.

Exercise: Download the annual US GDP.[1]

- Create a working directory, say **C:\Projects\GDPa**, for the analysis of the annual US GDP.

- Visit the website of the Federal Reserve Bank of St. Louis (**www.stlouisfed.org**) and look for the **FRED® Economic Data**.

- Search for **GDP** (gross domestic product) and select the annual time series **Real Gross Domestic Product**.

- **Download** the time series into your working directory using **CSV (data)** as file format.

The downloaded text file **GDPCA.csv** looks as follows:

```
DATE,GDPCA
1929-01-01,1056.6
1930-01-01,966.7
1931-01-01,904.8
1932-01-01,788.2
1933-01-01,778.3
1934-01-01,862.2
      ⋮
```

[1] This time series will be used also in other chapters. However, different versions will be obtained if the time series is downloaded at different times.

For the analysis of time series we need suitable software. We will use R, which is free software for statistical computing and graphics. Information for downloading, installing, and using R can be found on the R website **www.r-project.org** (see also Appendix A).

Exercise: Import the annual US GDP from the csv file **GDPCA.csv** into R.

- Launch R by double-clicking on the R icon.

- To select the working directory **C:\Projects\GDPa** type **setwd("C:/Projects/GDPa")** in the Commands window at the R command prompt > and press **Enter**. Note that R uses / to separate directories (like Unix) and not \ (like Windows).

- To read the time series into R enter the command **H <- read.csv("GDPCA.csv")**.

The assignment operator **<-** is used to save the data read by the function **read.csv**. The new object **H** contains the dates in its first column and the GDP values in its second column.

- To see the data stored in the object **H** all you have to do is to enter its name.

```
> H
        DATE    GDPCA
1  1929-01-01  1056.6
2  1930-01-01   966.7
3  1931-01-01   904.8
4  1932-01-01   788.2
5  1933-01-01   778.3
6  1934-01-01   862.2
                 ⋮
```

The first step in analyzing a time series is to plot it. A plot of the observations versus time is called a **time series plot**.

Exercise: Plot the annual postwar US GDP.

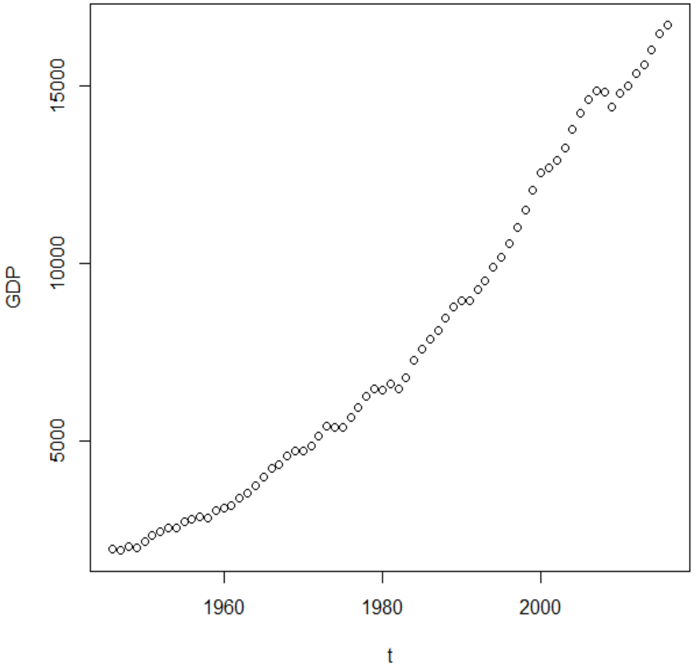
- Enter **H <- H[-(1:17),] to r**emove the first 17 years (1929,1930,…,1945) from the time series.

- Enter **Y <- H[,2]** to store the postwar US GDP values into **GDP**.

- Enter **N <- length(Y)** to store the length of the postwar time series into **N**.

- Enter **D <- 1946:(1946+N-1)** to create a vector of length **N** containing the calendar years 1946,1947,…,1946+N-1.

- Enter **plot(D,Y)** to plot the US GDP values against the dates.



Exercise: Download the monthly prices of the Nasdaq-listed stock Amazon.com, Inc. from Yahoo! Finance.

- Create a working directory, say **C:\Projects\AMZN**.

- Visit **www.yahoo.com** and click on the **Finance**.

- Enter the name **Amazon** or the symbol **AMZN** into the empty field and click on **Search**.

- Select **Historical Data**, **Time Period: Max**, and **Frequency: Monthly** and click on **Apply**.

- Right-click on **Download Data** and save the data in **C:\Projects\Amazon** as **AMZN.csv**.

Contents of **C:\Projects\AMZN\AMZN.csv**:

```
Date,Open,High,Low,Close,Adj Close,Volume
1997-05-01,1.968750,1.979170,1.312500,1.500000,1.500000,92180400
1997-06-01,1.510420,1.708330,1.375000,1.541670,1.541670,53163600
1997-07-01,1.541670,2.572920,1.510420,2.395830,2.395830,108614400
                                ⋮
```

The first line is the header line, which contains the column names. The first column contains the dates and the sixth column contains the closing prices adjusted for dividends and splits. The data are separated by commas.

Exercise: Import the monthly prices of Amazon from the file **C:\Projects\AMZN\AMZN.csv** into R.

- Launch R by double-clicking on the R icon.
- To select the working directory **C:\Projects\AMZN** enter **setwd("C:/Projects/AMZN")**.
- To import the data from the file **Amazon.csv** into R enter **H <- read.csv("AMZN.csv",header=TRUE, na.strings="null")**

na.strings="null" indicates that missing values in the file **AMZN.csv** are represented by the string "null" rather than by the symbol **NA**.

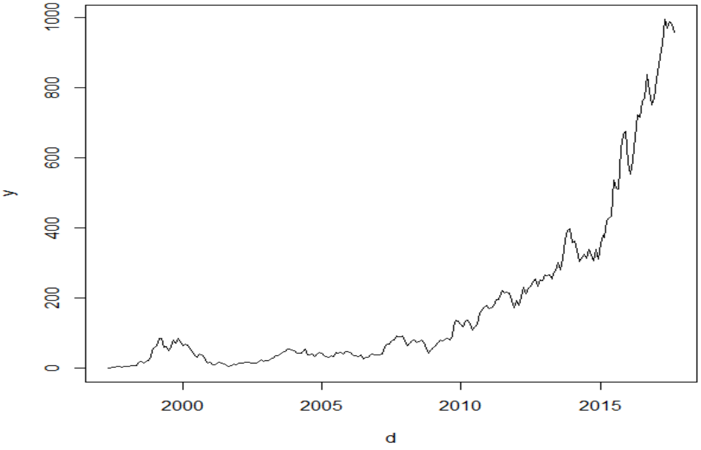
header=TRUE indicates that the file **AMZN.csv** contains the names of the variables as its first line.

Since the columns of the object **H** are named, we can access individual columns of **H** either by specifying their column numbers or by specifying their names.
For example, we may write **H$Date** and **H$Open** in place of **H[,1]** and **H[,2]**. Similarly, the first opening price can be accessed by **H[1,2]** or by **H$Open[1]**.

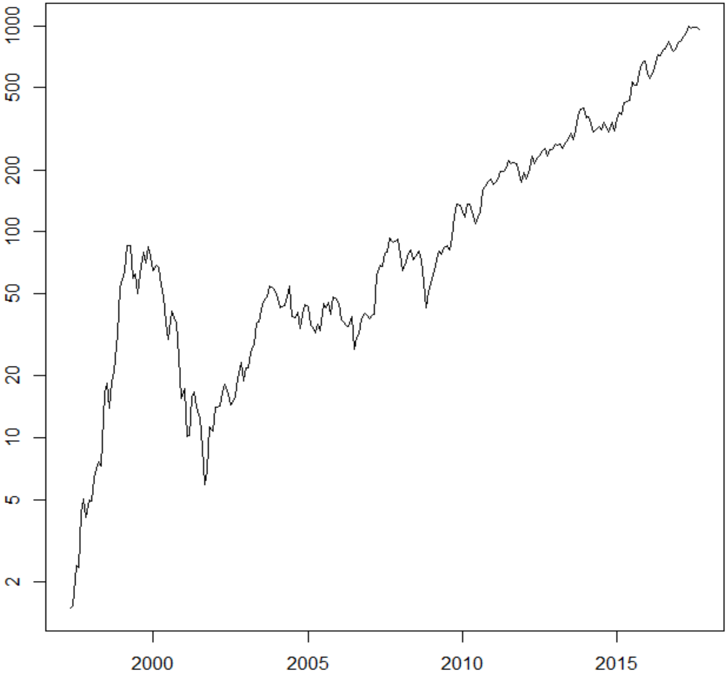
```
H <- na.omit(H) # rows with missing values are omitted

head(H)   # returns the first part of H

        Date    Open    High     Low   Close Adj.Close    Volume
1 1997-05-01 1.96875 1.97917 1.31250 1.50000   1.50000  92180400
2 1997-06-01 1.51042 1.70833 1.37500 1.54167   1.54167  53163600
3 1997-07-01 1.54167 2.57292 1.51042 2.39583   2.39583 108614400
4 1997-08-01 2.34375 2.41667 1.93750 2.33854   2.33854  31906800
5 1997-09-01 2.34375 4.81250 2.31250 4.33854   4.33854 107862000
6 1997-10-01 4.43750 5.50000 3.52083 5.08333   5.08333 105187200

tail(H)   # returns the last part of H

          Date   Open    High    Low  Close Adj.Close   Volume
240 2017-04-01 888.00  949.59 884.49 924.99    924.99 73539700
241 2017-05-01 927.80 1001.20 927.80 994.62    994.62 76148000
242 2017-06-01 998.59 1017.00 927.00 968.00    968.00 96102300
243 2017-07-01 972.79 1083.31 951.00 987.78    987.78 78796300
244 2017-08-01 996.11 1006.40 936.33 980.60    980.60 77261700
245 2017-09-01 984.20 1000.00 931.75 956.40    956.40 56741300
```

Exercise: Plot the monthly adjusted closing prices of Amazon.

```
N <- nrow(H)   # number of rows of object Y
D <- as.Date(H[,1])   # convert to dates
Y <- H[,6]   # adjusted closing prices
plot(D,Y,type="l")   # plot line instead of points
```



```
par(mar=c(2,2,1,1)) # lines of margin on 4 sides of plot
plot(D,Y,type="l",log="y")   # log scale on y-axis
```



For “professional” workers:

If you want to work always with current data, you need daily updates, which can easily be obtained with the help of the package tseries.

```
install.packages("tseries"); library(tseries)
```

```
A <- "AMZN"; T0 <- 1997-05-15; t0 <- "1970-01-01"
H <-  get.hist.quote(instrument=A,start=T0,
           end=Sys.Date(),
           quote=c("Open","High","Low","Close",
                        "Adjusted","Volume"),
           provider="yahoo",origin=t0,
           compression="d",retclass="zoo")
H <- na.omit(H); D <- time(H); N <- length(D)
Y <- as.numeric(as.vector(H$Adjusted))
```

Calculate Log-Returns:

```
Y <- log(Y); n <- N-1
y <- Y[2:N]-Y[1:n]); d <- D[2:N]
```

Plot log prices and log returns:

```
par(mfrow=c(2,1),mar=c(2,2,1,2))
plot(D,Y,type="l"); mtext(" (a)",side=4,cex=0.8,las=1)
plot(d,y,type="l"); mtext(" (b)",side=4,cex=0.8,las=1)
```

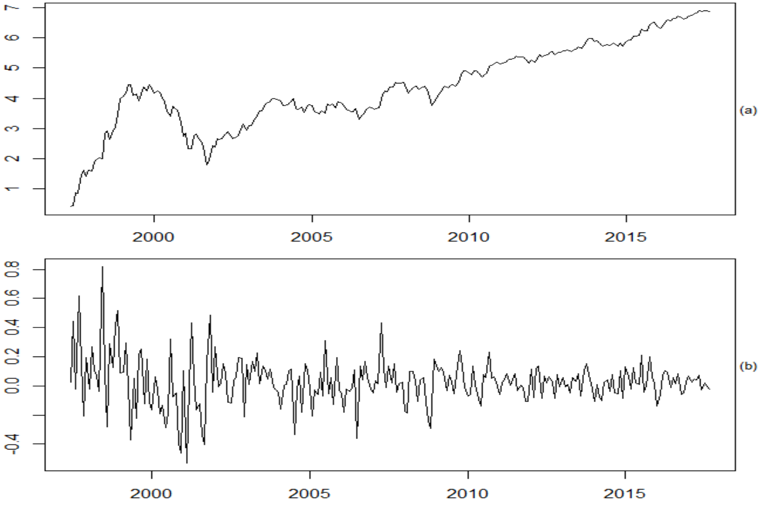


Figure 1: Log prices (a) and log returns (b) of Amazon