ASTA

The ASTA team

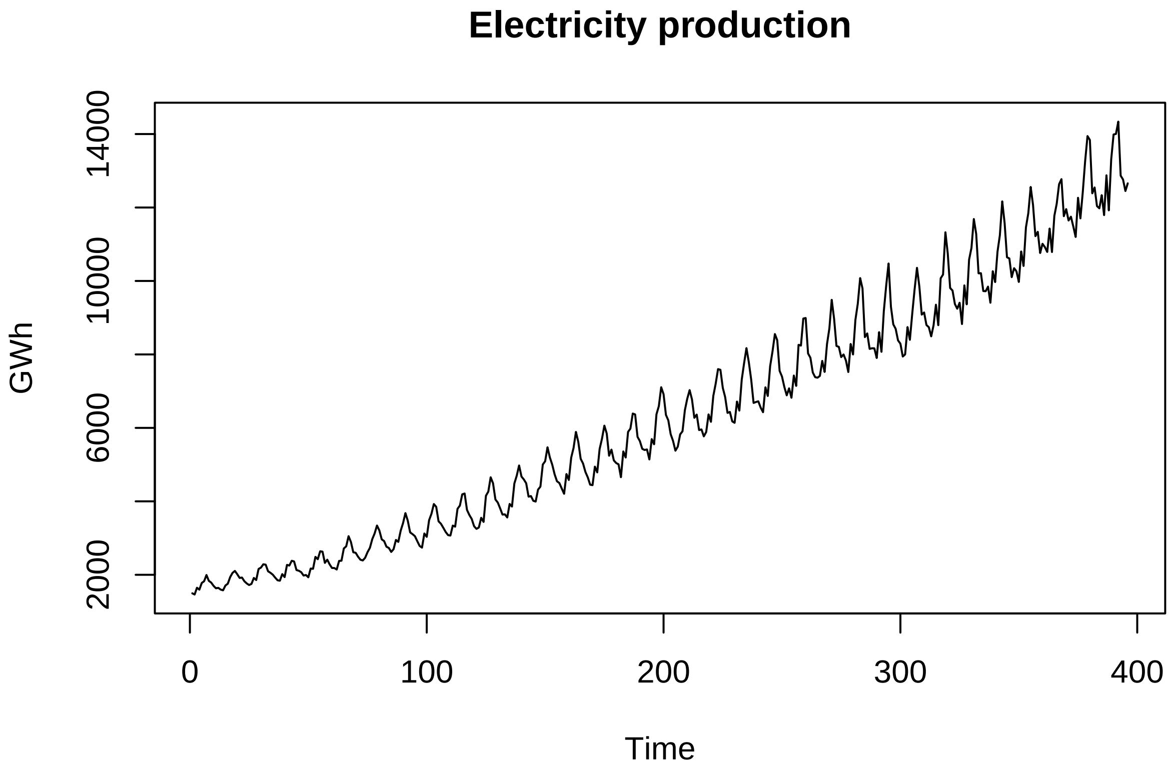
Contents

1 Introduction to stochastic processes

1.1 Data examples

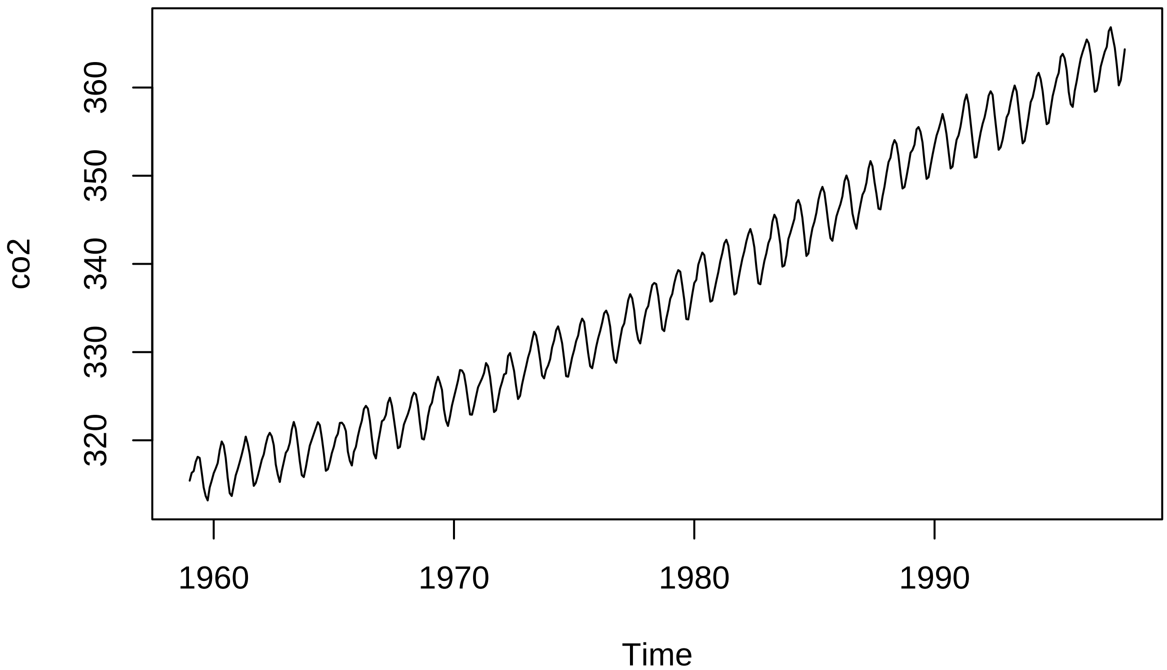
- A special type of data arises when we measure the same variable at different points in time with equal steps between time points.
- This data type is called a (discrete time) **stochastic process** or a **time series**
- One example is the time series of monthly electricity production (GWh) in Australia from Jan. 1958 to Dec. 1990 :

```r
CBEdata <- read.table("https://asta.math.aau.dk/eng/static/datasets?file=cbe.dat", header = TRUE)
CBE <- ts(CBEdata[,3])
plot(CBE, ylab="GWh",main="Electricity production")
```



- Another example is monthly measurements of the atmospheric CO_2 concentration measured at Mauna Loa 1959 - 1997:

```r
dat<-ts(co2)
plot(co2)
```



- Other examples:
  - Hourly wind speed measurements
  - Daily elspot prices
  - An electrical signal measured each millisecond
- Aim: Model, analyse and make predictions for such datasets.

1.2 Stochastic processes

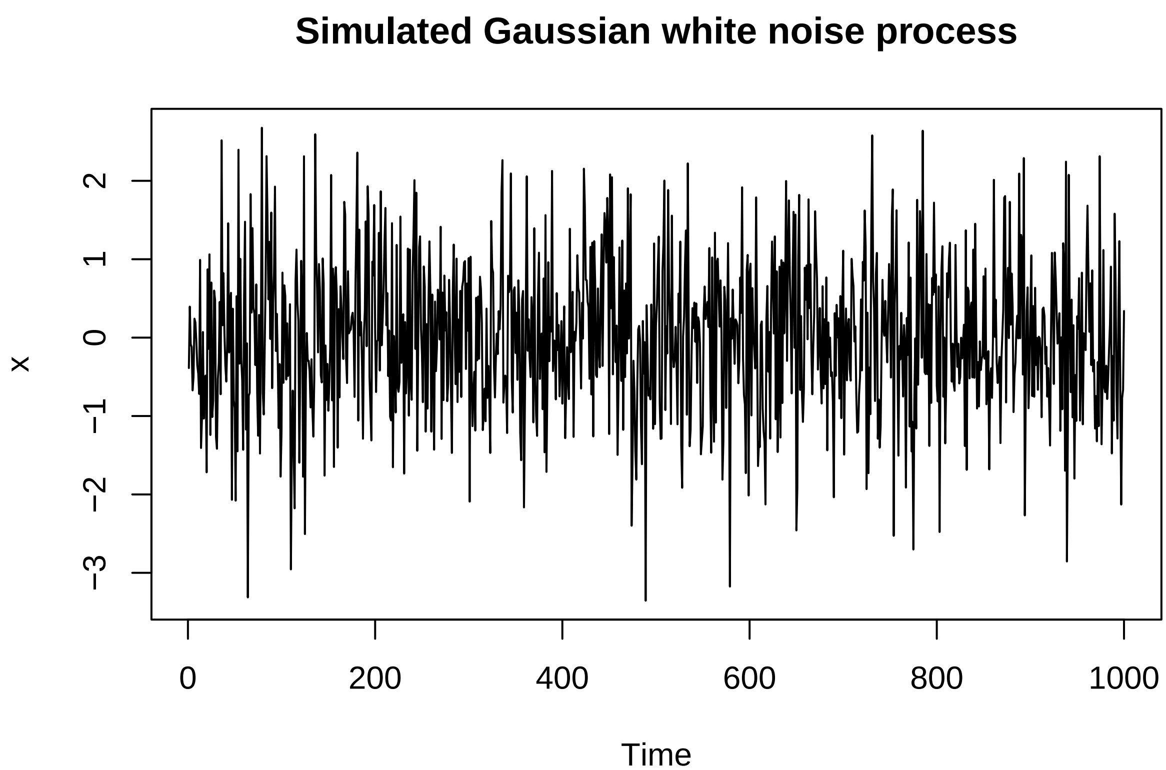
- We denote by X_t the variable at time t. We denote the time points by $t = 1, 2, 3, \dots, n$.
  - We will always assume the data is observed at equidistant points in time (i.e. time steps between consecutive observations are the same).
- Measurements that are close in time will typically be similar: observations are not statistically independent!
- Measurements that are far apart in time will typically be less correlated.

2 Important stochastic processes

2.1 Example 1: White noise

- A stochastic process is called a **white noise process** if the X_t are
  - statistically independent
  - identically distributed
  - have mean 0 and variance σ^2
- It is called **Gaussian white noise**, if
  - $X_t \sim norm(0, \sigma^2)$

```r
x = rnorm(1000,0,1)
ts.plot(x, main = "Simulated Gaussian white noise process")
```



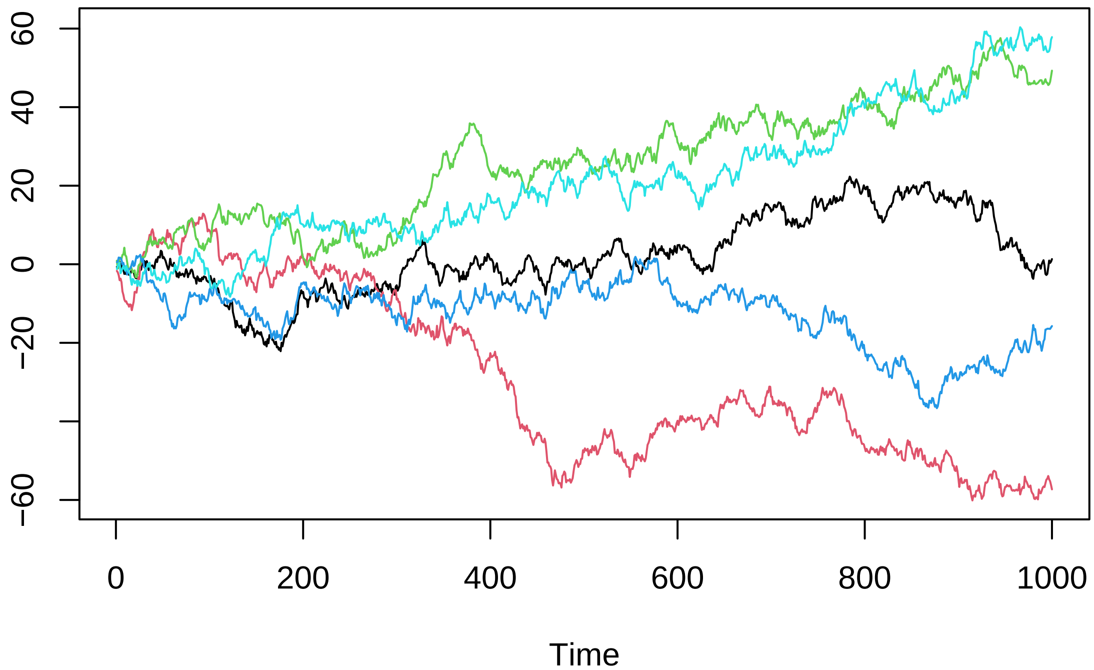
- White noise processes are the simplest stochastic processes.
- Real data does typically not have complete independence between measurements at different time points, so white noise is generally not a good model for real data, but it is a building block for more complicated stochastic processes.

---

3 Example 2: Random walk

- A **random walk** is defined by $X_t = X_{t-1} + W_t$, where W_t is white noise.
- Here are 5 simulations of a random walk:

```r
x = matrix(0,1000,5)
for (i in 1:5) x[,i] = cumsum(rnorm(1000,0,1))
ts.plot(x,col=1:5)
```



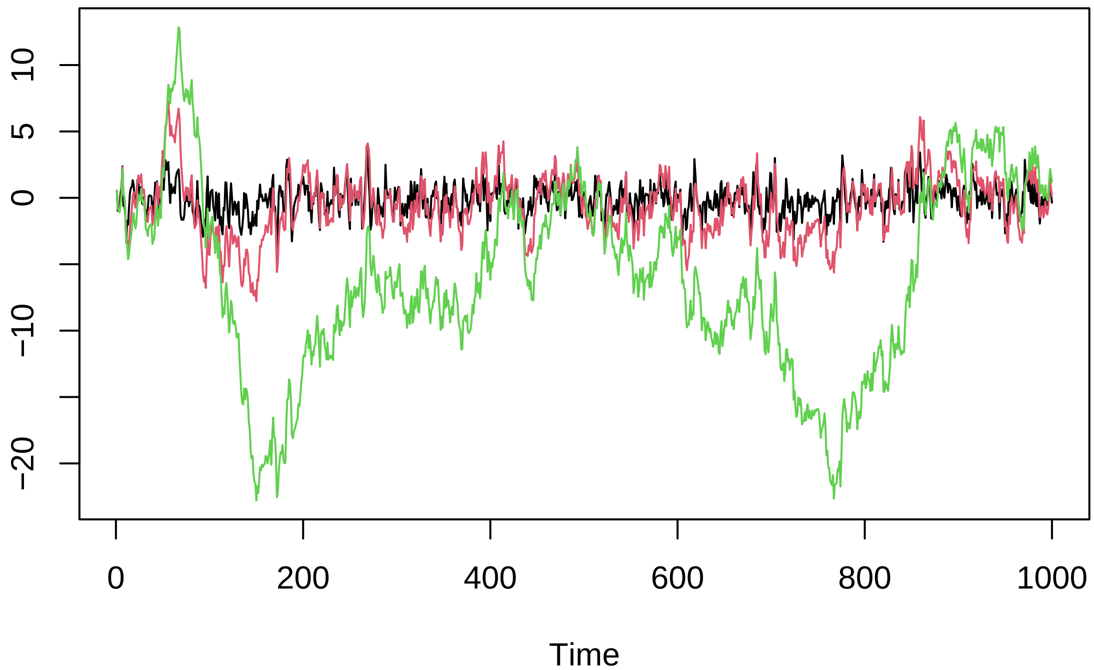
- The random walk may come back to zero after some time, but often it has a tendency to wander of in some random direction.

---

4 Example 3: First order autoregressive process

- A **first order autoregressive process**, AR(1), is defined by $X_t = \alpha X_{t-1} + W_t$, where W_t is white noise and $\alpha \in \mathbb{R}$.
  - Typically $-1 \leq \alpha \leq 1$
  - For $\alpha = 0$ we get white noise
  - For $\alpha = 1$ we get a random walk
- Simulation of 3 AR(1)-processes with different α values:

```
w = ts(rnorm(1000))
x1 = filter(w,0.5,method="recursive")
x2 = filter(w,0.9,method="recursive")
x3 = filter(w,0.99,method="recursive")
ts.plot(x1,x2,x3,col=1:3)
```



- Next time we will consider autoregressive processes in much more detail and higher order, where they become quite flexible models for data.

5 Mean, autocovariance and stationarity

5.1 Mean function

- The **mean function** of a stochastic process is given by

$$\mu_t = \mathbb{E}(X_t)$$

- A process is called first order stationary if $\mu_t = \mu$.
- **Examples:**
  - The white noise process: $\mu_t = 0$ by definition.
  - The random walk:

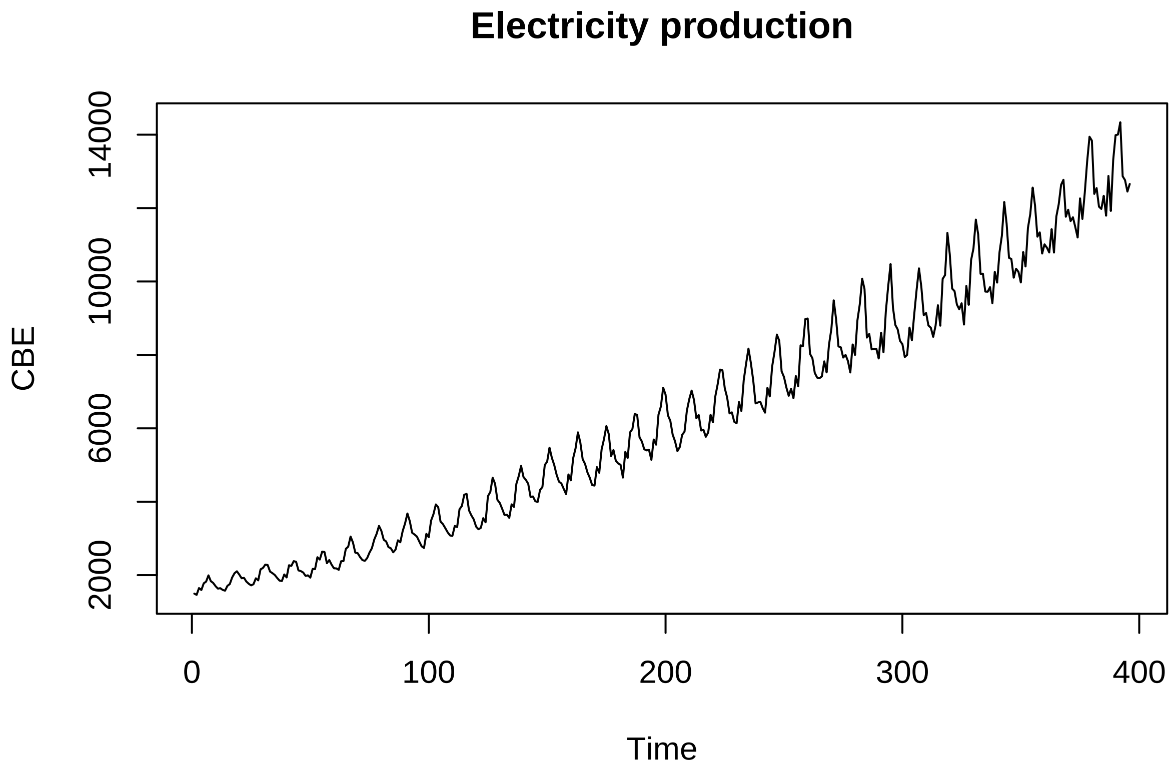
$$\mu_t = \mathbb{E}(X_t) = \mathbb{E}(X_{t-1} + W_t) = \mathbb{E}(X_{t-1}) + \mathbb{E}(W_t) = \mathbb{E}(X_{t-1}) = \mu_{t-1}$$

    So the random walk is first order stationary.
  - Similarly,

$$\mu_t = \mathbb{E}(X_t) = \mathbb{E}(\alpha X_{t-1} + W_t) = \alpha\mathbb{E}(X_{t-1}) + \mathbb{E}(W_t) = \alpha\mathbb{E}(X_{t-1}) = \alpha\mu_{t-1}$$

    The AR(1)-model is first order stationary if $\mu_0 = 0$ or $\alpha = 1$, otherwise not.
  - The electricity production in Australia did not look first order stationary.

```
plot(CBE,main="Electricity production")
```



- The mean function shows the mean behavior of the process, but individual simulations may move far away from this. For example, the random walk has a tendency to move far away from the mean. White noise on the other hand will stay close to the mean.

5.2 Autocovariance/autocorrelation functions

- The **autocovariance** function is given by

$$\gamma(t, t+h) = \text{Cov}(X_t, X_{t+h}) = \mathbb{E}((X_t - \mu_t)(X_{t+h} - \mu_{t+h}))$$

- h is called the **lag**.
- Note that

$$\gamma(t, t) = \text{Var}(X_t) = \sigma_t^2$$

  is the variance at time t.
- The **autocorrelation function (ACF)** is

$$\rho(t, t+h) = \text{Cor}(X_t, X_{t+h}) = \frac{\text{Cov}(X_t, X_{t+h})}{\sigma_t \sigma_{t+h}}$$

- It holds that $\rho(t, t) = 1$, and $\rho(t, t+h)$ is between -1 and 1 for any h.
- The autocorrelation function measures how correlated X_t and X_{t+h} are related:
  - If X_t and X_{t+h} are independent, then $\rho(t, t+h) = 0$
  - If $\rho(t, t+h)$ is close to one, then X_t and X_{t+h} tends to be either high or low at the same time.
  - If $\rho(t, t+h)$ is close to minus one, then when X_t is high X_{t+h} tends to be low and vice versa.

5.3 Stationarity

- We call a stochastic process **second order stationary** if

- the mean is constant, $\mu_t = \mu$
- the variance $\sigma_t^2 = \text{Var}(X_t, X_t)$ is constant.
- the autocorralation function only depends on the lag h:

$$\rho(t, t+h) = \rho(h)$$

- If a process is second order stationary, then also the autocovariance is stationary $\gamma(t, t+h) = \gamma(h)$, i.e. it is a function of only the lag and is easier to work with and plot.
- Intuitively, stationarity means that the process behaves in the same way no matter which time we look at.
- There are other kinds of stationarity, but *in this course, stationarity will always mean second order stationarity.*

5.4 Stationarity and autocorrelation - example

- Consider an AR(1) process $X_t = \alpha X_{t-1} + W_t$. We consider stationarity and autocorrelation for this process.
  - We have already seen that we need $\mu_t = 0$ to have first order stationarity.
- Now consider the variance. Since $X_t = \alpha X_{t-1} + W_t$,

$$\sigma_t^2 = \text{Var}(X_t) = \text{Var}(\alpha X_{t-1} + W_t) = \text{Var}(\alpha X_{t-1}) + \text{Var}(W_t) = \alpha^2 \text{Var}(X_{t-1}) + \text{Var}(W_t) = \alpha^2 \sigma_{t-1}^2 + \sigma^2$$

  - (Here we used that $\text{Var}(X+Y) = \text{Var}(X) + \text{Var}(Y)$ when X and Y are independent).
- If the variance is constant, then $\sigma_t^2 = \sigma_{t-1}^2$ and

$$\sigma_t^2 = \alpha^2 \sigma_t^2 + \sigma^2$$

  - We see that the variance can only be constant if $-1 < \alpha < 1$. In this case $\sigma_t^2 = \frac{\sigma^2}{1-\alpha^2}$.
  - For $|\alpha| \geq 1$, the variance will increase over time. The process is cannot be stationary (including random walk).
- To find the autocorrelation, first observe

$$X_{t+h} = \alpha X_{t+h-1} + W_{t+h} = \cdots = \alpha^h X_t + \sum_{i=0}^{h-1} \alpha^i W_{t+h-i}$$

- Then we find the autocovariance:

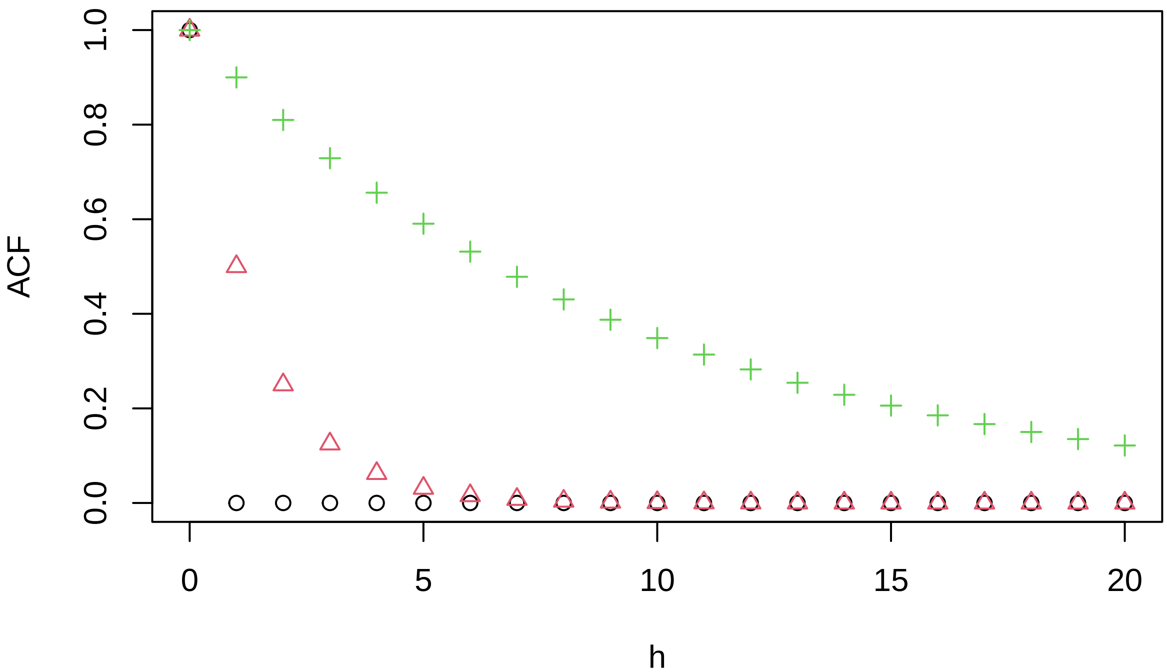
$$\gamma(t, t+h) = \text{Cov}(X_t, X_{t+h}) = \text{Cov}(X_t, \alpha^h X_t + \sum_{i=0}^{h-1} \alpha^i W_{t+h-i}) = \text{Cov}(X_t, \alpha^h X_t) + \text{Cov}(X_t, \sum_{i=0}^{h-1} \alpha^i W_{t+h-i}) = \alpha^h \text{Cov}(X_t, X$$

  - (Here we used the computation rules $\text{Cov}(X, Y+Z) = \text{Cov}(X,Y) + \text{Cov}(X,Z)$ and $\text{Cov}(X, aY) = a\text{Cov}(X,Y)$.)
- If the variance is constant, we can calculate the autocorrelation:

$$\frac{\text{Cov}(X_t, X_{t+h})}{\sigma_t \sigma_{t+h}} = \frac{\alpha^h \sigma^2/(1-\alpha^2)}{\sigma^2/(1-\alpha^2)} = \alpha^h.$$

- So: the AR(1)-model is stationary if $-1 < \alpha < 1$ and $\sigma_t^2 = \sigma^2/(1-\alpha^2)$ - otherwise not.
- The autocorrelation decays exponentially for a stationary AR(1)-model. This is illustrated for 3 different α values:

```r
h = 0:20
acf1 = 0^h  # AR(1) with alpha = 0 (or white noise)
acf2 = 0.5^h  # AR(1) with alpha = 0.5
acf3 = 0.9^h  # Ar(1) with alpha = 0.9
plot(matrix(rep(h,3),3),cbind(acf1,acf2,acf3),col=rep(1:3,each=length(h)),
     pch=rep(1:3,each = length(h)),xlab="h",ylab="ACF")
```



6 Estimation

6.1 Estimation

- The mean and autocovariance/autocorrelation functions are theoretical constructions defined for stochastic processes, but what about data? Here we have to estimate them.
- We will assume that the process is stationary.
- The (constant) mean can be estimated the usual way:

$$\hat{\mu} = \bar{x} = \frac{1}{n}\sum_{t=1}^{n} x_t$$

- The autocovariance function can be estimated as follows (remember it only depends on h, not on t in the case of stationarity):

$$\hat{\gamma}(h) = \frac{1}{n}\sum_{t=1}^{n-h}(x_t - \bar{x})(x_{t+h} - \bar{x})$$

- The (constant) variance is estimated as $\hat{\sigma}^2 = \hat{\gamma}(0)$.
- An estimate of the autocorrelation function is obtained as

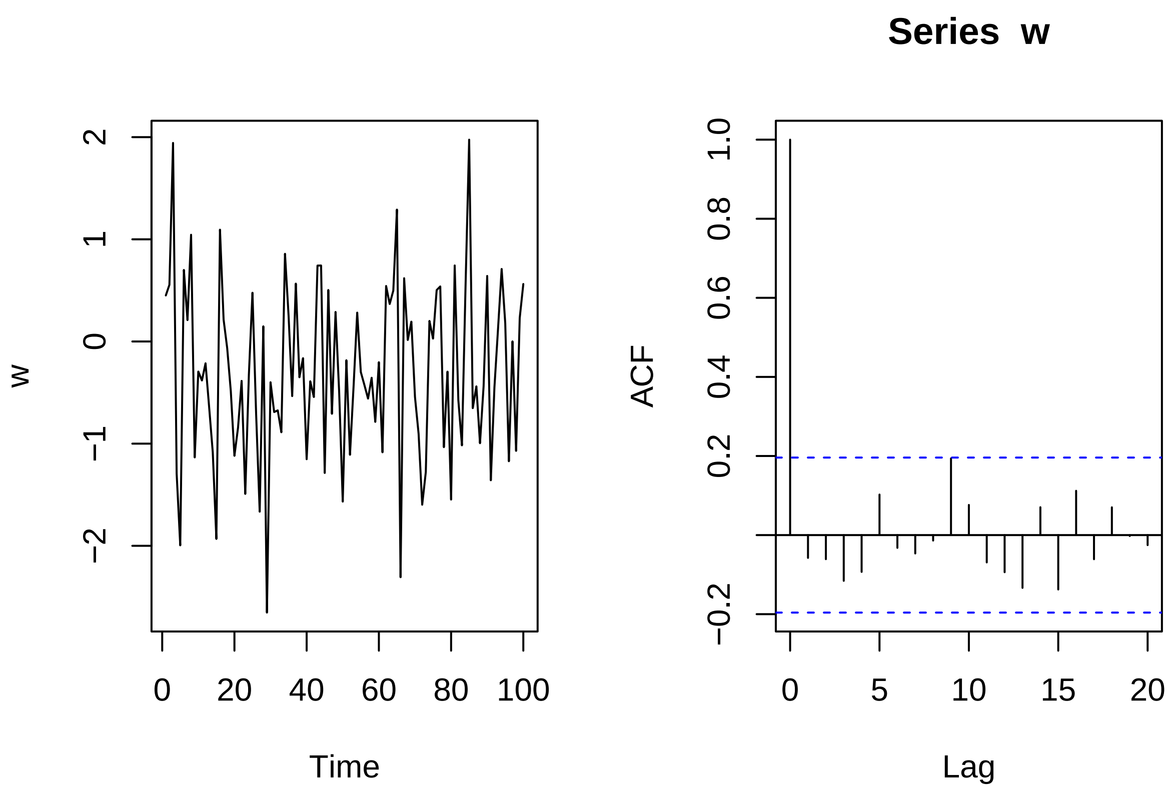
$$\hat{\rho}(h) = \frac{\hat{\gamma}(h)}{\hat{\gamma}(0)}$$

6.2 The correlogram

- A plot of the sample acf as a function of the lag is called a **correlogram**.
- To get an idea of how a correlogram looks, we make simulated data from different models and plot the correlograms below.

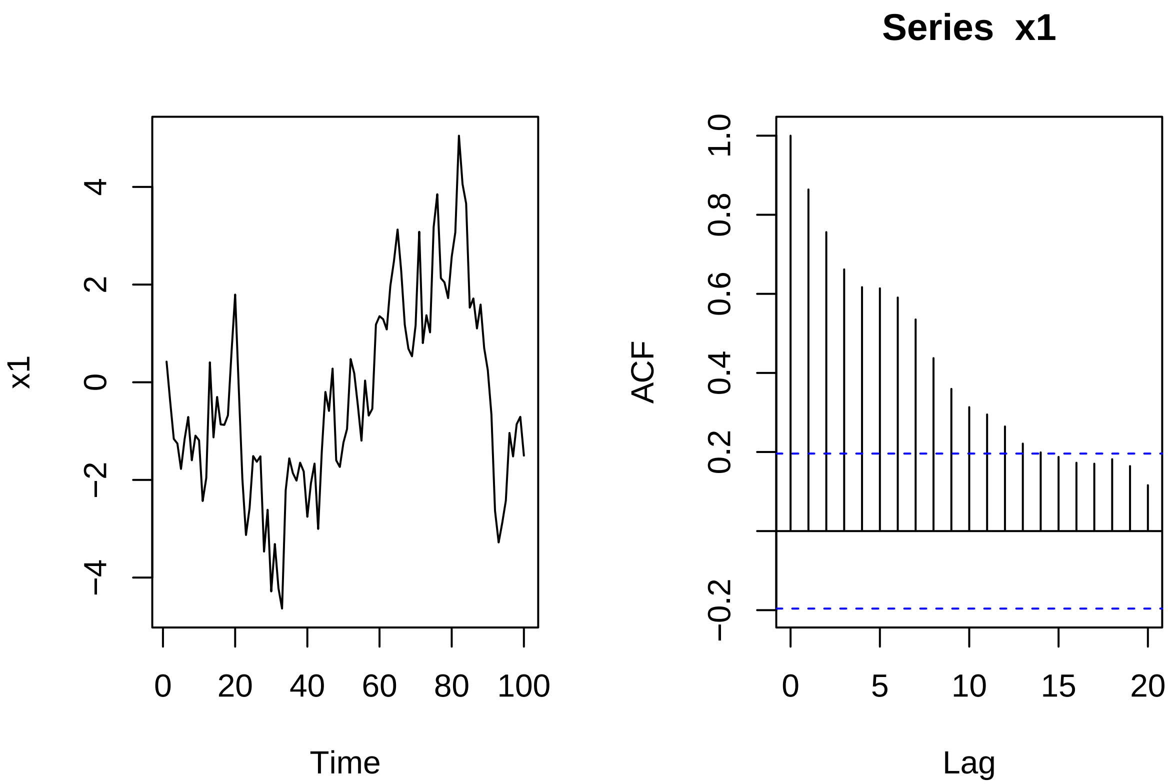
White noise:

```
w = ts(rnorm(100))
par(mfrow=c(1,2))
plot(w)
acf(w)
```



- The correlogram is always 1 at lag 1
- For white noise, the true autocorrelation drops to zero.
- The estimated autocorrelation is never exactly zero - hence we get the small bars.
- The blue lines is a 95% confidence band for a test that the true autocorrelation is zero.
- Remember that there is 5% chance of rejecting a true null hypothesis. Thus, 5% of the bars can be expected to exeed the blue lines.
- AR(1) process with $\alpha = 0.9$:

```
w = ts(rnorm(100))
x1 = filter(w,0.9,method="recursive")
par(mfrow=c(1,2))
plot(x1)
acf(x1)
```



- The true acf decays exponentially.

7 Non-stationary data

7.1 Check for stationarity

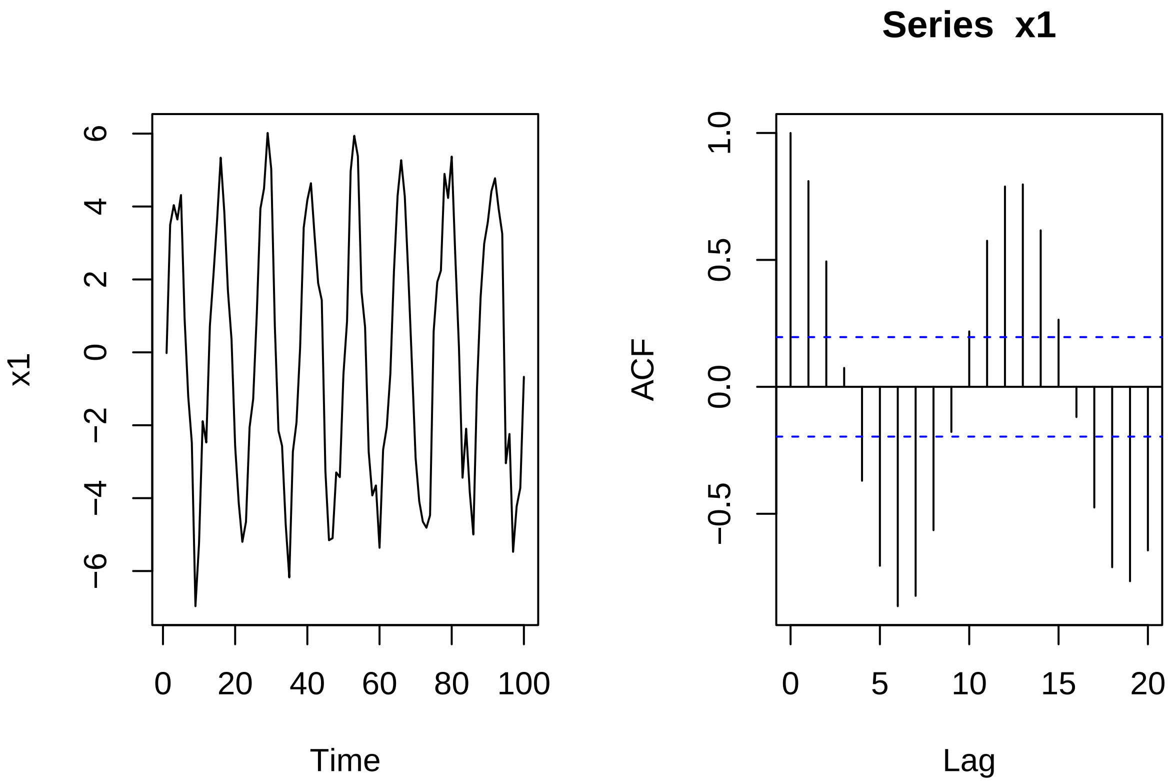
- We will primarily look at stationary processes the next time, but these will not always be good models for data.
- First we need to check whether the assumption of stationarity is okay.
  - One check is visual inspection of a plot of x_t vs t to see whether there is any indication of non-stationarity.
  - Another visual check is a plot of the correlogram. If this tends very slowly to zero, this indicates non-stationarity.
- Note: even though $\rho(h)$ is only well-defined for stationary models, we can plug any data (stationary or not) into the estimation formula. The estimate may help detecting deviations from stationarity.

7.2 Correlograms for non-stationary data

- Sine curve with added white noise:

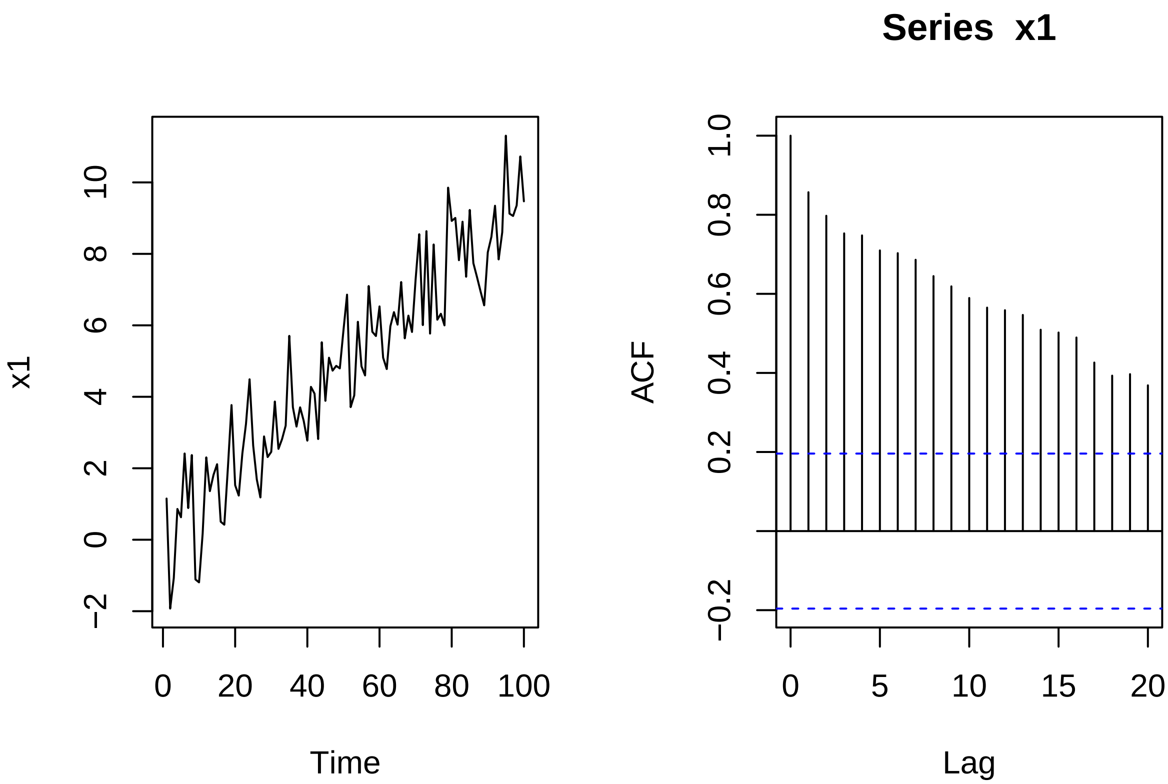
```
w = ts(rnorm(100))
x1 = 5*sin(0.5*(1:100)) + w
par(mfrow=c(1,2))
```

```
plot(x1)
acf(x1)
```



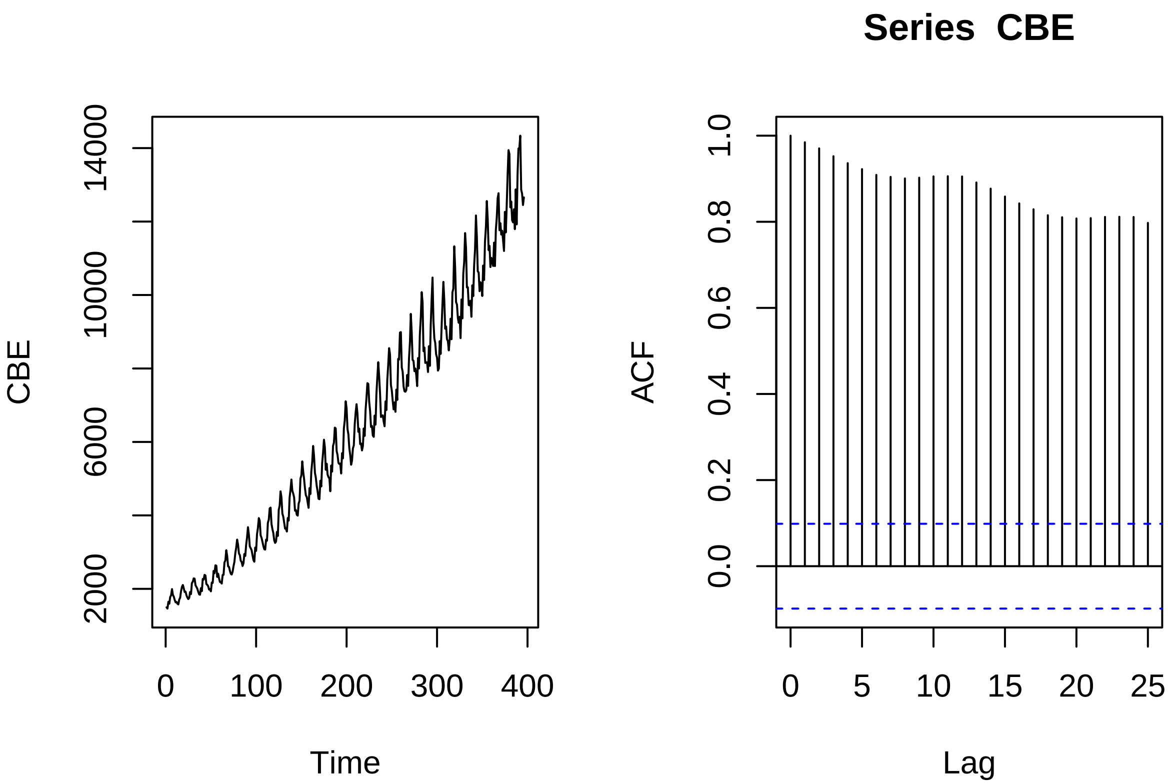
- The periodic mean of the process results in a periodic behavior of the correlogram.
- A periodic behavior in the correlogram suggests seasonal behavior in the process.
- Straight line with added white noise:

```
w = ts(rnorm(100))
x1 = 0.1*(1:100) + w
par(mfrow=c(1,2))
plot(x1)
acf(x1)
```



- The linear trend results in a slowly decaying, almost linear correlogram.
- Such a correlogram suggests a trend in the data.
- Data example: Electricity production.

```
par(mfrow=c(1,2))
plot(CBE)
acf(CBE)
```



- There seems to be an increasing trend in the data.
- There is a periodic behavior around the increasing trend.
- It is reasonable to believe that the period is 12 months.
- We have the model

$$X_t = m_t + s_t + Z_t$$

  where

  – m_t is the (deterministic) trend
  – s_t is a (deterministic) seasonal term ($s_t = s_{t+12}$)
  – Z_t is a random (hopefully) stationary part

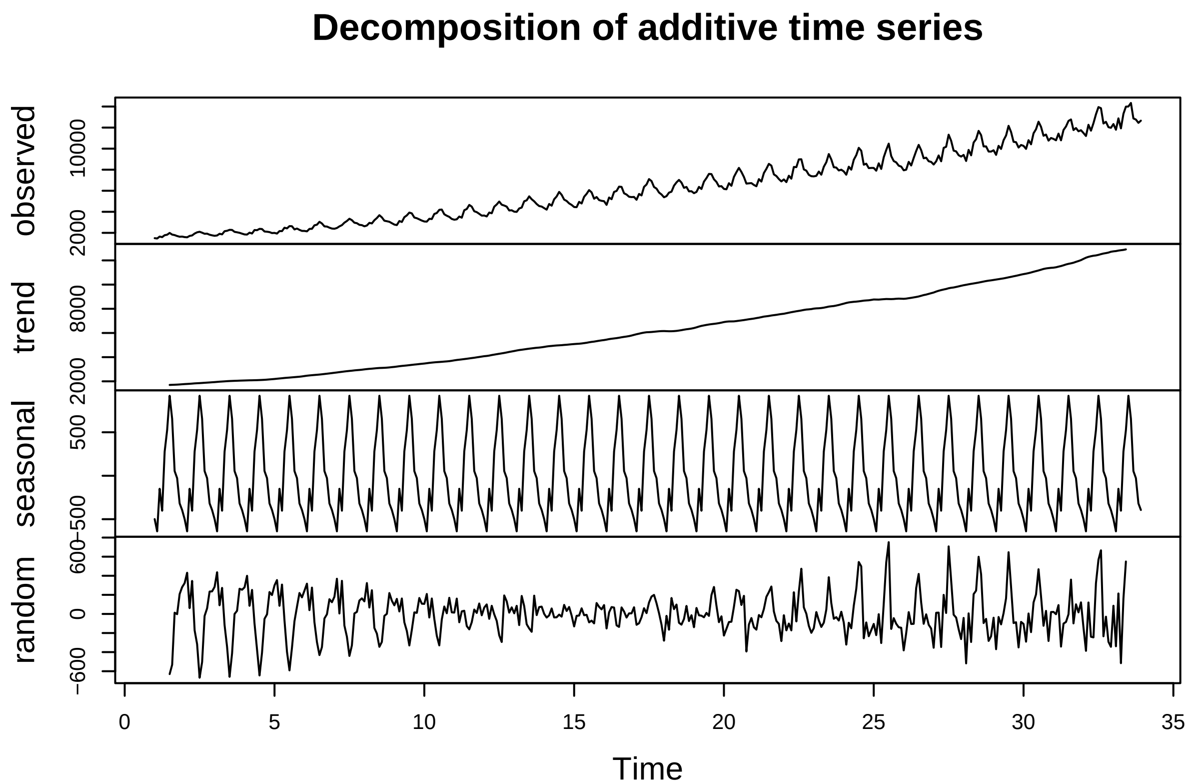
7.3 Detrending data

- The trend m_t in the data can be estimated by a **moving average**.
- In the case of monthly variation,

$$\hat{m}_t = \frac{\frac{1}{2}x_{t-6} + x_{t-5} + \cdots + x_t + \cdots + x_{t+5} + \frac{1}{2}x_{t+6}}{12}$$

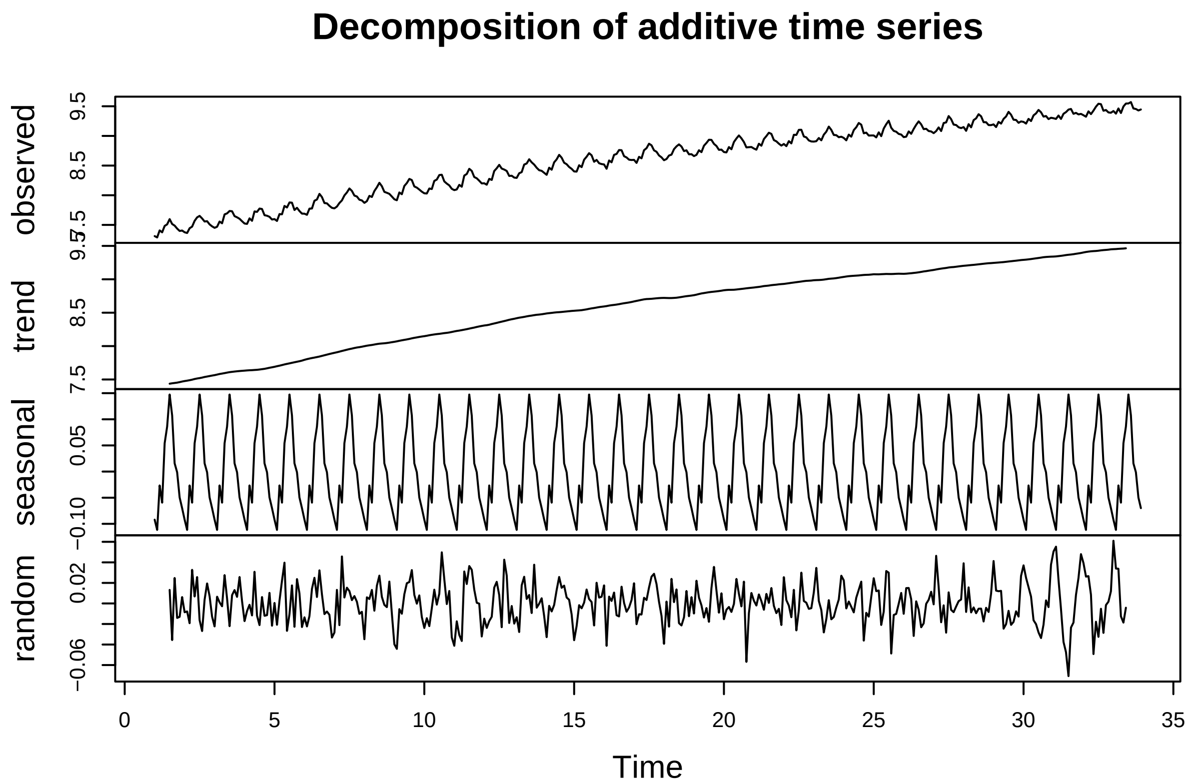
- We remove the trend by considering $x_t - \hat{m}_t$.
- Next we find the seasonal term s_t by averaging $x_t - \hat{m}_t$ over all measurements in the given month.
  – E.g., the value of s_t for January is given by averaging all values from January.
- We are left with the random part $\hat{z}_t = x_t - \hat{m}_t - \hat{s}_t$.
- For the Australian electricity data:

```r
CBE <- ts(CBE,frequency=12)
plot(decompose(CBE))
```

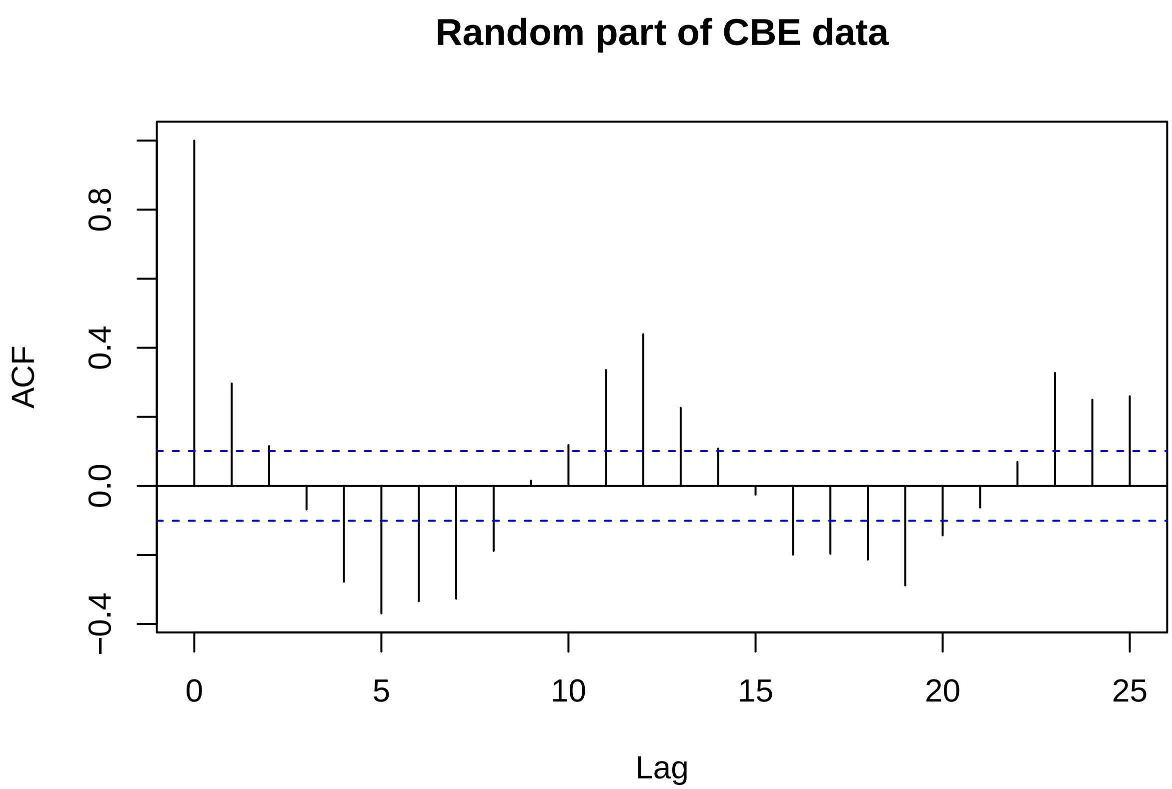


- The random term does not look stationary. The solution is to log-transform the data - see Ch. 1.5.5 in the book.

```r
logCBE <- ts(log(CBEdata[,3]),frequency=12)
plot(decompose(logCBE))
```



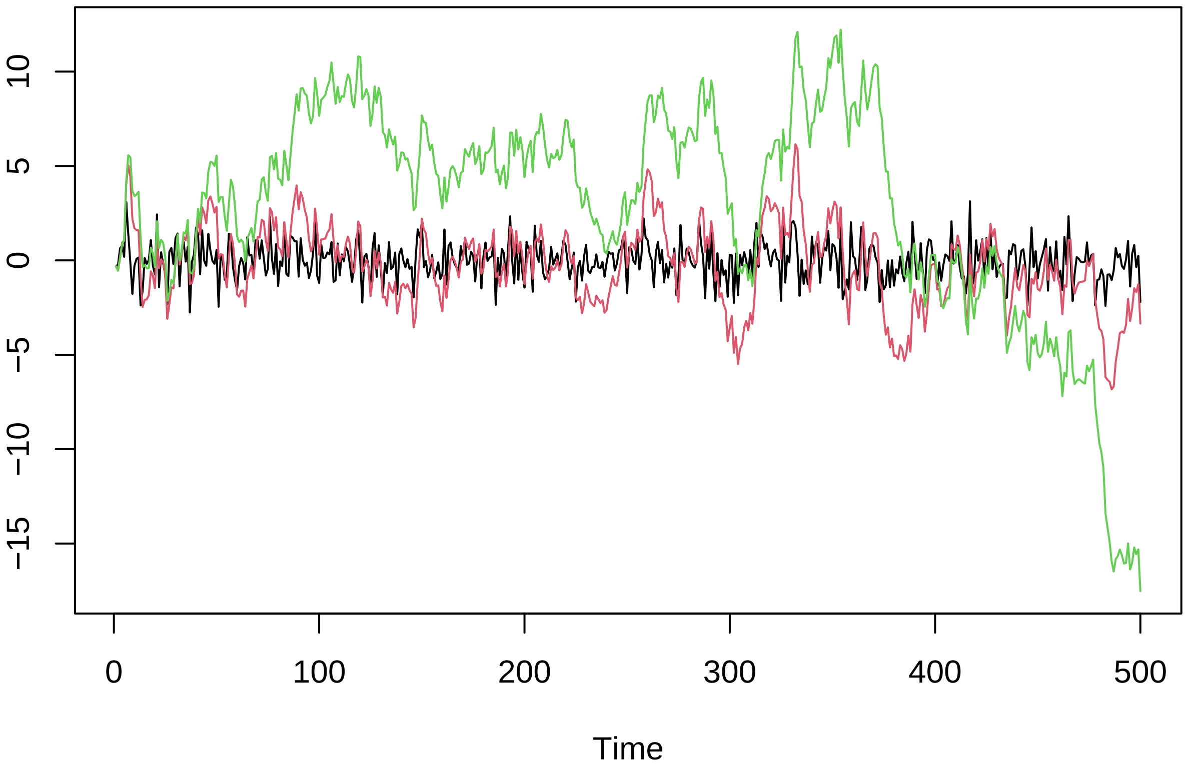

```
random<-decompose(logCBE)$random[7:382]
acf(random, main="Random part of CBE data")
```



8 Basic stochastic process models

8.1 Stochastic processes

- Last time we introduced stochastic processes



- We saw three basic models:
  - **White noise:** independent random variables - not very interesting by itself, but an essential building block in more complicated/realistic models.
  - **Random walk:** cumulatively adding white noise - non-stationary, tendency to wander off.
  - **Autoregressive process AR(1):** weaker dependence on previous value than random walk, is stationary or not depending on choice of parameters.
- Today we will consider more stochastic process models.
- First we take a recap of the three models above, and add some details.

9 White noise

9.1 White noise

- A time series W_t, $t = 1, \dots, n$ is **white noise** if the variables $W_1, W_2, \dots, W_n$ are
  - independent
  - identically distributed
  - have mean zero and variance σ^2
- From the definition it follows that white noise is a second order stationary process since

- The mean is constant ($= 0$)
- The variance function $\sigma^2(t) = \sigma^2$ is constant
- The autocorrelation $Cor(W_t, W_{t+k}) = 0$ for all $k \neq 0$, which does not depend on t.

- Hence, we have a well-defined mean,

$$\mu = 0,$$

autocovariance function

$$\gamma(k) = Cov(W_t, W_{t+k}) = \begin{cases} \sigma^2 \text{ for } k = 0, \\ 0 \text{ for } k \neq 0, \end{cases}$$

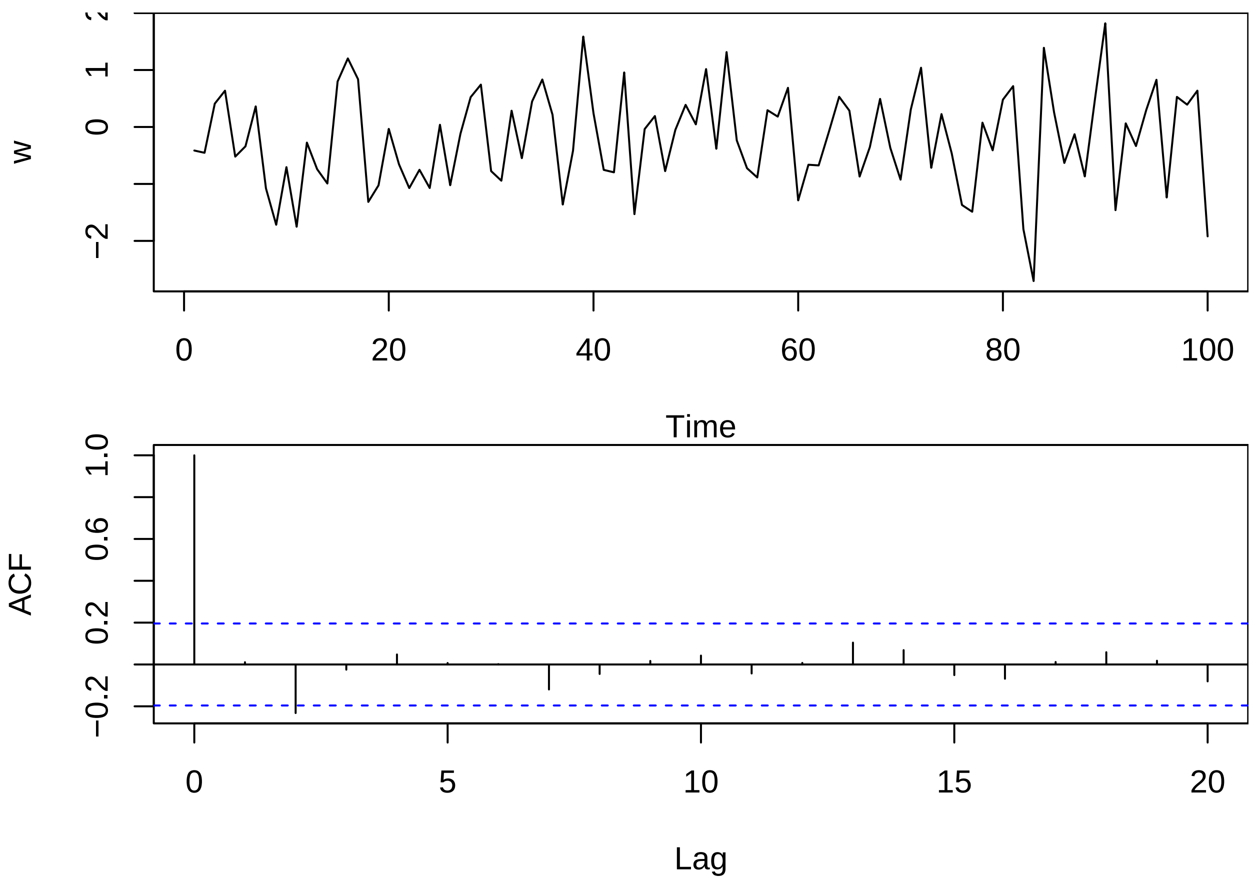
and autocorrelation function

$$\rho(k) = \begin{cases} 1 \text{ for } k = 0, \\ 0 \text{ for } k \neq 0. \end{cases}$$

9.2 Correlogram for white noise

- To understand how white noise behaves we can simulate it with R and plot both the series and the autocorrelation:

```
w <- rnorm(100, mean = 0, sd = 1)
par(mfrow = c(2,1), mar = c(4,4,0,0))
ts.plot(w)
acf(w)
```



- 95% of the estimated autocorrelations at lag $k > 0$ lie between the blue lines.

- It is a good idea to repeat this simulation and plot a few times to appreciate the variability of the results.

10 Random walk

10.1 Random walk

- A time series X_t is called a **random walk** if

$$X_t = X_{t-1} + W_t$$

  where W_t is a white noise series.
- Using $X_{t-1} = X_{t-2} + W_{t-1}$ we get

$$X_t = X_{t-2} + W_{t-1} + W_t$$

- Substituting for X_{t-2} we get

$$X_t = X_{t-3} + W_{t-2} + W_{t-1} + W_t$$

- Continuing this way, assuming we start at $X_0 = 0$,

$$X_t = W_1 + W_2 + \cdots + W_t$$

10.2 Properties of random walk

- A random walk X_t has a constant mean function

$$\mu(t) = 0.$$

- The variance function is given by

$$\sigma^2(t) = Var(X_t) = Var(W_1 + \cdots + W_t) = Var(W_1) + \cdots + Var(W_t) = t \cdot \sigma^2,$$

  which clearly depends on the time t, so the process is not stationary.
- Similarly, one can compute the autocovariance and autocorrelation function (see Cowpertwait and Metcalfe for details)

$$Cov(X_t, X_{t+k}) = t\sigma^2.$$

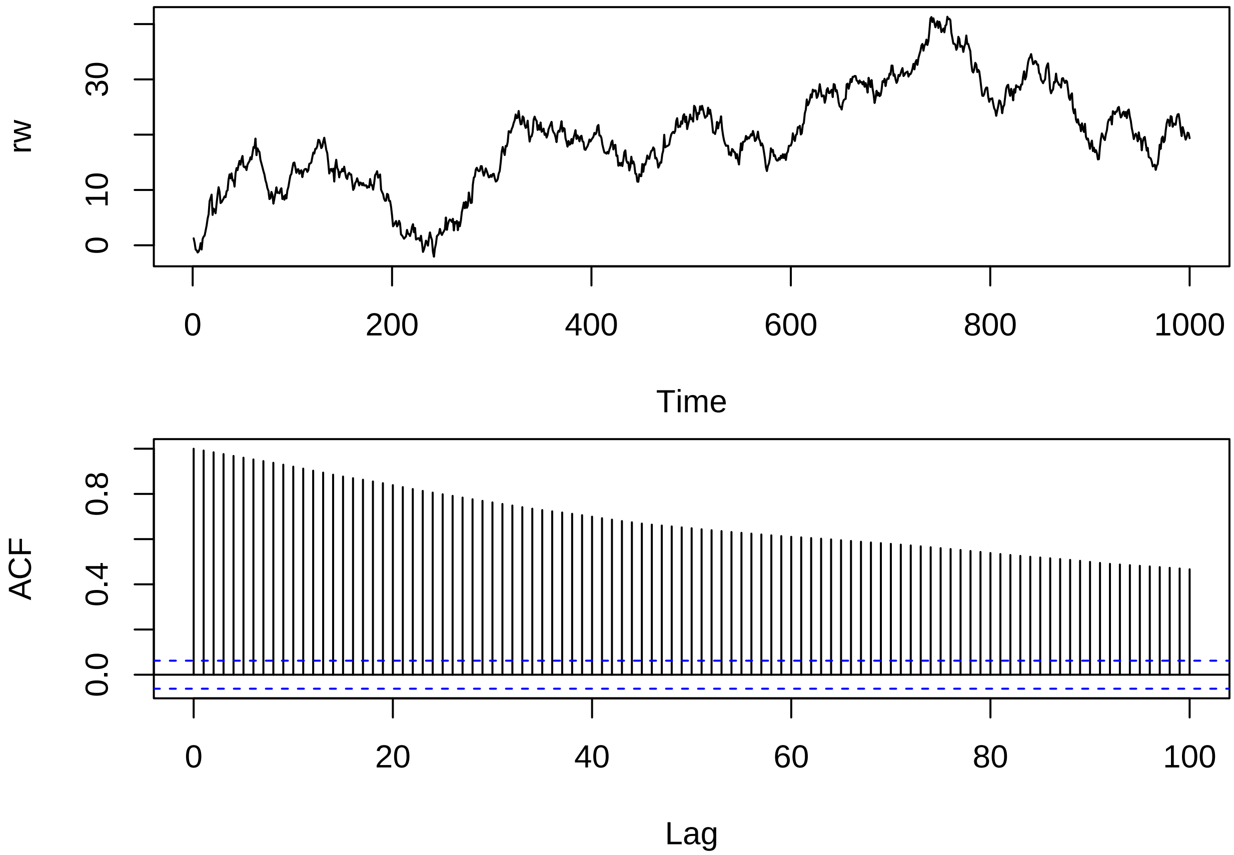
$$Cor(X_t, X_{t+k}) = \frac{1}{\sqrt{1 + k/t}}$$

- The autocorrelation is non-stationary.
- When t is large compared to k we have very high correlation (close to 1)
- We expect the correlogram of a reasonably long random walk to show very slow decay.

10.3 Simulation and correlogram of random walk

- We already know how to simulate Gaussian white noise W_t(with `rnorm`) and the random walk is just generated by repeatedly adding noise terms:

```r
w <- rnorm(1000)
rw<-w
for(t in 2:1000) rw[t]<- rw[t-1] + w[t]
par(mfrow = c(2,1), mar = c(4,4,0.5,0.5))
ts.plot(rw)
acf(rw, lag.max = 100)
```



- The slowly decaying acf for random walk is a classical sign of non-stationarity, indicating there may be some kind of trend. In this case there is no real trend, since the theoretical mean is constant zero, but we refer to the apparent trend which seems to change directions unpredictiably as a **stochastic trend**.

10.4 Differencing

- If a non-stationary time series shows a stochastic trend we can try to study the **time series of differences** and see if that is stationary and easier to understand:

$$\nabla X_t = X_t - X_{t-1}.$$

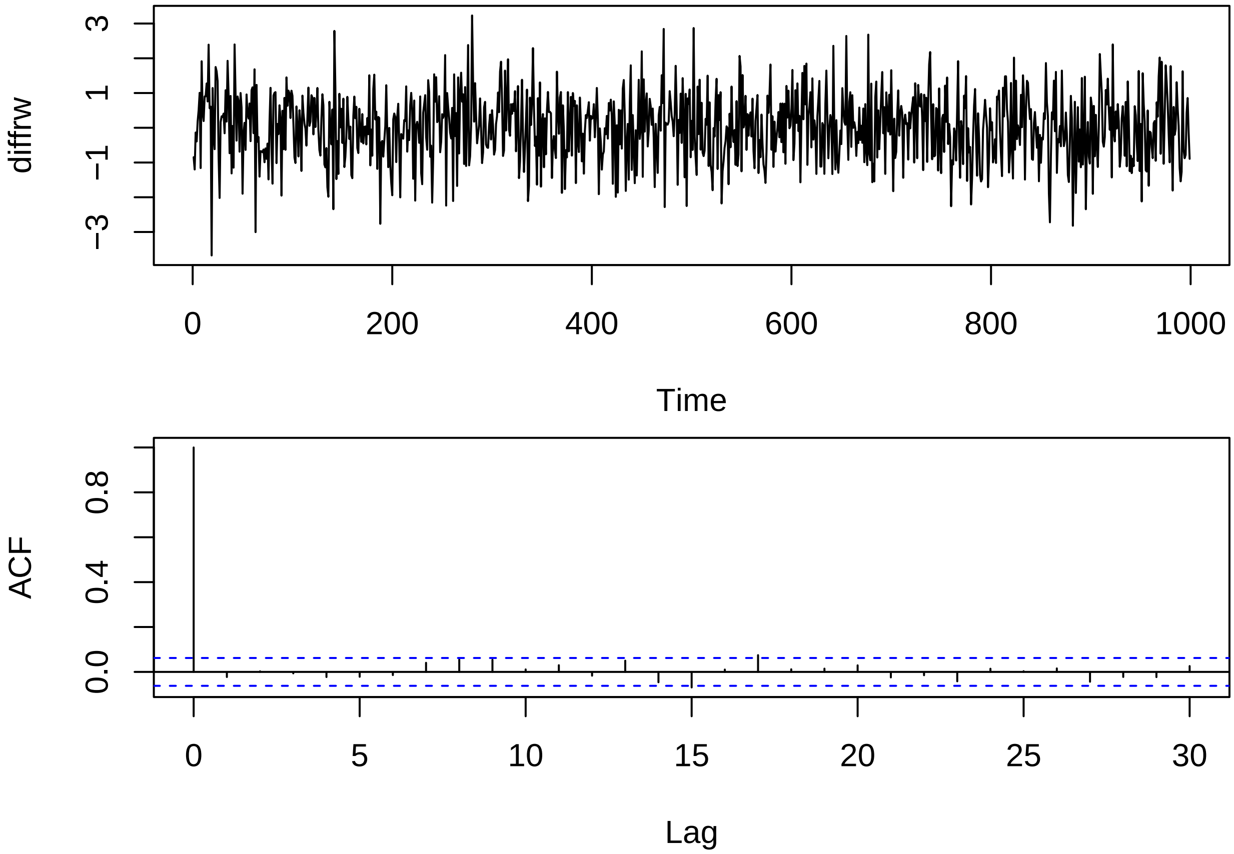
- Since we assume $X_0 = 0$ we get

$$\nabla X_1 = X_1.$$

- Specifically when we difference a random walk $X_t = X_{t-1} + W_t$ we recover the white noise series

$$\nabla X_t = W_t$$

```r
diffrw <- diff(rw)
par(mfrow = c(2,1), mar = c(4,4,0.5,0.5))
ts.plot(diffrw)
acf(diffrw, lag.max = 30)
```

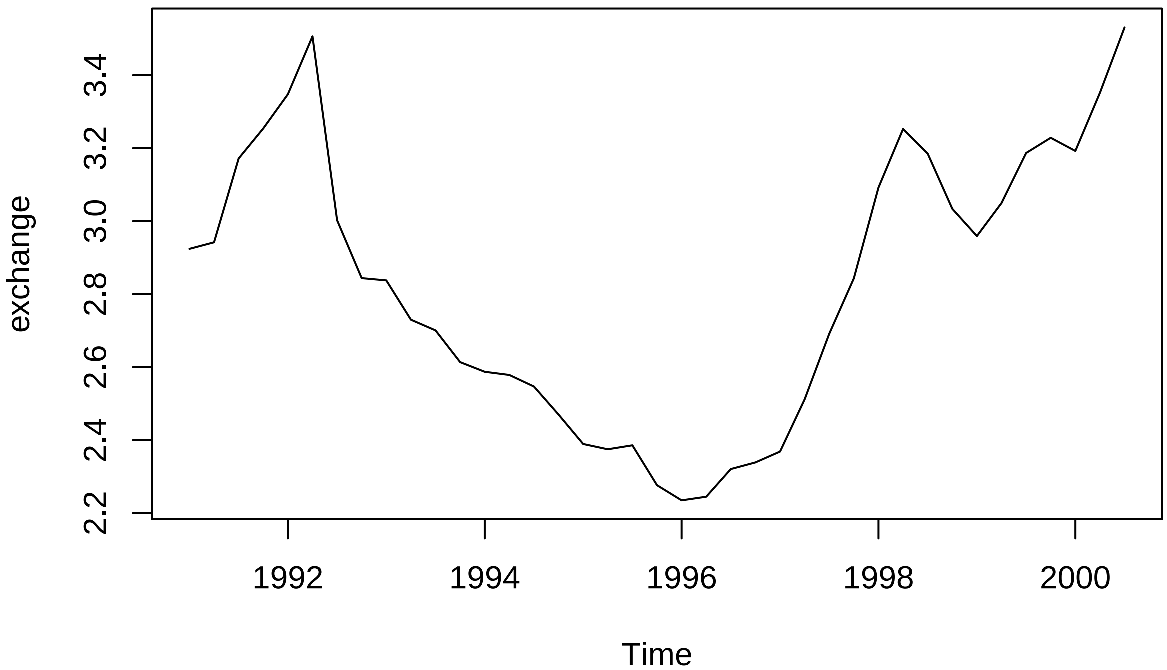


- If a time series needs to be differenced to become stationary, we say that the series is **integrated** (of order 1).

10.5 Example: Exchange rate

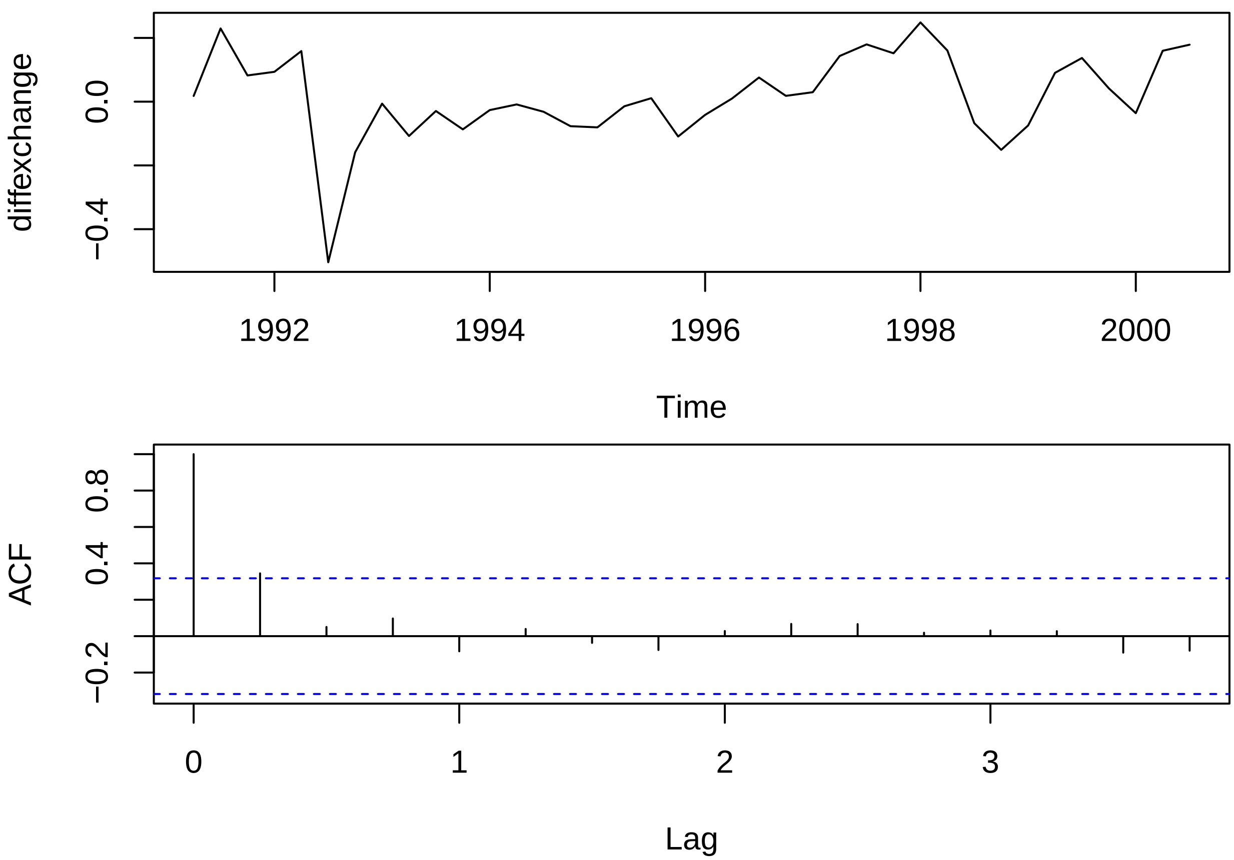
- We consider a data set for the exchange rate from GBP to NZD. We observe what looks like an unpredictable stochastic trend and we would like to see if it could reasonably be described as a random walk.

```r
www <- "https://asta.math.aau.dk/eng/static/datasets?file=pounds_nz.dat"
exchange_data <- read.table(www, header = TRUE)[[1]]
exchange <- ts(exchange_data, start = 1991, freq = 4)
plot(exchange)
```



- We difference the series and see if the differences looks like white noise:

```
diffexchange <- diff(exchange)
par(mfrow = c(2,1), mar = c(4,4,0.5,0.5))
plot(diffexchange)
acf(diffexchange)
```



- The first order difference looks reasonably stationary, so the original exchange rate series could be considered integrated of order 1. However, there is an indication of significant autocorrelation at lag 1, so a random walk might not be a completely satisfactory model for this dataset.

11 First order auto-regressive models AR(1)

11.1 Auto-regressive model of order 1: AR(1)

- If the correlogram shows a significant auto-correlation at lag 1, it means that X_t and X_{t-1} are correlated.
- The simplest way to model this, is the auto-regressive model of order one, AR(1):

$$X_t = \alpha_1 X_{t-1} + W_t$$

  where W_t is white noise and the auto-regressive coefficient α_1 is a parameter to be estimated from data.

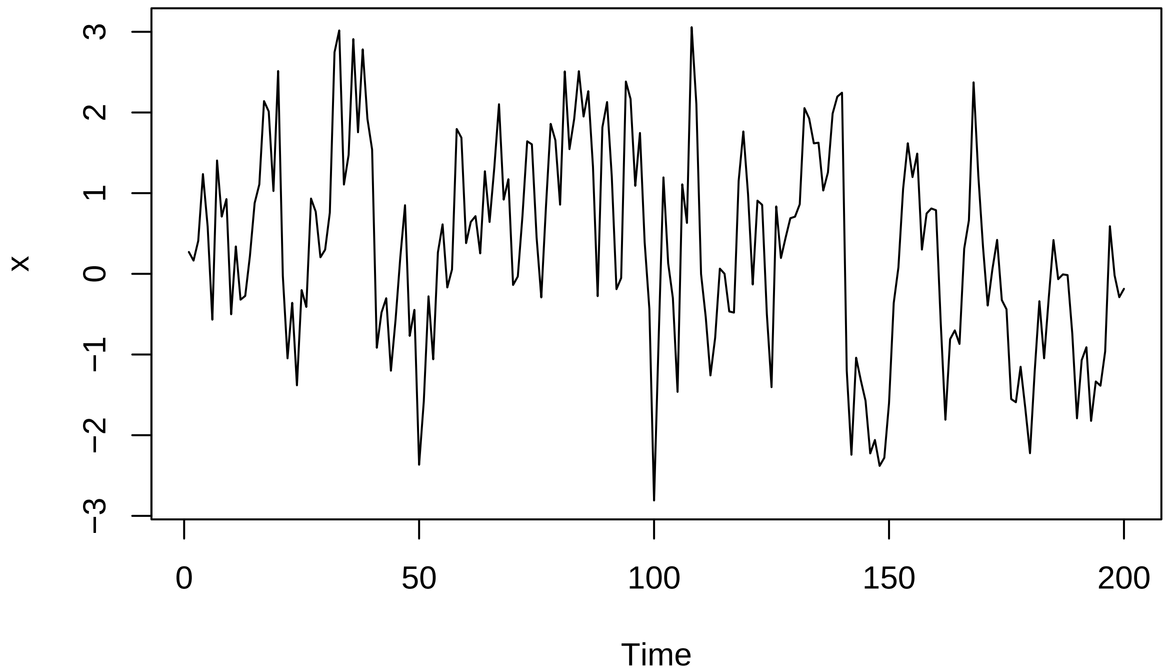
11.2 Properties of AR(1) models

- From last time: The model can only be stationary if $-1 < \alpha_1 < 1$ such that the dependence on the past decreases with time.
- For a stationary AR(1) model with $-1 < \alpha_1 < 1$ we found
  - $\mu(t) = 0$
  - $Var(X_t) = \sigma_t^2 = \sigma^2/(1-\alpha_1^2)$
  - $\gamma(k) = \alpha_1^k \sigma^2/(1-\alpha_1^2)$
  - $\rho(k) = \alpha_1^k$

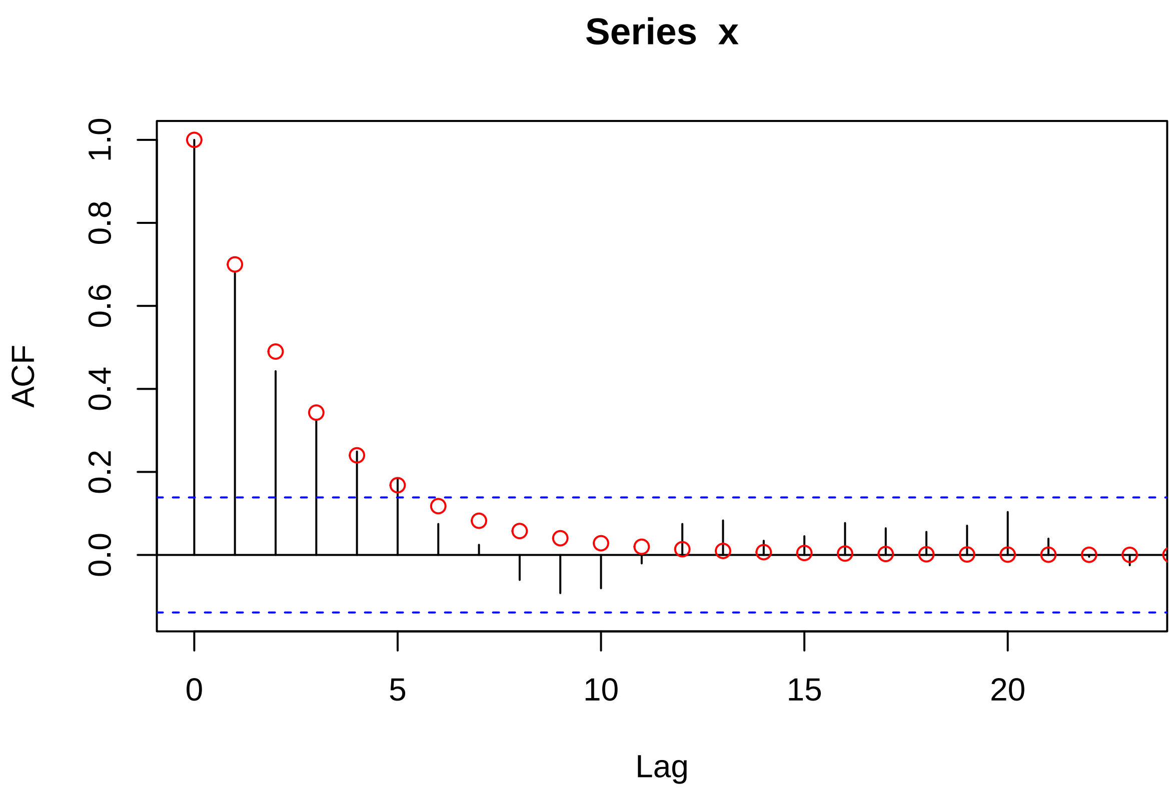
11.3 Simulation of AR(1) models

- R has a built-in function `arima.sim` to simulate AR(1) models (and other more complicated models called ARMA and ARIMA).
- It needs the model (i.e. the autoregressive coefficient α_1) and the desired number of time steps `n`.
- To simulate 200 time steps of AR(1) with $\alpha_1 = 0.7$ we do:

```
x <- arima.sim(model = list(ar = 0.7), n = 200)
plot(x)
```




```
acf(x)
k = 0:30
points(k, 0.7^k, col = "red")
```



- Here we have compared the empirical correlogram with the theoretical values of the model.

11.4 Fitted AR(1) models

- There are several ways to estimate the parameters in an AR(1) process. We use the so-called maximum likelihood estimation method (MLE).
- We skip the details of this, and simply use the function `ar`:

```
fit <- ar(x, order.max = 1, method="mle")
```

- The resulting object contains the value of the estimated parameter $\hat{\alpha}_1$ and a bunch of other information.
- In this example, the input data are artificial so we know that $\hat{\alpha}_1$ should ideally be close to 0.7:

```
fit$ar
```

```
## [1] 0.6764996
```

- An estimate of the variance (squared standard error) of $\hat{\alpha}_1$ is given in `fit$asy.var.coef`

```
fit$asy.var.coef
```

```
##             [,1]
## [1,] 0.002690941
```

- The estimated std. error is the square root of this:

```
se <- sqrt(fit$asy.var.coef)
se
```

```
##            [,1]
## [1,] 0.05187428
```

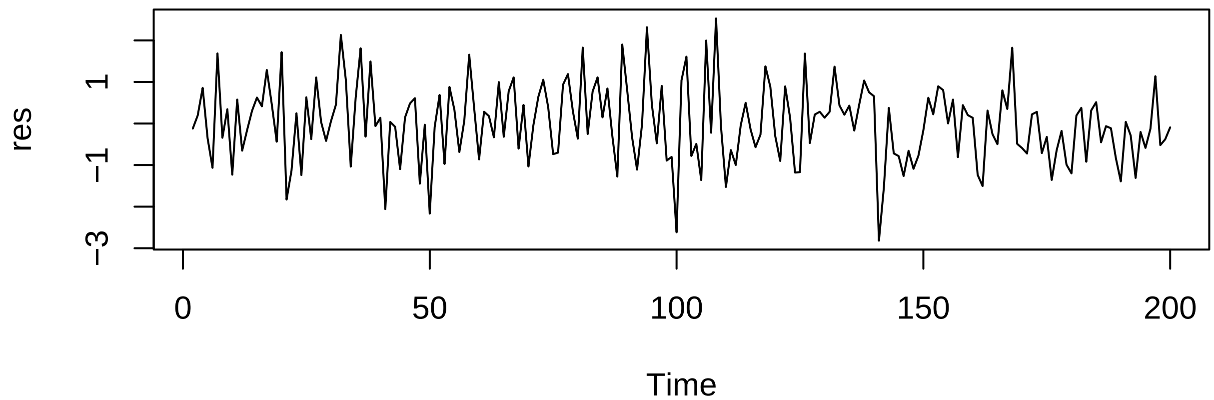
```
ci <- c(fit$ar - 2*se, fit$ar + 2*se)
ci
```

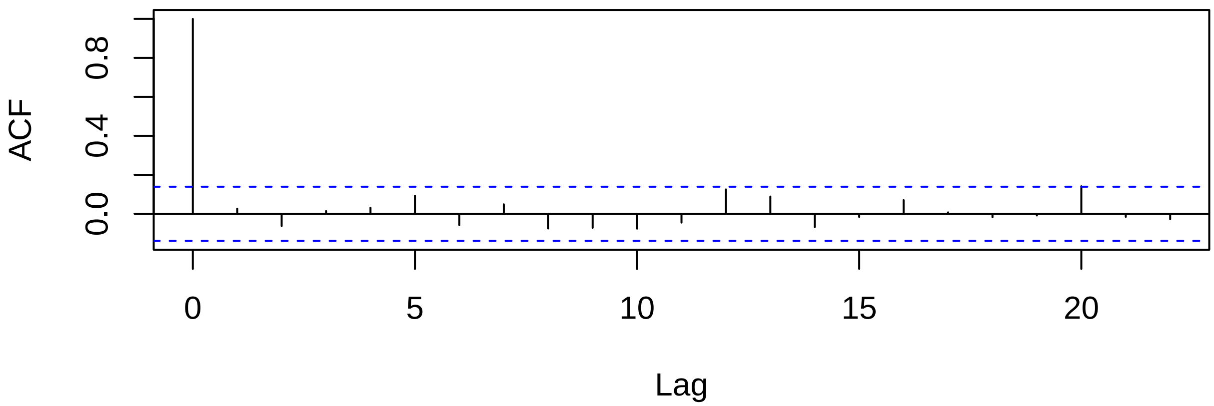
```
## [1] 0.5727511 0.7802482
```

11.5 Residuals for AR(1) models

- The AR(1) model defined earlier has mean 0. However, we cannot expect data to fulfill this.
- This is fixed by subtracting the average $\bar{X}$ of the data before doing anything else, so the model that is fitted is actually:
$$X_t - \bar{X} = \alpha_1 \cdot (X_{t-1} - \bar{X}) + W_t$$
- The fitted model can be used to make predictions. To predict X_t from X_{t-1}, we use that W_t is white noise, so we expect it to be zero on average:
$$\hat{x}_t - \bar{x} = \hat{\alpha}_1 \cdot (x_{t-1} - \bar{x}),$$
so
$$\hat{x}_t = \bar{x} + \hat{\alpha}_1 \cdot (x_{t-1} - \bar{x}), \quad t \geq 2.$$
- We can estimate the white noise terms as usual by the model residuals:
$$\hat{w}_t = x_t - \hat{x}_t, \quad t \geq 2.$$
- If the model is correct, the residuals should look like a sample of white noise. We check this by plotting the acf:

```
res <- na.omit(fit$resid)
par(mfrow = c(2,1), mar = c(4,4,1,1))
plot(res)
acf(res)
```





- It naturally looks good for this artificial dataset.

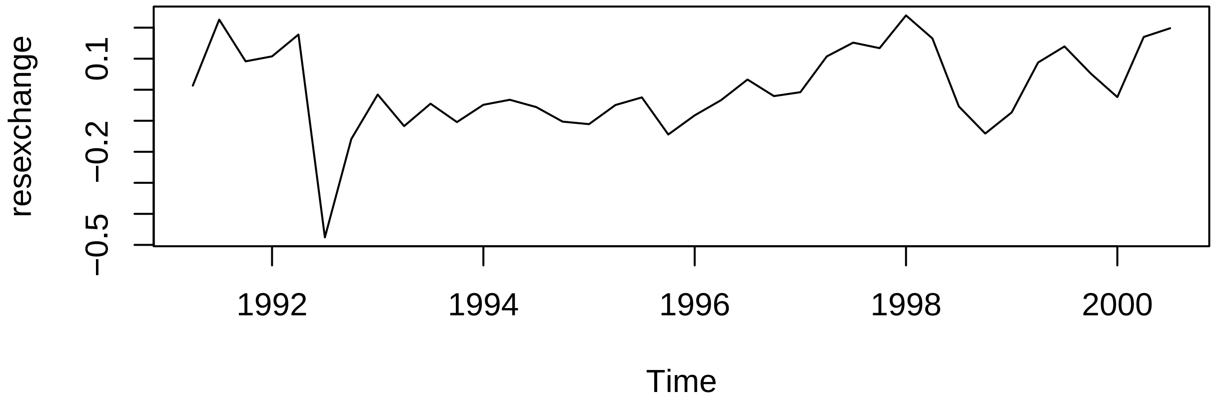
11.6 AR(1) model fitted to exchange rate

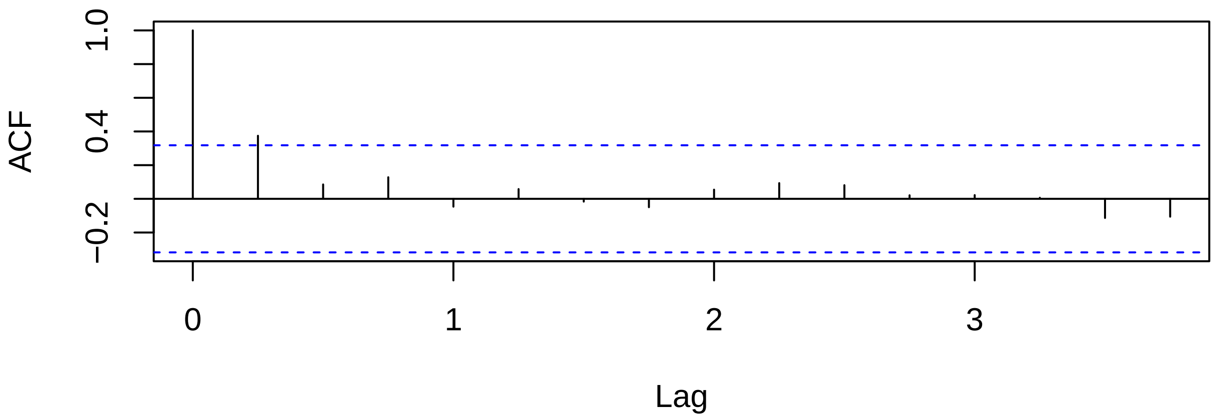
- A random walk is an example of an AR(1) model with $\alpha_1 = 1$. It didn't provide an ideal fit for the exchange rate dataset, so we might suggest a stationary AR(1) model with α_1 as a parameter to be estimated from data:

```
fitexchange <- ar(exchange, order.max = 1, method="mle")
fitexchange$ar
```

```
## [1] 0.9437125
```

```
resexchange <- na.omit(fitexchange$resid)
par(mfrow = c(2,1), mar = c(4,4,1,1))
plot(resexchange)
acf(resexchange)
```



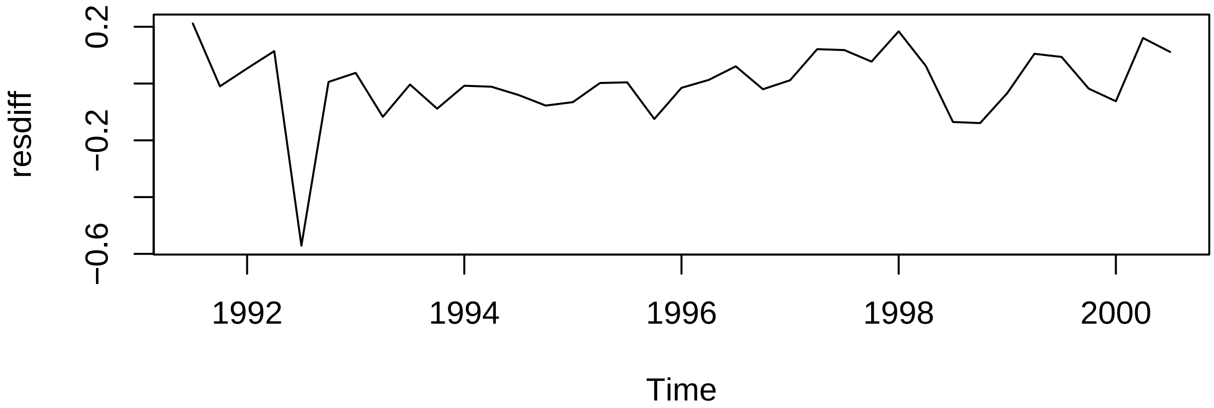


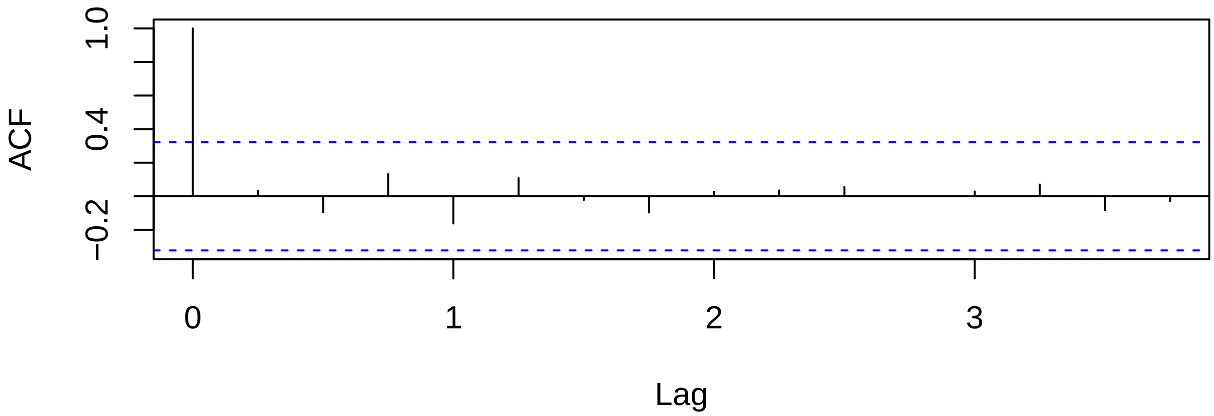
- This does not appear to really provide a better fit than the random walk model proposed earlier.
- An alternative would be to propose an AR(1) model for the differenced time series $\nabla X_t = X_t - X_{t-1}$:

```
dexchange <- diff(exchange)
fitdiff <- ar(dexchange, order.max = 1,method="mle")
fitdiff$ar
```

```
## [1] 0.3495624
```

```
resdiff <- na.omit(fitdiff$resid)
par(mfrow = c(2,1), mar = c(4,4,1,1))
plot(resdiff)
acf(resdiff)
```





11.7 Prediction/forecasting from AR(1) model

- We can use a fitted AR(1) model to predict the next value of an observed time series $x_1, \ldots, x_n$.

$$\hat{x}_{n+1} = \bar{x} + \hat{\alpha}_1 \cdot (x_n - \bar{x}).$$

- This can be iterated to predict x_{n+2}

$$\hat{x}_{n+2} = \bar{x} + \hat{\alpha}_1 \cdot (\hat{x}_{n+1} - \bar{x}),$$

  and we can continue this way.

- Prediction is performed by `predict` in R.
- E.g. for the AR(1) model fitted to the exchange rate data, the last observation is in third quarter of 2000. If we want to predict 1 year ahead to third quarter of 2001:

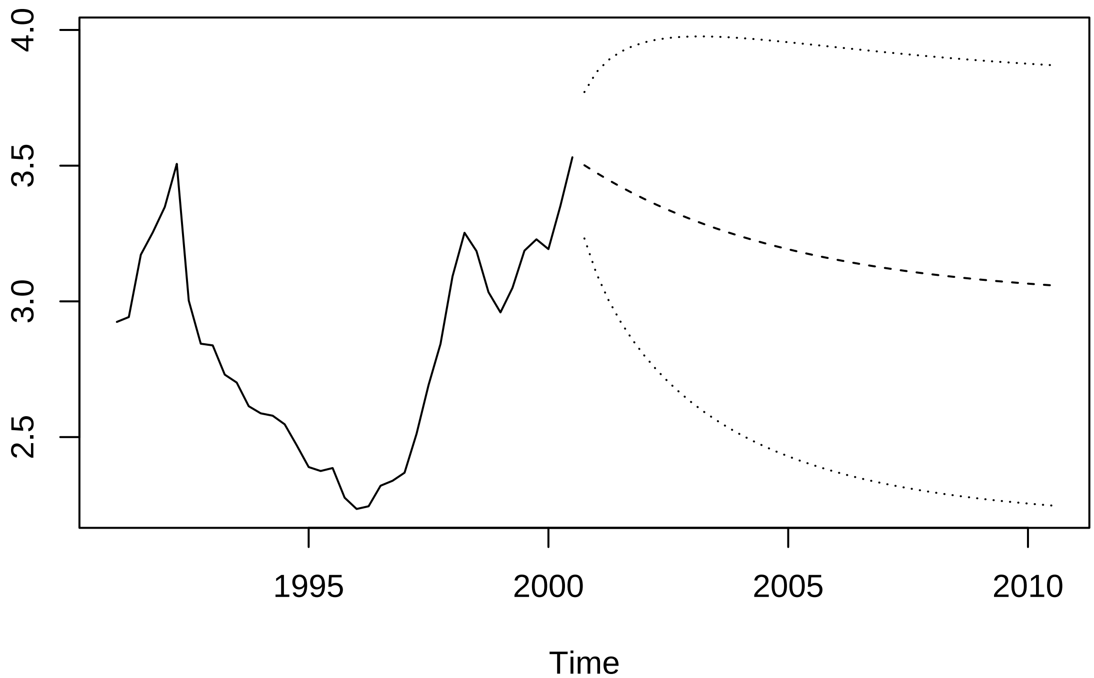
```
pred1 <- predict(fitexchange, n.ahead = 4)
pred1
```

```
## $pred
##          Qtr1     Qtr2     Qtr3     Qtr4
## 2000                            3.501528
## 2001 3.473716 3.447469 3.422699
##
## $se
##           Qtr1      Qtr2      Qtr3      Qtr4
## 2000                               0.1348262
```

```
## 2001 0.1853845 0.2208744 0.2482461
```

- Note how the prediction returns both the predicted value and a standard error for this value.
- We can use this to say that we are 95% confident that the exchange rate in third quarter of 2001 would be within $3.42 \pm 2 \cdot 0.25$.
- We can plot a prediction and approximate 95% pointwise prediction intervals with `ts.plot` (where we use a 10 year prediction – which is a very bad idea – to see how it behaves in the long run):

```
pred10 <- predict(fitexchange, n.ahead = 40)
lower10 <- pred10$pred-2*pred10$se
upper10 <- pred10$pred+2*pred10$se
ts.plot(exchange, pred10$pred, lower10, upper10, lty = c(1,2,3,3))
```



12 Auto-regressive models of higher order (AR(p)-models)

12.1 Auto-regressive models of higher order

- The first order auto-regressive model can be generalised to higher order by adding more lagged terms to explain the current value X_t.
- An **AR(p) process** is
$$X_t = \alpha_1 X_{t-1} + \alpha_2 X_{t-2} + \cdots + \alpha_p X_{t-p} + W_t$$
where W_t is white noise and $\alpha_1, \alpha_2, \ldots, \alpha_p$ are parameters to be estimated from data.
- To simplify the notation, we introduce the **backshift operator** B, that takes the time series one step back, i.e.
$$BX_t = X_{t-1}$$

- This can be used repeatedly, for example $B^2X_t = BBX_t = BX_{t-1} = X_{t-2}$. We can then make a "polynomial" of backshift operators

$$\alpha(B) = 1 - \alpha_1 B - \alpha_2 B^2 - \cdots - \alpha_p B^p$$

and write the AR(p) process as

$$W_t = \alpha(B)X_t.$$

12.2 Stationarity for AR(p) models

- Not all AR(p) models are stationary. To check that a given AR(p) model is stationary we consider the **characteristic equation**

$$1 - \alpha_1 z - \alpha_2 z^2 - \cdots - \alpha_p z^p = 0.$$

- The characteristic equation can be written with the polynomial α from previous slide, but with z inserted instead of B, i.e.

$$\alpha(z) = 0.$$

- The process is stationary if and only if all roots of the characteristic equation have an absolute value that is greater than 1.
- Solving a p-order polynomial is hard for high values of p, so we will let R do this for us.

12.3 Estimation of AR(p) models

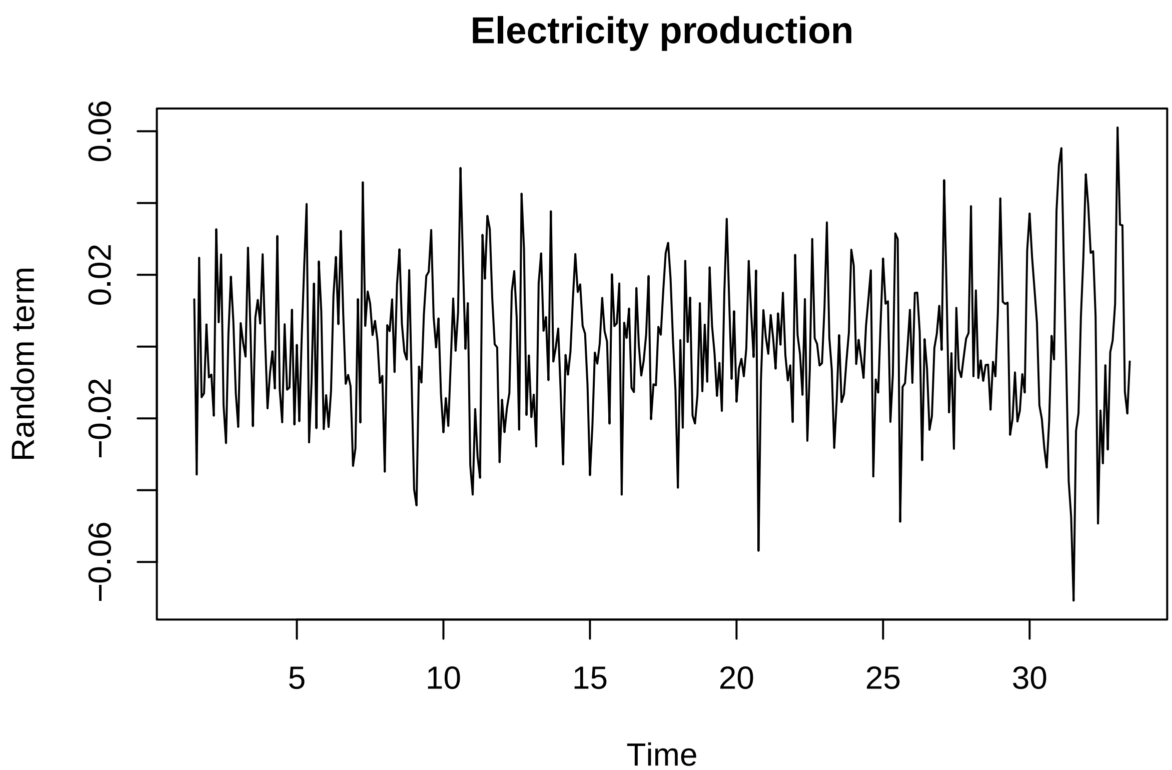
- For an AR(p) model there are typically two things we need to estimate:

1. The maximal non-zero lag p in the model.
2. The autoregressive coefficients/parameters $\alpha_1, \ldots, \alpha_p$.

- To select p we can use **AIC (Akaike's Information Criterion)**.
  - More complex models can always fit a dataset better
  - AIC is essentially a balance between model simplicity and good fit.
  - The AIC results in a single real number, where smaller is better.
  - The ar function in R uses AIC to automatically select the value for p by calculating the AIC for all models with p between 1 and some chosen maximal value, and picking the one with the smallest AIC.
- Once the order p is chosen, the estimates $\hat{\alpha}_1, \ldots, \hat{\alpha}_p$ can be computed by the ar function.
  - The corresponding standard errors can be found as the square root of the diagonal of the matrix stored as `asy.var.coef` in the fitted model object.

12.4 Example of AR(p) model for electricity data

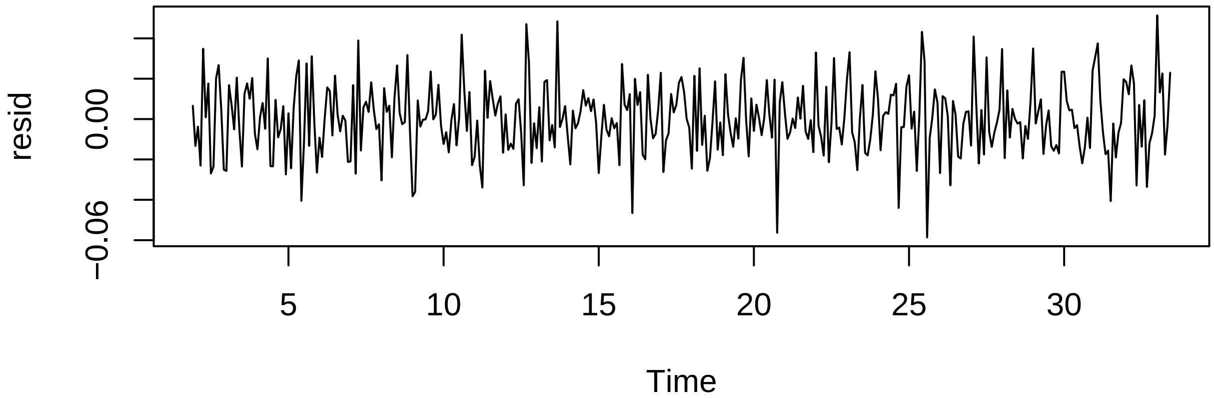
- We try to fit an AR(p) model to the detrended data for the log of the electricity production in Australia:

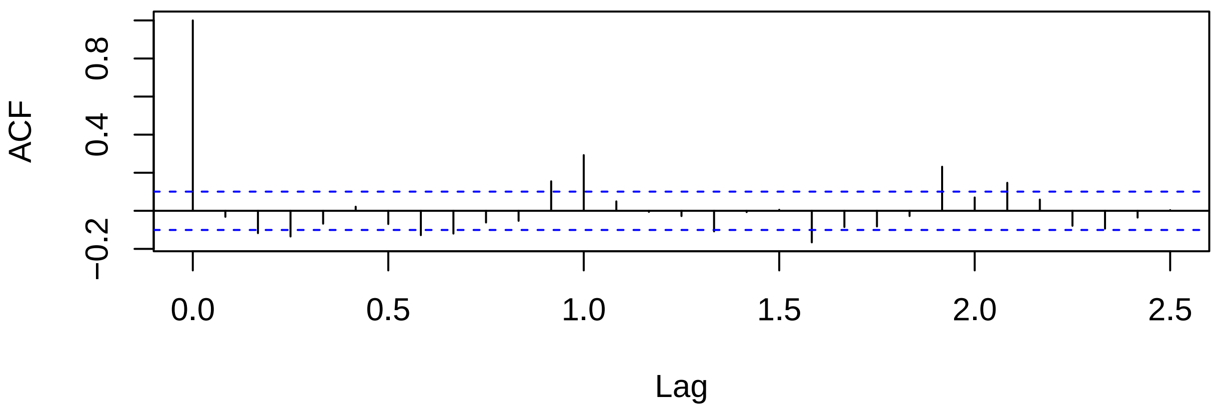



```
fit <- ar(random, order.max = 5,method="mle")
fit
```

```
##
## Call:
## ar(x = random, order.max = 5, method = "mle")
##
## Coefficients:
##       1        2        3        4        5
##  0.2047   0.0714  -0.0280  -0.1975  -0.2440
##
## Order selected 5  sigma^2 estimated as  0.0003103
```

```
resid <- na.omit(fit$resid)
par(mfrow = c(2,1), mar = c(4,4,1,1))
plot(resid)
acf(resid, lag.max = 30)
```





- There are too many significant autocorrelations. Suggests the model is not a good fit.
- We are not assured that the estimated model will be stationary. To check stationarity we solve $\alpha(z) = 0$:

```
abs(polyroot(c(1,-fit$ar)))
```

```
## [1] 1.148425 1.415987 1.415987 1.148425 1.549525
```

- All absolute values of the roots are greater than 1, indicating stationarity.

13 Moving average models (MA(q)-models)

13.1 The moving average model

- A **moving average process** of order q, MA(q), is defined by

$$X_t = W_t + \beta_1 W_{t-1} + \beta_2 W_{t-2} + \cdots + \beta_q W_{t-q}$$

  where W_t is a white noise process with mean 0 and variance σ_w^2 and $\beta_1, \beta_2, \ldots, \beta_q$ are parameters to be estimated.

- The moving average process can also be written using the backshift operator:

$$X_t = \beta(B) W_t$$

  where the polynomial $\beta(B)$ is given by

$$\beta(B) = 1 + \beta_1 B + \beta_2 B^2 + \cdots + \beta_q B^q.$$

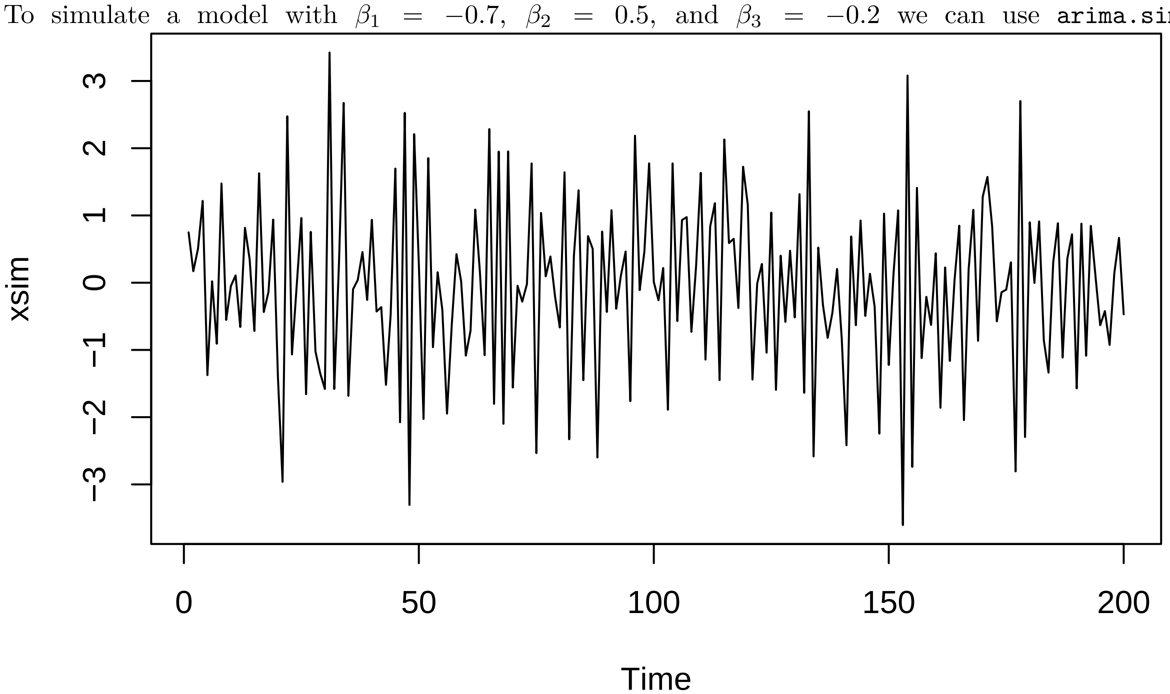
- Since a moving average process is a finite sum of stationary white noise terms it is itself stationary with mean and variance:
  - Mean $\mu(t) = 0$.
  - Variance $\sigma^2(t) = \sigma_w^2(1 + \beta_1^2 + \beta_2^2 + \cdots + \beta_q^2)$.
- The autocorrelation function is

$$\rho(k) = \begin{cases} 1 & k = 0 \\ \sum_{i=0}^{q-k} \beta_i \beta_{i+k} / \sum_{i=0}^{q} \beta_i^2 & k = 1, 2, \ldots, q \\ 0 & k > q \end{cases}$$

  where $\beta_0 = 1$.

13.2 Simulation of MA(q) processes

- An MA(q) process can be simulated by first generating a white noise process W_t and then transforming it using the MA coefficients.
- To simulate a model with $\beta_1 = -0.7$, $\beta_2 = 0.5$, and $\beta_3 = -0.2$ we can use `arima.sim`:



- The theoretical autocorrelations are in this case:

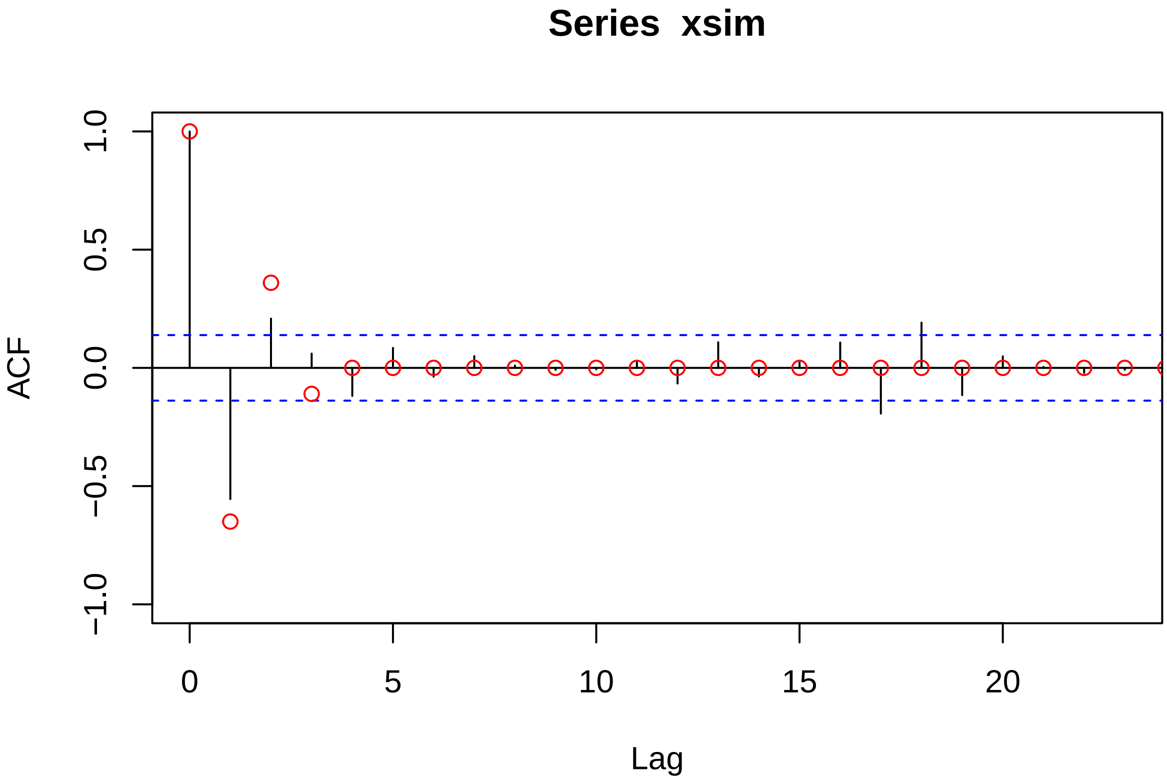
$$\rho(1) = \frac{1 \cdot (-0.7) + (-0.7) \cdot 0.5 + 0.5 \cdot (-0.2)}{1 + (-0.7)^2 + 0.5^2 + (-0.2)^2} = -0.65$$

$$\rho(2) = \frac{1 \cdot 0.5 + (-0.7) \cdot (-0.2)}{1 + (-0.7)^2 + 0.5^2 + (-0.2)^2} = 0.36$$

$$\rho(3) = \frac{1 \cdot (-0.2)}{1 + (-0.7)^2 + 0.5^2 + (-0.2)^2} = -0.11$$

  and $\rho(k) = 0$ for $k > 3$.
- We plot the acf of the simulated series together with the theoretical one.

```r
acf(xsim,ylim=c(-1,1))
points(0:25, c(1,-.65, .36, -.11, rep(0,22)), col = "red")
```



13.3 Estimation of MA(q) models

- To estimate the parameters of an MA(3) model we use `arima`:

```r
xfit <- arima(xsim, order = c(0,0,3))
xfit
```

```
##
## Call:
## arima(x = xsim, order = c(0, 0, 3))
##
## Coefficients:
##           ma1     ma2      ma3  intercept
##       -0.6258  0.3876  -0.0868    -0.0604
## s.e.   0.0719  0.0886   0.0706     0.0507
##
## sigma^2 estimated as 1.124:  log likelihood = -295.76,  aic = 601.52
```

- The function `arima` does not include automatic selection of the order of the model so this has to be chosen beforehand or selected by comparing several proposed models and choosing the model with the minimal AIC.

```r
AIC(xfit)
```

```
## [1] 601.5189
```

14 Auto-regressive moving average models (ARMA)

14.1 Mixed models: Auto-regressive moving average models

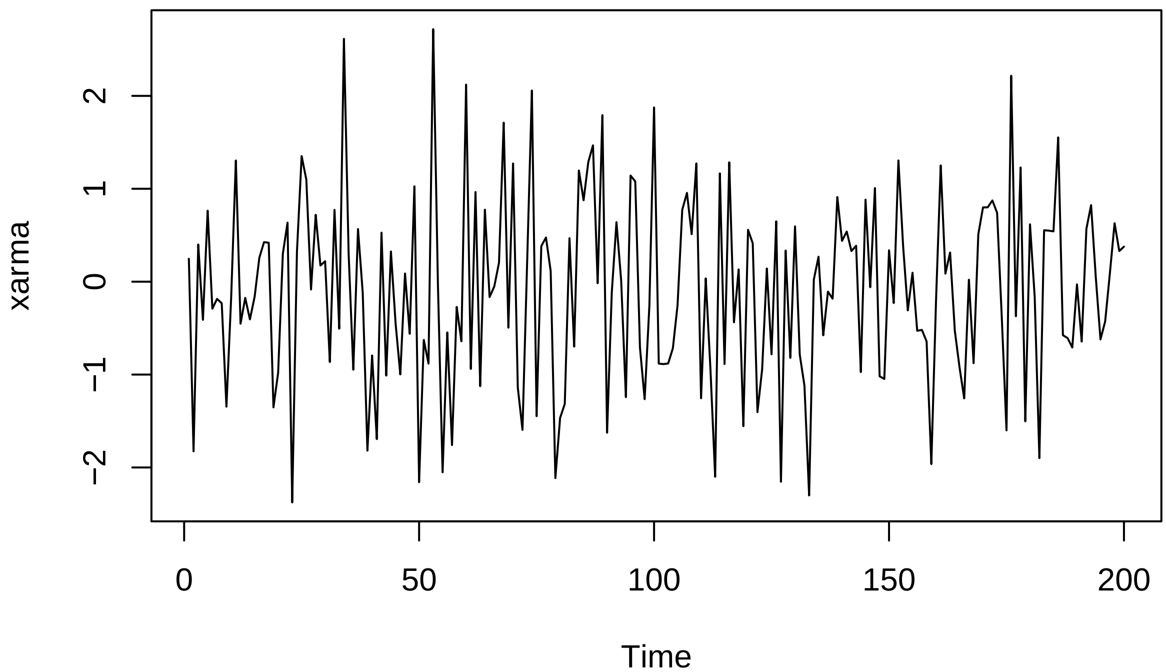
- A time series X_t follows an auto-regressive moving average (ARMA) process of order (p, q), denoted $ARMA(p, q)$, if

$$X_t = \alpha_1 X_{t-1} + \alpha_2 X_{t-2} + \cdots + \alpha_p X_{t-p} + W_t + \beta_1 W_{t-1} + \beta_2 W_{t-2} + \cdots + \beta_q W_{t-q}$$

  where W_t is a white noise process and $\alpha_1, \alpha_2, \ldots, \alpha_p, \beta_1, \beta_2, \ldots, \beta_q$ are parameters to be estimated.

- We can simulate an ARMA model with `arima.sim`. E.g. an ARMA(1,1) model with $\alpha_1 = -0.6$ and $\beta_1 = 0.5$:

```
xarma <- arima.sim(model = list(ar = -0.6, ma = 0.5), n = 200)
plot(xarma)
```



- Estimation is done with `arima` as before.

14.2 Example with exchange rate data

For the exchange rate data we may e.g. suggest either a AR(1), MA(1) or ARMA(1,1) model. We can compare fitted models using `AIC` (smaller is better):

```
exchange_ar <- arima(exchange, order = c(1,0,0))
AIC(exchange_ar)
```

```
## [1] -37.40417
```

```
exchange_ma <- arima(exchange, order = c(0,0,1))
AIC(exchange_ma)
```

```
## [1] -3.526895
```

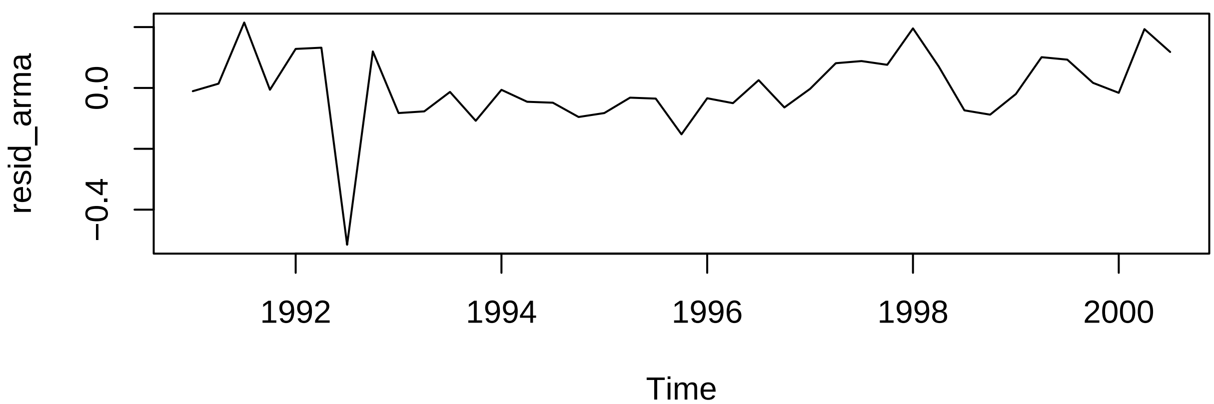
```r
exchange_arma <- arima(exchange, order = c(1,0,1))
AIC(exchange_arma)
```

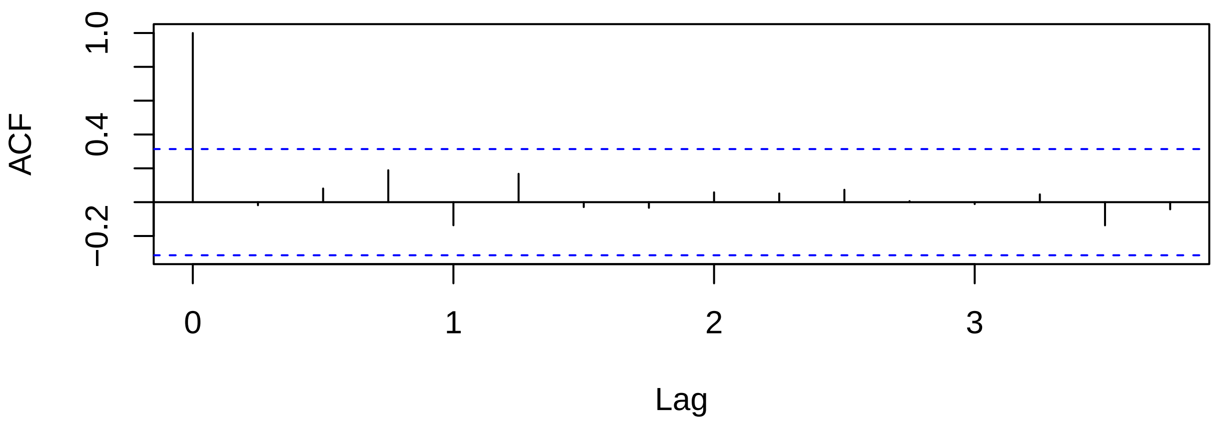
```
## [1] -42.27357
```

```r
exchange_arma
```

```
##
## Call:
## arima(x = exchange, order = c(1, 0, 1))
##
## Coefficients:
##          ar1     ma1  intercept
##       0.8925  0.5319     2.9597
## s.e.  0.0759  0.2021     0.2435
##
## sigma^2 estimated as 0.01505:  log likelihood = 25.14,  aic = -42.27
```

```r
par(mfrow = c(2,1), mar = c(4,4,1,1))
resid_arma <- na.omit(exchange_arma$residuals)
plot(resid_arma)
acf(resid_arma)
```





15 Models with exogenous variables

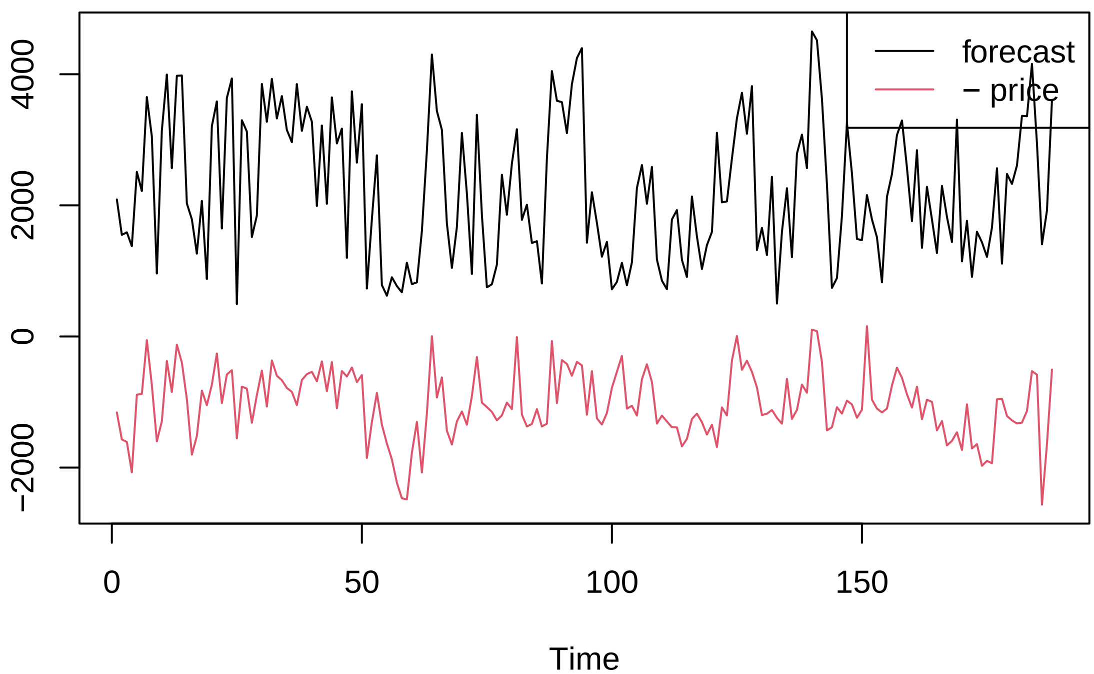
15.1 Exogenous variables

- The ARMA processes are flexible models for a time series Y_t, $t = 1, \dots, n$ evolving randomly over time, but they do not include the possibility that anything is influencing Y_t.
- An exogeneous variable is another variable, say X_t, that influences the behaviour of Y_t
  - Wind power production Y_t is influenced by the wind speed X_t
  - The velocity of a DC motor Y_t is influenced by the input voltage X_t
- Here X_t may be another stochastic process, which we do not model, but only consider as given, or it might be something we can control.

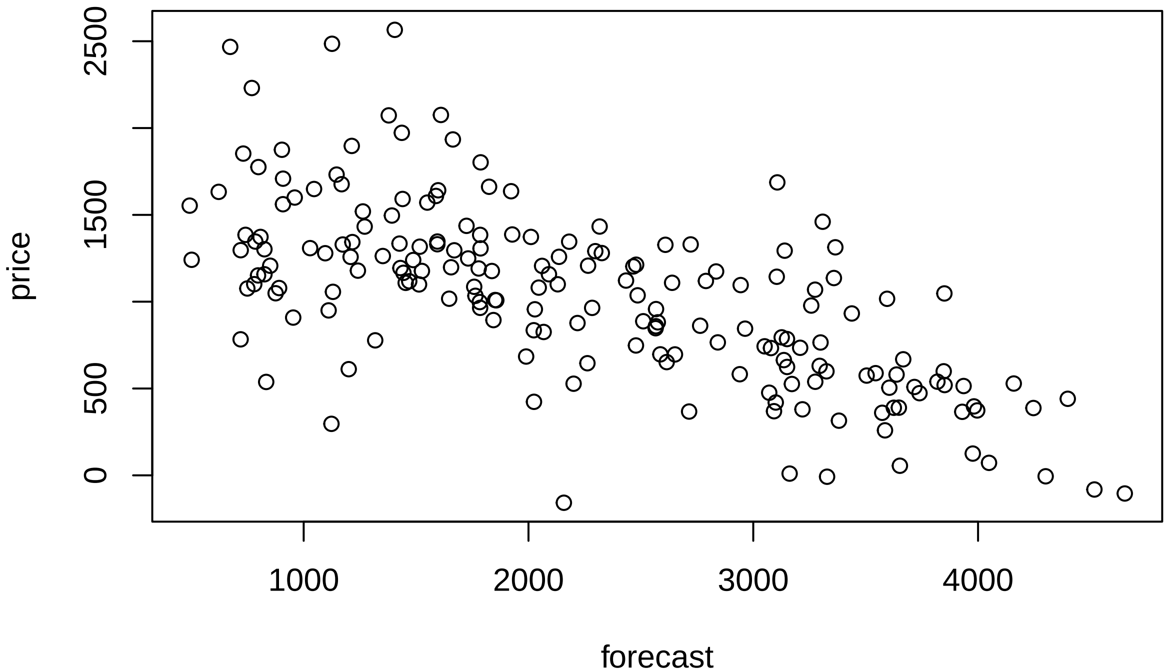
15.2 Data example

- The dataset below contains data from Jan 7 to Jul 13 2022 on two variables
  - forecast: Total day ahead forecasted wind and solar energy production
  - price: Day ahead elspot prices with weekly variation removed

```
elspot<-read.csv("https://asta.math.aau.dk/eng/static/datasets?file=elspot.csv", header = TRUE)
forecast<-elspot[,2]
price<-elspot[,3]
ts.plot(ts(forecast),ts(-price),col=1:2)
legend("topright",legend=c("forecast","- price"),col=1:2,lty=1)
```




```
plot(forecast,price)
```



15.3 Regression models with exogenous variables

- We can combine regression models with ARMA models to obtain a stochastic process which is influenced by exogenous variables.
- Consider a linear regression of Y_t on X_t, but where the noise term is an ARMA process:

$$Y_t = \gamma_0 + \gamma_1 X_t + \epsilon_t, \qquad \alpha(B)\epsilon_t = \beta(B)W_t$$

- If we isolate $\epsilon_t = Y_t - \gamma_0 - \gamma_1 X_t$ and insert into the ARMA expression, we get something that looks more like an ARMA process, but with Y_t adjusted by the exogenous variable:

$$\alpha(B)(Y_t - \gamma_0 - \gamma_1 X_t) = \beta(B)W_t$$

- The purpose of fitting such a model is both to obtain a good model for the evolution of the data and to obtain an understanding of the relation between Y_t and X_t.
- Above, X_t is a single stochastic process, but we can also include multiple stochastic processes by making a multiple regression model with an ARMA model for the errors.

15.4 Example

- As an example consider a simple linear regression combined with an AR(1) process for noise terms:

$$Y_t = \gamma_0 + \gamma_1 X_t + \epsilon_t, \qquad \epsilon_t = \alpha_1 \epsilon_{t-1} + W_t$$

  or, since $\epsilon_{t-1} = Y_{t-1} - \gamma_0 - \gamma_1 X_{t-1}$,

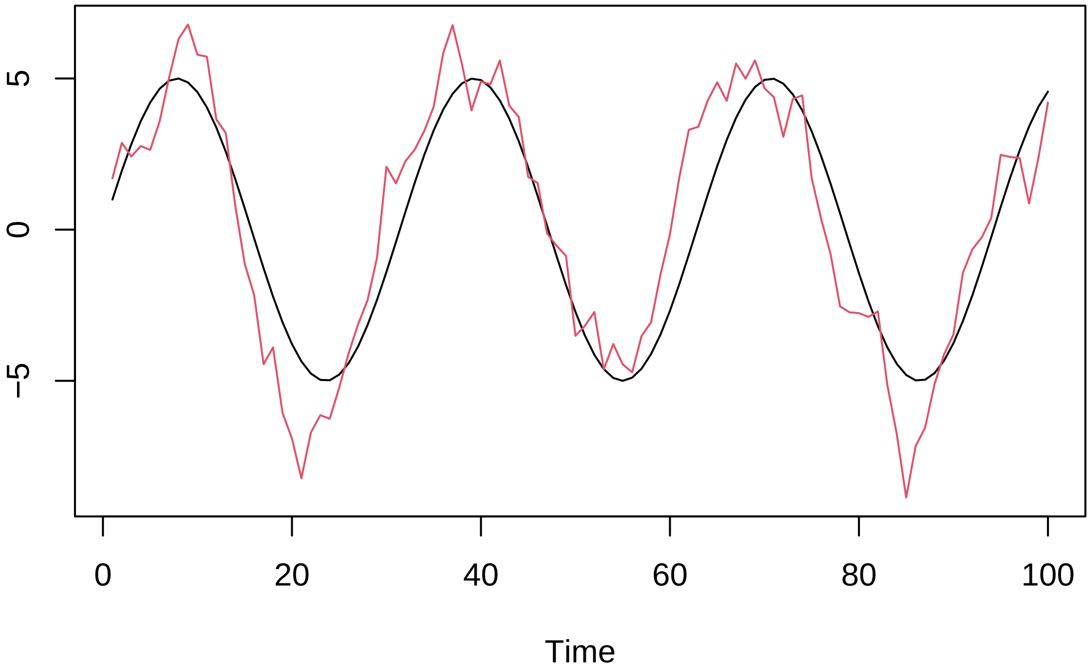
$$Y_t = \alpha_1 Y_{t-1} + (1 - \alpha_1)\gamma_0 + \gamma_1(X_t - \alpha_1 X_{t-1}) + W_t$$

- Notice that the model behaves like an AR(1) process, but instead of having a constant mean of 0, its mean is constantly adjusted by the exogenous variable.

15.5 Simulation of the example

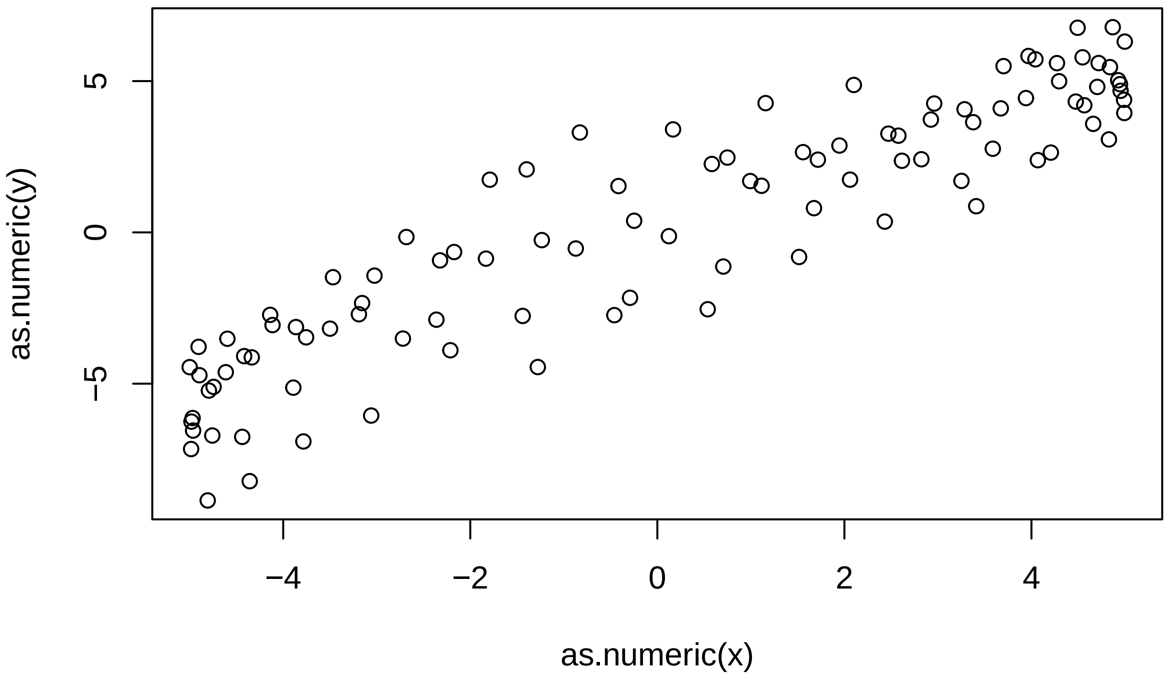
- We simulate some data resembling the example, where we let X_t follow a sine curve:

```
alpha = 0.9; gamma = 1; n = 100
x = as.ts(5*sin(1:n/5))
eps = arima.sim(model=list(ar=alpha),n=n)
y = gamma*x+eps
ts.plot(x,y,col=1:2)
```



- We should think of the red curve as some data we want to model, and the black curve as another variable which we believe may influence the data.
- We can also plot X_t against Y_t to get a view of the relation between the two variables.

```
plot(as.numeric(x),as.numeric(y))
```



15.6 Estimation and model checking

- We can estimate the parameters using the arima function in R.
- We fit a linear regression model with AR(1) noise to the simulated data (i.e. the true model used for simulation):

```
mod=arima(y,order=c(1,0,0),xreg=x); mod
```

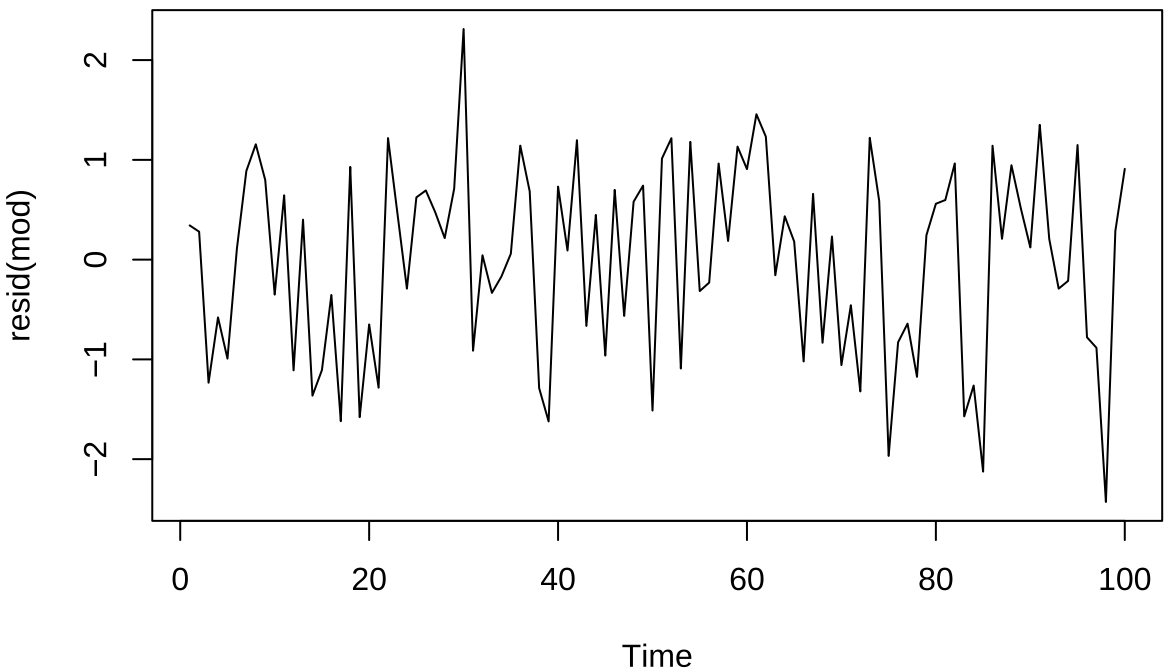
```
##
## Call:
## arima(x = y, order = c(1, 0, 0), xreg = x)
##
## Coefficients:
##          ar1  intercept       x
##       0.8069     0.0679  1.0551
## s.e.  0.0569     0.4795  0.1018
##
## sigma^2 estimated as 0.923:  log likelihood = -138.41,  aic = 284.82
```

- The fitted model becomes

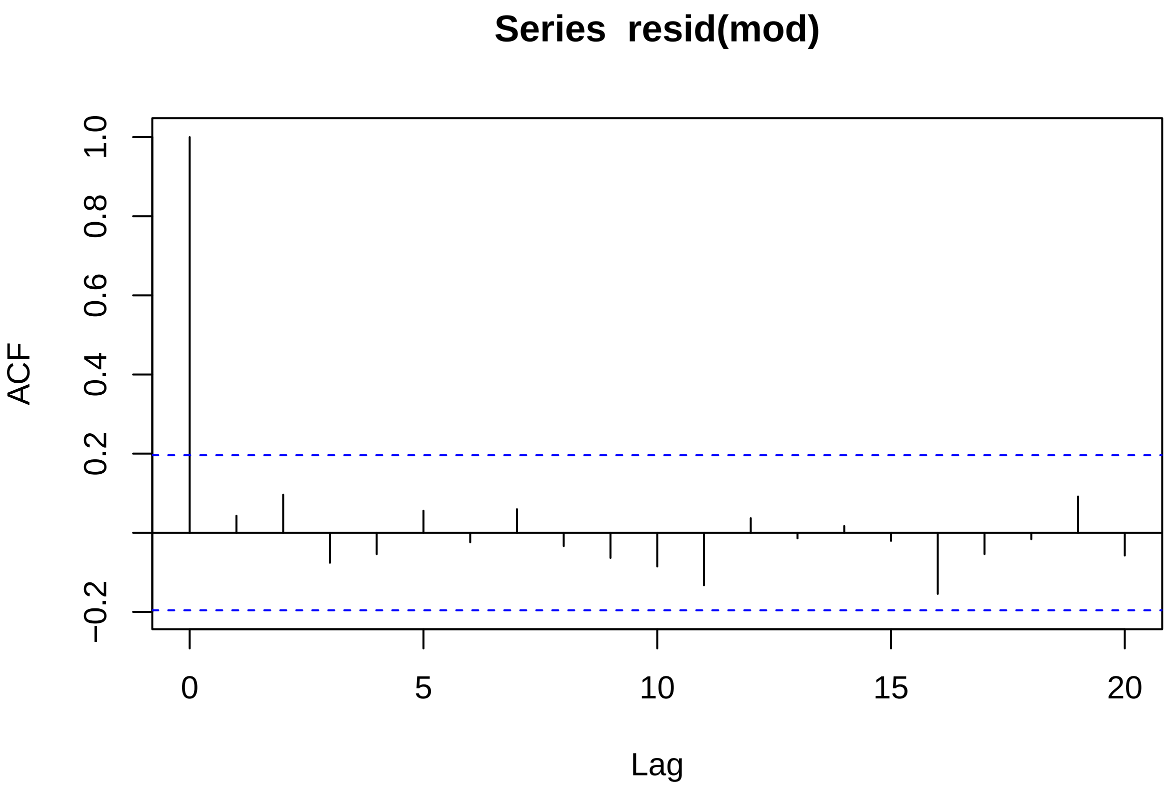
$$Y_t = 0.0679 + 1.0551 \cdot X_t + \epsilon_t, \qquad \epsilon_t = 0.8069 \cdot \epsilon_{t-1} + W_t$$

- The errors $\hat{\epsilon}_t = y_t - 0.0679 + 1.0551 \cdot x_t$ should behave like an AR(1)-model with $\hat{\alpha} = 0.8069$.
  - So the residuals $\epsilon_t - 0.8069 \cdot \epsilon_{t-1}$ should look like white noise.

```r
plot(resid(mod))
```




```r
acf(resid(mod))
```



15.7 Fitting AR(1) model to data example

- Recall the elspot price dataset

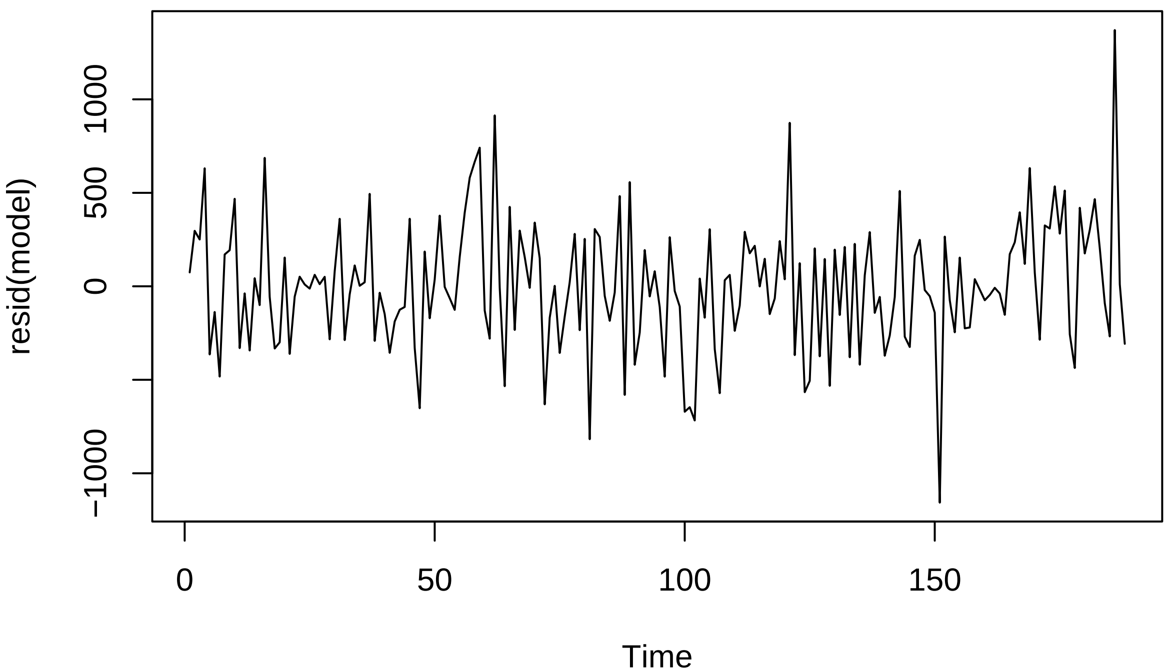
```
forecast<- ts(forecast)
price<-ts(price)
model=arima(price,order=c(1,0,0),xreg=forecast); model
```

```
##
## Call:
## arima(x = price, order = c(1, 0, 0), xreg = forecast)
##
## Coefficients:
##          ar1  intercept  forecast
##       0.3886  1715.8412   -0.3053
## s.e.  0.0680    73.2894    0.0271
##
## sigma^2 estimated as 117486:  log likelihood = -1364.2,  aic = 2736.41
```

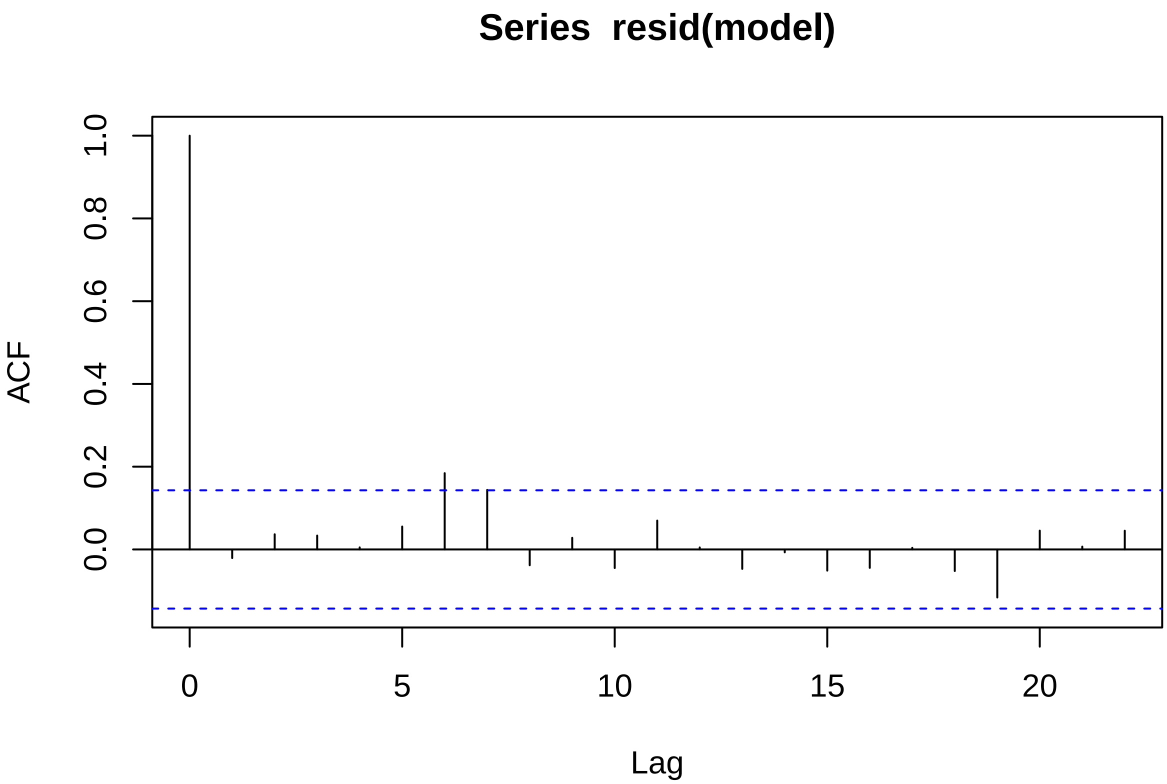
- So we get the model

$$\text{price}_t = 1715.8412 - 0.3053 \cdot \text{forecast}_t + \epsilon_t, \qquad \epsilon_t = 0.3886 \cdot \epsilon_{t-1} + W_t.$$

```
plot(resid(model))
```




```
acf(resid(model))
```

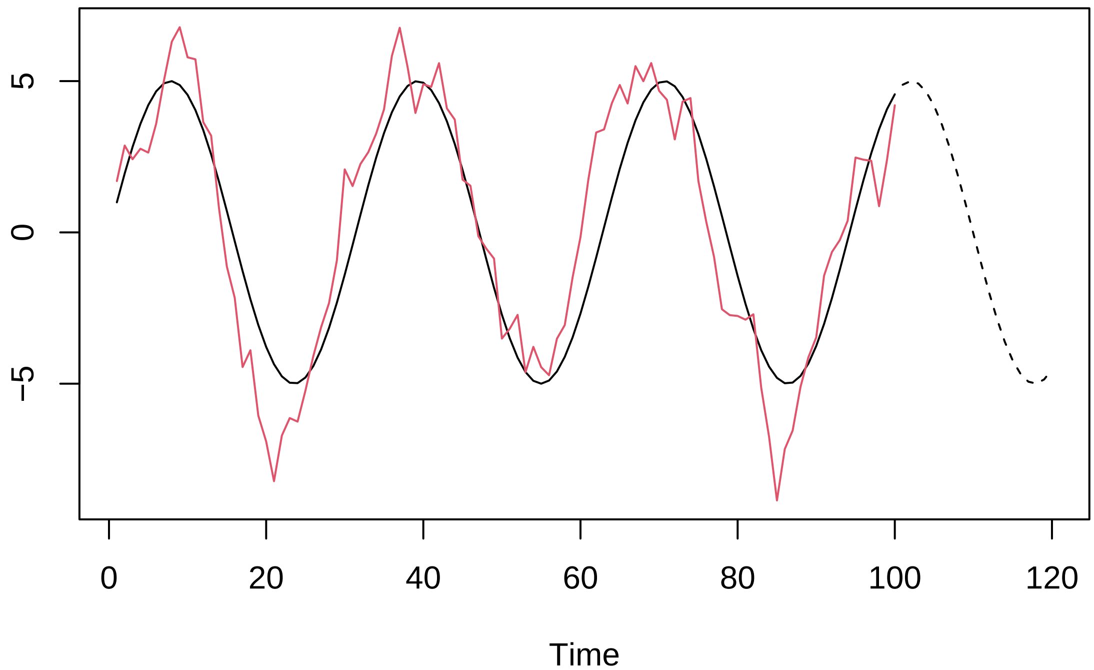


- Residuals indicate that there could be some weekly variation not accounted for.

15.8 Prediction

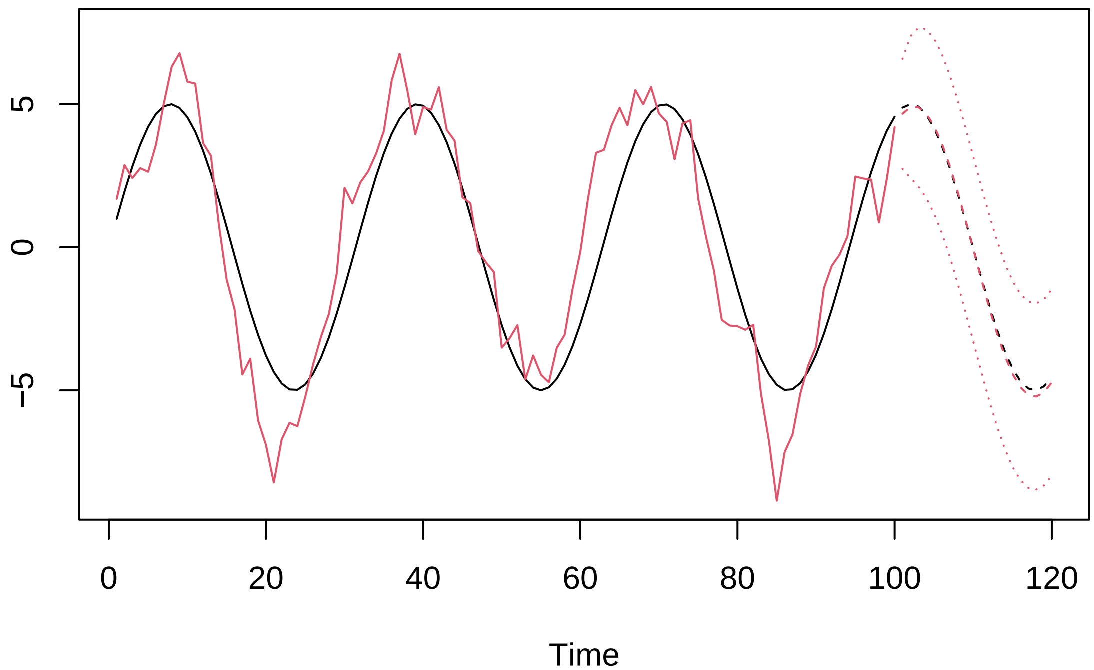
- Prediction can only be performed if we know the behavior of X_t for future time points, for example if we are able to control it.
- For the previous example we assume that the sine curve continues:

```
nnew = 20
xnew = lag(as.ts(5*sin(((n+1):(n+nnew))/5)),-n)
ts.plot(x,y,xnew,col=c(1,2,1),lty=c(1,1,2))
```



- We use the predict function.

```
p = predict(mod,n.ahead=nnew,newxreg=xnew)
ts.plot(x,y,xnew,p$pred,p$pred+2*p$se,p$pred-2*p$se,col=c(1,2,1,2,2,2),lty=c(1,1,2,2,3,3))
```

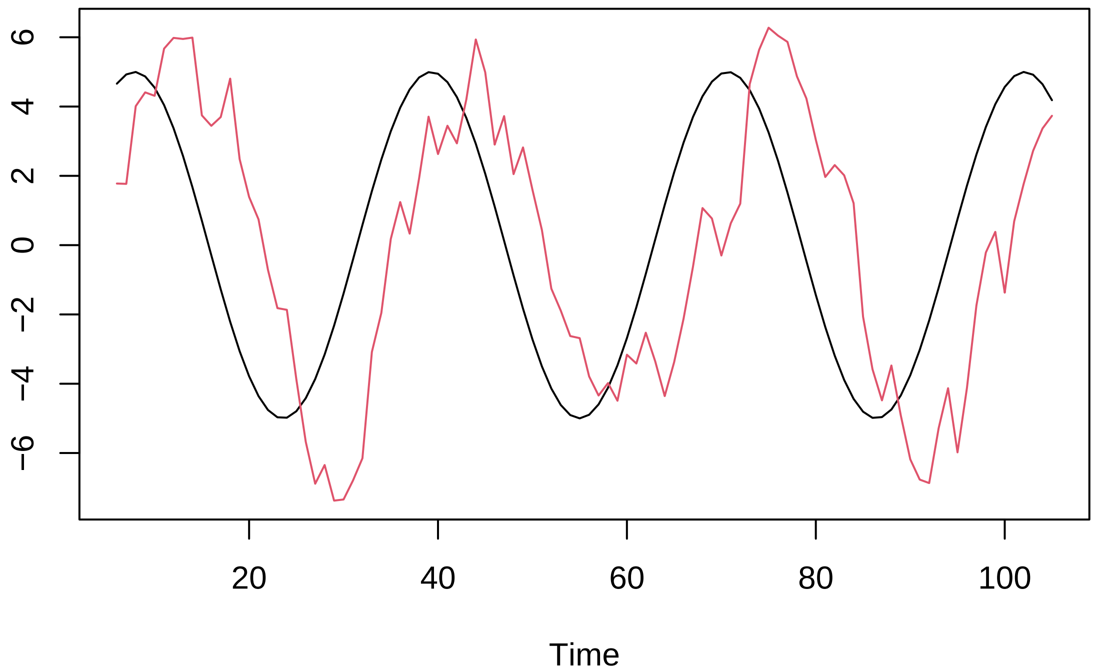


15.9 An example with delay

- If we model the influence of X_t on Y_t, it may take some time before Y_t responds to a change in X_t.
  - Say the delay is k time steps.

- We want to model the effect of X_{t-k} on Y_t.
  - We may not know the delay k, so we may need to estimate it first.
- We simulate a dataset with a built-in delay, and then we model this afterwards.

```
alpha = 0.5; gamma = 1; n = 100; delay = 5
x = as.ts(5*sin(1:(n+delay)/5))
eps = arima.sim(model=list(ar=alpha),n=n+delay)
y = gamma*lag(x,-delay)+eps
dat_lag = ts.intersect(x,y)
ts.plot(dat_lag[,1],dat_lag[,2],col=1:2)
```



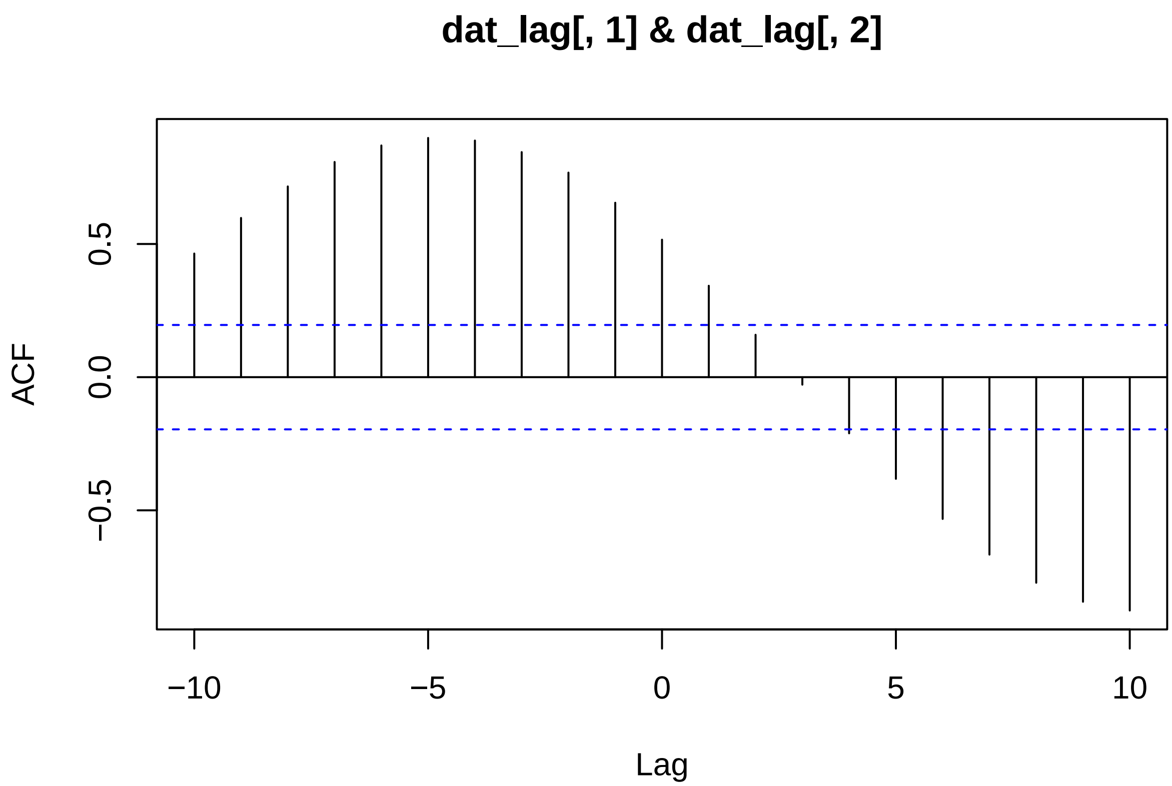
15.10 The cross-correlation function

- The **cross correlation function** is used for checking the relation between two time series at different time points:

$$\rho_{xy}(t+k,t) = \text{Cor}(X_{t+k}, Y_t).$$

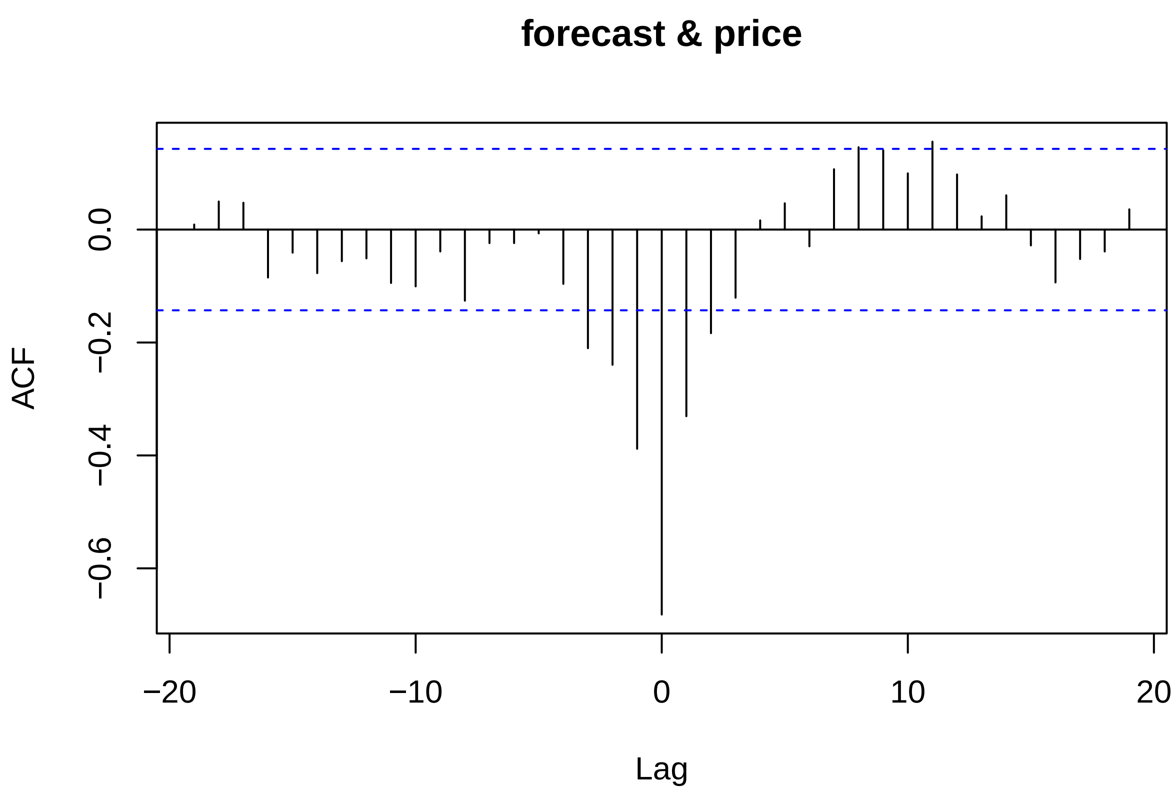
- Values that are close to 1 or -1 indicate that the two time series are closely related if X_t is delayed by k time steps.
- Cross-correlation function for the simulated data

```
cc = ccf(dat_lag[,1],dat_lag[,2],lag.max=10)
```



- Cross-correlation function for the elspot data:

```
ccf(forecast,price)
```



15.11 Fitting models with lag

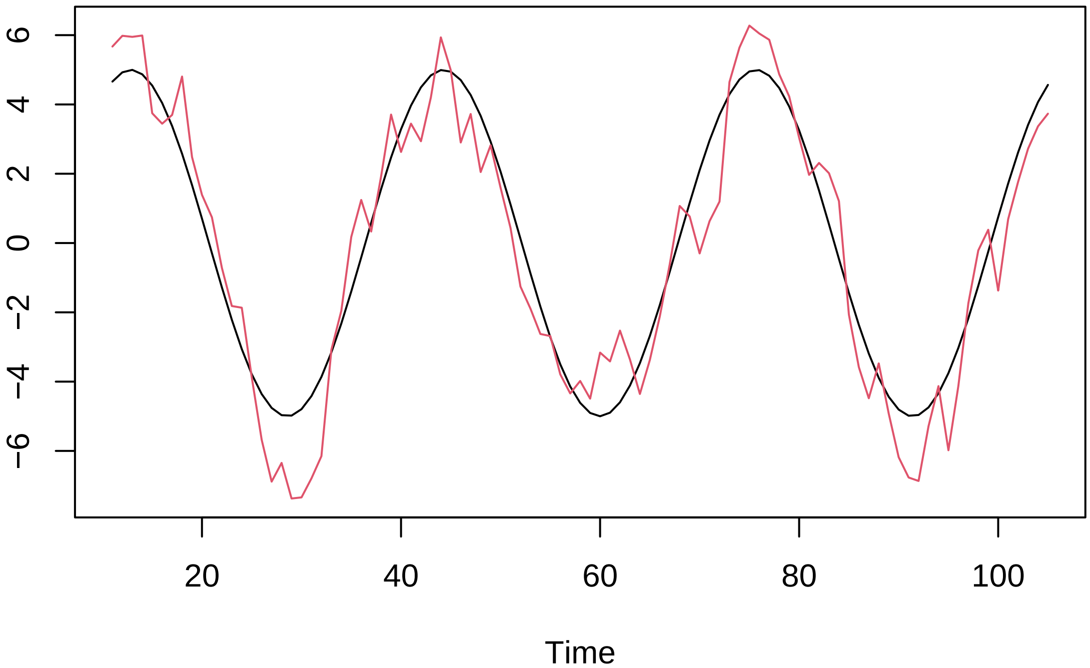
- We estimate the lag to be the one where the cross-correlation function is maximal:

```
estlag = cc$lag[which(cc$acf==max(abs(cc$acf)))]
estlag
```

```
## [1] -5
```

- Plotting the data with this lags can be useful to check the choice:

```
dat_shifted = ts.intersect(lag(as.ts(dat_lag[,1]),estlag),dat_lag[,2] )
ts.plot(dat_shifted[,1],dat_shifted[,2],col=1:2)
```



- We can now fit a model with this lag:

```
mod=arima(dat_shifted[,2],order=c(1,0,0),xreg=dat_shifted[,1]); mod
```

```
##
## Call:
## arima(x = dat_shifted[, 2], order = c(1, 0, 0), xreg = dat_shifted[, 1])
##
## Coefficients:
##          ar1  intercept  dat_shifted[, 1]
##       0.5938    -0.2047            1.0526
## s.e.  0.0820     0.2347            0.0615
##
## sigma^2 estimated as 0.8884:  log likelihood = -129.39,  aic = 266.79
```

15.12 ARMAX models

- An alternative way of including exogenous variables into an ARMA model is an ARMAX model.
- The $ARMAX(p, q, b)$ model is an $ARMA(p, q)$ model including b terms of an exogenous variable, i.e. it

is defined by

$$Y_t = \sum_{i=1}^{p} \alpha_i Y_{t-i} + \sum_{i=1}^{b} \gamma_i X_{t-i} + W_t + \sum_{i=1}^{q} \beta_i W_{t-i}$$

- Using the backshift operator, this can be written as

$$\alpha(B)Y_t = \gamma(B)X_t + \beta(B)W_t$$

with $\alpha(B) = 1 - \sum_{i=1}^{p} \alpha_i B^i$, $\beta(B) = 1 + \sum_{i=1}^{q} \beta_i B^i$, and $\gamma(B) = \sum_{i=1}^{b} \gamma_i B^i$.
- Compare with the regression with ARMA noise:

$$\alpha(B)(Y_t - \gamma_0 - \gamma_1 X_t) = \beta(B)W_t \quad \Rightarrow \quad \alpha(B)Y_t = \alpha(B)(\gamma_0 + \gamma_1 \gamma X_t) + \beta(B)W_t$$

- The difference is only how the model includes the exogenous variable.
- It is mostly a matter of taste which kind of model you should choose.
- Only the regression with ARMA noise is included into R as standard.

16 Continuous time processes

16.1 Discrete vs. continuous time

There are two fundamentally different model classes for time series data.

- Discrete time stochastic processes
  - Variables given at equally spaced time points
- Continuous time stochastic processes
  - Variables that evolve over a continuous time scale

So far we have only looked at the discrete time case. We will finish todays lecture by looking a bit at the continuous time case, just to give you an idea of this topic.

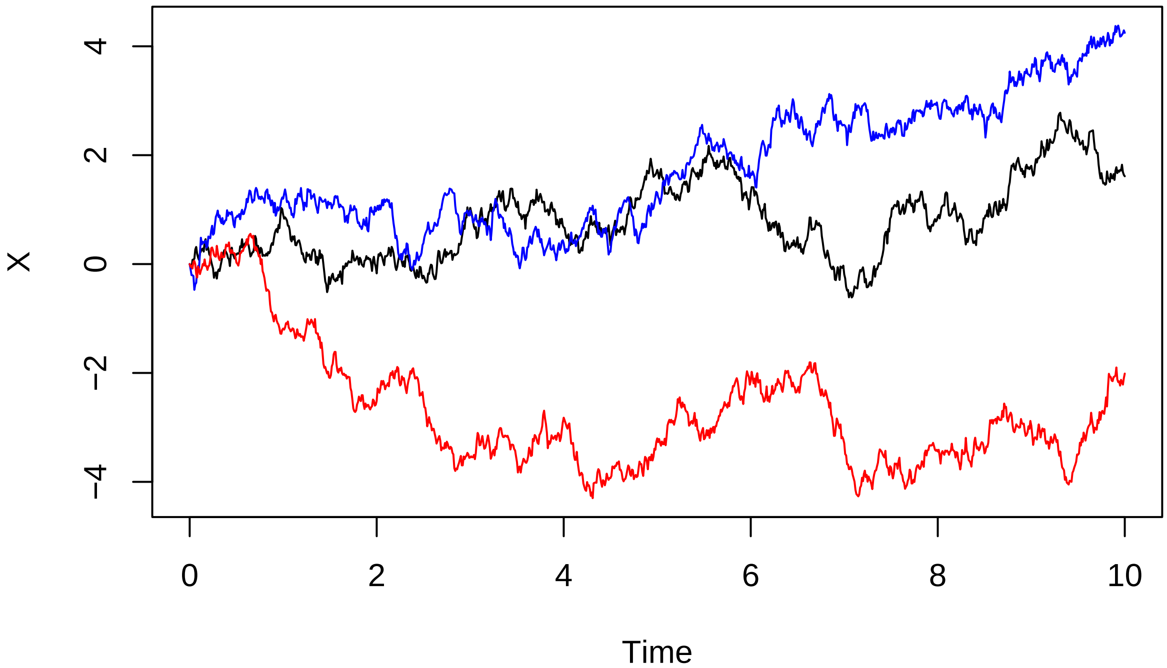
16.2 Continuous time stochastic processes

- In this setup we see the underlying X_t as a continuous function of t for t in some interval $[0, T]$.
- In principle we imagine that there are infinitely many data points, simply because there are infinitely many time points between 0 and T.
- In practice we will always only have finitely many data points.
- But we can imagine that the real data actually contains all the data points. We are just not able to measure them (and to store them in a computer).
- With a model for all datapoints, we are - through simulation - able to describe the behaviour of data. Also between the observations.

16.3 The Wiener process

- A key example of a process in continuous time will be the so–called **Wiener process** or **Brownian motion**.
- Here are three simulated realizations (black, blue and red) of this process: here

```
## Package 'Sim.DiffProc', version 4.9
## browseVignettes('Sim.DiffProc') for more informations.
```



- A Wiener process has the following properties:
  - It starts in 0: $B_0 = 0$.
  - It has independent increments: For $0 < s < t$ it holds that $B_t - B_s$ is independent of everything that has happened up to time s, that is B_u for all $u \leq s$.
  - It has normally distributed increments: For $0 < s < t$ it holds that the increment $B_t - B_s$ is normally distributed with mean zero and variance $t - s$:

$$B_t - B_s \sim \texttt{norm}(0, t - s).$$

- The intuition of this process is that it somehow changes direction all the time: How the process changes after time s will be independent of what has happened before time s. So whether the process should increase or decrease after s will not be affected by how much it was increasing or decreasing before. This gives the very bumpy behaviour over time.

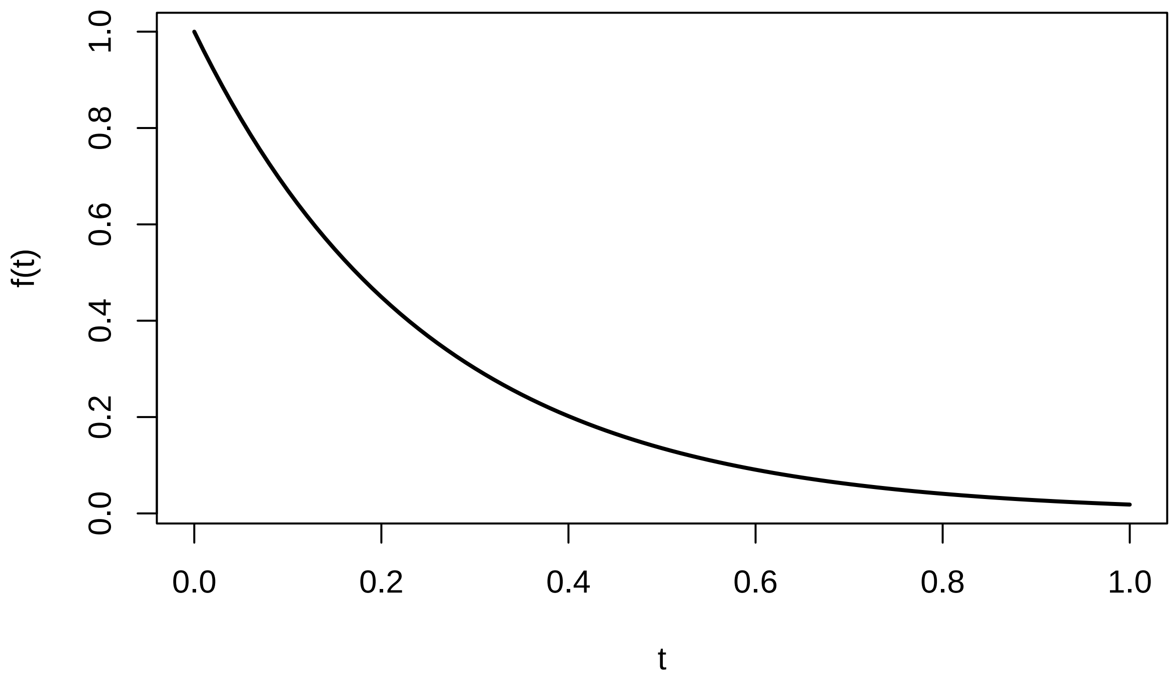
16.4 Stochastic differential equations

- A common way to define a continuous time stochastic process model is through a stochastic differential equation (SDE) which we will turn to shortly, but before doing so we will recall some basic things about ordinary differential equations.
- **Example:** Suppose f is an unknown differentiable function satisfying the differential equation

$$\frac{df(t)}{dt} = -4f(t)$$

with initial condition $f(0) = 1$. This equation has the solution

$$f(t) = \exp(-4t)$$



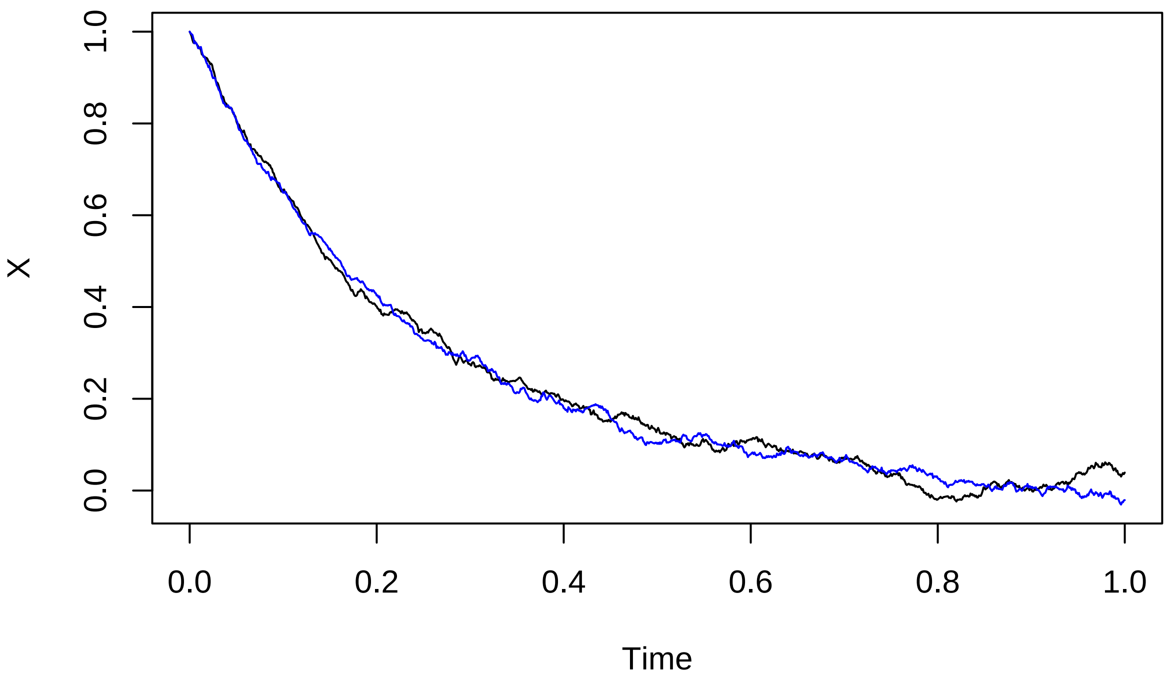
- With a slightly unusual notation we can rewrite this as

$$df(t) = -4 \cdot f(t)dt$$

- This equation has the following (hopefully intuitive) interpretation:
  - When time increases by a small amount dt (from t to $t + dt$) the value of f changes (approximately) by $-4f(t) \cdot dt$.
- So when t is increased, then f is decreased, and the decrease is proportional to the value of $f(t)$. That is why f decreases slower and slower, when t is increased.
- We say that the function has a **drift** towards zero, and this drift is determined by the value of the function.

16.5 Stochastic differential equations

- It will probably never be true that data behaves exactly like the exponentially decreasing curve on the previous slide.
- Instead we will consider a model, where some random noise from a Wiener process has been added to the growth rate. Two different (black/blue) simulated realizations can be seen below



- The type of process that is simulated above is described formally by the equation

$$dX_t = -4X_t dt + 0.1dB_t$$

- This is called a **Stochastic Differential Equation** (SDE), and the processes simulated above are called solutions of the stochastic differential equation.
- The SDE $dX_t = -4X_t dt + 0.1dB_t$ has two terms:
  - $-4X_t dt$ is the **drift term**.
  - $0.1dB_t$ is the **diffusion term**.
- The intuition behind this notation is very similar to the intuition in the equation $df(t) = -4 \cdot f(t)\, dt$ for an ordinary differential equation. When the time is increased by the small amount dt, then the process X_t is increased by $-4X_t\, dt$ AND by how much the process $0.1B_t$ has increased on the time interval $[t, t + dt]$.
- So this process has a **drift** towards zero, but it is also pushed in a random direction (either up or down) by the Wiener process (more precisely, the process $0.1B_t$)