Stochastic processes II

The ASTA team

Contents

1 Basic stochastic models

1.1 White noise

Suppose y_t is a time series and we have a model which gives us predictions $\hat{y}_t$. Then the residuals are $x_t = y_t - \hat{y}_t$ for each time point $t = 1, \ldots, n$. If the model has correctly captured the trend, seasonality and any autocorrelation in the original process y_t then the correlogram (acf) of the residual process x_t should show no obvious patterns. Such a residual process with no autocorrelation or other structure is called *white noise*.

1.1.1 Definition and properties of *white noise*

A time series w_t, $t = 1, \ldots, n$ is *white noise* if the variables $w_1, w_2, \ldots, w_n$ are *independent* and *identically* distributed with a mean of zero.

From the definition it follows that white noise is a second order stationary process since the variance function $\sigma^2(t) = \sigma^2$ is the same constant for all t and the autocovariance is $Cov(w_t, w_{t+k}) = 0$ for all $k \neq 0$ which does not depend on t. We summaries this as:

$$\mu = 0$$

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

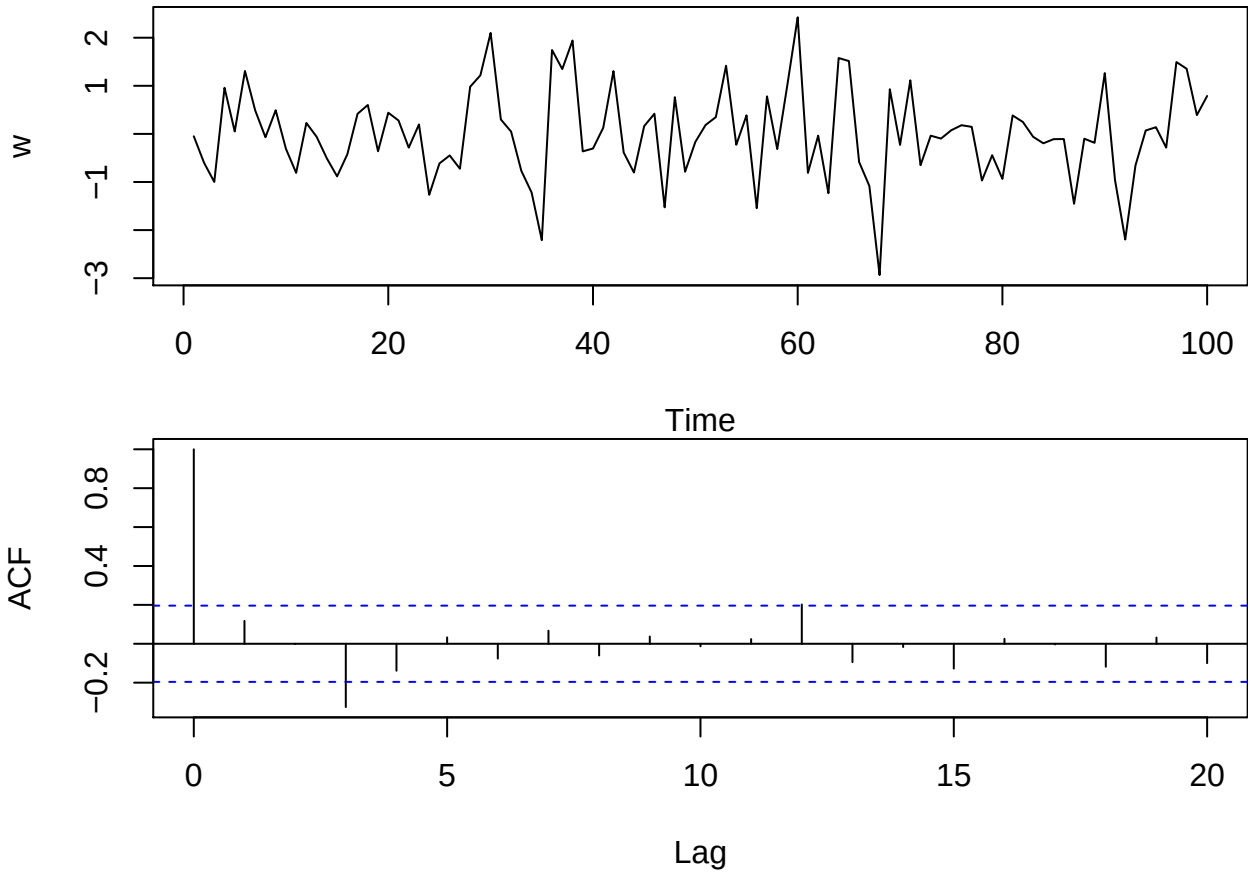
$$\rho(k) = \begin{cases} 1 \text{ for } k = 0 \\ 0 \text{ for } k \neq 0 \end{cases}$$

Often we will also assume the distribution of each w_t is Gaussian (i.e. $w_t \sim N(0, \sigma)$) and then we call it *Gaussian white noise*.

1.1.2 Simulation of 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)
```



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

1.2 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 we would get an infinite sum of white noise

$$x_t = w_t + w_{t-1} + w_{t-2} + w_{t-3} + \dots$$

However, we will assume we have a fixed starting point $x_0 = 0$ such that

$$x_t = w_1 + w_2 + \cdots + w_t$$

1.2.1 Properties of random walk

A random walk x_t has a constant mean function

$$\mu(t) = 0$$

since the random walk at time t is a sum of t white noise terms that all have mean zero.

However, the variance function

$$\sigma^2(t) = t \cdot \sigma^2$$

clearly depends on the time t, so the process is not stationary. The variance function is derived from the general fact that for **independent random variables**, y_1 and y_2, the variance of the sum is

$$Var(y_1 + y_2) = Var(y_1) + Var(y_2).$$

Thus,

$$Var(x_t) = Var(w_1 + w_2 + \cdots + w_t) = \sigma^2 + \sigma^2 + \cdots + \sigma^2 = t\sigma^2$$

The non-stationary autocovariance function is

$$Cov(x_t, x_{t+k}) = t\sigma^2, \quad k = 0, 1, \dots$$

which only depends on how many white noise terms x_t and x_{t+k} have in common (t) and not how far they are separated (k).

By combining the two results we obtain the non-stationary autocorrelation function

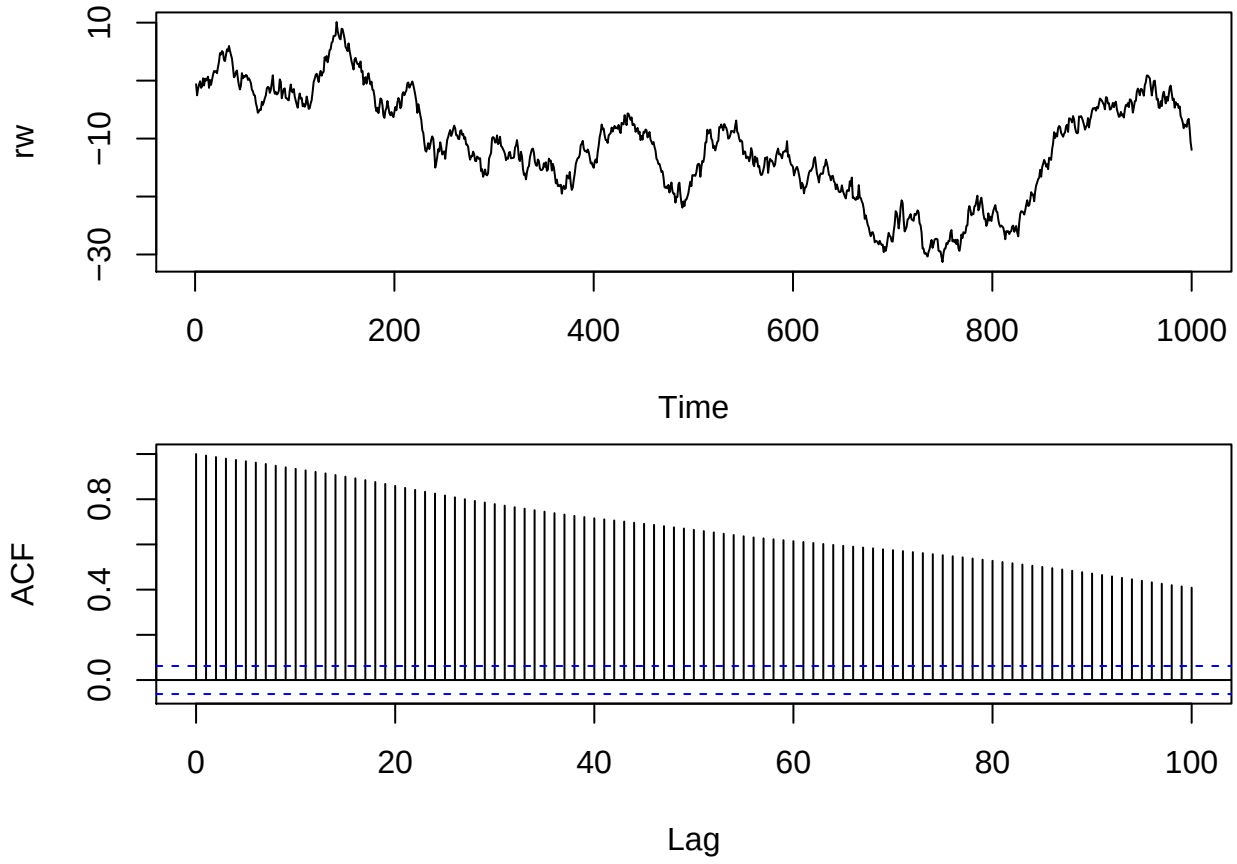
$$Cor(x_t, x_{t+k}) = \frac{Cov(x_t, x_{t+k})}{\sqrt{Var(x_t)Var(x_{t+k})}} = \frac{t\sigma^2}{\sqrt{t\sigma^2(t+k)\sigma^2}} = \frac{1}{\sqrt{1+k/t}}$$

When t is large compared to k we have very high correlation (close to one) and even though the process is not stationary we expect the correlogram of a reasonably long random walk to show very slow decay.

1.2.2 Simulation of random walk

We already know how to simulate Gaussian white noise (with `rnorm`) and the random walk is just a cumulative sum of white noise:

```r
w <- rnorm(1000)
rw <- cumsum(w)
par(mfrow = c(2,1), mar = c(4,4,0.5,0.5))
ts.plot(rw)
acf(rw, lag.max = 100)
```



1.2.3 Differencing

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.

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

$$\nabla x_t = x_t - x_{t-1}.$$

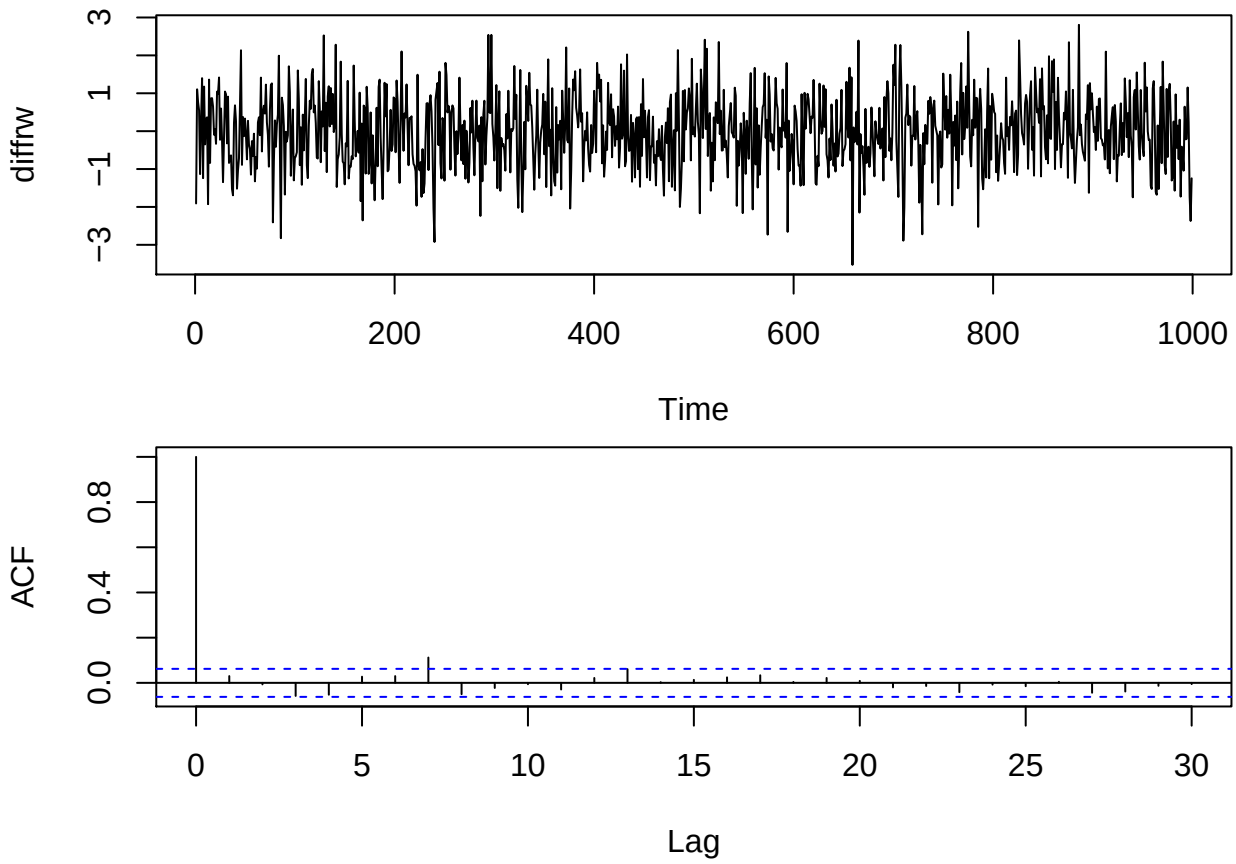
Since we assume/define $x_0 = 0$ we get

$$\nabla x_1 = x_1$$
$$\nabla x_2 = x_2 - x_1$$
$$\nabla x_3 = x_3 - x_2$$

etc.

Specifically when we difference a random walk 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)
```

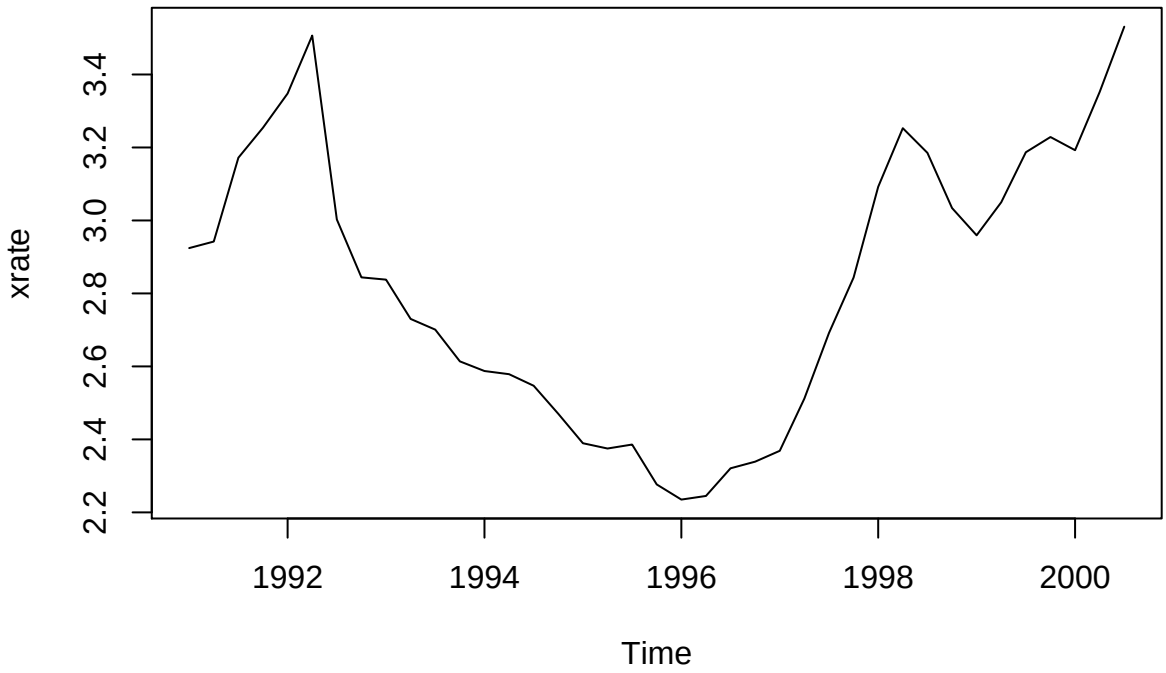


In general if a time series needs to be differenced to become stationary we say that the series is integrated (of order 1).

1.2.4 Example: Exchange rate

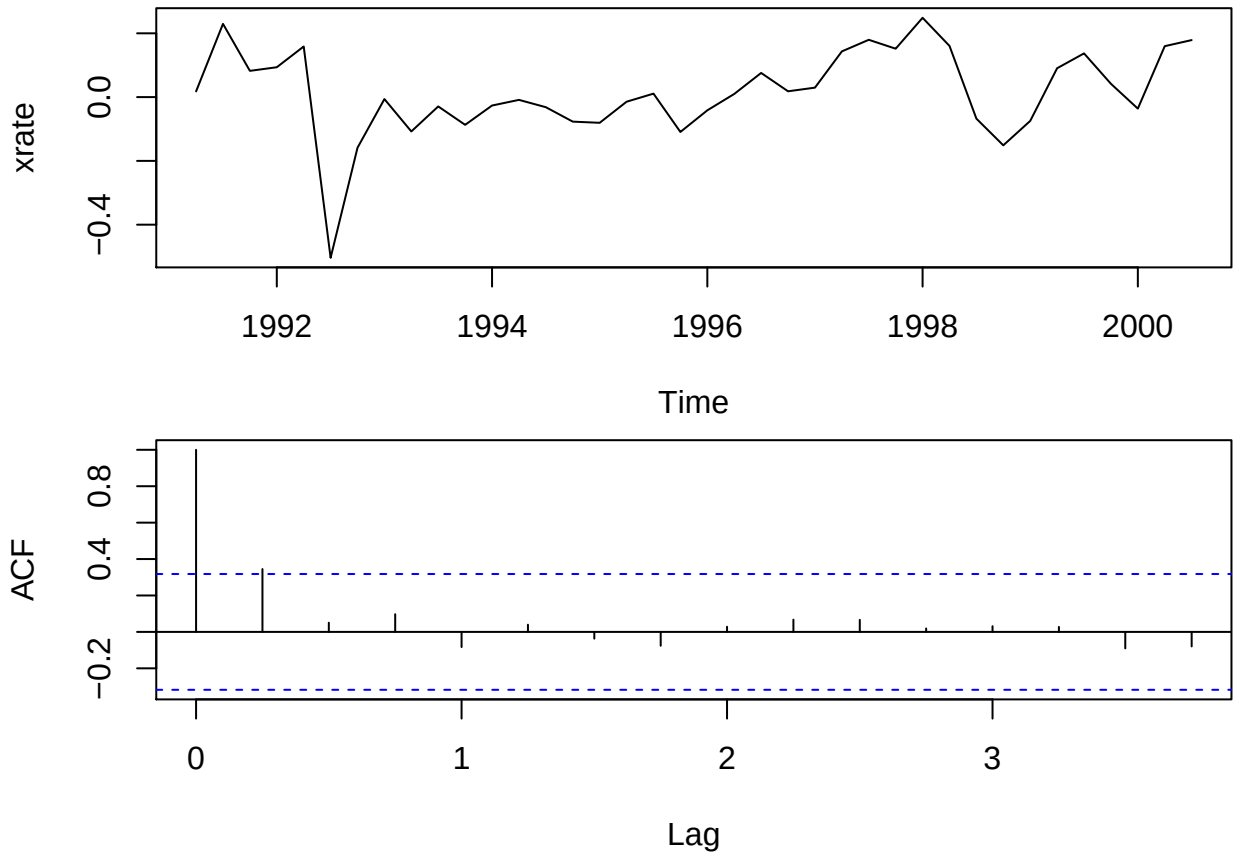
We have previously studied the exchange rate from GBP to NZD and observed what looked like an unpredictable stochastic trend and we would like to see if it could reasonably be described as a random walk.

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



To this end we difference the series and see if the difference 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.

2 Auto-regressive (AR) models

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

A significant auto-correlation at lag 1 means that x_t and x_{t-1} are correlated so the previous value x_{t-1} can be used to predict the current value x_t. This is the idea behind an 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.

The model is only stationary if $-1 < \alpha_1 < 1$ such that the dependence of the past decreases with time.

2.1.1 Properties of AR(1) models

For a stationary AR(1) model with $-1 < \alpha_1 < 1$ it can be shown that

- $\mu(t) = 0$
- $Var(x_t) = \sigma^2(t) = \sigma^2/(1-\alpha_1^2)$

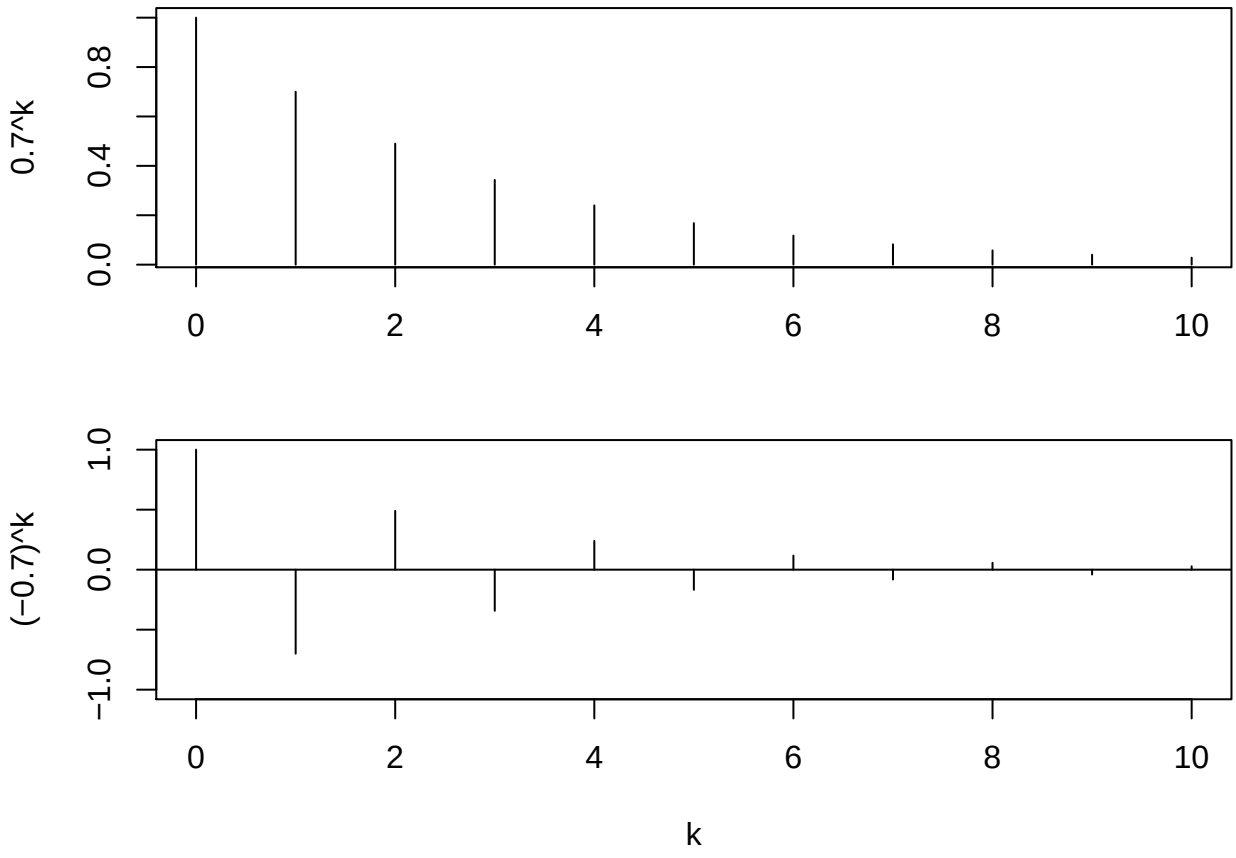
- $\gamma(k) = \alpha_1^k \sigma^2/(1-\alpha_1^2)$
- $\rho(k) = \alpha_1^k$

Furthermore, the partial autocorrelation of an AR(1) model is zero for all lags $k \geq 2$.

Below are the theoretical correlograms for the following AR(1) models:

- Model 1: $x_t = 0.7x_{t-1} + w_t$
- Model 2: $x_t = -0.7x_{t-1} + w_t$

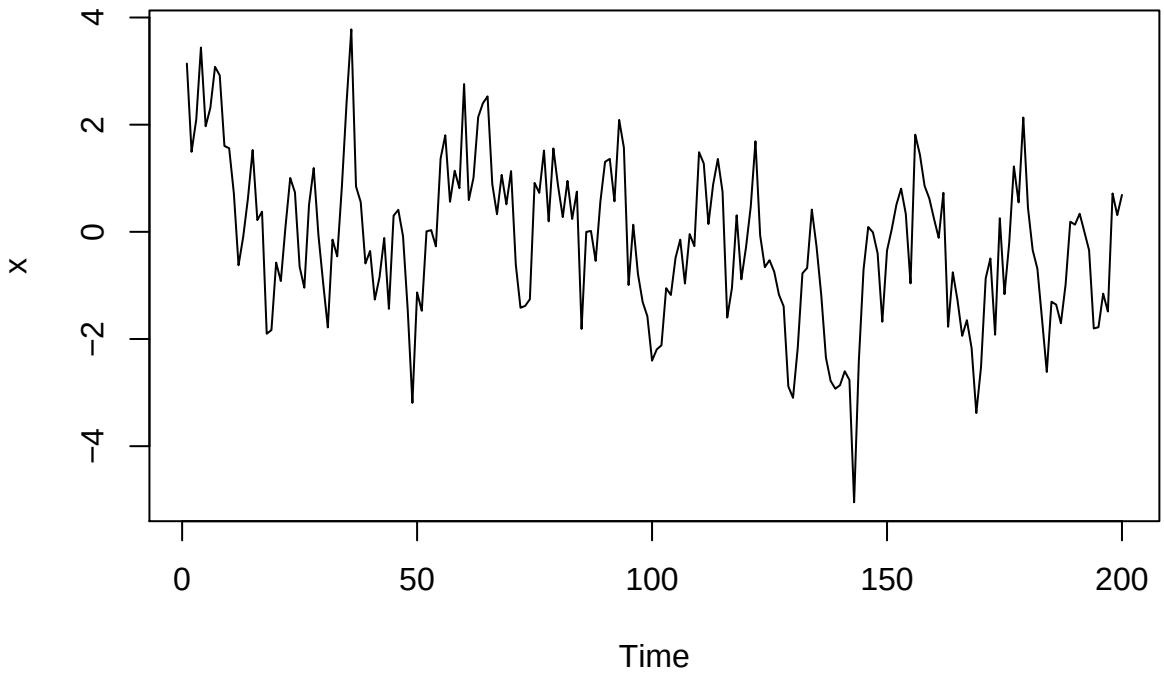
```
k <- 0:10
par(mfrow = c(2,1), mar = c(4,4,0.5,0.5))
plot(k, 0.7^k, type = "h", xlab = "")
plot(k, (-0.7)^k, type = "h", ylim = c(-1,1))
abline(h=0)
```



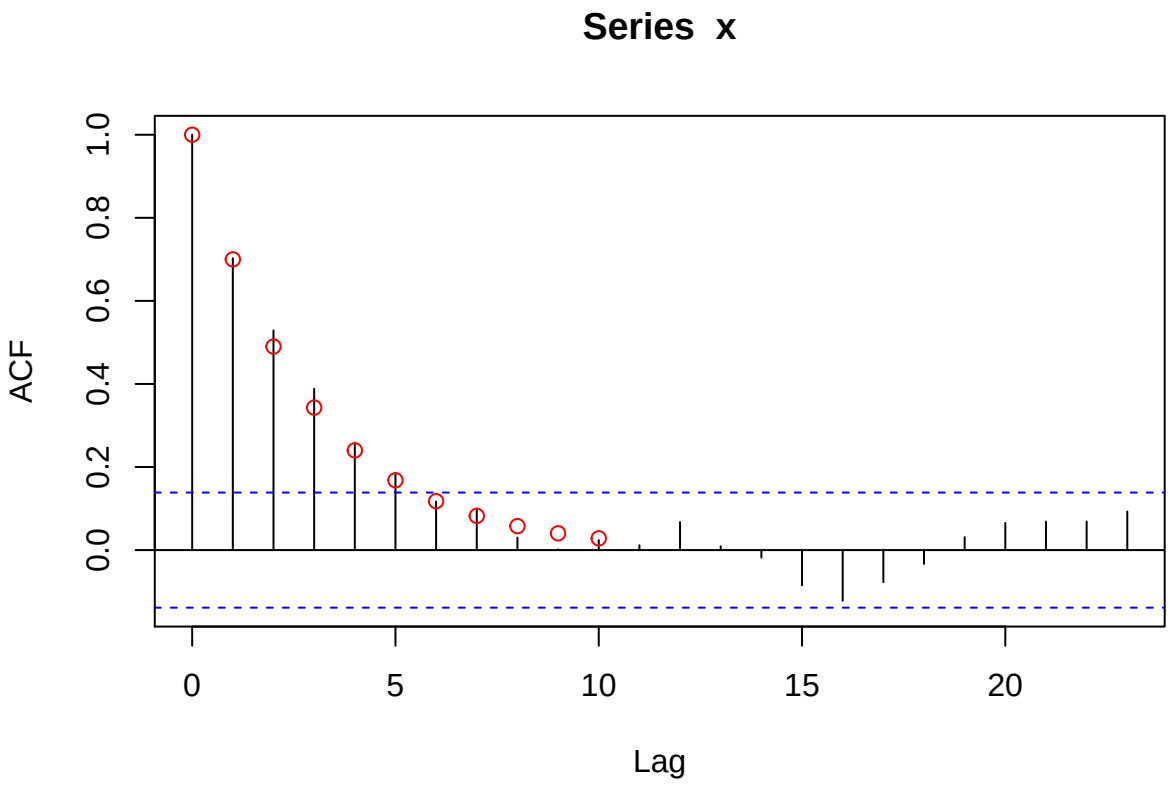
2.1.2 Simulation of AR(1) models

R has a built-in function `arima.sim` to simulate AR(1) 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:

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




```r
acf(x)
points(k, 0.7^k, col = "red")
```



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

2.1.3 Fitted AR(1) models

To estimate the autoregressive parameter α_1 we use the function `ar`:

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

The resulting object contains the value of the estimated parameter and a bunch of other information. In this case the input data are artificial so we know we should ideally get a value close to 0.7:

```
fit$ar
```

```
## [1] 0.7027681
```

An estimate of the variance of the estimate $\hat{\alpha}_1$ is given in `fit$asy.var.coef` (the estimated std. error is the square root of this):

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

```
##              [,1]
## [1,] 0.002556146
```

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

```
##             [,1]
## [1,] 0.05055834
```

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

```
## [1] 0.6016515 0.8038848
```

There are several different methods that can be used to estimate the model, but we ignore the details of this. However, first step in the estimation is to subtract the mean $\bar{x}$ of the time series before doing anything else, so the model that is fitted is actually:

$$x_t - \bar{x} = \alpha_1 \cdot (x_{t-1} - \bar{x}) + w_t$$

To predict the value of x_t given 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 the predictions are given by

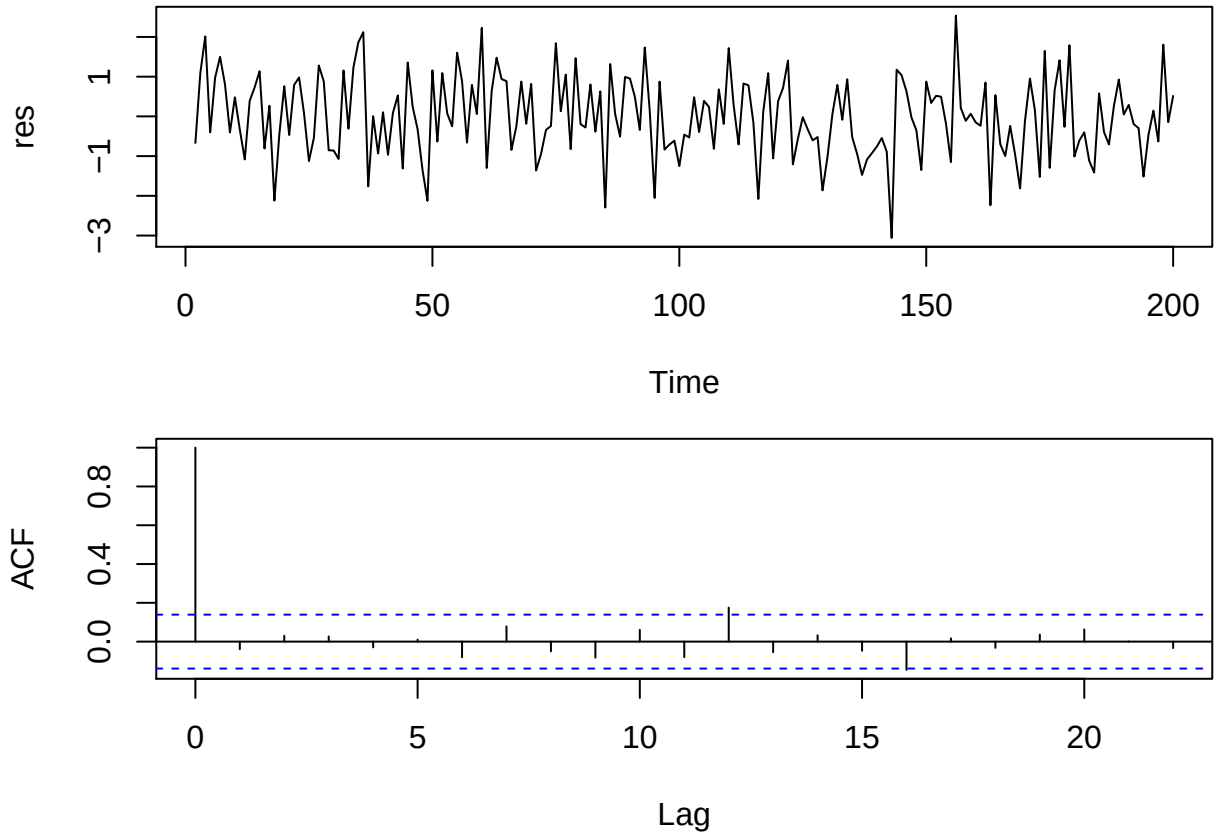
$$\hat{x}_t = \bar{x} + \hat{\alpha}_1 \cdot (x_{t-1} - \bar{x}), \quad t \geq 2.$$

Given the predictions we can estimate the model errors as usual by the model residuals:

$$\hat{w}_t = x_t - \hat{x}_t, \quad t \geq 2.$$

If we believe the model describes the dataset well the residuals should look like a sample of white noise:

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



This naturally looks good for this artificial dataset.

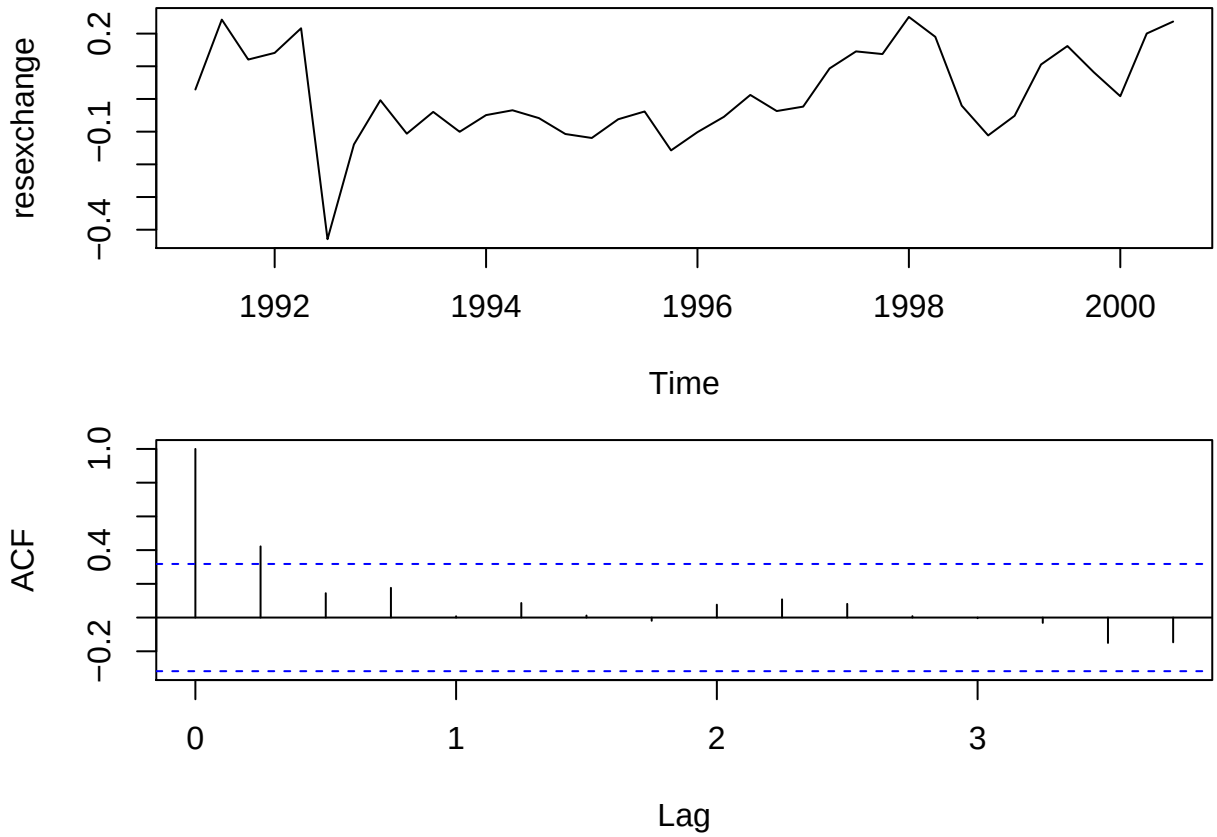
2.1.4 AR(1) model fitted to exchange rate

A random walk is an example of a AR(1) model with $\alpha_1 = 1$, and it is non-stationary. This 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)
fitexchange$ar
```

```
## [1] 0.890261
```

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



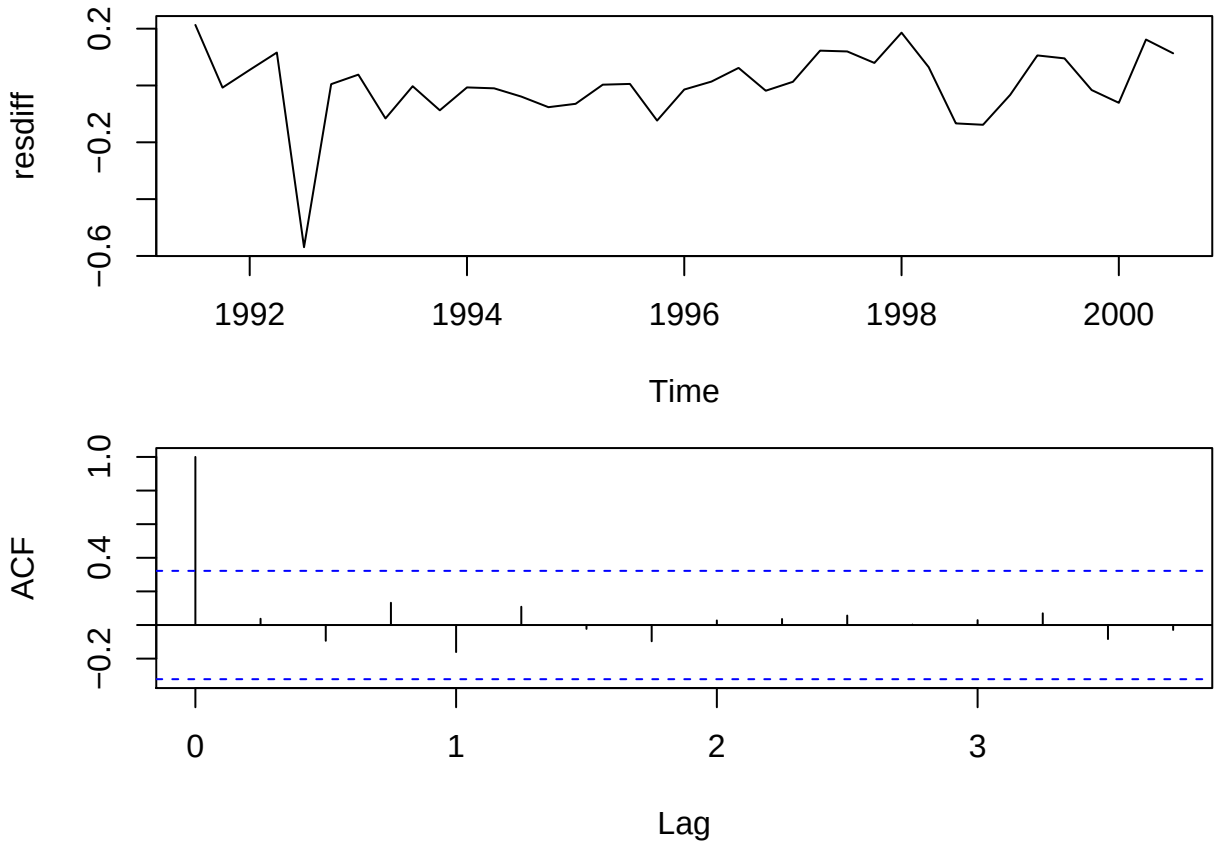
This does not appear to really provide a better fit than the random walk model proposed earlier.

An alternative would be to propose a 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)
fitdiff$ar
```

```
## [1] 0.3451507
```

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



2.1.5 Prediction from AR(1) model

We can use a fitted AR(1) model to predict future values of a time series. If the last observed time point is t then we predict x_{t+1} using the equation given previously:

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

If we want to predict x_{t+2} we use

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

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 (probably a bad idea due to the stochastic trend):

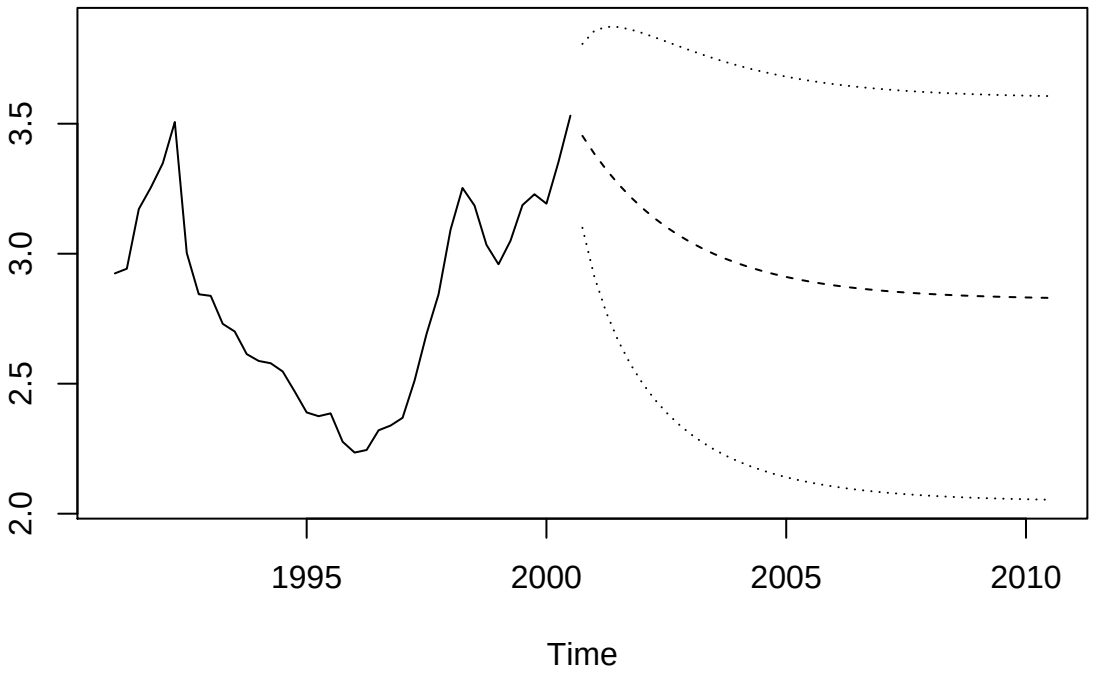
```
pred1 <- predict(fitexchange, n.ahead = 4)
pred1
```

```
## $pred
##          Qtr1     Qtr2     Qtr3     Qtr4
## 2000                           3.453332
## 2001 3.384188 3.322631 3.267830
##
## $se
##           Qtr1      Qtr2      Qtr3      Qtr4
## 2000                                0.1767767
## 2001 0.2366805 0.2750411 0.3020027
```

Note how the prediction returns both the predicted value and a standard error for this value. So we predict that the exchange rate in third quarter of 2001 would be within 3.27 ± 0.6 with approximately 95% probability.

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))
```



2.2 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.

The parameters cannot be chosen arbitrarily if we want the model to be stationary. To check that a given AR(p) model is stationary we must find all the roots of the characteristic equation

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

and check that the absolute value of all the roots is greater than 1.

2.2.1 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 $\alpha_1, \ldots, \alpha_p$.

For the first point we note the following theoretical property of AR(p) processes:

For an AR(p) process the theoretical value of the partial autocorrelation at lag k is α_k. Thus, for all lags greater than p the theoretical partial autocorrelation is zero. So heuristically we can choose the maximal lag p of an AR(p) process by looking at when the estimated partial autocorrelation function is close to zero.

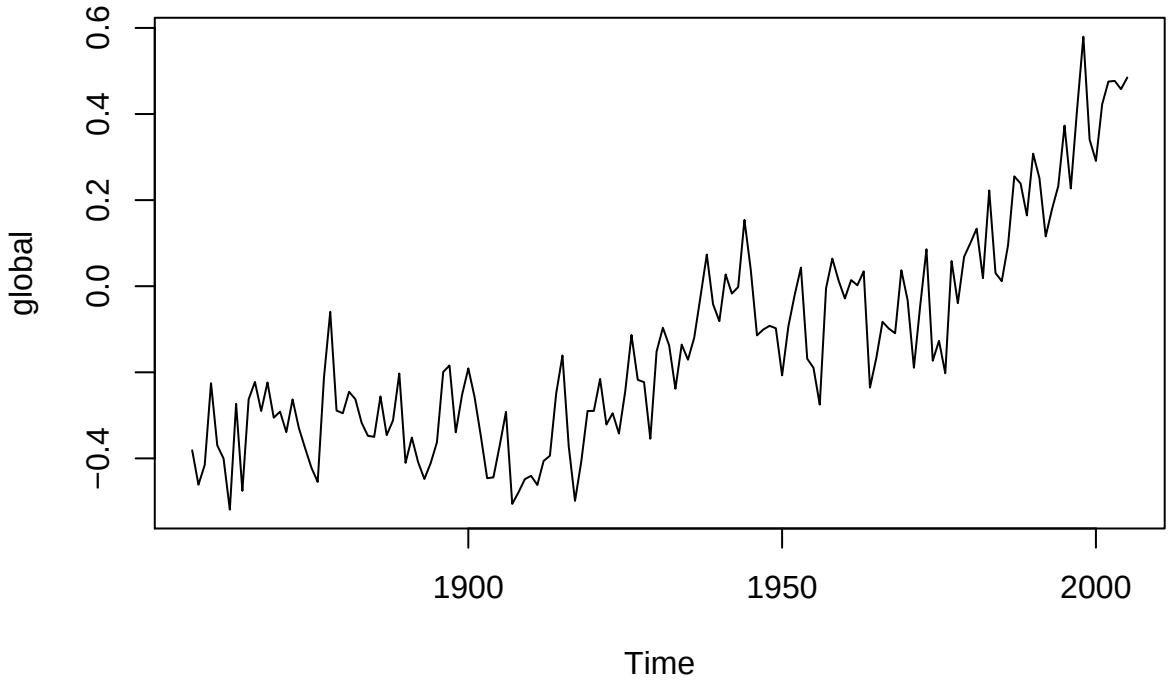
In practice it can be hard to say exactly when the pacf is sufficienty close to zero and more advanced methods are often used. The function `ar` in R uses AIC (Akaike's Information Criterion) to automatically select the value for p.

Once the order is chosen and the estimates $\hat{\alpha}_1, \ldots, \hat{\alpha}_p$ are found 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.

2.2.2 Example of AR(p) model

We use an example of monthly global temperatures expressed as anomalies from the monthly average in 1961-1990. We reduce the dataset to the yearly mean temperature, and fit an AR(p) model to this. A good fit would indicate that the higher temperatures over the last decade could be explained by a purely stochastic process which just has dependence on the temperature anomalies from previous year and eventually might as well start decreasing again. **However, this does not mean that there is no climate crisis!** There is lots of scientific evidence of this based on much more complicated models and more detailed data.

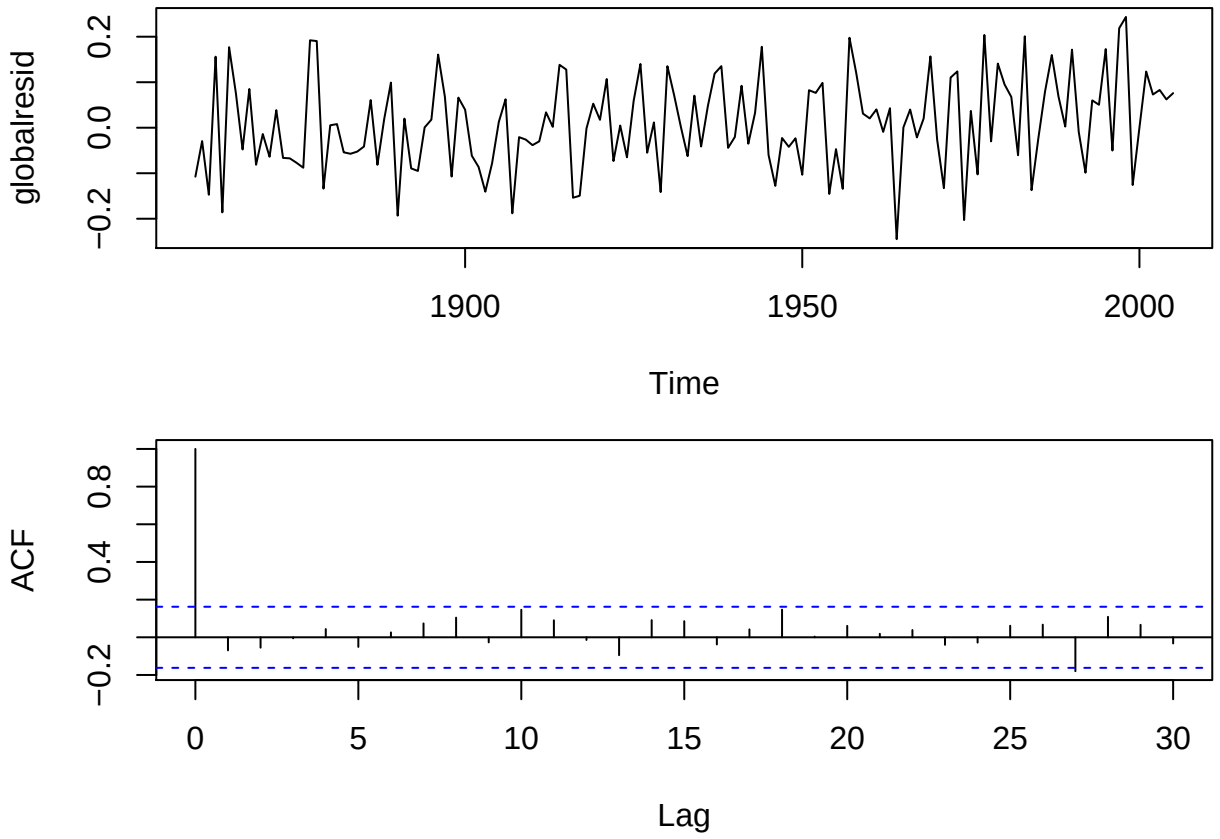
```
global_data <- scan("https://asta.math.aau.dk/eng/static/datasets?file=global.dat")
global_monthly <- ts(global_data, st = c(1856,1), end = c(2005,12), freq = 12)
global <- aggregate(global_monthly, FUN = mean)
plot(global)
```




```
globalfit <- ar(global, order.max = 10)
globalfit
```

```
##
## Call:
## ar(x = global, order.max = 10)
##
## Coefficients:
##      1       2       3       4
## 0.6825  0.0032  0.0672  0.1730
##
## Order selected 4  sigma^2 estimated as  0.01371
```

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



3 Moving average models

Another class of models are moving average (MA) models. An 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 zero and variance σ_w^2 and $\beta_1, \beta_2, \ldots, \beta_q$ are parameters to be estimated.

Since a moving average process is a finite sum of stationary white noise terms it is itself stationary and therefore the mean and variance is time-invariant (same constant mean and variance for all t):

- 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, for $k \geq 0$, is

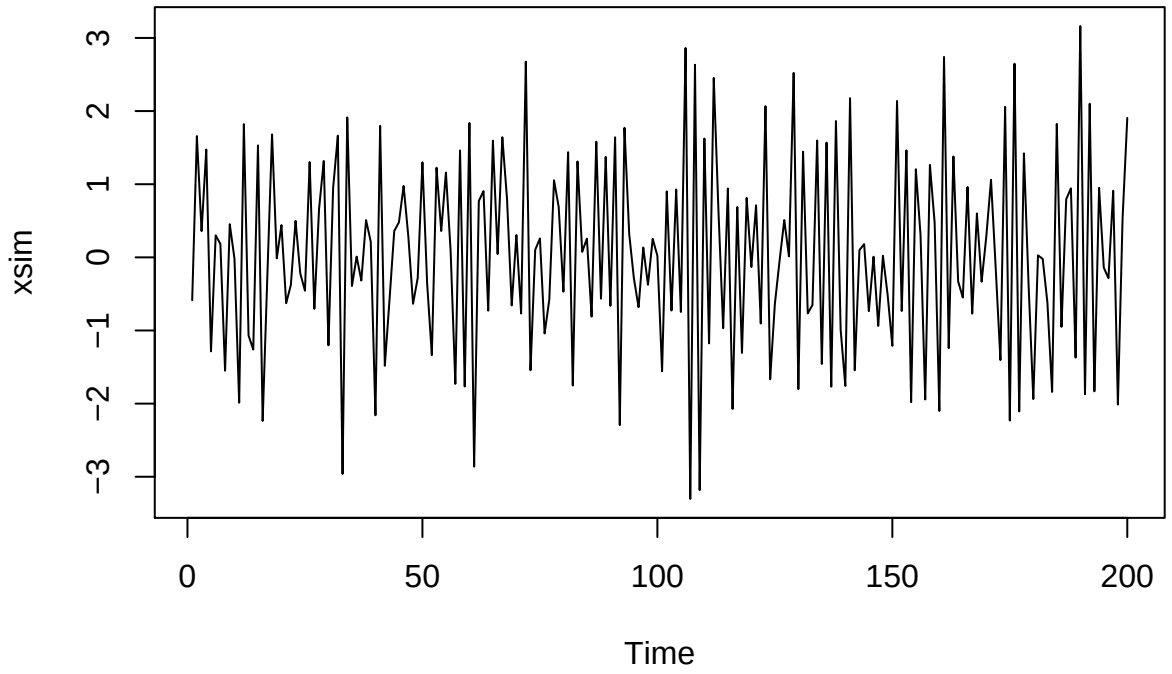
$$\rho(k) = \begin{cases} 1 & k = 0 \\ \sum_{i=0}^{q-k} \beta_i \beta_{i+k} / \sum_{i=0}^{q} \beta_i^2 & k = 1, 2, \ldots, q \\ 0 & k > q \end{cases}$$

where $\beta_0 = 1$.

3.1 Simulation of MA(q) processes

To simulate a MA(q) process we just need the white noise process w_t and then transform it using the MA coefficients. If we e.g. want to simulate a model with $\beta_1 = -0.7$, $\beta_2 = 0.5$, and $\beta_3 = -0.2$ we can use `arima.sim`:

```
xsim <- arima.sim(list(ma = c(-.7, .5, -.2)), n = 200)
plot(xsim)
```



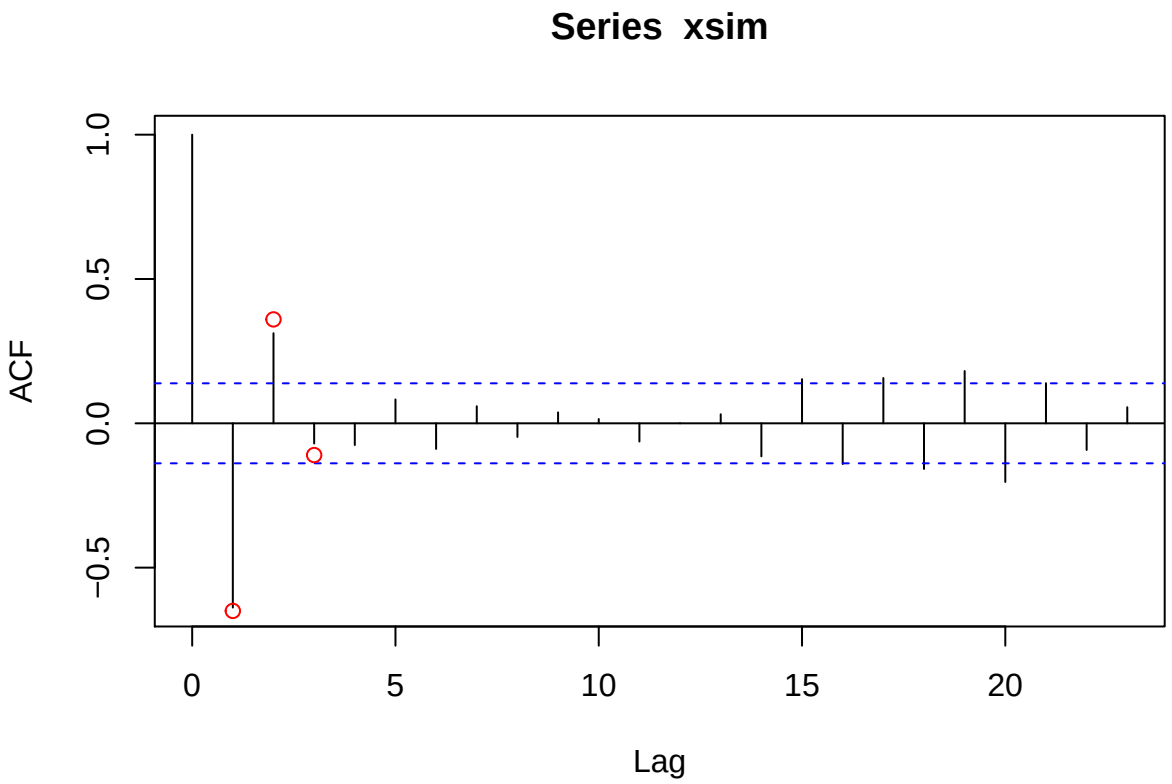
The theoretical autocorrelations are in this case:

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

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

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

```
acf(xsim)
points(1:3, c(-.65, .36, -.11), col = "red")
```



3.1.1 Estimation of MA(q) models

To estimate the parameters of a MA(q) model we use `arima`:

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

```
##
## Call:
## arima(x = xsim, order = c(0, 0, 3))
##
## Coefficients:
##           ma1     ma2      ma3  intercept
##       -0.7478  0.4605  -0.2267     0.0358
## s.e.   0.0708  0.0805   0.0752     0.0346
##
## sigma^2 estimated as 1.001:  log likelihood = -284.25,  aic = 578.51
```

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 using the AIC.

4 Mixed models: Auto-regressive moving average models

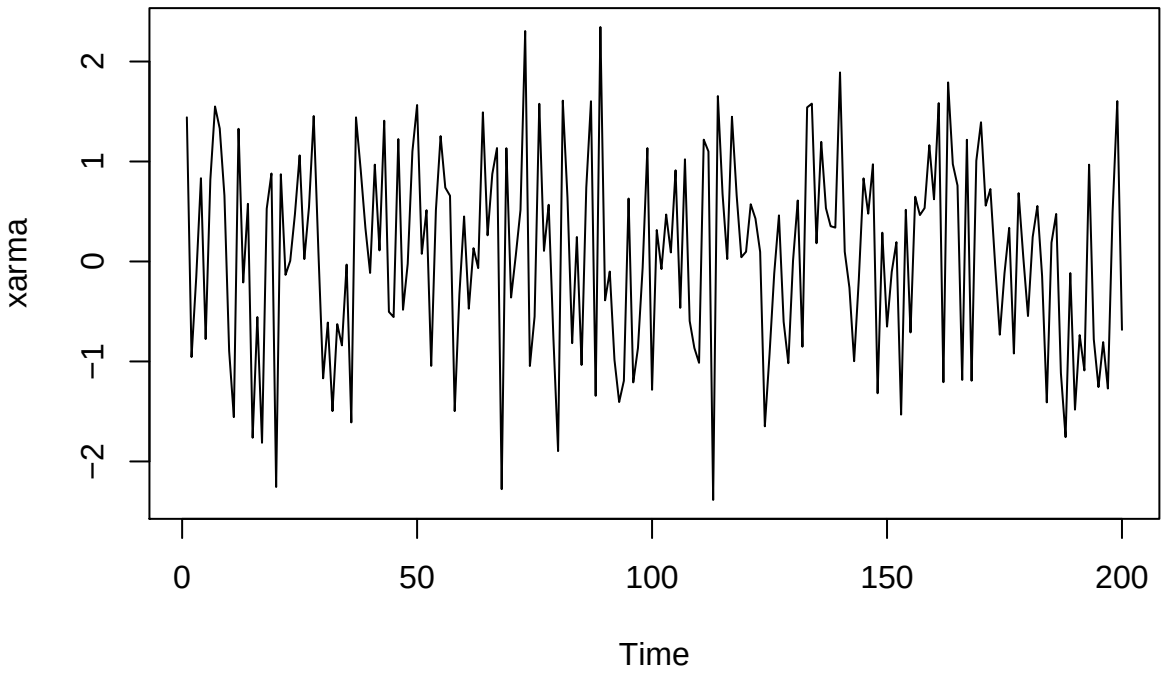
A time series x_t follows an auto-regressive moving average (ARMA) process of order (p, q), denoted $ARMA(p, q)$, if

$$x_t = \alpha_1 x_{t-1} + \alpha_2 x_{t-2} + \cdots + \alpha_p x_{t-p} + w_t + \beta_1 w_{t-1} + \beta_2 w_{t-2} + \cdots + \beta_q w_{t-q}$$

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:

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



Estimation is done with `arima` as before.

4.0.1 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 model 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
```

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

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

```
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
```

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