# Module 5 - Model Specification

## MATH1318 Time Series Analysis

Dr. Haydar Demirhan - RMIT University School of Science

*All the contents in this presentation are mainly based on the textbook of MATH1318 Time Series Analysis course: 'Cryer and Chan, Time Series Analysis with R, Springer, 2008.' Other resources than the textbook are cited accordingly.*

# Introduction

ARIMA models constitute a large class of models for the analysis of stationary and nonstationary time series.

In this module, we will study how to choose appropriate values for the parameters $p$, $d$, and $q$ of ARIMA(p,d,q) models by using a bunch of tools.

For this aim, we will focus on

- the properties of sample ACF and PACF,
- the partial and extended autocorrelation function, and
- Bayesian Information Criterion (BIC).

Then, we will examine some simulated time series data and specify ARIMA models.

# The Autocorrelation Function

We will use the ACF to observe the main characteristics of ARMA models.

Our goal is to recognize, to the extent possible, patterns in the sample ACF that are characteristic of the known patterns in the true ACF for common ARMA models.

However, we have only an estimate of true ACF from the observed data.

Therefore, we need to investigate the sampling properties of ACFs to facilitate the comparison of estimated correlations with theoretical correlations.

The properties given below are mostly based on Wei (2006).

Suppose $\{Y_t\}$ is **white noise**. Then,

$$Var(r_k) \approx \frac{1}{n} \text{ and } Corr(r_k, r_j) \approx 0 \text{ for } k \neq j.$$

In the sample ACF of a white noise series, all autocorrelations should be insignificant at all lags.

Suppose $\{Y_t\}$ is generated by an **AR(1)** process. Then, we have $\rho_k = \phi^k$.

In the sample ACF of an AR(1) series, there are exponential decays depending on the sign of $\phi$.

If $0 < \phi < 1$, then all autocorrelations are positive.

If $-1 < \phi < 0$, sign of the autocorrelations shows an alternating pattern starting with a negative value.

Suppose $\{Y_t\}$ is generated by an **AR(2)** process. Then, we have $\rho_k = \phi_1 \rho_{k-1} + \phi_2 \rho_{k-2}, k \geq 1$.

In the sample ACF of an AR(2) series, there are exponential decays if the roots of the above equation are real and a damped sine wave if the roots are complex.

Suppose $\{Y_t\}$ is generated by an **AR(p)** process. Then,

The ACF of AR(p) series tails off as a mixture of exponential decays or damped sine waves depending on the roots of autocorrelation equation.

A damped sine wave appears if some of the roots are complex.

Suppose $\{Y_t\}$ is generated by an **MA(1)** process. Then we have

$$\begin{aligned} \rho_k &= \frac{-\theta}{1+\theta^2}, \quad k = 1, \\ &= 0, \qquad k > 1. \end{aligned}$$

In the sample ACF of an MA(1) series, there will be a significant autocorrelation at lag 1 and those at the rest of lags will be insignificant.

So, the sample ACF should cut off after lag 1.

Suppose $\{Y_t\}$ is generated by an **MA(2)** process. Then we have

$$\begin{aligned}
\rho_k &= \frac{-\theta_1(1-\theta_2)}{1+\theta_1^2+\theta_2^2}, \quad k=1,\\
&= \frac{\theta_2}{1+\theta_1^2+\theta_2^2}, \quad k=2,\\
&= 0, \quad k>2.
\end{aligned}$$

All autocorrelations after lag 2 will be insignificant in the sample ACF of a MA(2) series.

Suppose $\{Y_t\}$ is generated by an **MA(q)** process. Then we have

$$\begin{aligned} \rho_k &= \sigma_a^2(-\theta_k + \theta_1\theta_{k+1} + \cdots + \theta_{q-k}\theta_q), \quad k = 1, 2, \ldots, q \\ &= 0, \quad k > q. \end{aligned}$$

All autocorrelations after lag q will be insignificant in the sample ACF of a MA(q) series.

Suppose $\{Y_t\}$ is generated by an **ARMA(1,1)** process. Then we have

$$\begin{aligned}
\rho_k &= 1, && k = 0, \\
\rho_k &= \frac{(\phi-\theta)(1-\phi\theta)}{1+\theta^2-2\theta\phi}, && k = 1, \\
&= \phi\rho_{k-1}, && k \geq 2.
\end{aligned}$$

In the sample ACF of an ARMA(1,1) series, we expect to have exponential decays.

Beyond $\rho_1$ the ACF of ARMA(1,1) follows the same pattern with that of AR(1) process.

Suppose $\{Y_t\}$ is generated by an **ARMA(p,q)** process.

For the general ARMA(p,q) processes, the sample ACF will tail off after lag $q$ like an AR(p) process.

# The Partial Autocorrelation Function

For an observed time series, we need to be able to estimate the partial autocorrelation function at a variety of lags.

When we estimate autocorrelation between $Y_t$ and $Y_{t+k}$ **given** $Y_{t+1}, Y_{t+2}, \ldots, Y_{t+k-1}$ , we get an estimate of **partial autocorrelation between** $Y_t$ **and** $Y_{t+k}$. We denote the partial autocorrelation by $\phi_{kk}$ .

The properties given below are mostly based on Wei (2006).

Suppose $\{Y_t\}$ is **white noise**. Then, because $Var(\phi_{kk}) \approx 1/n$, $2/\sqrt{n}$ can be used as the critical bound to test hypothesis of a white noise process.

In the sample PACF of a white noise series, all partial autocorrelations should be insignificant at all lags.

Suppose $\{Y_t\}$ is generated by an **AR(1)** process. Then, we have

$$\begin{aligned} \phi_{kk} &= \rho_1 = \phi, \quad k = 1, \\ &= 0, \quad k > 1. \end{aligned}$$

In the sample PACF of an AR(1) series will show a significant positive of negative spike at lag 1 depending on the sign of $\phi$ and then cut off.

Suppose $\{Y_t\}$ is generated by an **AR(2)** process. Because of the fact that $\rho_k = \phi_1 \rho_{k-1} + \phi_2 \rho_{k-2}, k \geq 1$.

In the sample PACF of an AR(2) series, the partial autocorrelations at the first two lags will be significant and the rest will be insignificant.

Suppose $\{Y_t\}$ is generated by an **AR(p)** process. Then

In the sample PACF of an AR(p) series, the estimates of partial autocorrelations at the lags $1, 2, \ldots, q$ will be significant and than they will vanish after lag $q$.

Suppose $\{Y_t\}$ is generated by an **MA(1)** process. Then we have

$$\phi_{kk} = \frac{-\theta^k(1-\theta^2)}{1-\theta^{2(k+1)}}$$

for $k \geq 1$.

The sample PACF of a MA(1) series tails off after lag 1 in one of two forms depending on the sign of $\theta$.

Suppose $\{Y_t\}$ is generated by an **MA(2)** process. Then,

There are exponential decays in the PACF of MA(2) series if the roots of the autocorrelation equation are real and a damped sine wave if the roots are complex.

Suppose $\{Y_t\}$ is generated by an **MA(q)** process. Then,

The PACF of MA(q) series tails off as a mixture of exponential decays or damped sine waves depending on the roots of autocorrelation equation.

Suppose $\{Y_t\}$ is generated by an **ARMA(1,1)** process. Then we have

$$\begin{aligned}
\rho_k &= 1, && k = 0, \\
\rho_k &= \frac{(\phi-\theta)(1-\phi\theta)}{1+\theta^2-2\theta\phi}, && k = 1, \\
&= \phi\rho_{k-1}, && k \geq 2.
\end{aligned}$$

In the sample PACF of an ARMA(1,1) series exhibits similar patterns as MA(1) and AR(1) process.

Suppose $\{Y_t\}$ is generated by an **ARMA(p,q)** process.

Because ARMA(p,q) process contains MA(q) process as a special case, its PACF will be a mixture of exponential decays or damped sine waves.

The following table summarizes the general behavior of the ACF and PACF for ARMA models.

| | AR($p$) | MA($q$) | ARMA($p, q$), $p>0$, and $q>0$ |
|---|---|---|---|
| **ACF** | Tails off | Cuts off after lag $q$ | Tails off |
| **PACF** | Cuts off after lag $p$ | Tails off | Tails off |

# The Extended Autocorrelation Function

The sample ACF and PACF provide effective tools for identifying pure AR(p) or MA(q) models.

However, for a mixed ARMA model, its theoretical ACF and PACF have infinitely many nonzero values, making it difficult to identify mixed models from the sample ACF and PACF.

Many graphical tools have been proposed to make it easier to identify the ARMA orders.

The extended autocorrelation (EACF) method is one of those methods to identify the order of autoregressive and moving average components of ARMA models.

It is supposed to have good sampling properties for **moderately large sample sizes**.

The EACF method uses the fact that if the AR part of a mixed ARMA model is known, “filtering out” the autoregression from the observed time series results in a pure MA process that enjoys the cutoff property in its ACF.

In the EACF of an ARMA model, a table is displayed with the element in the $k$th row and $j$th column equal to the symbol “$\times$” if the lag $j+1$ sample correlation of corresponding autoregressive residuals is significantly different from 0, and 0 otherwise.

In such a table, an ARMA(p,q) process will have a theoretical pattern of a triangle of zeroes, with the upper left-hand vertex corresponding to the ARMA orders. **Theoretical** EACF for an ARMA(1,1) model is given below:

| *AR/MA* | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | x | x | x | x | x | x | x | x | x | x | x | x | x | x |
| 1 | x | 0* | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | x | x | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | x | x | x | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | x | x | x | x | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | x | x | x | x | x | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | x | x | x | x | x | x | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | x | x | x | x | x | x | x | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The upper left-hand vertex of the triangle of zeros corresponds to AR(1) and MA(1) components, we identify the model as ARMA(1,1) from this display.

Notice that the **sample** EACF will never be this clear-cut.

# Specification of Some Simulated Time Series

Now, we will consider specification of $p$, $d$, and $q$ parameters of ARIMA(p,d,q) models in practice.

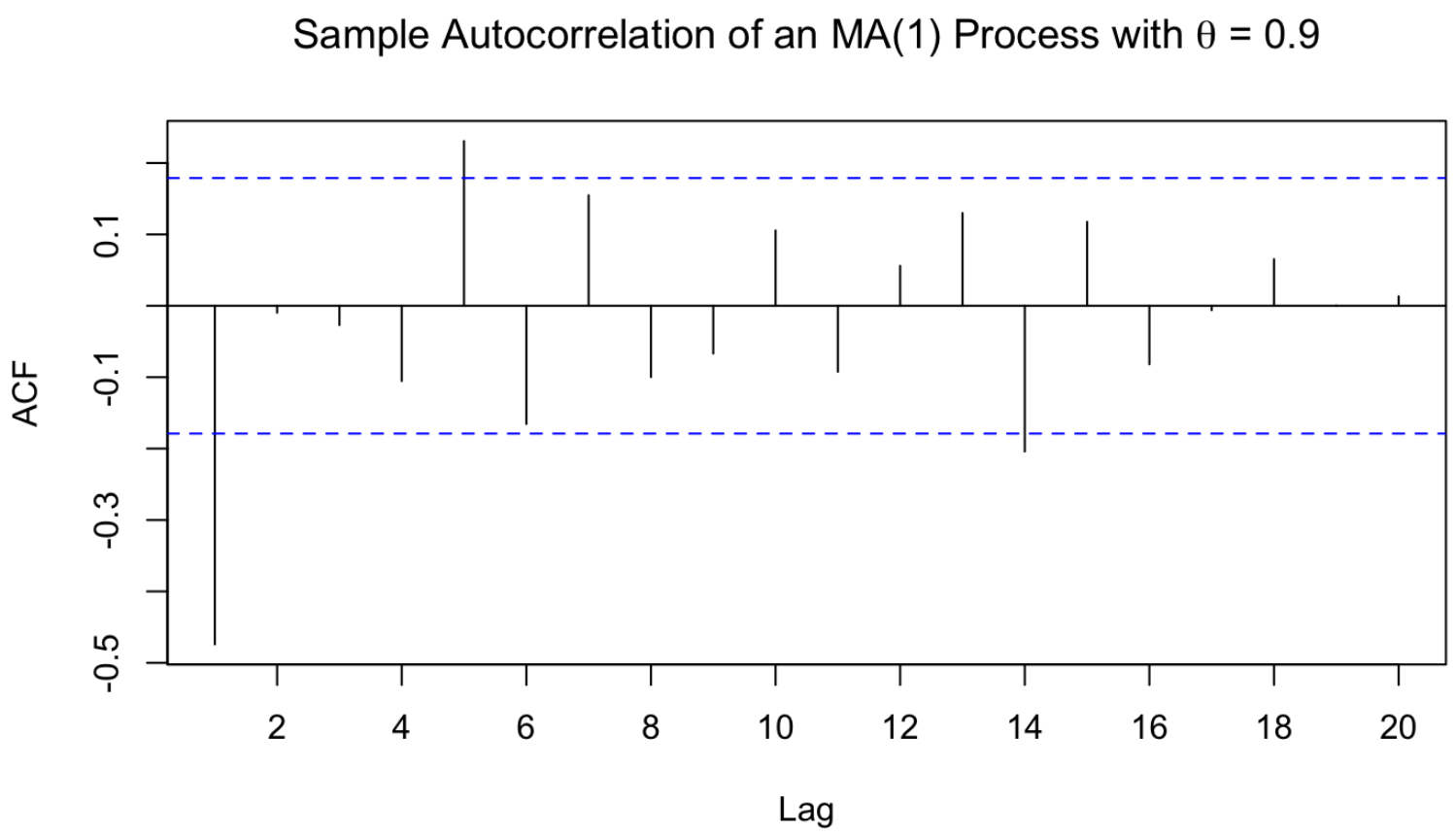
We will use simulated datasets for all illustrations.

Thus, we know the exact values of $p$, $d$, and $q$ beforehand.

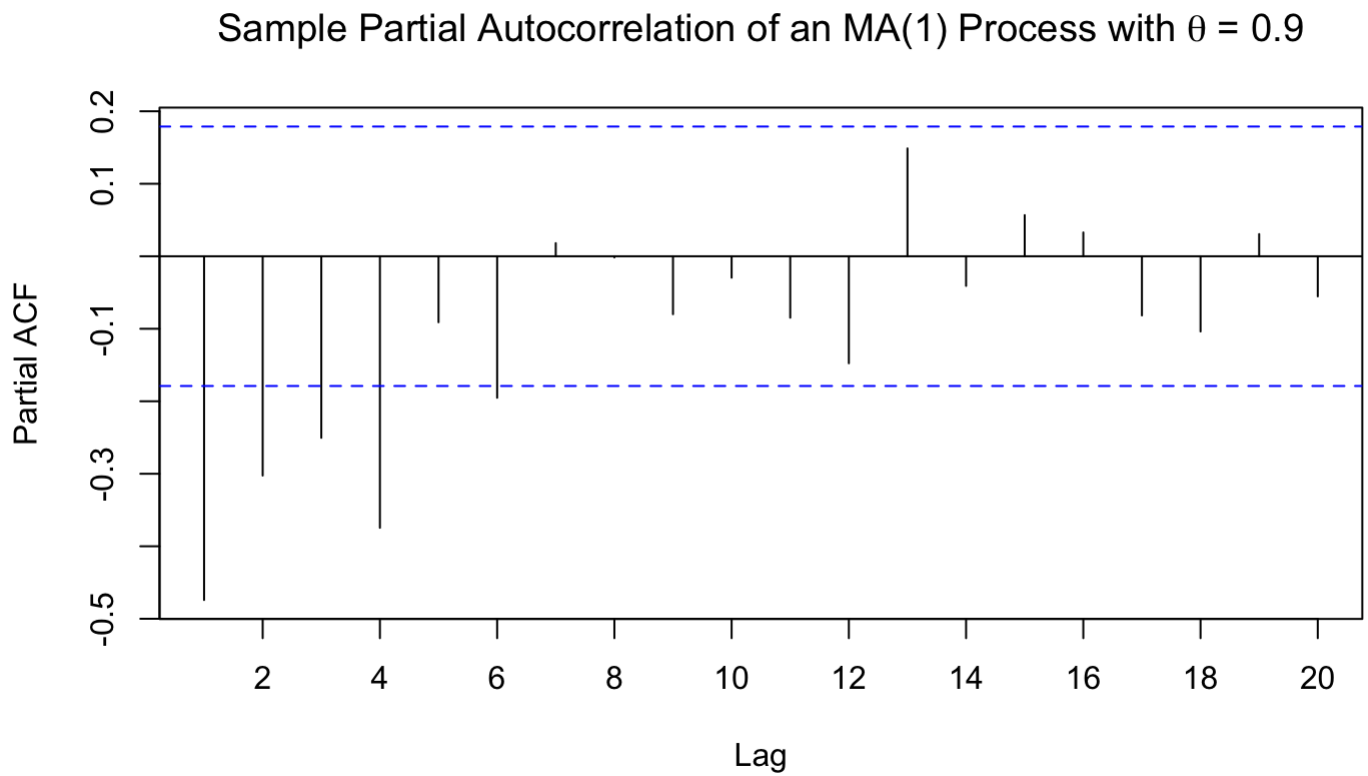
First, we will see some examples for MA(q) processes and then study AR(p) processes and so on.

The sample autocorrelation out to lag 20 for the simulated time series that is simulated from an MA(1) process with length 120.

```
data(ma1.1.s); acf(ma1.1.s,xaxp=c(0,20,10), main=expression(paste("Sample Autocorrelation
```



```
pacf(ma1.1.s,xaxp=c(0,20,10), main=expression(paste("Sample Partial Autocorrelation of an
```



The dashed horizontal lines in the plot are based on the approximate large sample standard error $(1/\sqrt{n})$ that applies to a white noise process.

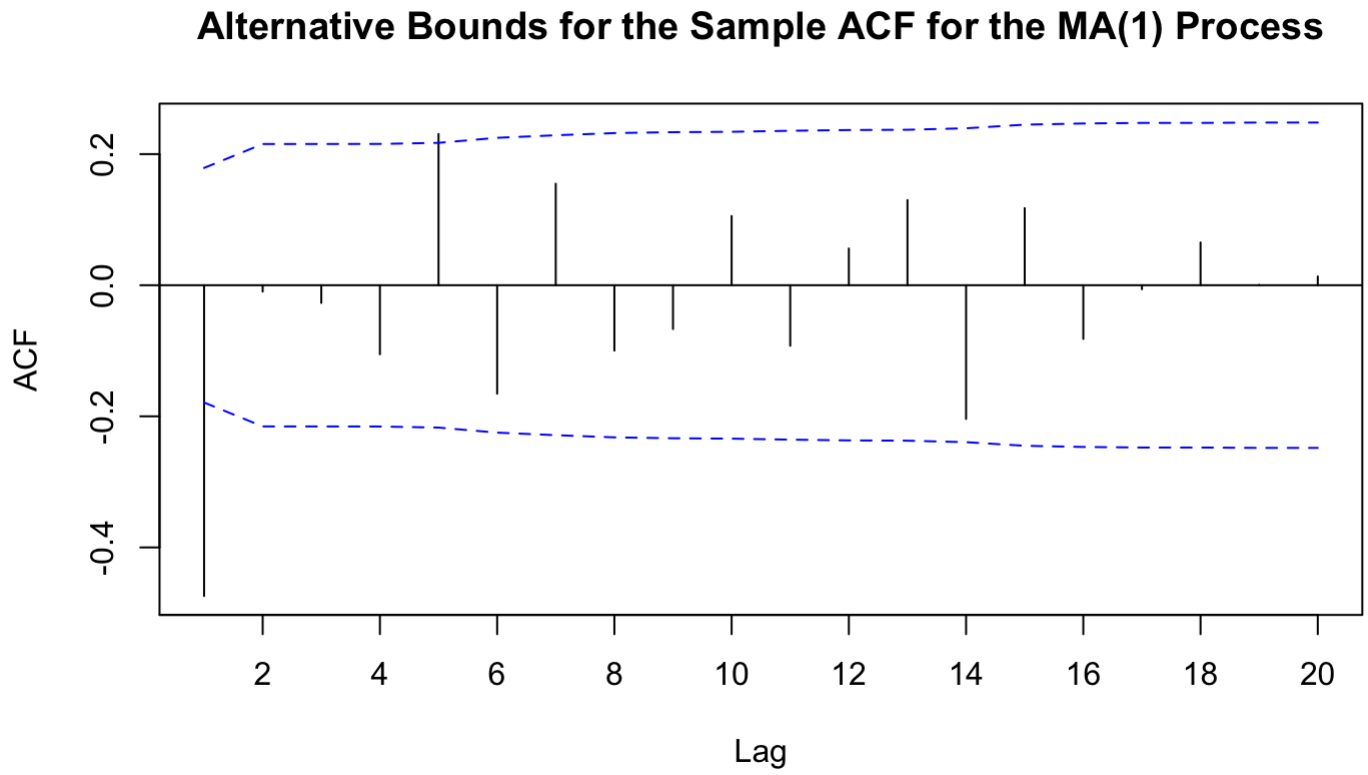
So, they are plotted at $\pm 2/n = \pm 0.1826$, and intended to give critical values for testing whether or not the autocorrelation coefficients are significantly different from zero.

Notice that the sample ACF values exceed these rough critical values at lags 1, 5, and 14.

Although the true autocorrelations at lags 5 and 14 are both zero due to the nature of the MA(1) process.

The next plot displays the ACF with critical bound calculated in an alternative way that they are based on plus and minus two of the more complex standard errors. Thus, the uncertainty increases in higher lags and we get wider critical bounds for large $k$.

```
acf(ma1.1.s,ci.type='ma',xaxp=c(0,20,10),
    main="Alternative Bounds for the Sample ACF for the MA(1) Process")
```



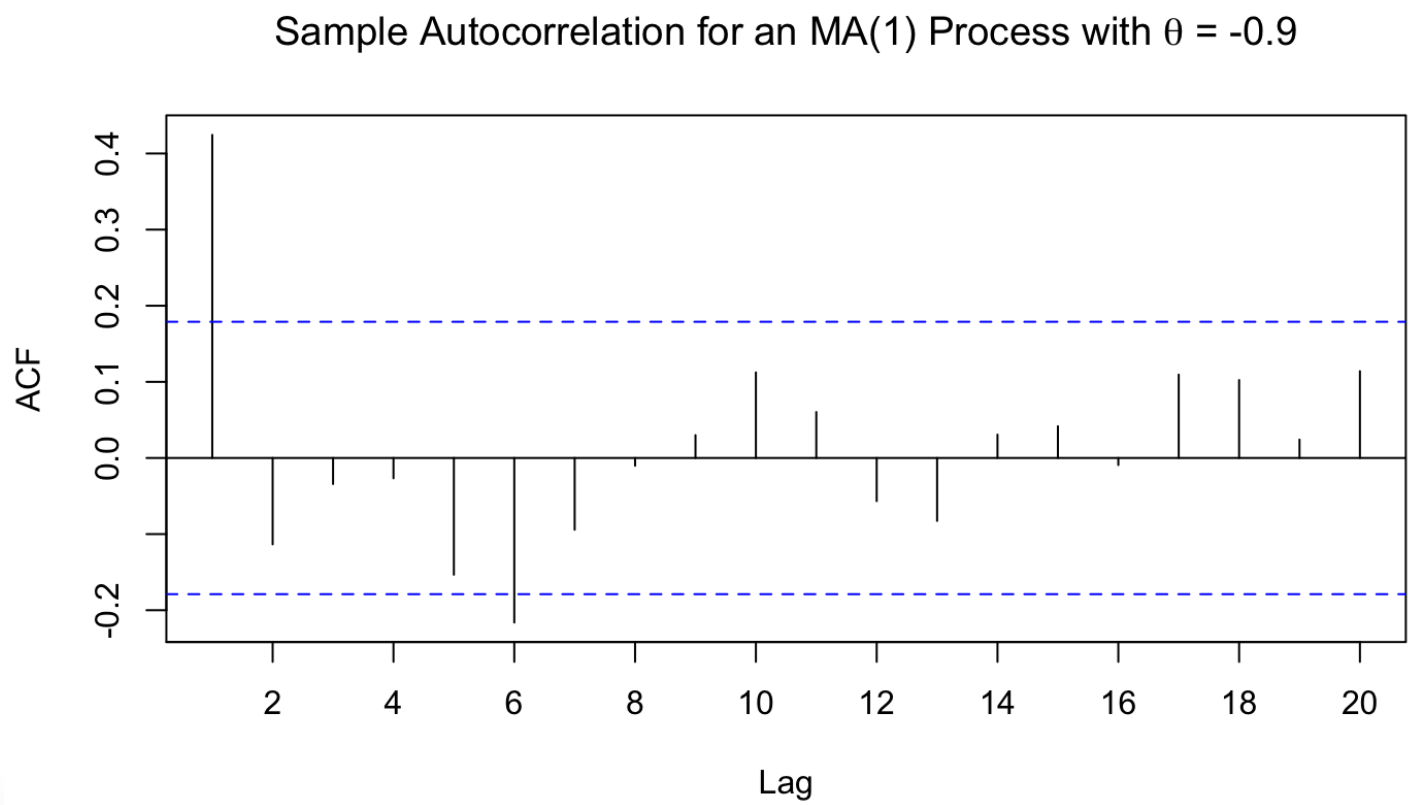
With these alternative bounds, the sample ACF value at lag 14 is insignificant as expected and the one at lag 5 is just barely significant.

> Because the lag 1 autocorrelation is still highly significant and the rest of the sample autocorrelations are insignificant, the information given in these two plots taken together leads us to consider an MA(1) model for this series.

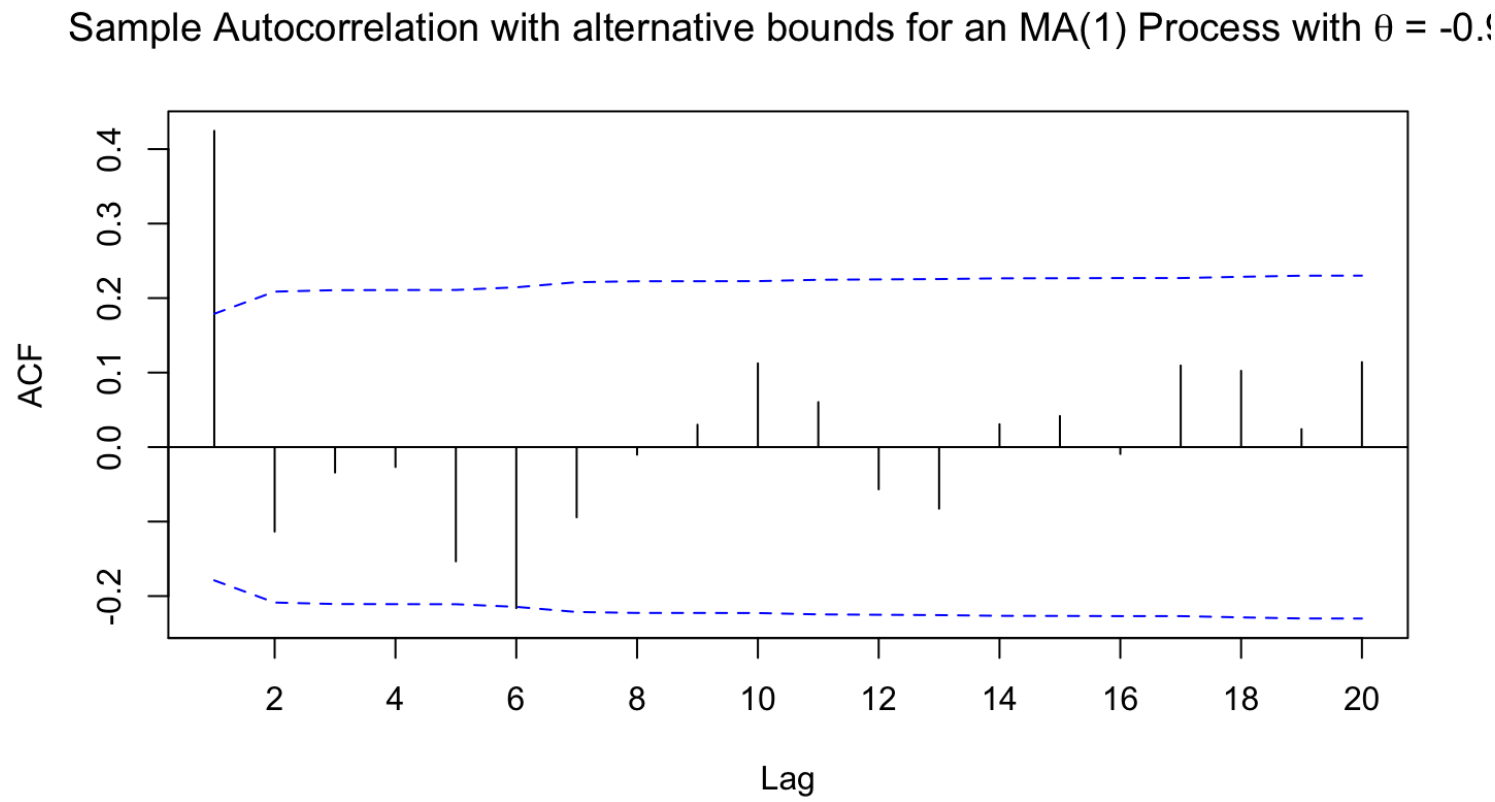
Notice that the model identified here is still a tentative one.

As a second example, the following plot shows the ACF plot of a simulated MA(1) process with $\theta = -0.9$.

```
data(ma1.2.s)
acf(ma1.2.s,xaxp=c(0,20,10),main=expression(
  paste("Sample Autocorrelation for an MA(1) Process with ", theta, " = -0.9")))
```

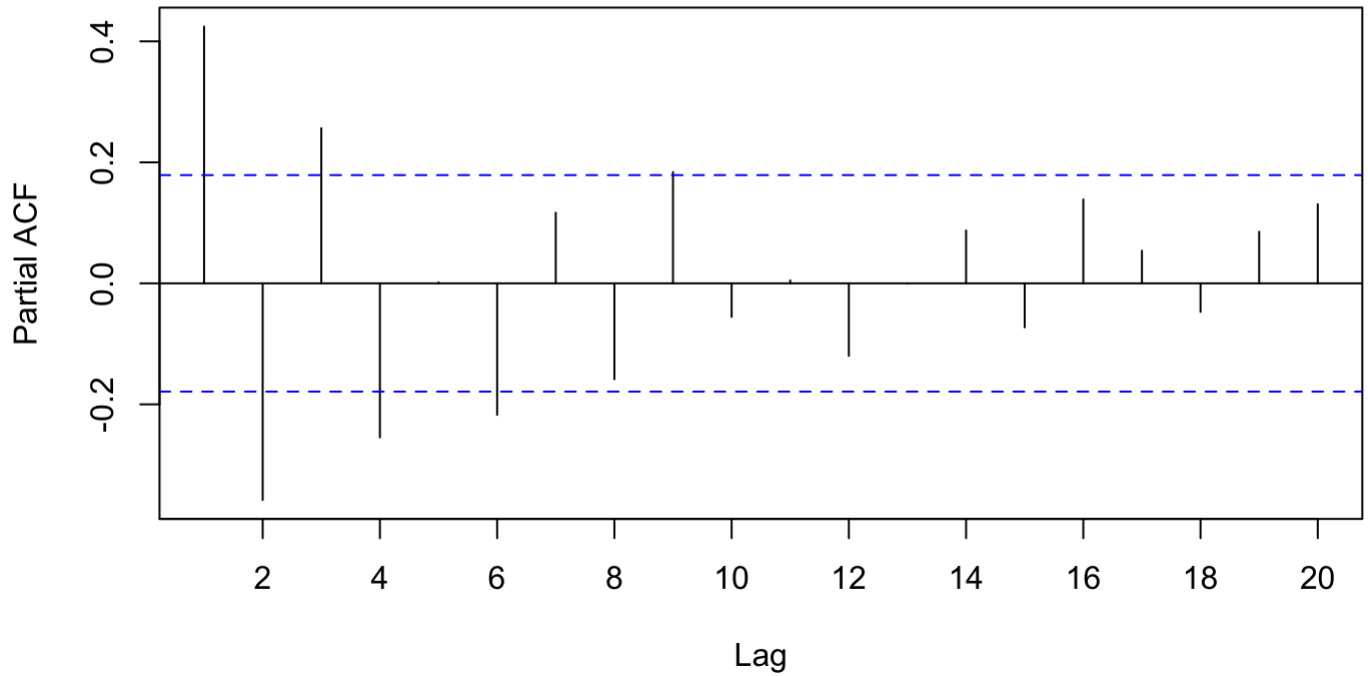


```
acf(ma1.2.s,ci.type='ma',xaxp=c(0,20,10),main=expression(
  paste("Sample Autocorrelation with alternative bounds for an MA(1) Process with ", theta
```



```
pacf(ma1.2.s,xaxp=c(0,20,10),main=expression(paste("Sample Partial Autocorrelation with al
```

Sample Partial Autocorrelation with alternative bounds for an MA(1) Process with $\theta$ =

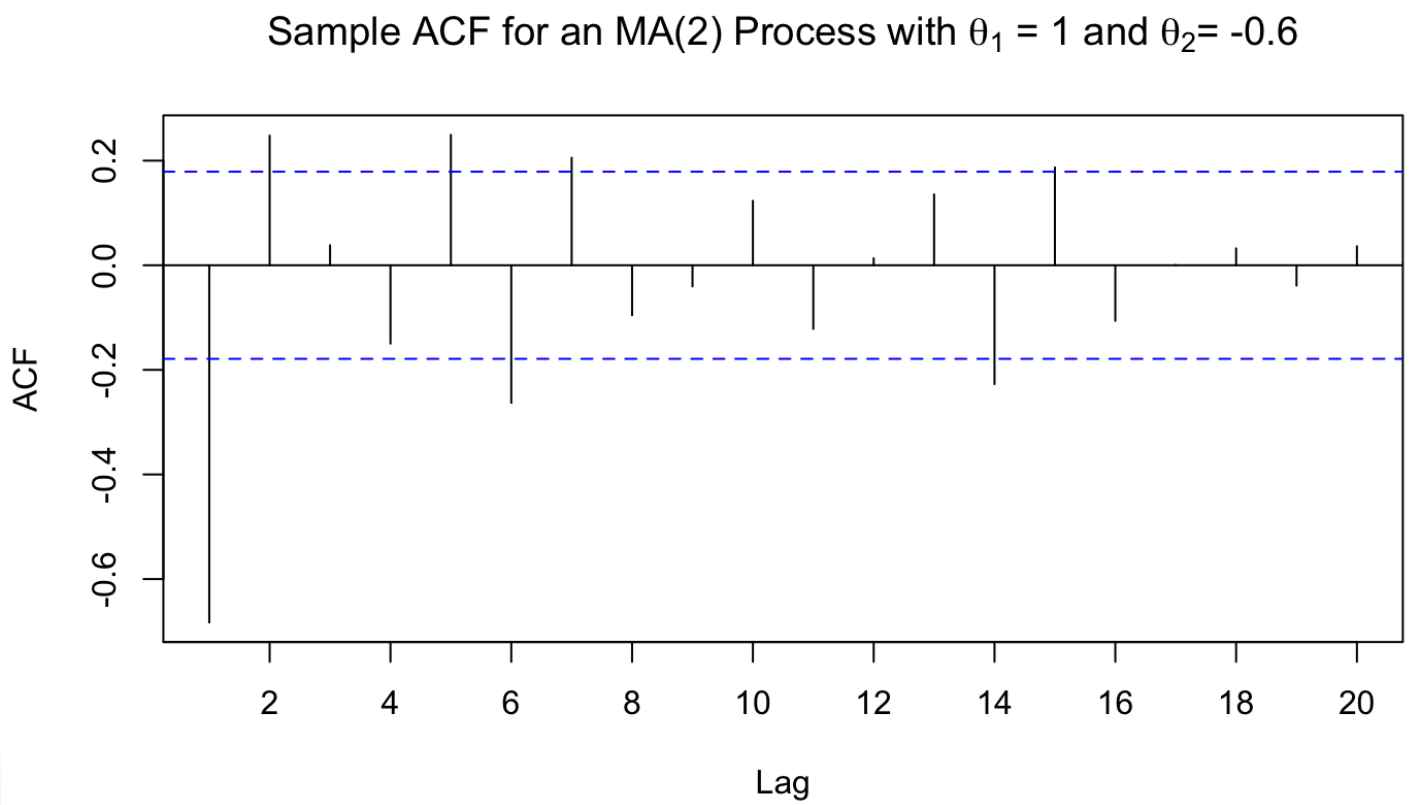


The critical values based on the very approximate standard errors point to an MA(1) model for this series also.

> Because the lag 1 autocorrelation is highly significant and the rest of the sample autocorrelations are (nearly) insignificant, we consider an MA(1) model for this series.

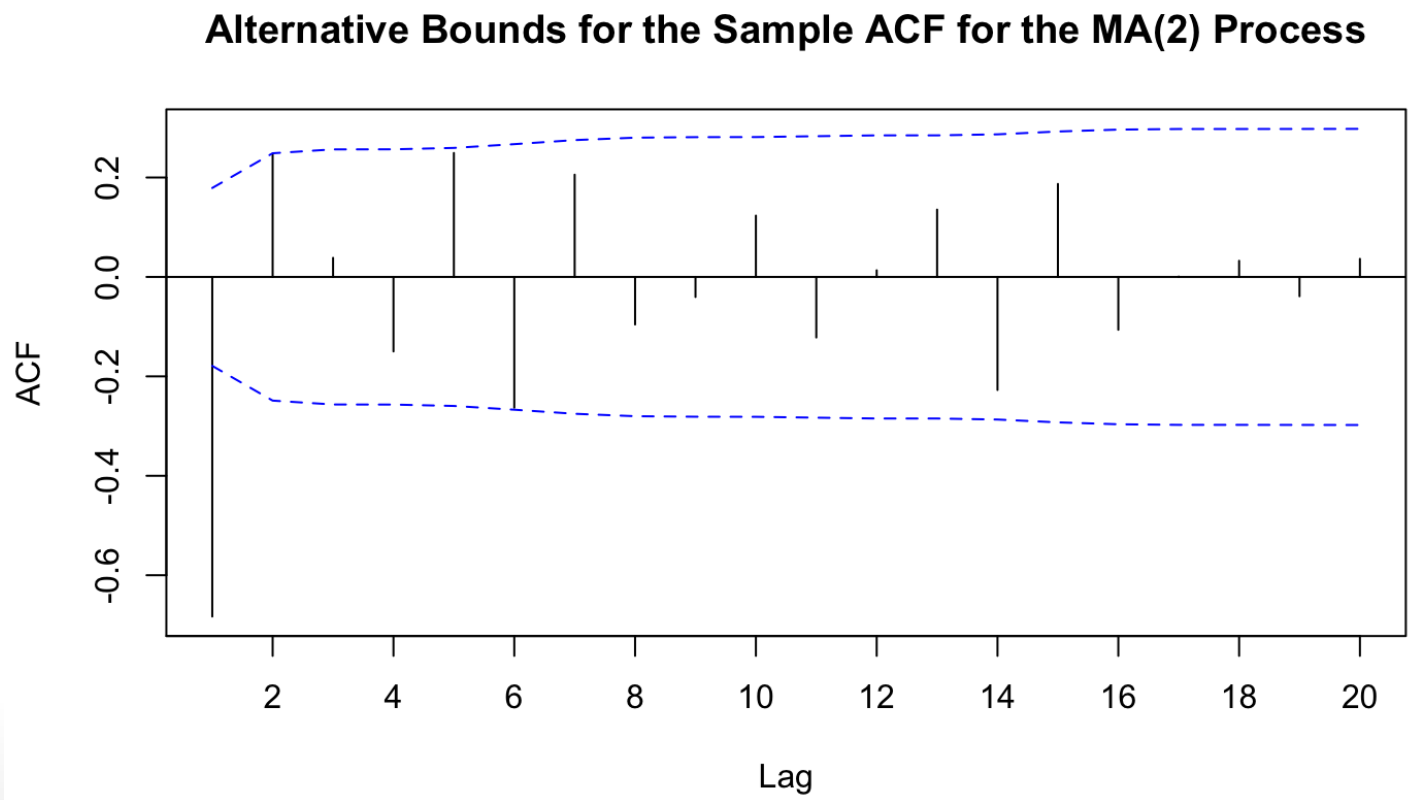
For the third example, the following plot shows the ACF plot of a series simulated from MA(2) process with $\theta_1 = 1$ and $\theta_2 = -0.6$.

```
data(ma2.s)
acf(ma2.s,xaxp=c(0,20,10),main=expression(paste(
  "Sample ACF for an MA(2) Process with ",theta[1] ," = 1 and ",
  theta[2], "= -0.6")))
```

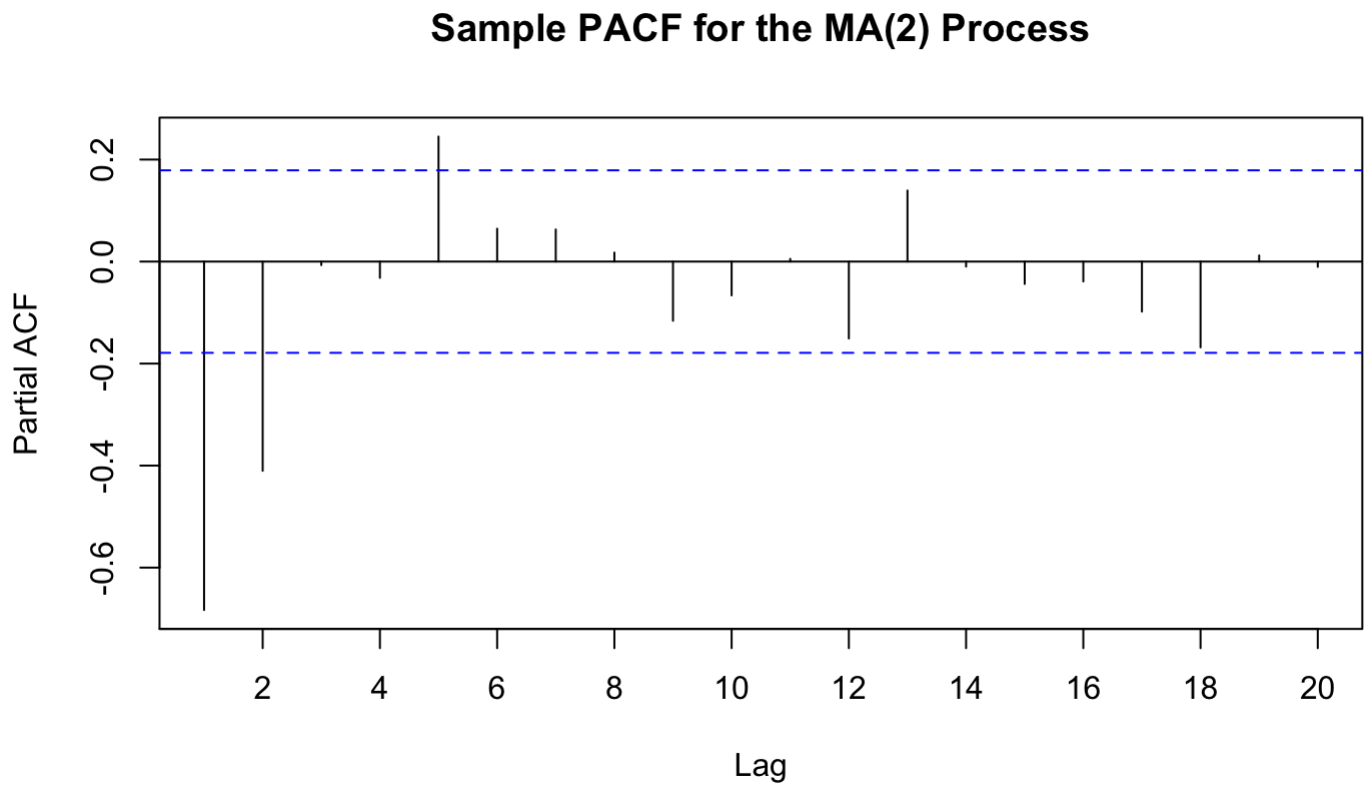


When we use simple standard errors, we observe significant autocorrelations at lags 1, 2, 5, 7, and 14.

```
acf(ma2.s,ci.type='ma',xaxp=c(0,20,10), main="Alternative Bounds for the Sample ACF for th
```



```
pacf(ma2.s,xaxp=c(0,20,10), main="Sample PACF for the MA(2) Process")
```

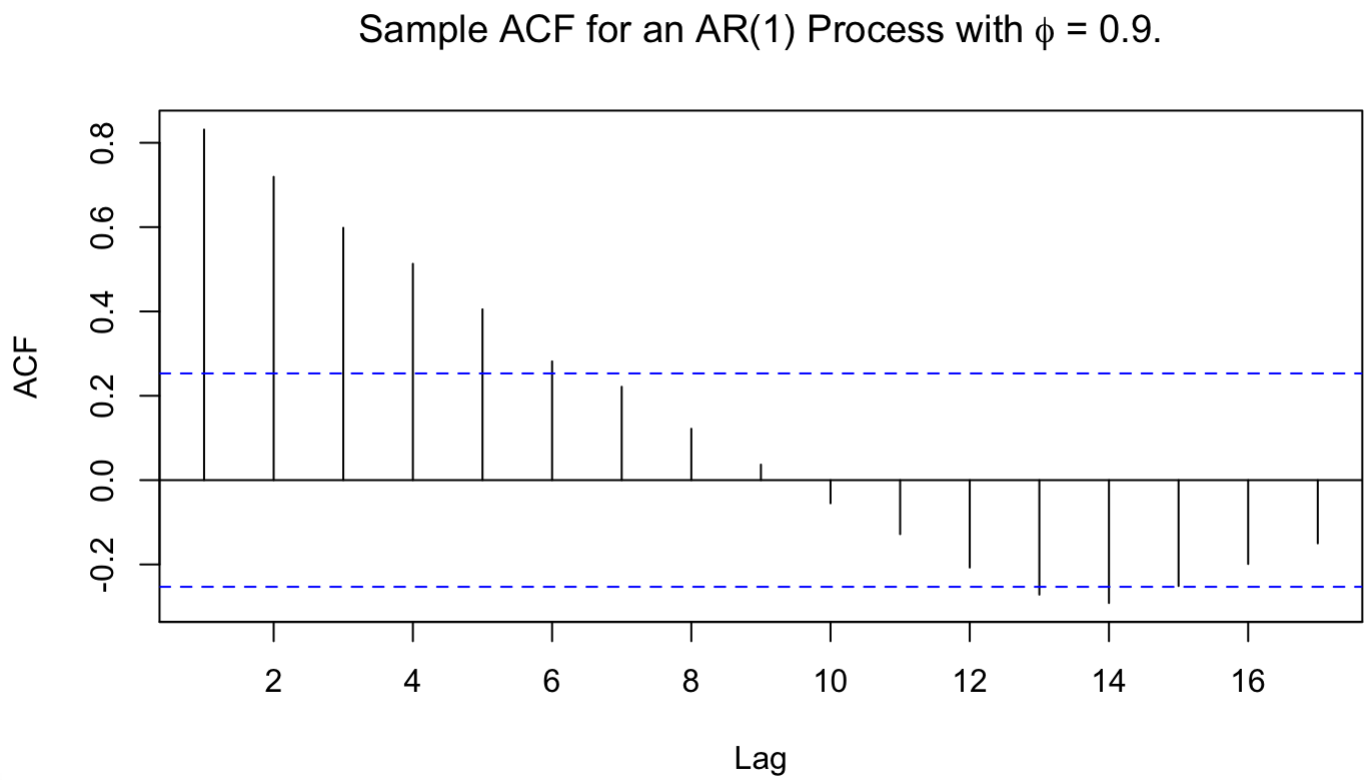


In ACF plot, the lag 2 ACF is slightly great than the critical value implying that it is not significant, and it appears that an MA(1) may be applicable.

We will have to wait until we get further along in the model-building process to see that the MA(2) model the correct one is the most appropriate model for these data.

The following plot shows the sample ACF of an AR(1) process simulated with $\phi = 0.9$.

```
data(ar1.s)
acf(ar1.s,xaxp=c(0,20,10),main=expression(paste(
  "Sample ACF for an AR(1) Process with ", phi, " = 0.9.")))
```

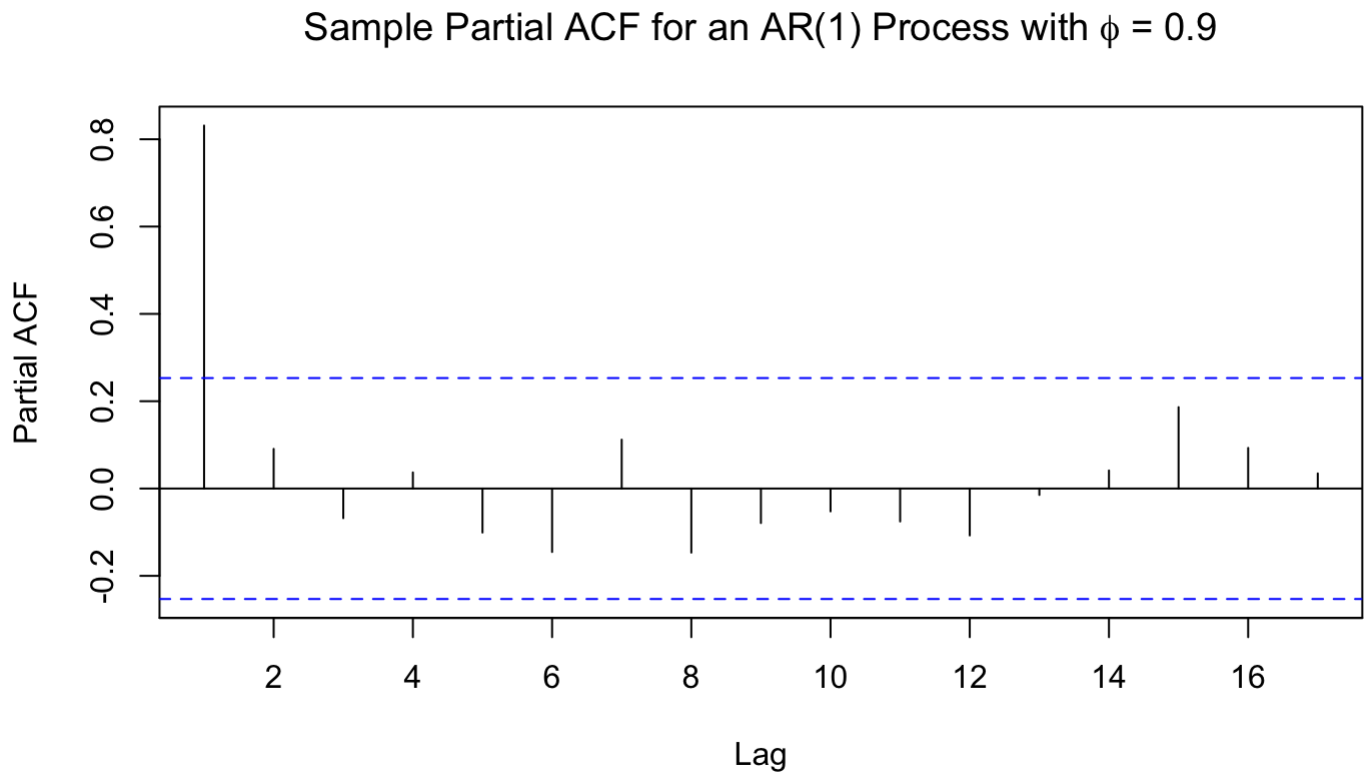


The positive sample ACF values at lags 1, 2, and 3 reflect the strength of the lagged relationships. The sample ACF decreases more linearly than exponentially as theory suggests.

Also contrary to theory, the sample ACF goes negative at lag 10 and remains so for many lags.

For the AR(p) processes, we also need to take a look at sample PACF plot to identify the order of the AR(p) process.

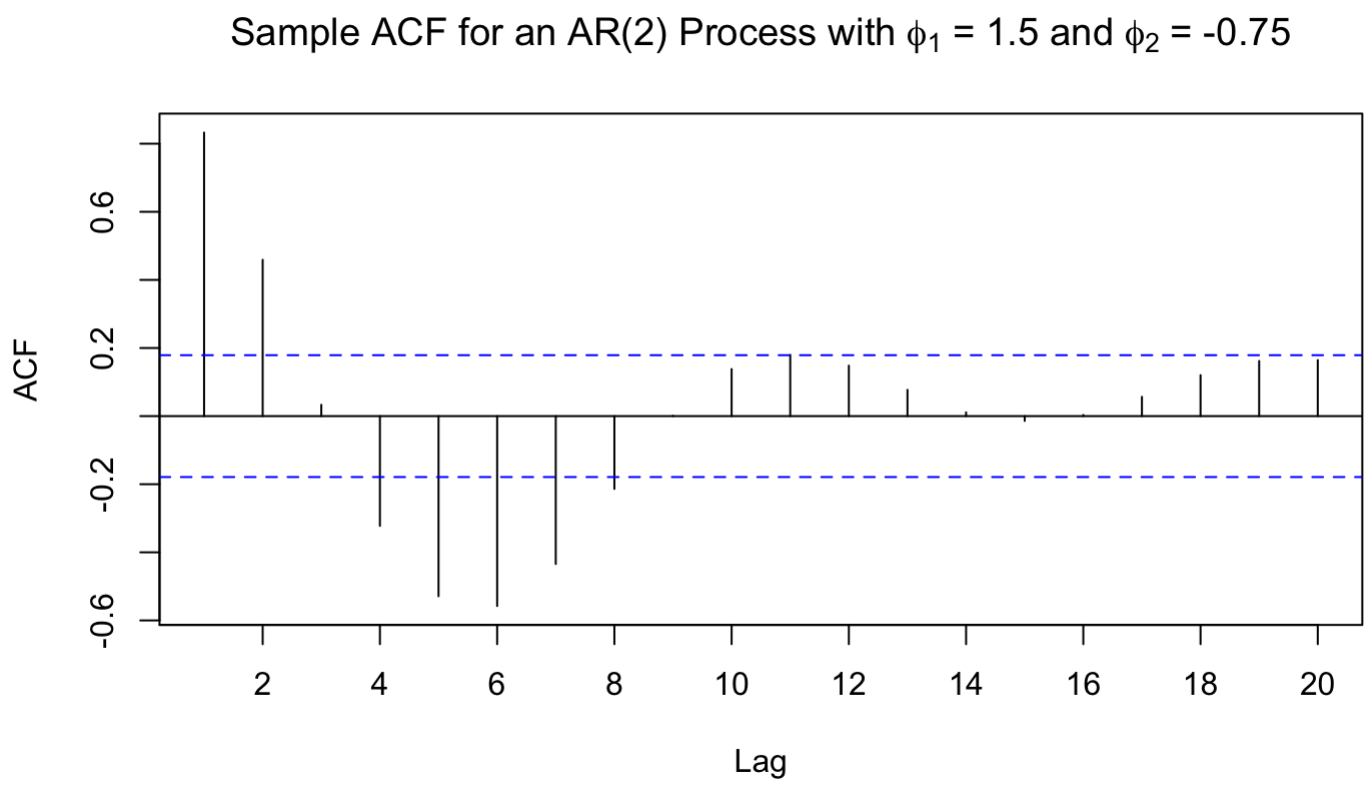
```
pacf(ar1.s,xaxp=c(0,20,10),main=expression(paste(
  "Sample Partial ACF for an AR(1) Process with ", phi, " = 0.9")))
```



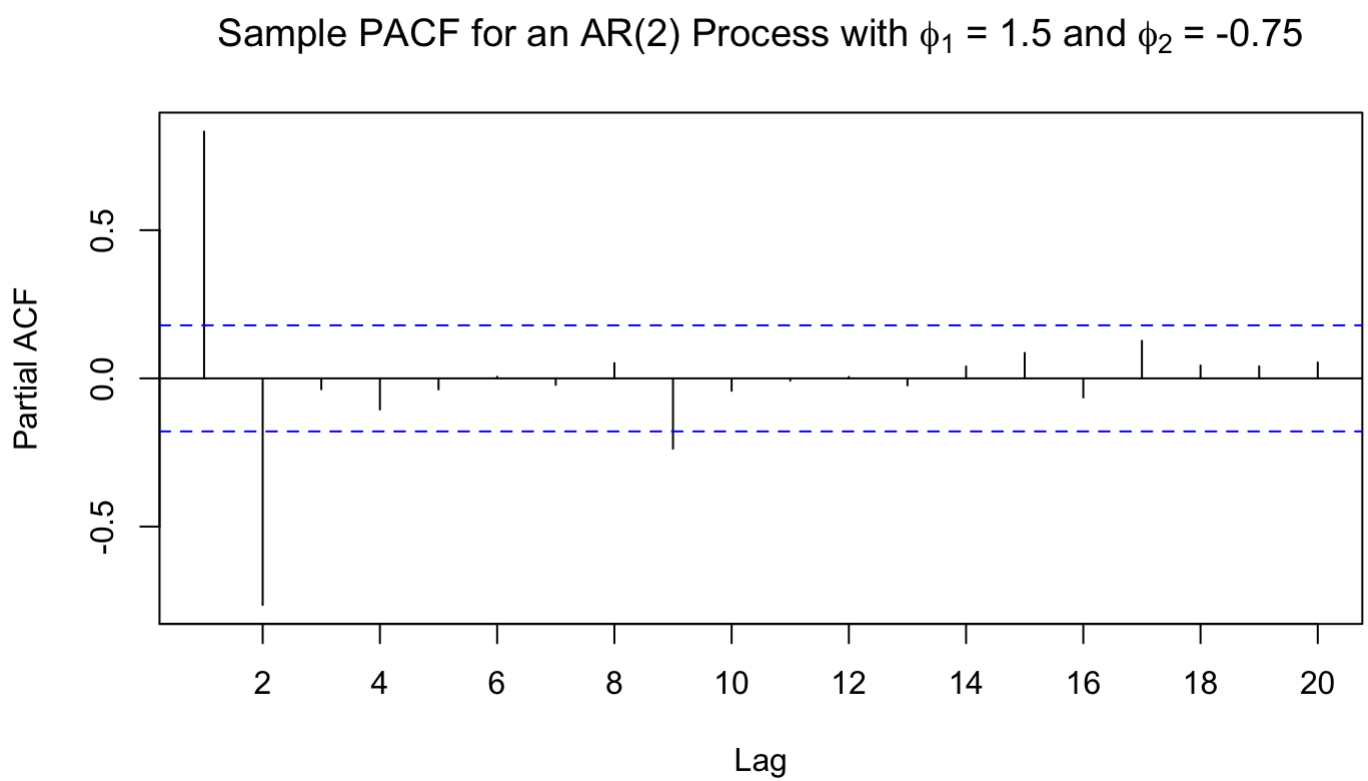
Now, sample PACF gives a clear picture of the AR(1) process with a significant partial autocorrelation at lag 1 and insignificant ones at the rest of lags.

The following plots display the sample ACF and PACF for a simulated AR(2) series.

```
data(ar2.s)
acf(ar2.s,xaxp=c(0,20,10),main=expression(paste("Sample ACF for an AR(2) Process with ", p
```



```
pacf(ar2.s,xaxp=c(0,20,10),main=expression(paste("Sample PACF for an AR(2) Process with ",
```

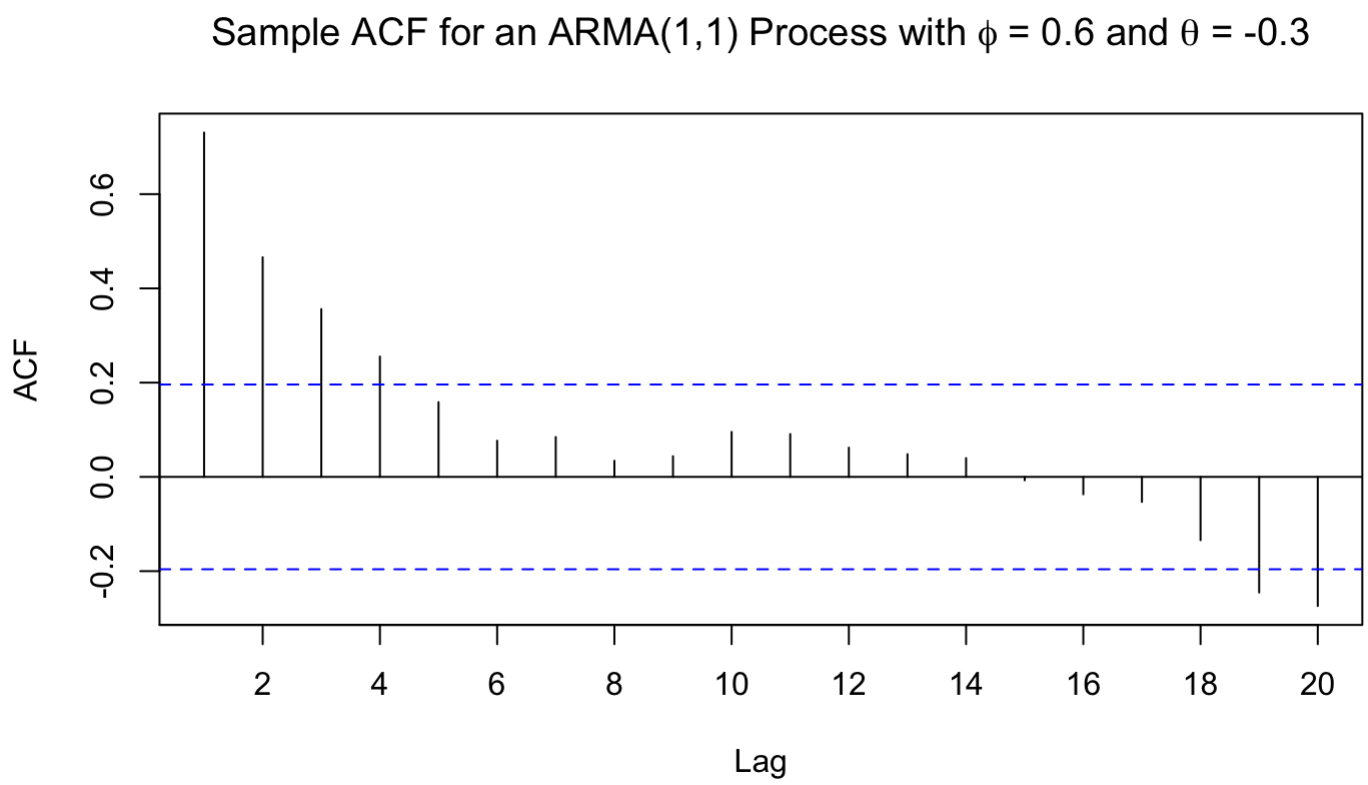


We have a damped sinusoidal wave in ACF and partial autocorrelations at the first two lags are highly significant.

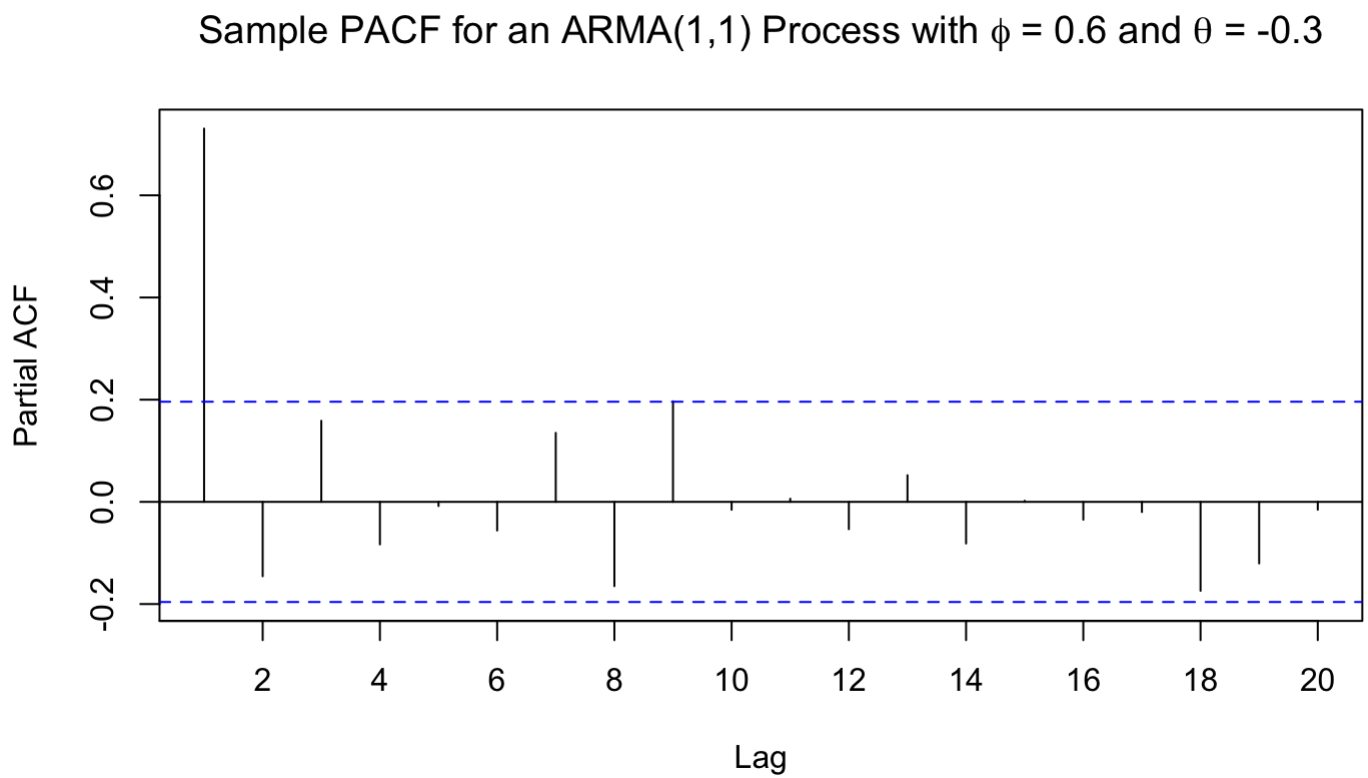
The nearly significant partial autocorrelation at lag 9 would need to be investigated further during model diagnostics.

The following plots display the sample ACF and PACF for a simulated ARMA(1,1) series.

```
data(arma11.s)
acf(arma11.s,xaxp=c(0,20,10),main=expression(paste("Sample ACF for an ARMA(1,1) Process wi
```



```
pacf(arma11.s,xaxp=c(0,20,10),main=expression(paste("Sample PACF for an ARMA(1,1) Process
```



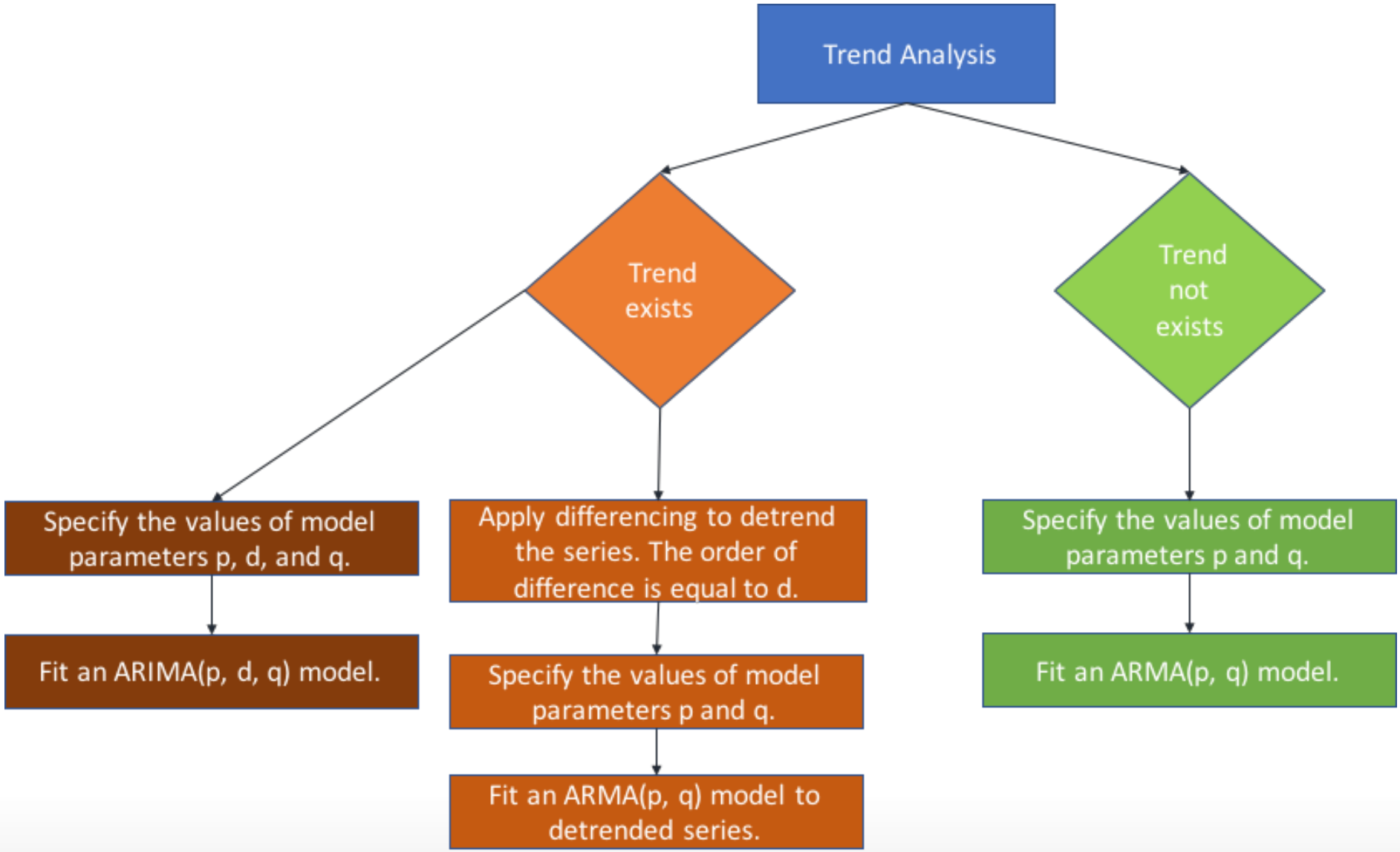
The decay in ACF and highly significant partial autocorrelation at lag 1 suggests an AR(1) model for this series. Then, we can consider EACF to make a further inference.

```
eacf(arma11.s) # Sample EACF of the simulated ARMA(1,1) process.
```

```
## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 x x x x o o o o o o o  o  o  o
## 1 x o o o o o o o o o o  o  o  o
## 2 x o o o o o o o o o o  o  o  o
## 3 x x o o o o o o o o o  o  o  o
## 4 x o x o o o o o o o o  o  o  o
## 5 x o o o o o o o o o o  o  o  o
## 6 x o o o x o o o o o o  o  o  o
## 7 x o o o x o o o o o o  o  o  o
```

It is inferred from the EACF that while clearly $q = 1$, we are not quite sure about whether $p = 1$ or $p = 2$.

# Nonstationarity

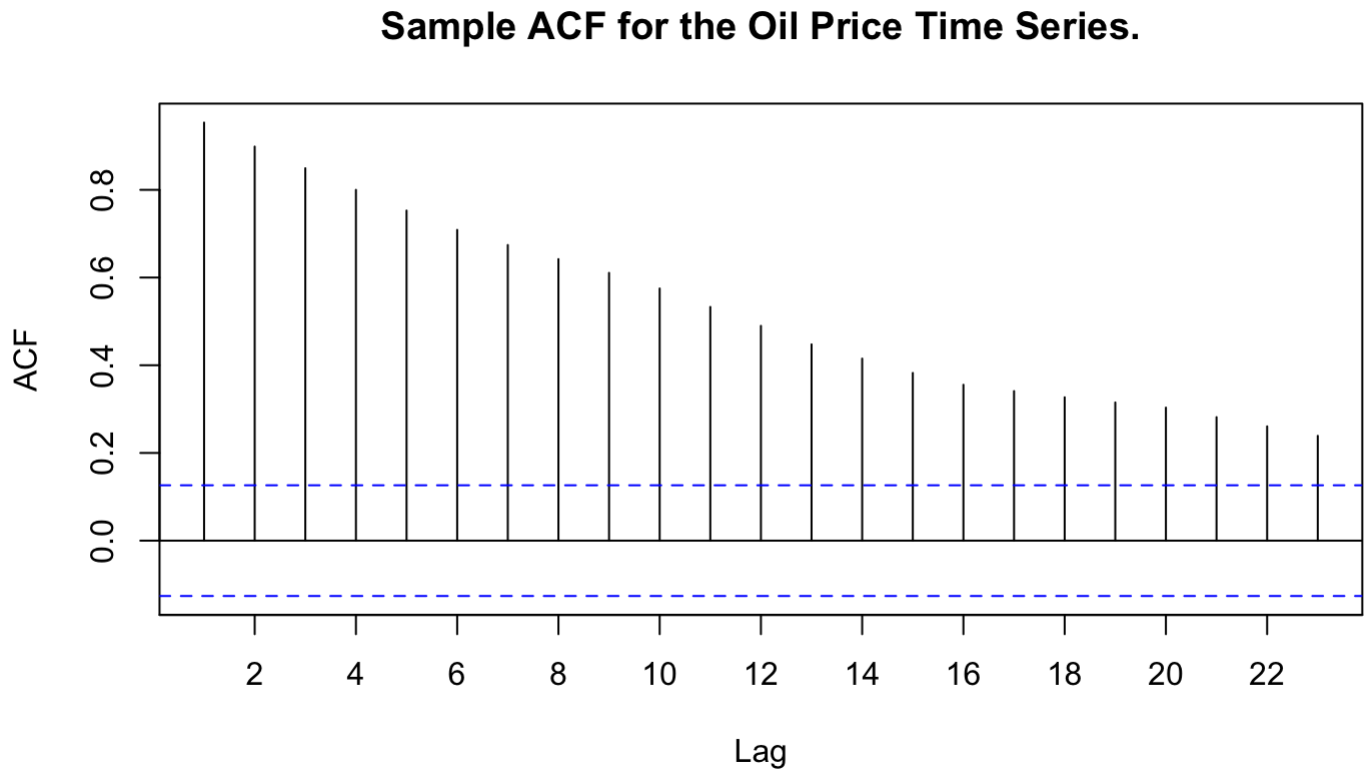

Trend Analysis
Trend exists
Trend not exists
Specify the values of model parameters p, d, and q.
Fit an ARIMA(p, d, q) model.
Apply differencing to detrend the series. The order of difference is equal to d.
Specify the values of model parameters p and q.
Fit an ARMA(p, q) model to detrended series.
Specify the values of model parameters p and q.
Fit an ARMA(p, q) model.

In addition to time series plots, the sample ACF also indicates nonstationarity by failing to die out rapidly as the lags increase.

This is due to the tendency for nonstationary series to drift slowly, either up or down, with apparent “trends.”

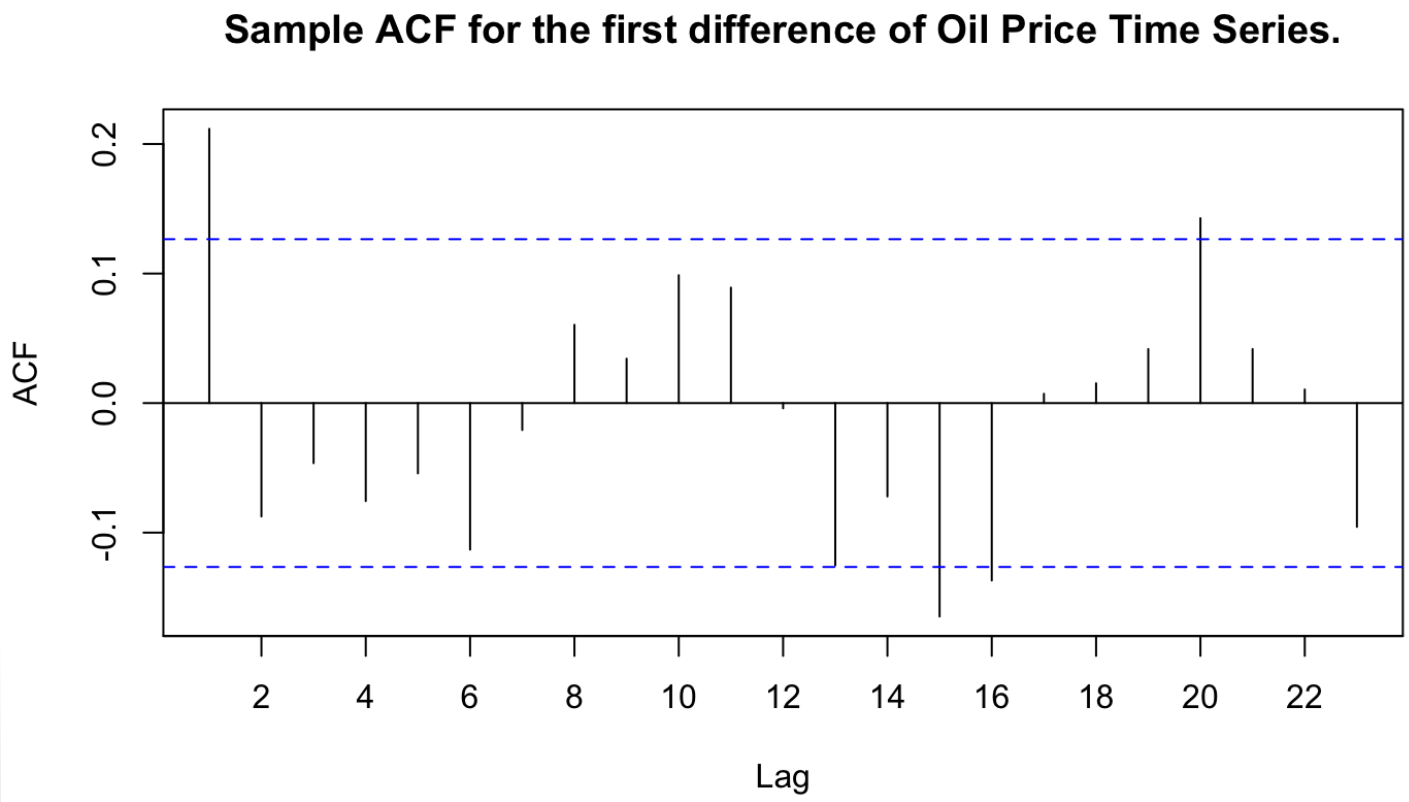
The sample ACF plot of the logarithms of oil prices series is shown below. There is a slow linear decrease with increasing lag.

```
data(oil.price)
acf(as.vector(log(oil.price)),xaxp=c(0,24,12), main="Sample ACF for the Oil Price Time Ser
```

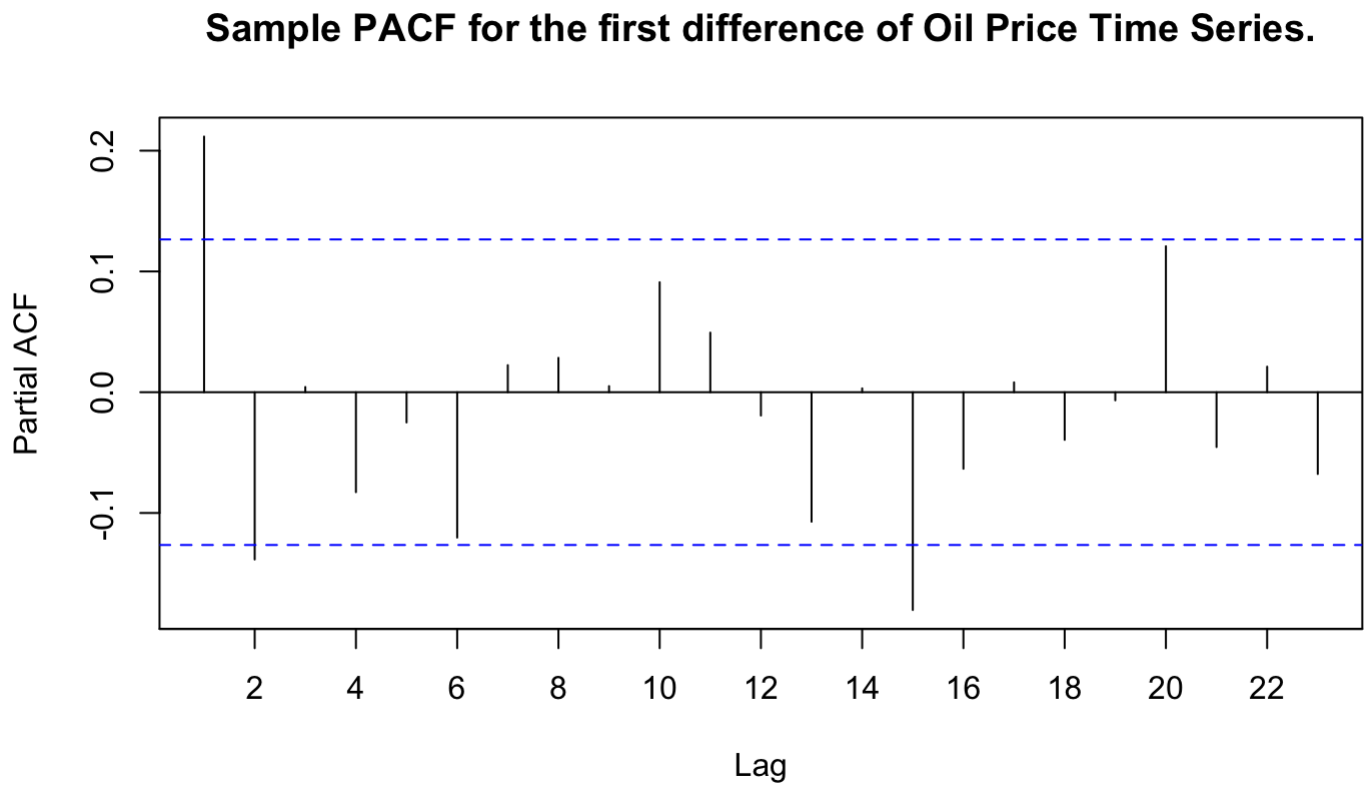


Although we get a more sensible ACF picture, after taking the first difference of the series, there is no clear sense in the PACF that we have significant partial autocorrelations at lags 2 and 15.

```
acf(diff(as.vector(log(oil.price))),xaxp=c(0,24,12), main="Sample ACF for the first differ
```



```
pacf(diff(as.vector(log(oil.price))),xaxp=c(0,24,12), main="Sample PACF for the first diff
```



When we display EACF of the oil price series, we can say that the IMA(1,1) model would be suitable for this series.

```
eacf(diff(as.vector(log(oil.price)))) # Sample EACF of the oil price series.
```

```
## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 x o o o o o o o o o o  o  o  o
## 1 x x o o o o o o o o x  o  o  o
## 2 o x o o o o o o o o o  o  o  o
## 3 o x o o o o o o o o o  o  o  o
## 4 o x x o o o o o o o o  o  o  o
## 5 o x o x o o o o o o o  o  o  o
## 6 o x o x o o o o o o o  o  o  o
## 7 x x o x o o o o o o o  o  o  o
```

# Overdifferencing

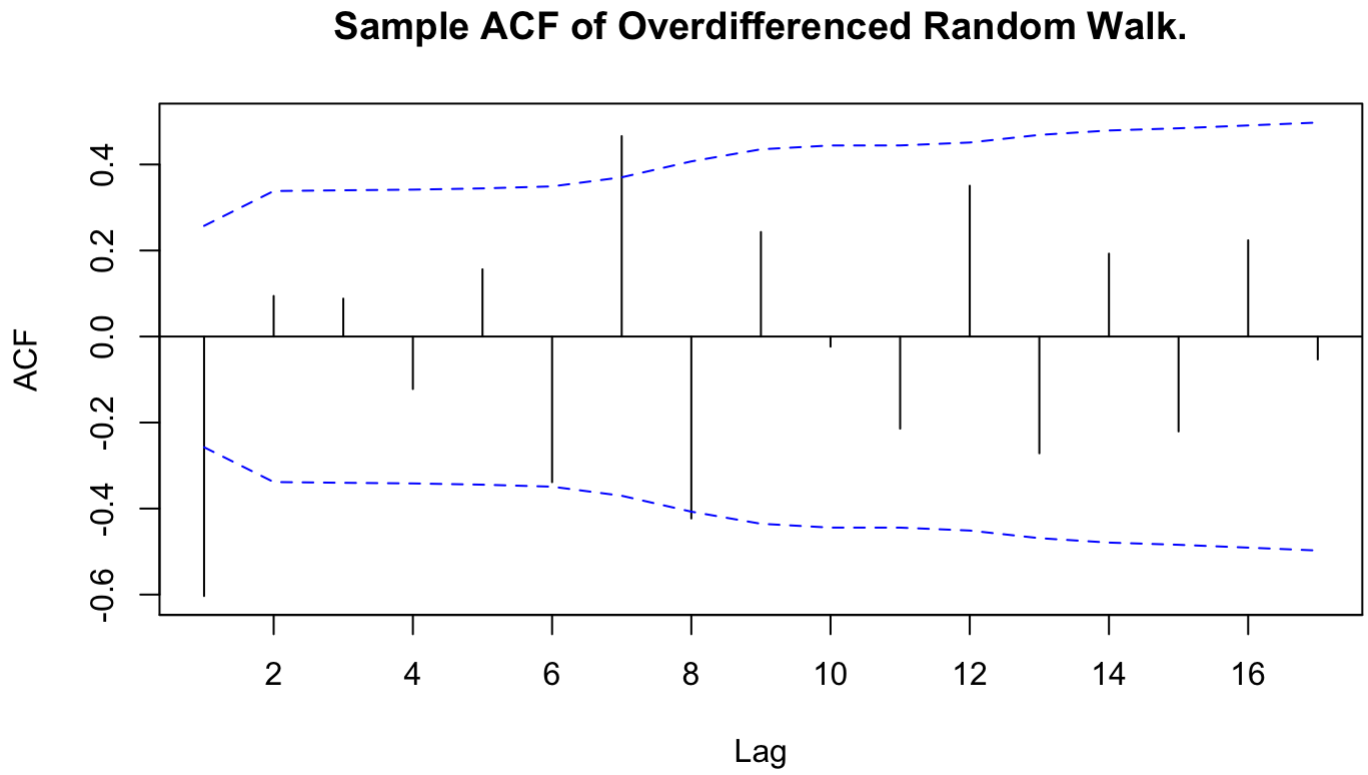
Although the difference of any stationary time series is also stationary, overdifferencing introduces unnecessary correlations into a series and the modeling process unnecessarily gets complicated.

Overdifferencing also creates a noninvertible model that create serious problems when we attempt to estimate their parameters.

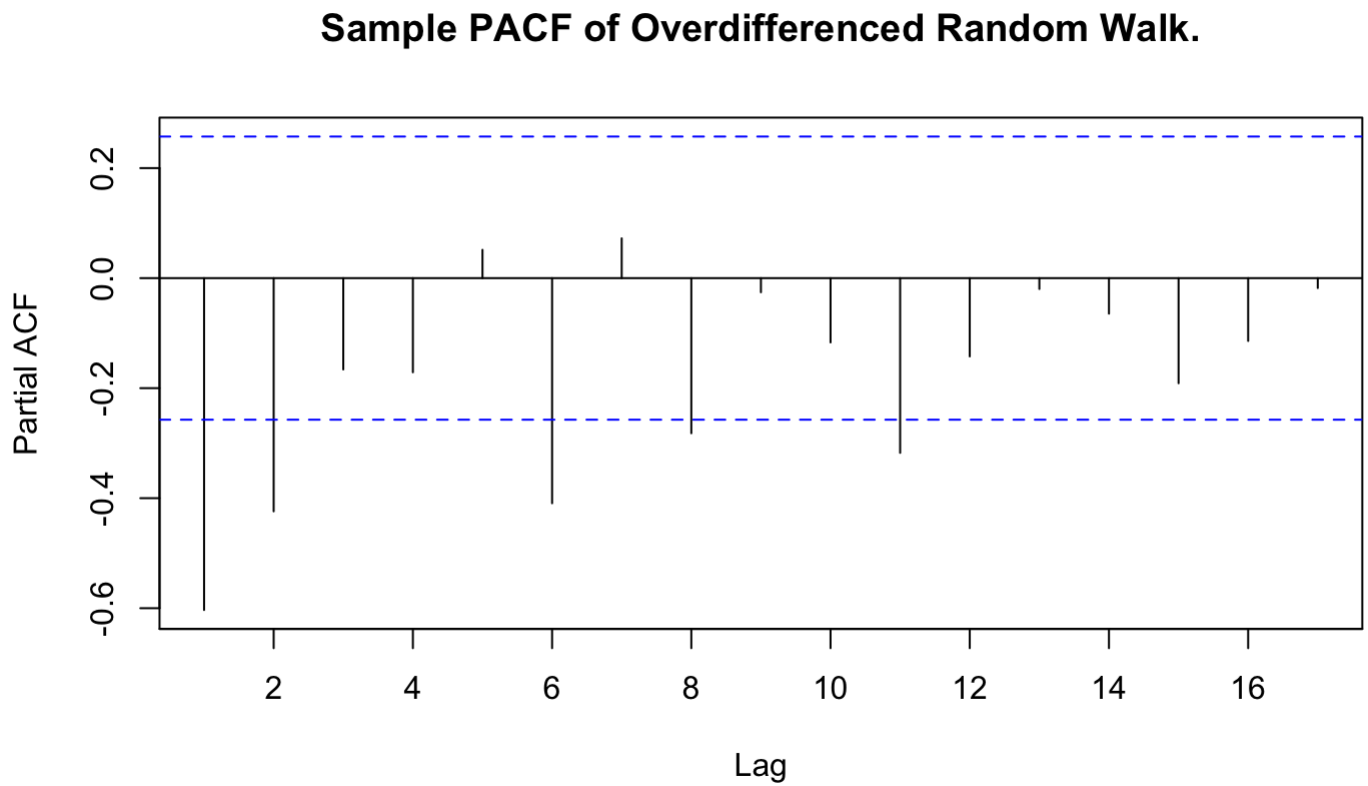
When we take two differences of random walk series, we get the following ACF plot. If we have taken only the first difference, we would get a white noise series.

Now, we can specify an IMA(2,1) model. We also have a significant sample ACF value at lag 7 to think about and deal with.

```
data(rwalk)
acf(diff(rwalk,difference=2),ci.type='ma', xaxp=c(0,18,9),main="Sample ACF of Overdifferen
```

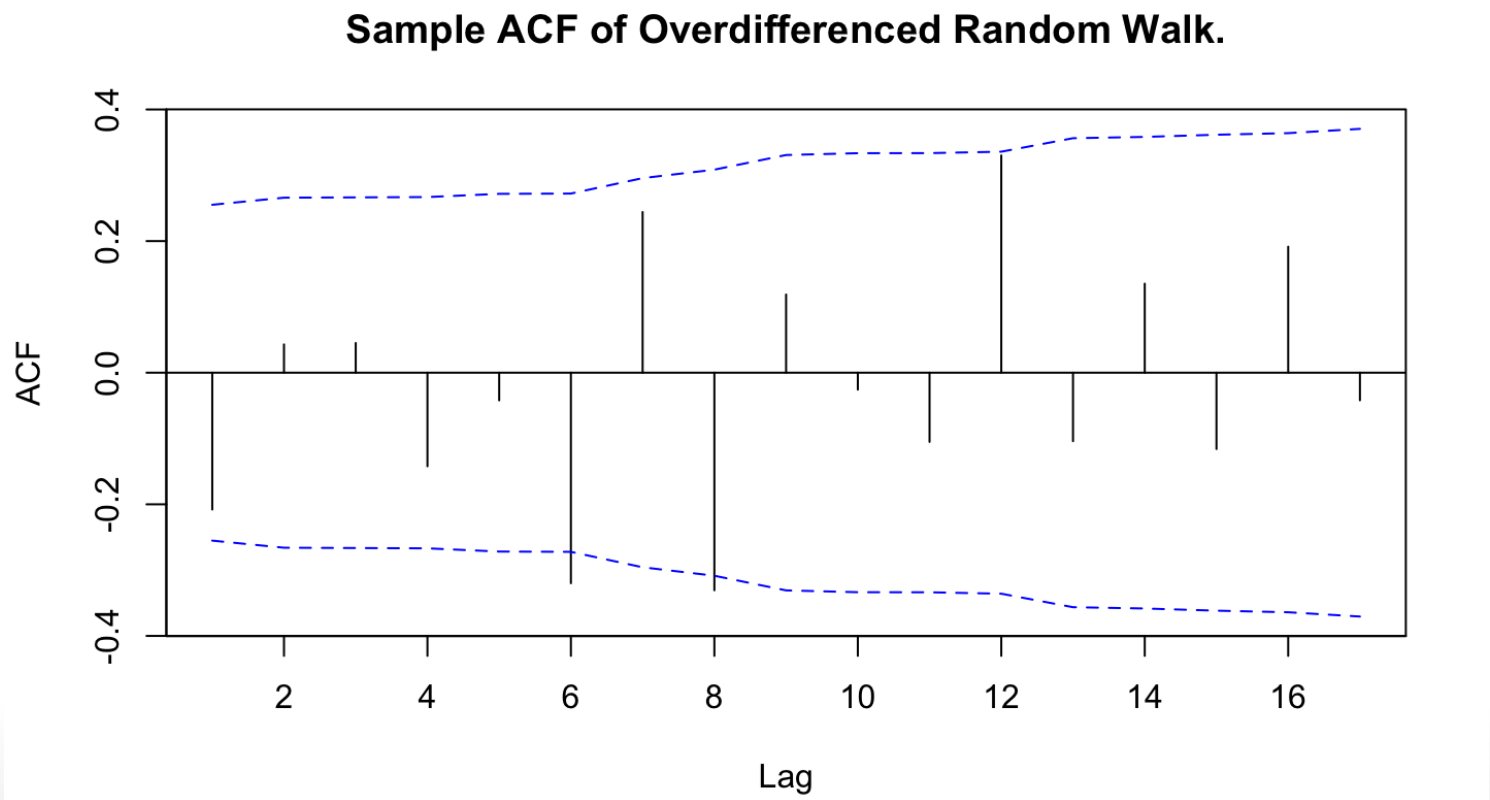


```
pacf(diff(rwalk,difference=2), xaxp=c(0,18,9),main="Sample PACF of Overdifferenced Random
```

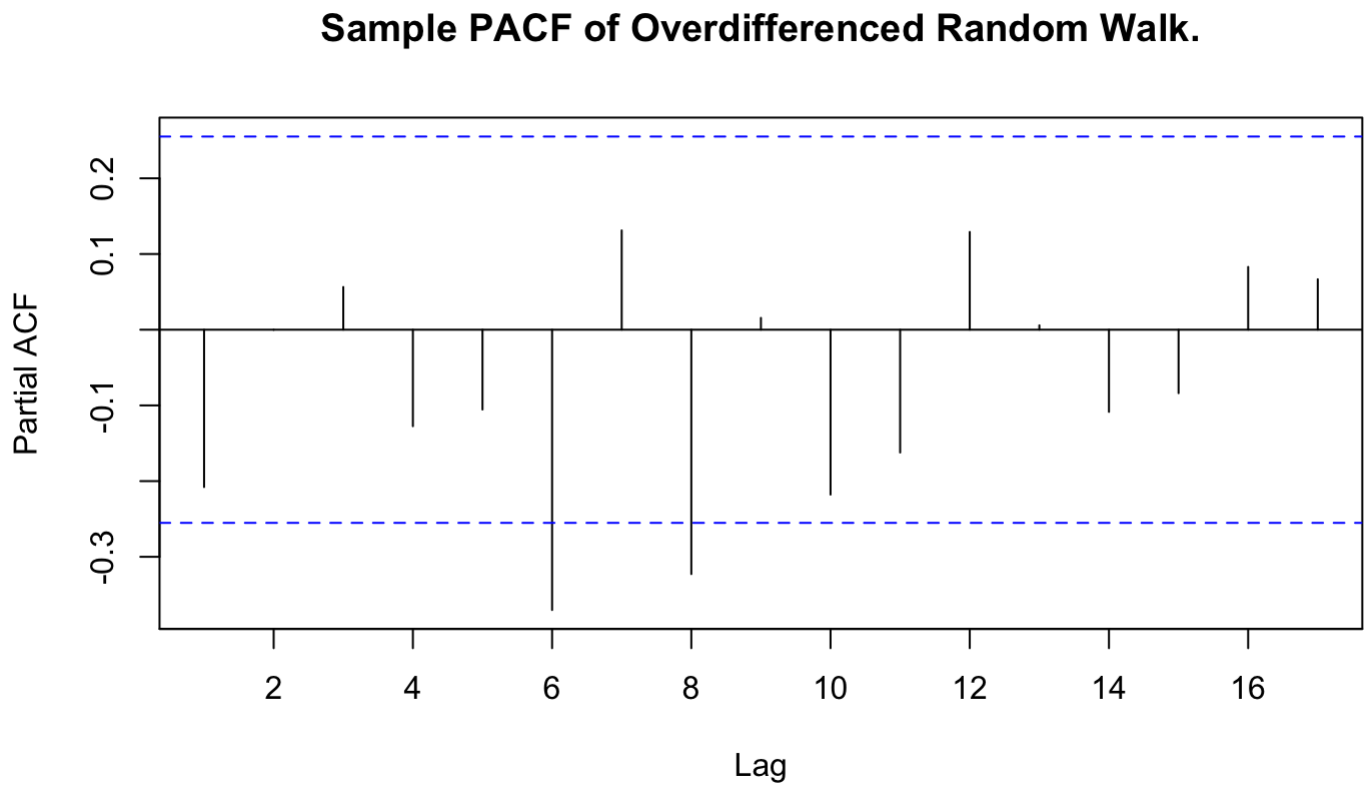


In contrast, the sample ACF of the first difference of the random walk series, shown below, exhibits a white noise pattern.

```
acf(diff(rwalk),ci.type='ma', xaxp=c(0,18,9),main="Sample ACF of Overdifferenced Random Wa
```



```
pacf(diff(rwalk), xaxp=c(0,18,9),main="Sample PACF of Overdifferenced Random Walk.")
```



To avoid overdifferencing, we recommend looking carefully at each difference in succession and keeping the principle of parsimony always in mind- **models should be simple, but not too simple**.

# The Dickey-Fuller Unit-Root Test

The Dickey-Fuller unit-root test is used to test the null hypothesis that **the process is difference nonstationary** (the process is nonstationary but becomes stationary after first differencing).

This is equivalent to stating that the AR characteristic polynomial has a unit root.

The alternative hypothesis is that the process is stationary.

The unit root hypothesis can be tested by regressing the first difference of the observed time series on lag 1 of the observed series and on the past $k$ lags of the first difference of the observed series.

This can be done by carrying out the ADF test with the detrended data.

In some cases, the process may be trend nonstationary in the sense that it has a deterministic trend (for example, some linear trend) but otherwise is stationary.

A unit-root test may be conducted with the aim of discerning difference stationarity from trend stationarity.

**Trend stationary**: The mean trend is deterministic. If you remove the trend by fitting a model to series, the residuals will be stationary.

**Difference stationary**: The mean trend is stochastic. If you take differences of the series, it will yield a stationary series.

Time series with a deterministic trend always revert to the trend in the long run. Forecast intervals have a constant width.

Time series with a stochastic trend never recover from shocks to the system. Forecast intervals grow over time. Source: Matlab 2017a Documentation

In practice, the process may not be a finite-order AR process after differencing, but it may be closely approximated by some AR process with the AR order increasing with the sample size.

To carry out the test with the random walk data, we must determine $k$ using the following code:

```
ar(diff(rwalk))
```

```
##
## Call:
## ar(x = diff(rwalk))
##
## Coefficients:
##       1         2         3         4         5         6         7         8
## -0.1622  -0.1234   0.0036  -0.1961  -0.1597  -0.3418   0.0652  -0.3228
##
## Order selected 8  sigma^2 estimated as  0.8394
```

We found that $k = 8$, in which case the ADF test with no intercept (constant) nor time trend is implemented with the following code chunk:

```
library(fUnitRoots)
adfTest(rwalk, lags = 8, type = "nc", title = NULL,description = NULL)
```

```
##
## Title:
##  Augmented Dickey-Fuller Test
##
## Test Results:
##   PARAMETER:
##     Lag Order: 8
##   STATISTIC:
##     Dickey-Fuller: 1.1975
##   P VALUE:
##     0.9363
##
## Description:
##  Tue Apr  2 09:10:56 2019 by user:
```

The ADF test with an intercept (constant) but no time trend is implemented with the following code chunk:

```
adfTest(rwalk, lags = 8, type = "c")
```

```
##
## Title:
##  Augmented Dickey-Fuller Test
##
## Test Results:
##   PARAMETER:
##     Lag Order: 8
##   STATISTIC:
##     Dickey-Fuller: -0.6006
##   P VALUE:
##     0.8281
##
## Description:
##  Tue Apr  2 09:10:56 2019 by user:
```

The ADF test with an intercept (constant) and a time trend is implemented with the following code chunk:

```
adfTest(rwalk, lags = 8, type = "ct")
```

```
##
## Title:
##  Augmented Dickey-Fuller Test
##
## Test Results:
##   PARAMETER:
##     Lag Order: 8
##   STATISTIC:
##     Dickey-Fuller: -2.2892
##   P VALUE:
##     0.4579
##
## Description:
##  Tue Apr  2 09:10:56 2019 by user:
```

The ADF test with an unknown true order, an intercept (constant) and a time trend implying that the process equals linear time trend plus stationary error is as follows:

```
adfTest(rwalk, lags = 0, type = "ct")
```

```
##
## Title:
##  Augmented Dickey-Fuller Test
##
## Test Results:
##   PARAMETER:
##     Lag Order: 0
##   STATISTIC:
##     Dickey-Fuller: -3.4904
##   P VALUE:
##     0.0501
##
## Description:
##  Tue Apr  2 09:10:56 2019 by user:
```

For all cases, we observed p-values greater than 0.1.

So, we cannot reject the null hypothesis stating the nonstationarity.

Another way of applying the ADF test is to use the package called `tseries`.

The `tseries` package contains the function

```
adf.test(x, alternative = c("stationary", "explosive"),
         k = trunc((length(x)-1)^(1/3)))
```

to implement the ADF test.

Notice here that the value of lag-length is found by the formula

`k = trunc((length(x)-1)^(1/3))`.

This is another general approach to find the value of lag-length in ADF test.

For the `rwalk` series, we get

```
trunc((length(rwalk)-1)^(1/3))
```

```
## [1] 3
```

as the value of lag-length, which was found as 8 with the first approach.

```
adf.test(rwalk, alternative = c("stationary")) # Omit k to use the default formula
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  rwalk
## Dickey-Fuller = -3.9699, Lag order = 3, p-value = 0.01701
## alternative hypothesis: stationary
```

Now, we got a considerably smaller p-value for the test!

So, the ADF test is very sensitive to the parameter “lag.”

We need to look at the most suitable lag to conclude about the whole series.

Therefore, there are several different approaches for the suitable specification of the value of “lag” in the literature, and each approach has its own drawbacks.

Actually, the value of "lag’ is related to the power of the test. If you set it to a higher value than the suitable one, the power of the test decreases.

If you set it too small, you will look at the series at an early lag and the remaining ones will affect the result.

# Other Specification Methods

Akaike's (1973) Information Criterion (AIC) is one of the most commonly used information criteria for model specification and selection.

AIC tends to select the model that minimizes

$$AIC = -2\log(\text{maximum likelihood}) + 2k,$$

where $k = p + q + 1$ if the model contains an intercept or constant term and $k = p + q$ otherwise.

The additive factor $2k$ serves as a "penalty function" for the addition of each term to help ensure selection of parsimonious models and to avoid choosing models with too many parameters.

Because AIC is a biased statistic, there are corrected versions of AIC in the literature.

One of those corrected versions Of AIC, namely $\text{AIC}_c$ is

$$AIC_c = AIC + \frac{2(k+1)(k+2)}{n-k-2}$$

where $n$ is the (effective) sample size and again $k$ is the total number of parameters as above excluding the noise variance.

Simulation results by Hurvich and Tsai (1989) suggest that for cases with $k/n > 0.1$, the $\text{AIC}_c$ outperforms many other model selection criteria, including both the AIC and BIC.

Another widely used criterion in model selection is **Bayesian Information Criterion (BIC)** or Schwartz's Bayesian Criterion given as

$$BIC = -2\log(\text{maximum likelihood}) + k\log(n).$$

If the true process follows an ARMA(p,q) model, then it is known that the orders Specified- by minimizing the BIC are consistent.

However, if the true process is not a finite-order ARMA process, then minimizing AIC among an increasingly large class of ARMA models enjoys the appealing property that it will lead to an optimal ARMA model that is closest to the true process among the class of models under study.

Regardless of whether we use the AIC or BIC, the methods require carrying out maximum likelihood estimation.

However, maximum likelihood estimation for an ARMA model is prone to numerical problems due to multimodality of the likelihood function and the problem of overfitting when the AR and MA orders exceed the true orders.

Hannan and Rissanen (1982) proposed an interesting and practical solution to this problem. In their approach, we

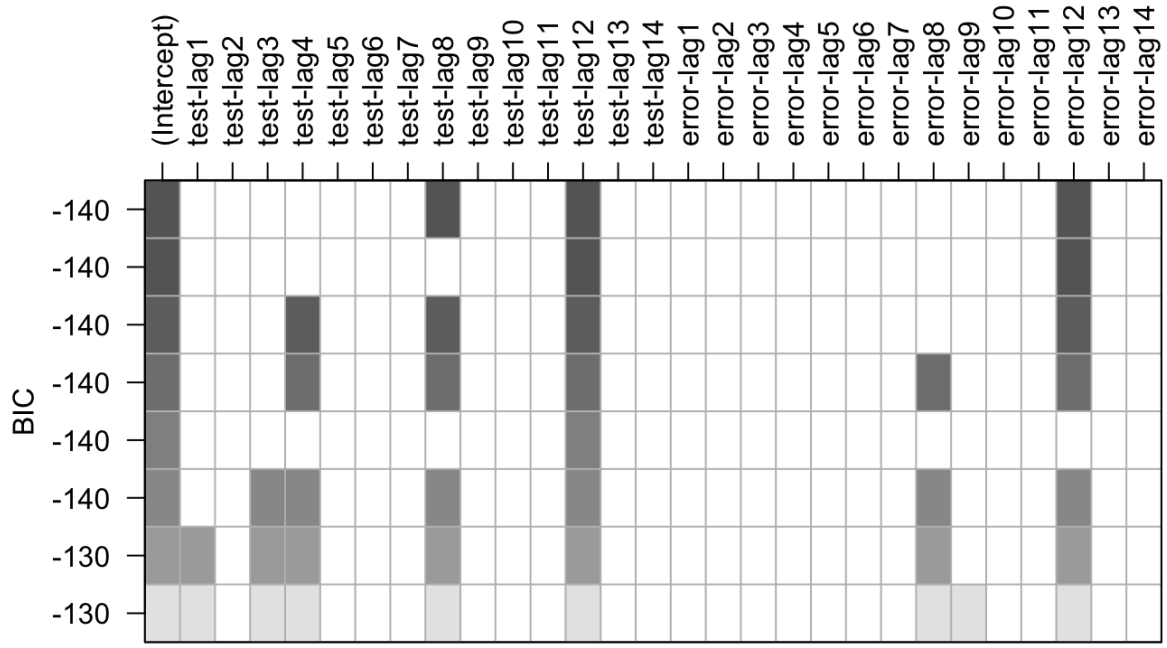
- fit a high-order AR process with the order determined by minimizing the AIC and
- use the residuals from the first step as proxies for the unobservable error terms.

Thus, an ARMA(k,j) model can be approximately estimated by regressing the time series on its own lags 1 to $k$ together with the lags 1 to $j$ of the residuals from the high order autoregression; the BIC of this autoregressive model is an estimate of the BIC obtained with maximum likelihood estimation.

It would be helpful to examine a few best subset ARMA models (in terms of, for example, BIC) in order to arrive at some helpful tentative models for further study.

The pattern of which lags of the observed time series and which of the error process enter into the various best subset models can be summarized in a display like that shown below.

```
set.seed(92397)
test = arima.sim(model=list(ar=c(rep(0,11),.8),ma=c(rep(0,11),0.7)),n=120)
res = armasubsets(y=test,nar=14,nma=14,y.name='test',ar.method='ols')
plot(res)
```



Each row in the exhibit corresponds to a subset ARMA model where the cells of the variables selected for the model are shaded.

The models are sorted according to their BIC, with better models (lower BIC) placed in higher rows and with darker shades.

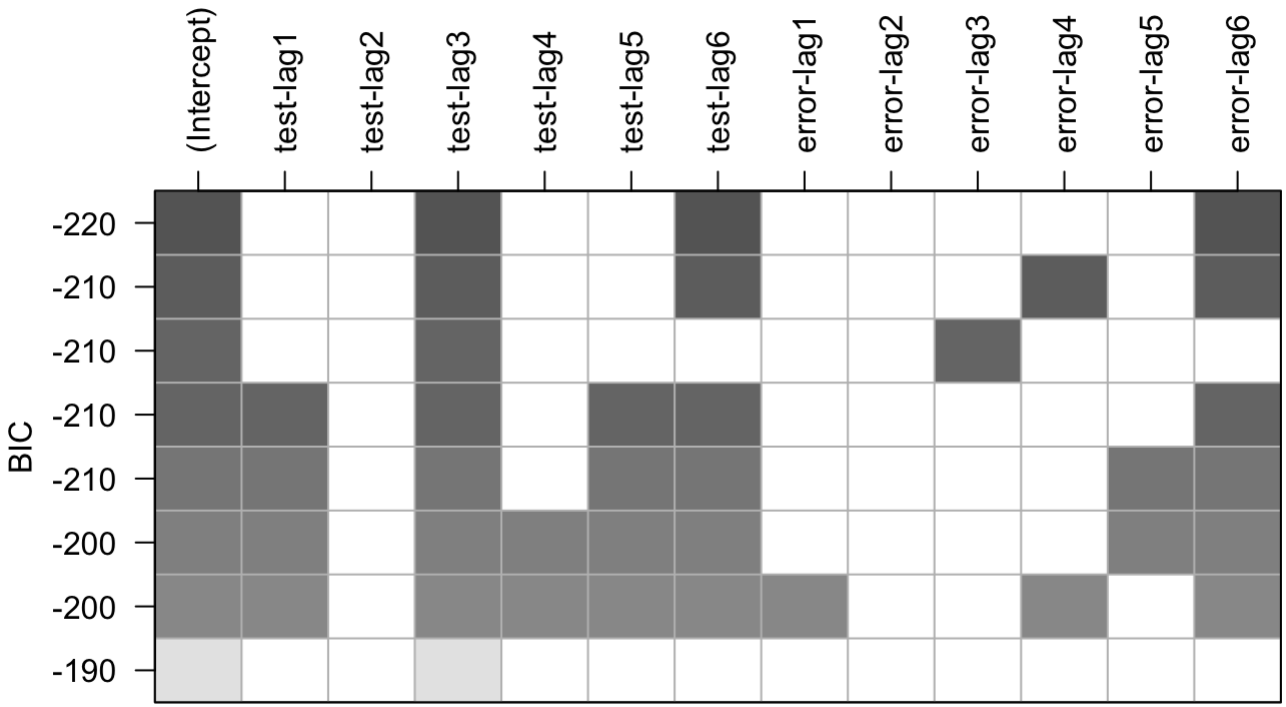
The top row tells us that the subset ARMA(14,14) model with the smallest BIC contains only lags 8 and 12 of the observed time series and lag 12 of the error process.

The next best model contains lag 12 of the time series and lag 8 of the errors, while the third best model contains lags 4, 8, and 12 of the time series and lag 12 of the errors. In our simulated time series, the second best model is the true subset model.

However, the BIC values for these three models are all very similar, and all three (plus the fourth best model) are worthy of further study.

However, lag 12 of the time series and that of the errors are the two variables most frequently found in the various subset models summarized

```
test = arima.sim(model=list(ar=c(rep(0,2),.8),ma=c(rep(0,2),0.7)),n=120)
res = armasubsets(y=test,nar=6,nma=6,y.name='test',ar.method='ols')
plot(res)
```



# Specification of Some Actual Time Series

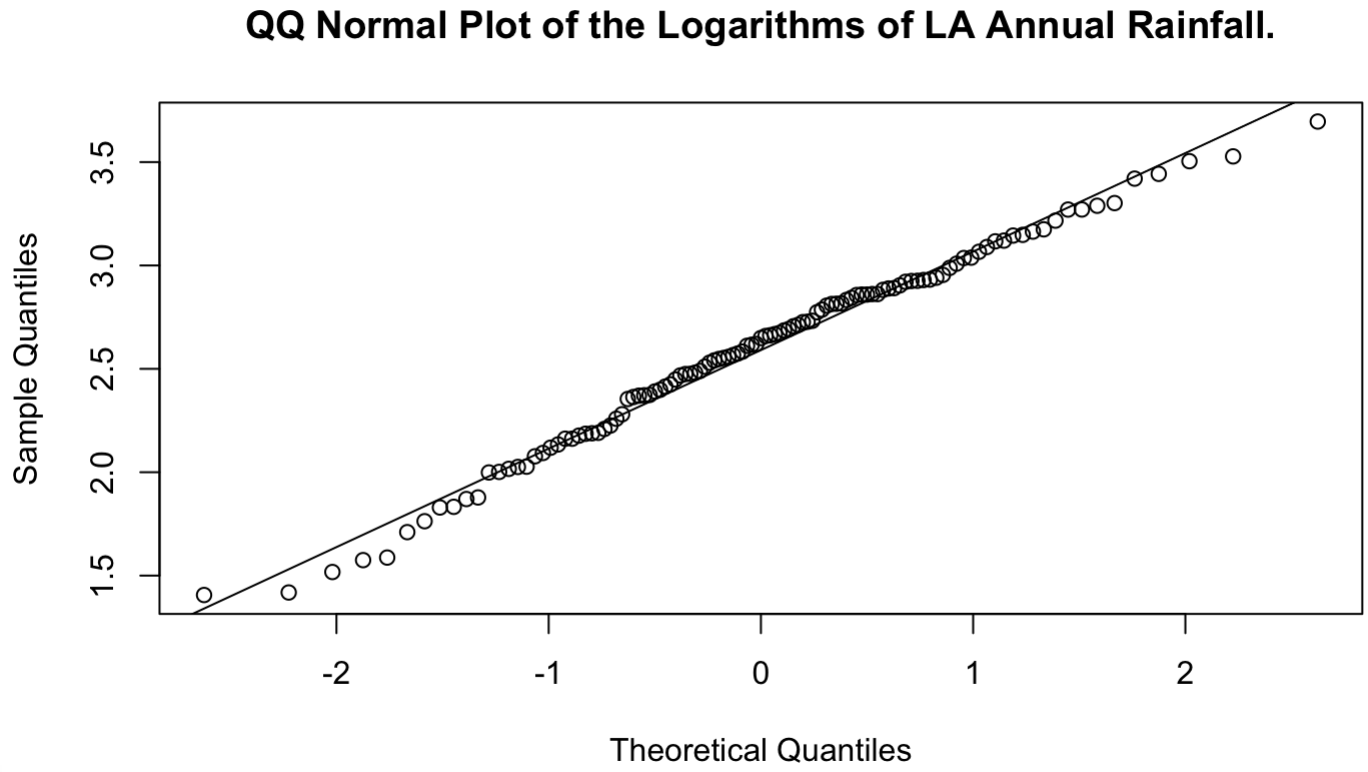
# The Los Angeles Annual Rainfall Series

We will revisit annual total rainfall amounts for the Los Angeles series for this example.

It was discussed that those rainfall amounts were not normally distributed.

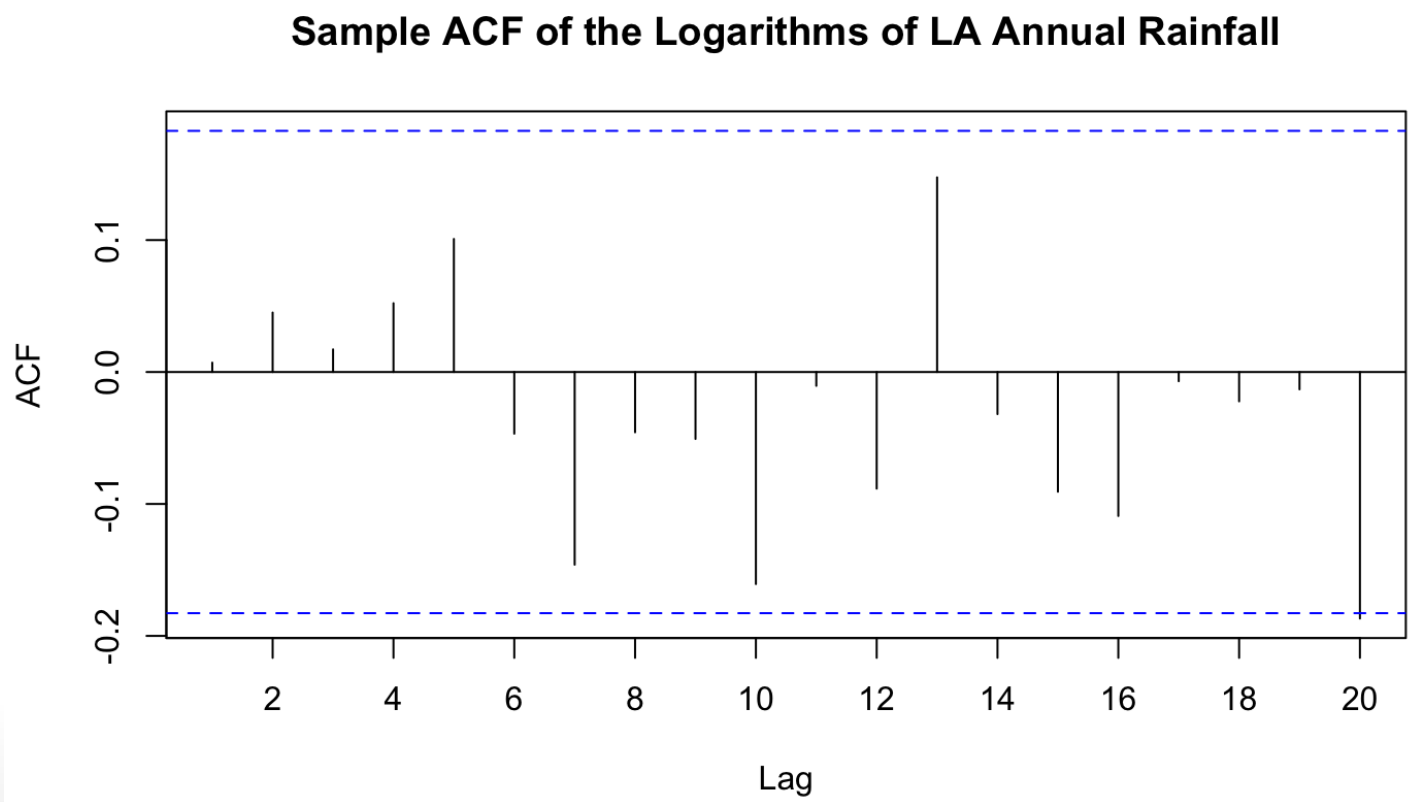
So we apply the natural log transform to the series and display the QQ plot.

```
data(larain)
qqnorm(log(larain),main="QQ Normal Plot of the Logarithms of LA Annual Rainfall.")
qqline(log(larain))
```

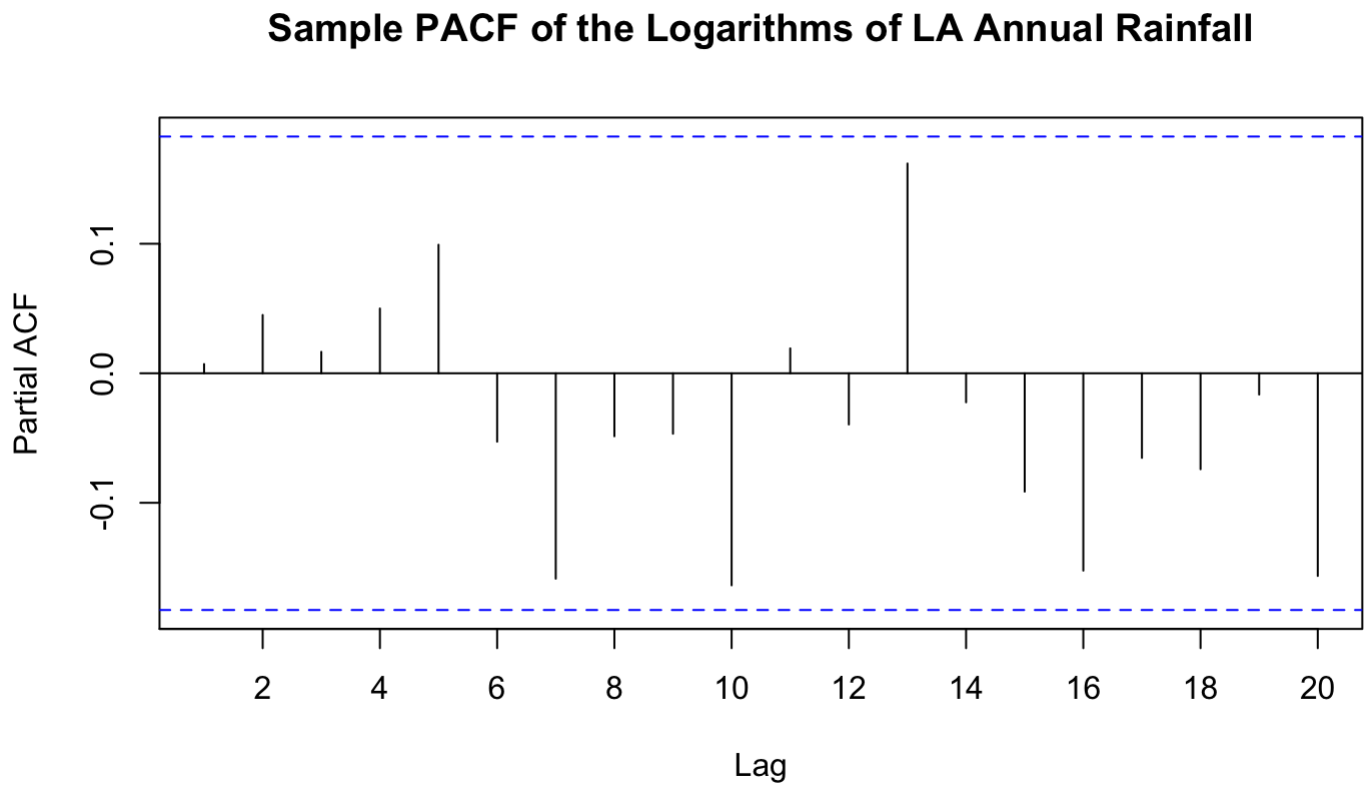


It seems that the log transform improves the fit to the normal distribution. Sample ACF and PACF of the series are:

```
acf(log(larain),xaxp=c(0,20,10),main = "Sample ACF of the Logarithms of LA Annual Rainfall
```



```
pacf(log(larain),xaxp=c(0,20,10),main = "Sample PACF of the Logarithms of LA Annual Rainfa
```



The log transformation has improved the normality, but there is no discernible dependence in this time series.

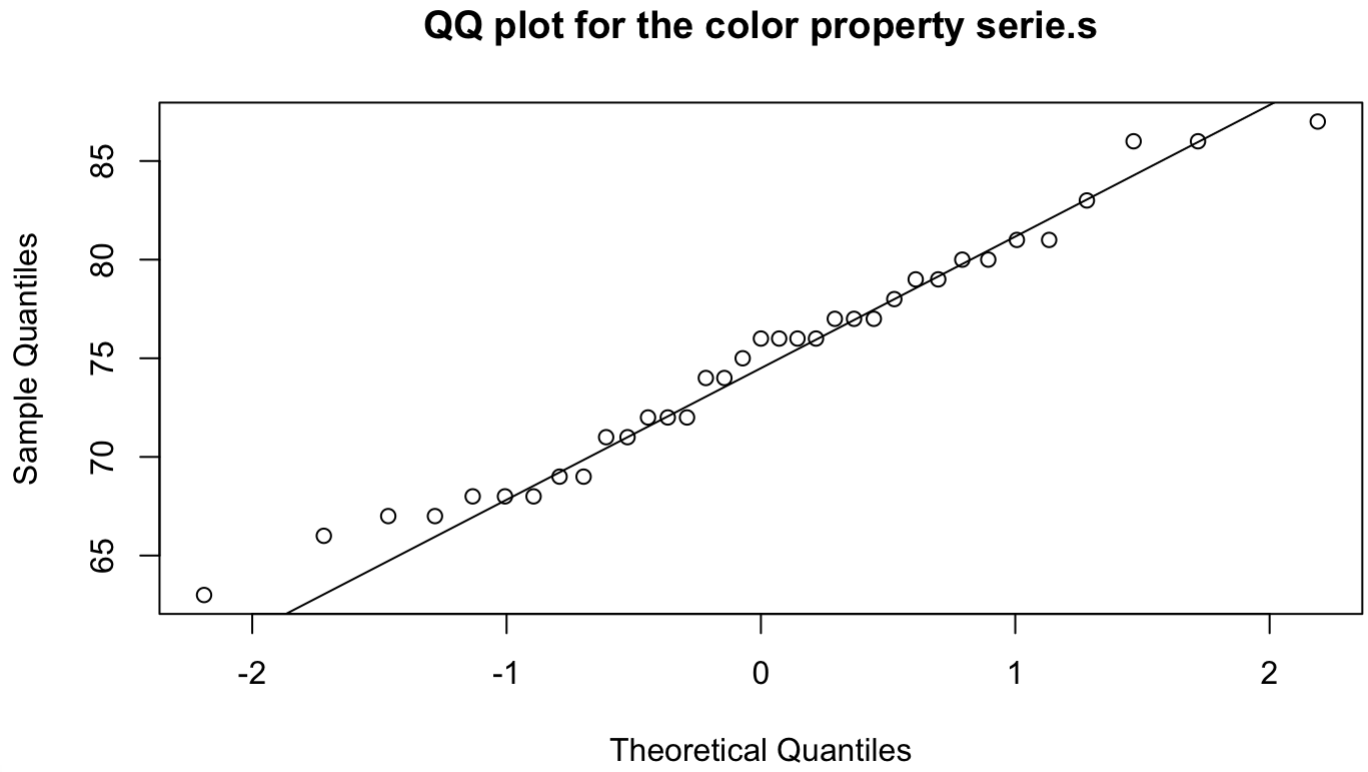
We could model the logarithm of annual rainfall amount as independent, normal random variables with mean 2.58 and standard deviation of 0.478.

# The Chemical Process Color Property Series

The industrial chemical process color property series was displayed in module 1.

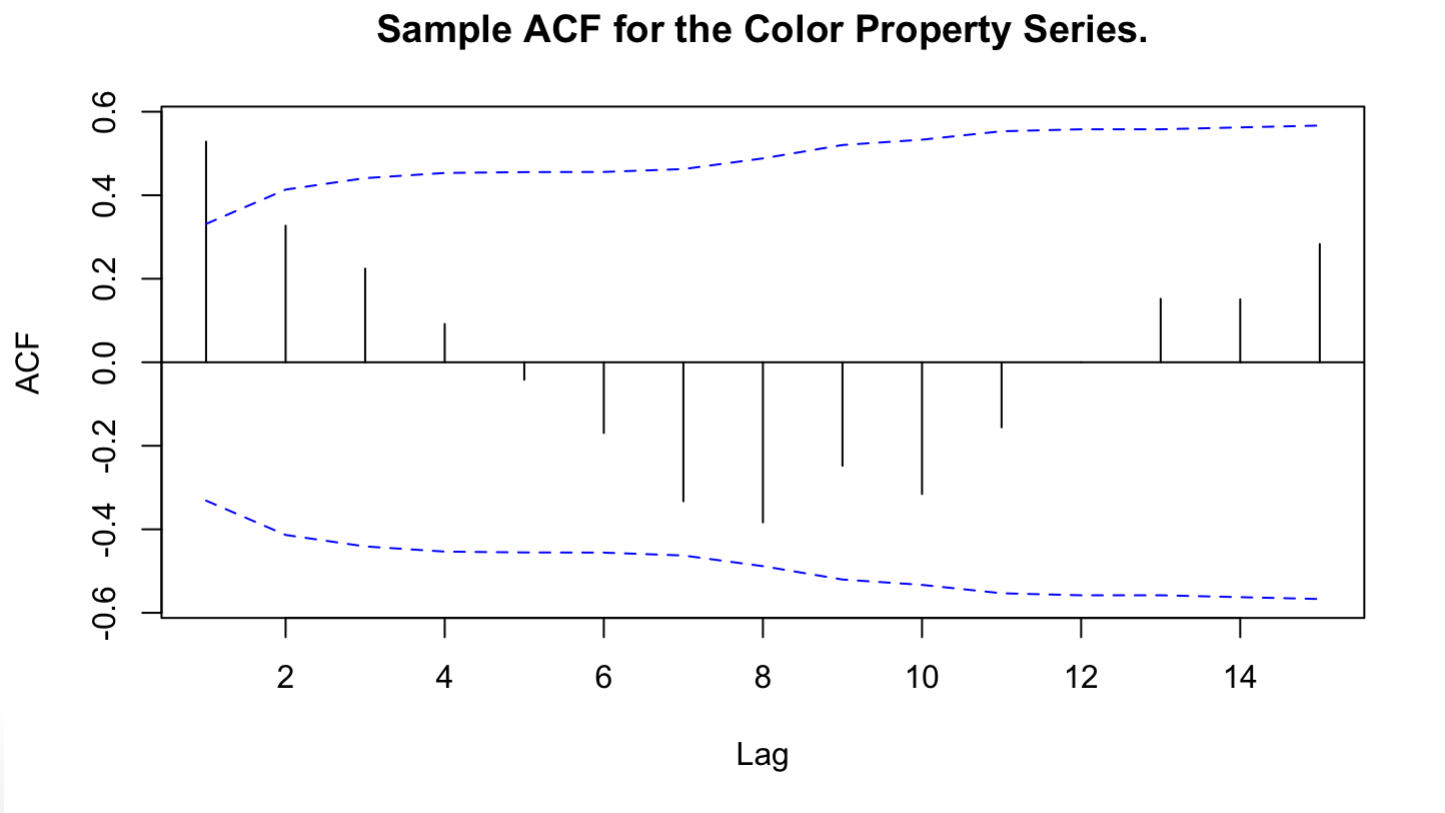
The corresponding QQ plot is shown below. There is a satisfactory fit to the normal distribution seen in the QQ plot.

```
data(color)
qqnorm(color,main="QQ plot for the color property serie.s")
qqline(color)
```

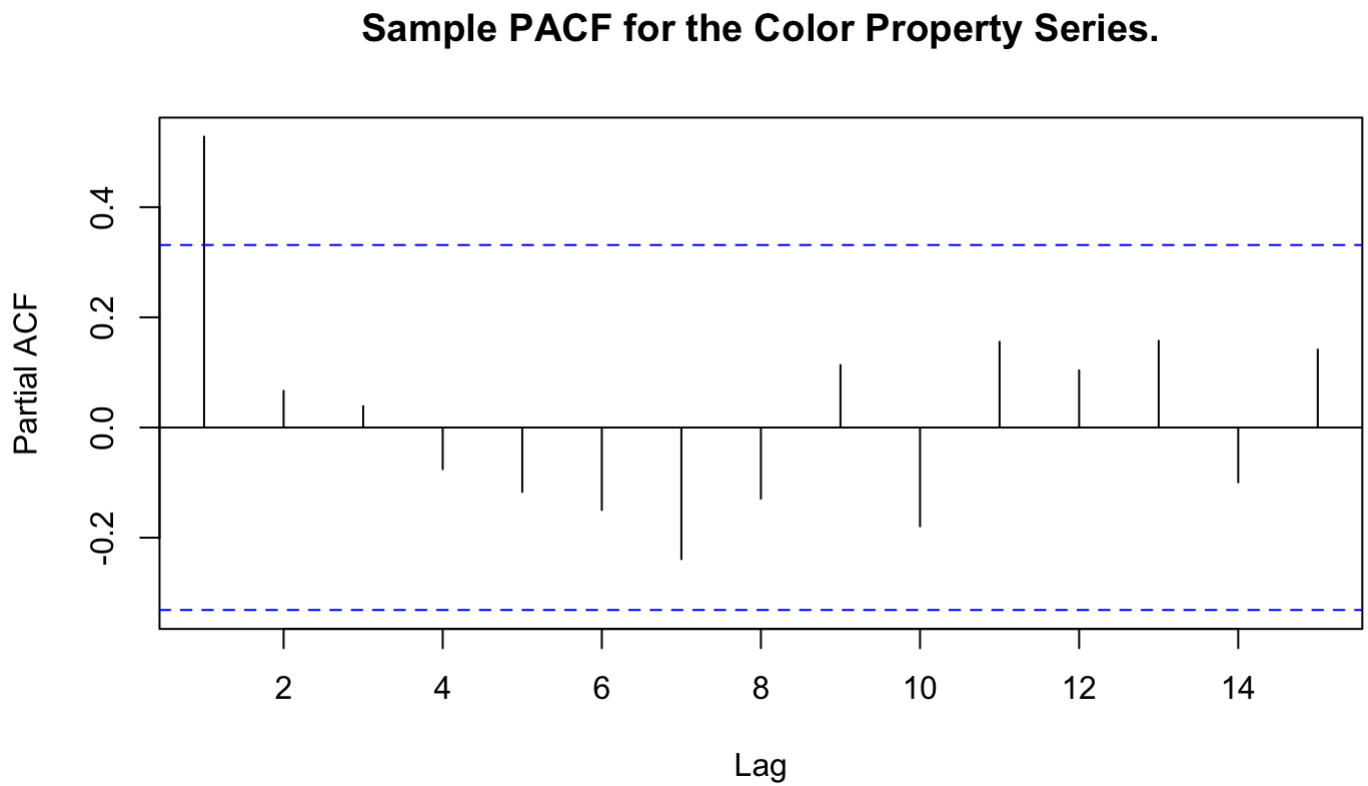


The sample ACF and PACF plots for the color property series are given below.

```
acf(color,ci.type='ma',main="Sample ACF for the Color Property Series.")
```



```
pacf(color,main="Sample PACF for the Color Property Series.")
```



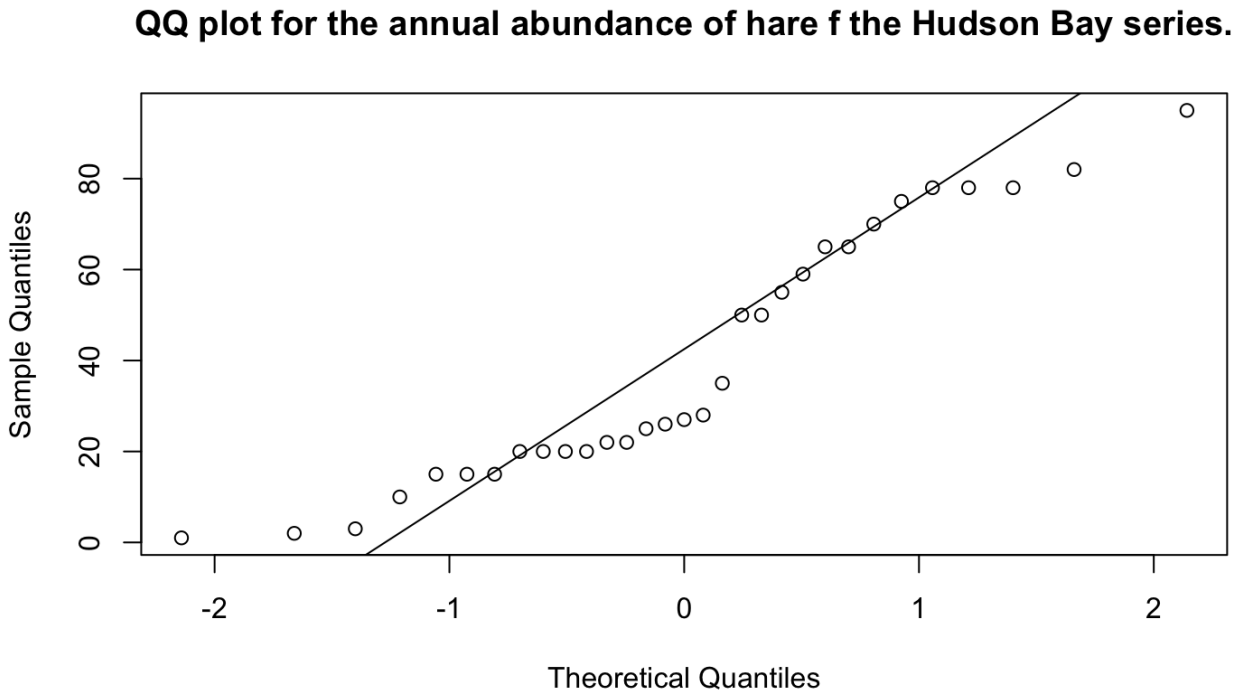
We have a damped sine wave appearance ACF along with the only significant partial autocorrelation at the first lag.

Thus, we see an AR(1) model from these plots.

# The Annual Abundance of Canadian Hare Series

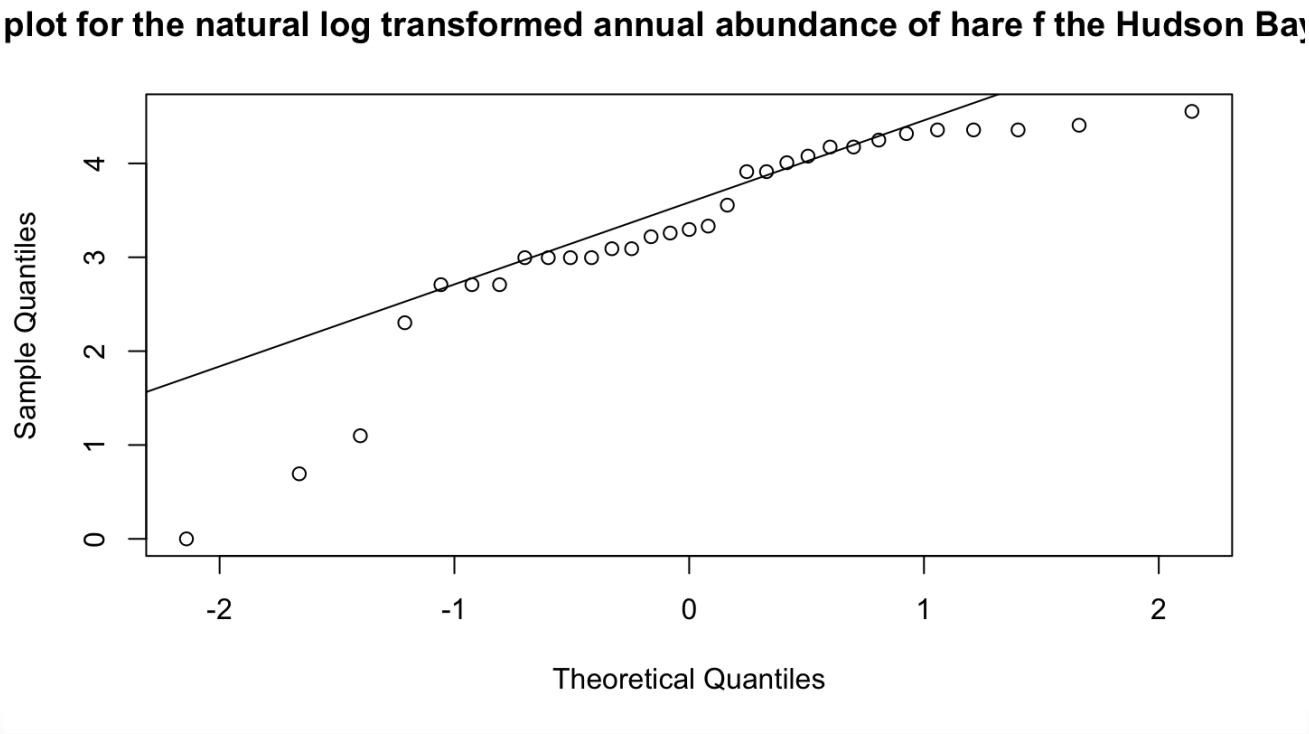
The QQ plot for the time series of the annual abundance of hare of the Hudson Bay is given below.

```
data(hare)
qqnorm(hare,main="QQ plot for the annual abundance of hare f the Hudson Bay series.")
qqline(hare)
```



QQ plot suggests non-normality in the hare series. So, we try the natural log transformation firstly.

```
qqnorm(log(hare),main="QQ plot for the natural log transformed annual abundance of hare f
```

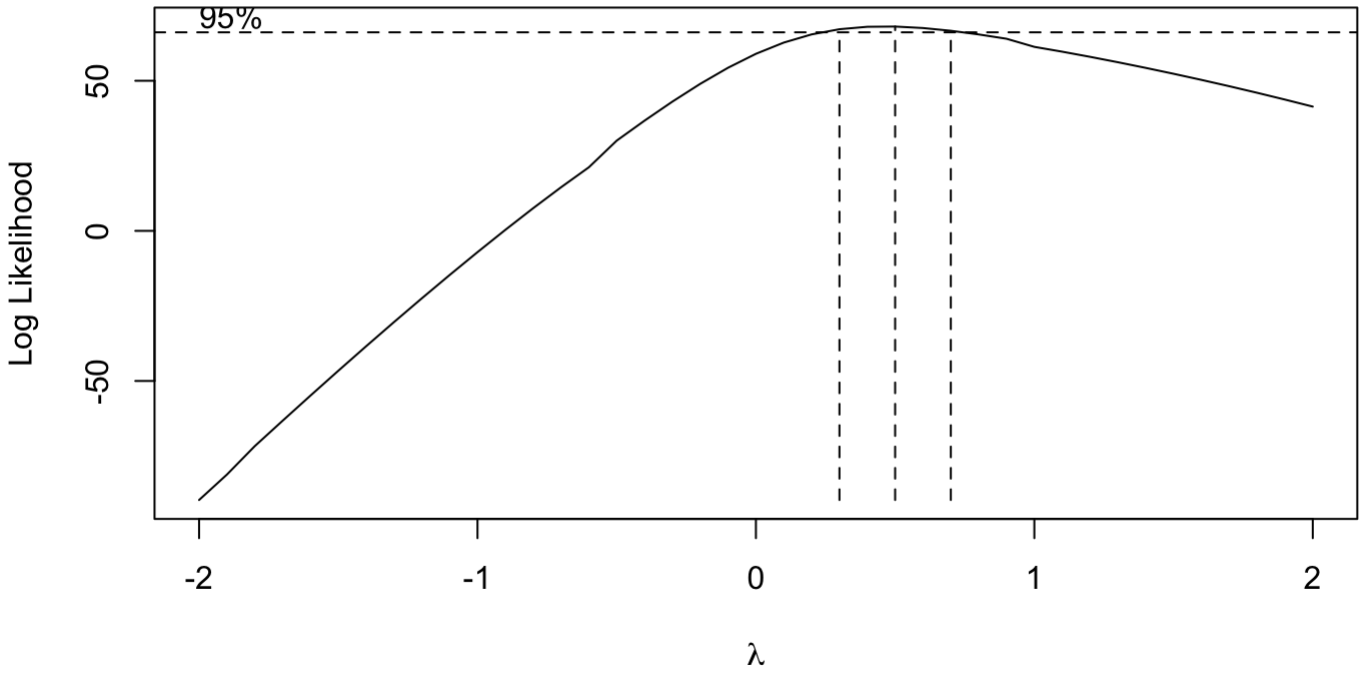


It seems that the log transformation does not work for this series.

The next step to take is to make a Box-Cox transformation.

To find the optimal value of the $\lambda$ parameter in Box-Cox transformation, we plot the following:

```
BoxCox.ar(hare)
```

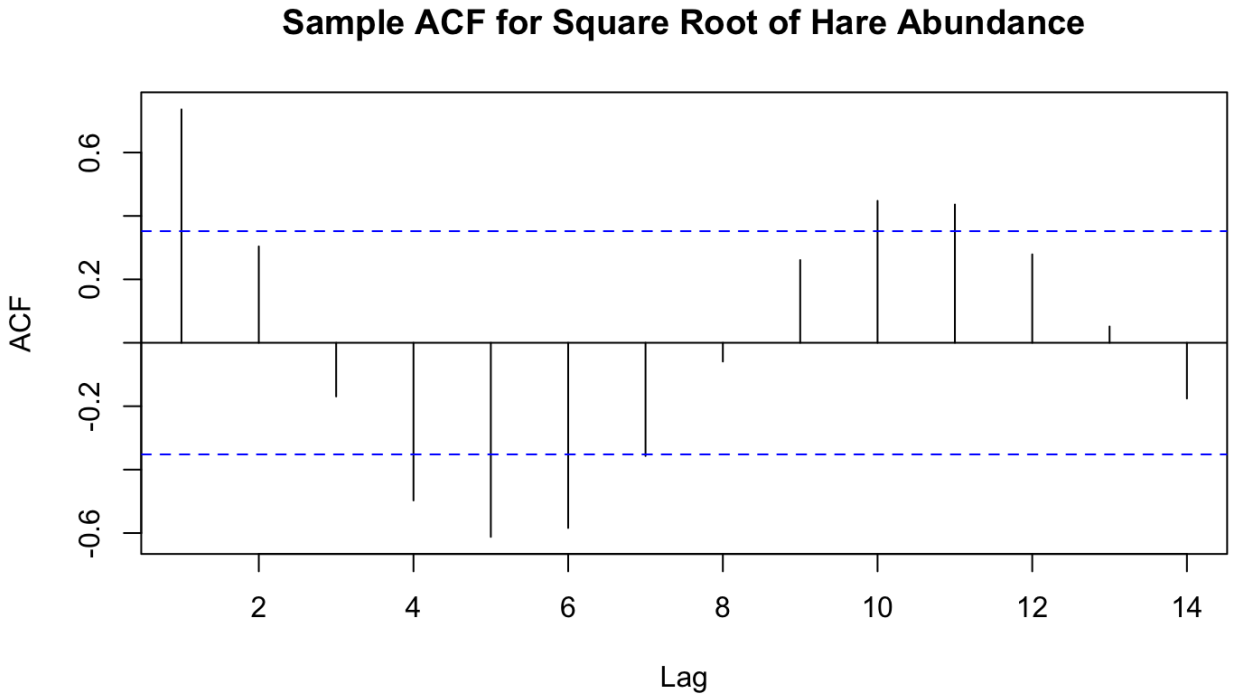


The maximum occurs at $\lambda = 0.4$, but a square root transformation with $\lambda = 0.5$ is well within the confidence interval for $\lambda$.

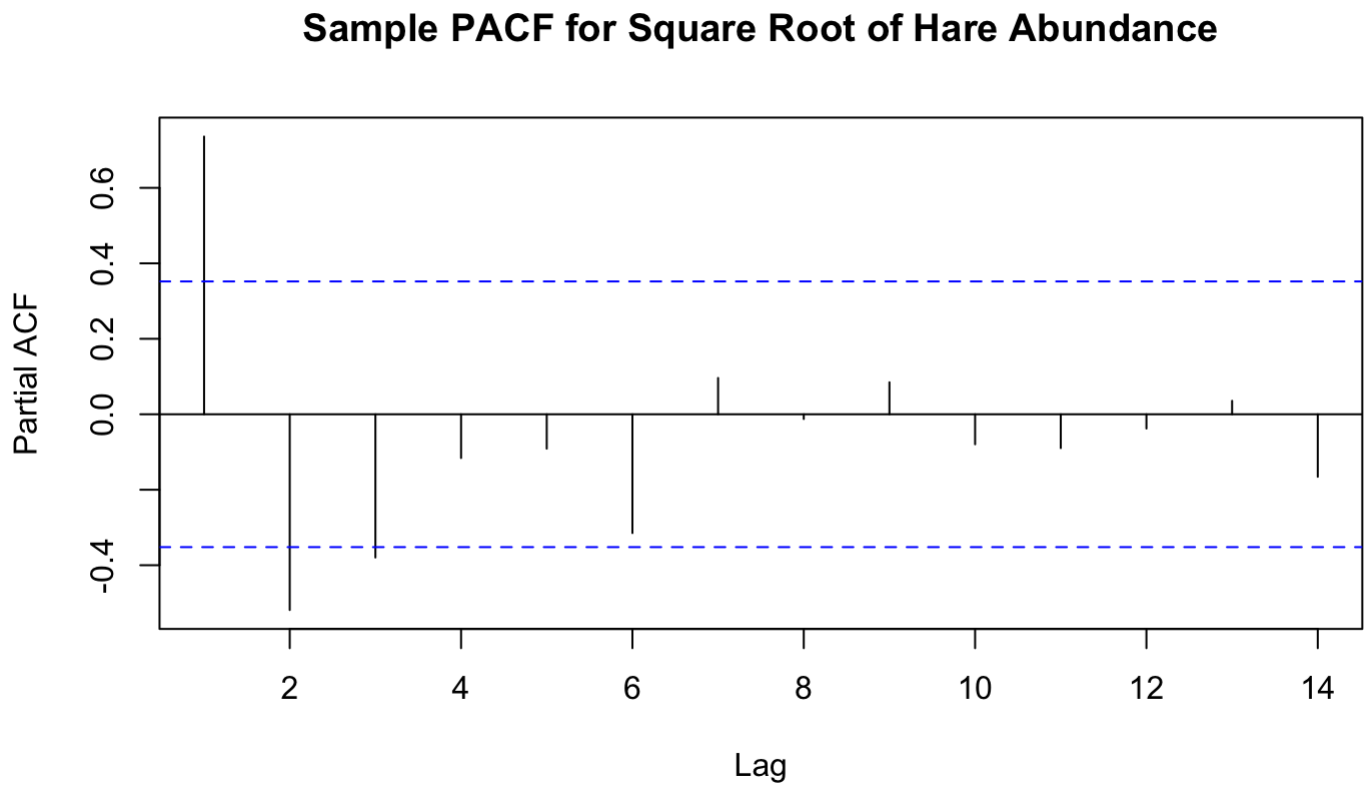
We will take the square root of the abundance values for all further analyses.

The sample ACF and PACF for the transformed series are shown below.

```
acf(hare^.5,main="Sample ACF for Square Root of Hare Abundance")
```



```
pacf(hare^.5,main="Sample PACF for Square Root of Hare Abundance")
```



There is a strong indication of damped oscillatory behavior in the ACF and two highly significant partial autocorrelations at the very first lags in the PACF.

So, the ACF and PACF suggest an AR(2) process for this time series.

```
sq.hare = sqrt(hare)
eacf(sq.hare,ar.max = 7, ma.max = 7)
```

```
## AR/MA
##   0 1 2 3 4 5 6 7
## 0 x o o x x x o o
## 1 x o o x x x o o
## 2 o o o o o o o o
## 3 o o o o o o o o
## 4 o o o o o o o o
## 5 x o o o o o o o
## 6 x o o o o o o o
## 7 x o o o o o o o
```

# The Oil Price Series

We applied the difference of natural log transformation to the monthly oil price time series to make it stationary.

```
log.oil = log(oil.price)
ar(diff(log.oil))
```

```
##
## Call:
## ar(x = diff(log.oil))
##
## Coefficients:
##       1        2
##  0.2410  -0.1385
##
## Order selected 2  sigma^2 estimated as  0.006767
```

```
adfTest(log.oil, lags = 2, type = "ct", title = NULL,description = NULL)
```

```
##
## Title:
##  Augmented Dickey-Fuller Test
##
## Test Results:
##   PARAMETER:
##     Lag Order: 2
##   STATISTIC:
##     Dickey-Fuller: -1.9401
##   P VALUE:
##     0.6011
##
## Description:
##  Tue Apr  2 09:10:57 2019 by user:
```

The software implementation of the Augmented Dickey-Fuller unit-root test applied to the logs of the original prices with $k = 6$ leads to a p-value of 0.6011.

With stationarity as the alternative hypothesis, this provides strong evidence of nonstationarity and the appropriateness of taking a difference of the logs.

We give EACF for the difference of logarithms for this series below.

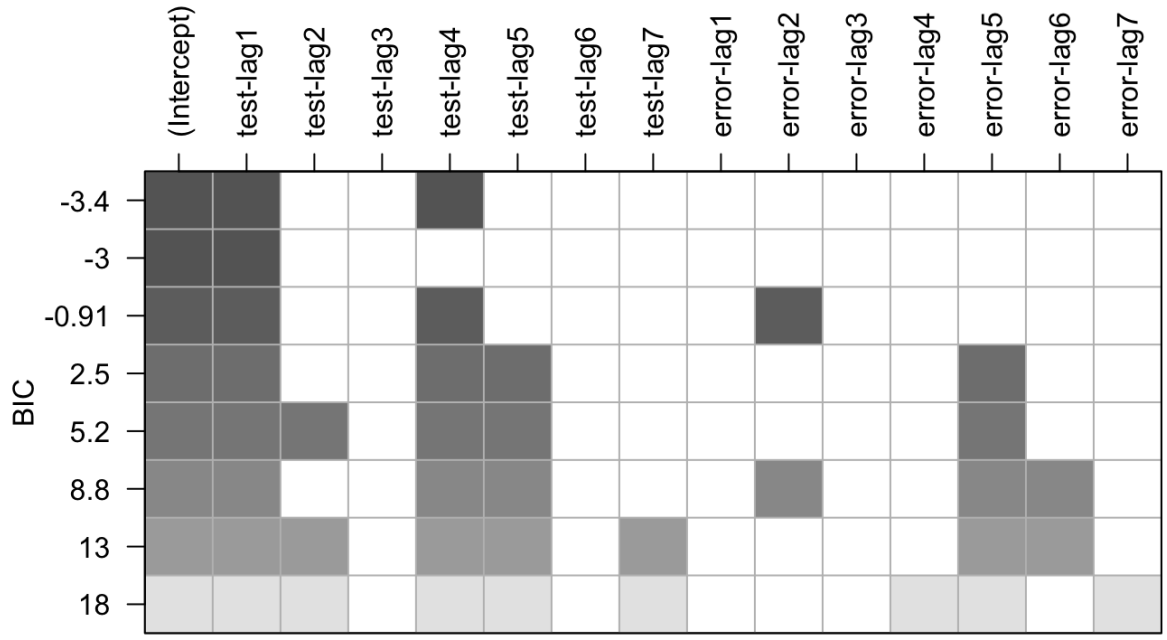
EACF table suggests ARIMA(0,1,1) model for the differences.

```
eacf(diff(log(oil.price)))
```

```
## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 x o o o o o o o o o o  o  o  o
## 1 x x o o o o o o o o x  o  o  o
## 2 o x o o o o o o o o o  o  o  o
## 3 o x o o o o o o o o o  o  o  o
## 4 o x x o o o o o o o o  o  o  o
## 5 o x o x o o o o o o o  o  o  o
## 6 o x o x o o o o o o o  o  o  o
## 7 x x o x o o o o o o o  o  o  o
```

The results of the best subsets ARMA approach are given below.

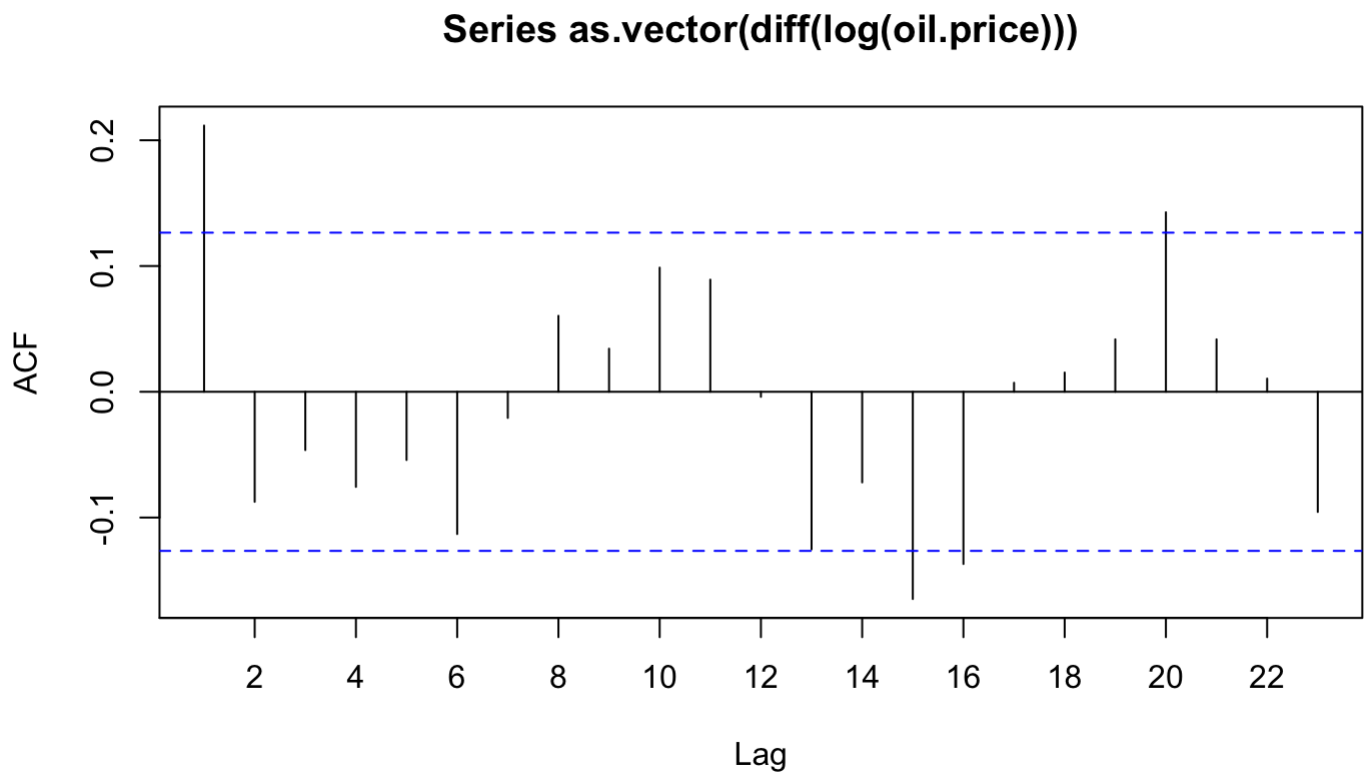
```
res=armasubsets(y=diff(log(oil.price)),nar=7,nma=7,y.name='test', ar.method='ols')
plot(res)
```



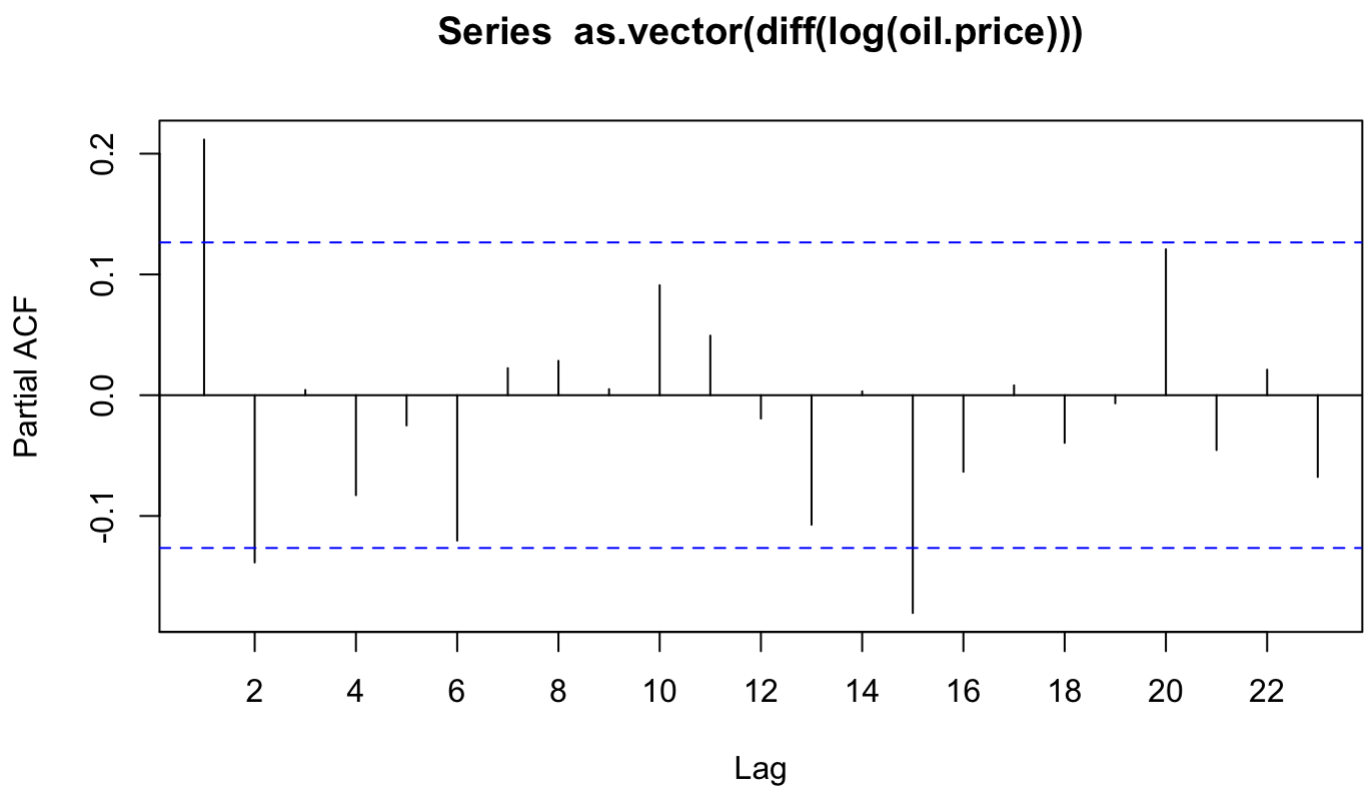
Because we get the smallest BIC at lags 1 and 4, $Y_t = \nabla \log(Oil_t)$ should be modeled in terms of $Y_{t-1}$ and $Y_{t-4}$ and that no lags are needed in the error terms. The second best model omits the lag 4 term so that an ARIMA(1,1,0) model on the logarithms should also be investigated further.

The sample ACF and PACF plots for the oil price series are given in the following plots.

```
acf(as.vector(diff(log(oil.price))),xaxp=c(0,22,11))
```



```
pacf(as.vector(diff(log(oil.price))),xaxp=c(0,22,11))
```



Because we have one significant pike at the first lag in ACF and two significant pikes at the first two lags in PACF, we can consider MA(1) or AR(2) models on the difference of log of series.

# Summary

In this module, we considered the problem of specifying reasonable but simple models for observed times series. In particular,

- we investigated the sample autocorrelation function, the sample partial autocorrelation function, and the sample extended autocorrelation function for choosing the orders (p, d, and q) for ARIMA(p,d,q) models.
- The Dickey-Fuller unit-root test was introduced to help distinguish between stationary and nonstationary series.
- Model selection criteria are used to identify the most promising model.
- These ideas were all illustrated with both simulated and actual time series.

# What's next?

In the next module, we will focus on the estimation of model parameters using

- methods of moments,
- maximum likelihood, and
- Bootstrapping.

# Task 5

Please complete the tasks given in `Module 5: Tasks` for the fifth task.

The required files are available in the Canvas shell of the course via

`Course webpage` -> `Modules` -> `Module 5: Tasks`

# References

Wei. W.W.S. Time Series Analysis: Univariate and Multivariate Methods. Wiley, New York, 2006.

Thanks for your attendance!
Please use Socrative.com with room *TIMESERIES* to give anonymous feedback!