chapter 9 In-Class

Chapter 9 - Time series

```
library("Ecdat")
```

```
## Loading required package: Ecfun

##
## Attaching package: 'Ecfun'

## The following object is masked from 'package:base':
##
##     sign

##
## Attaching package: 'Ecdat'

## The following object is masked from 'package:datasets':
##
##     Orange
```

```
library("fGarch")
```

```
## Loading required package: timeDate

## Loading required package: timeSeries

## Loading required package: fBasics

##

## Rmetrics Package fBasics

## Analysing Markets and calculating Basic Statistics

## Copyright (C) 2005-2014 Rmetrics Association Zurich

## Educational Software for Financial Engineering and Computational Science

## Rmetrics is free software and comes with ABSOLUTELY NO WARRANTY.

## https://www.rmetrics.org --- Mail to: info@rmetrics.org
```

```
library("evir")
library("forecast")
```

```
## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following object is masked from 'package:timeSeries':
##
##     time<-

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## This is forecast 7.3
```

```
##
## Attaching package: 'forecast'
## The following object is masked from 'package:Ecfun':
##
##     BoxCox
```

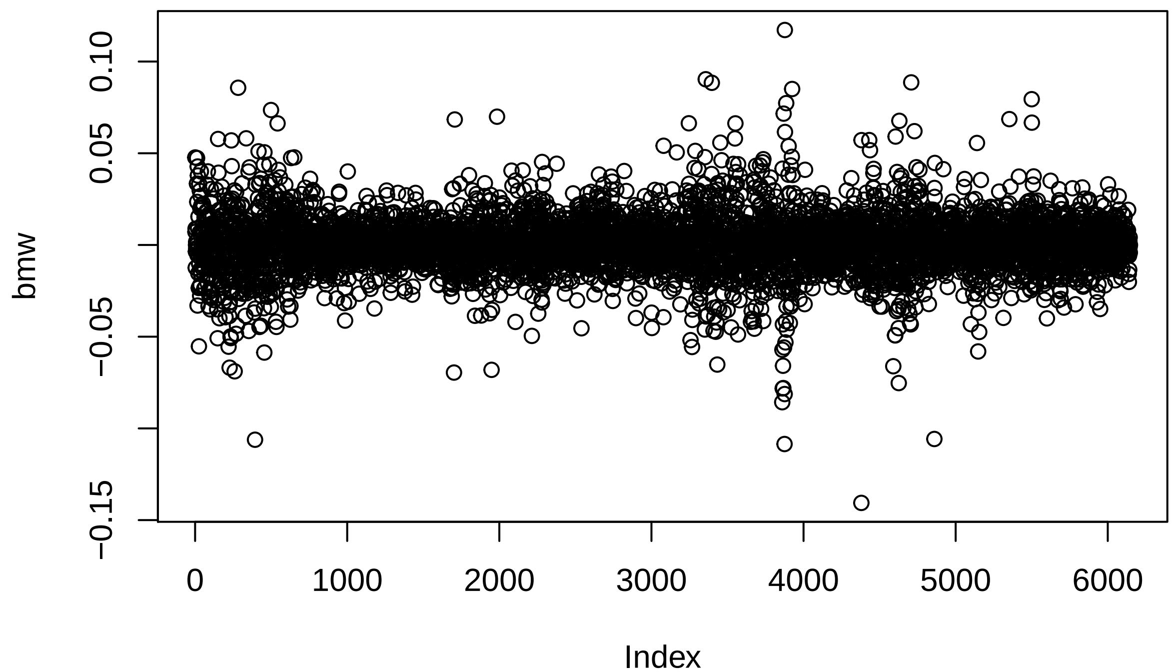
```r
data(CRSPday, package="Ecdat")
data(bmw)
summary(bmw)
```

```
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## -0.1406000 -0.0066560  0.0000000  0.0003407  0.0071260  0.1172000
```

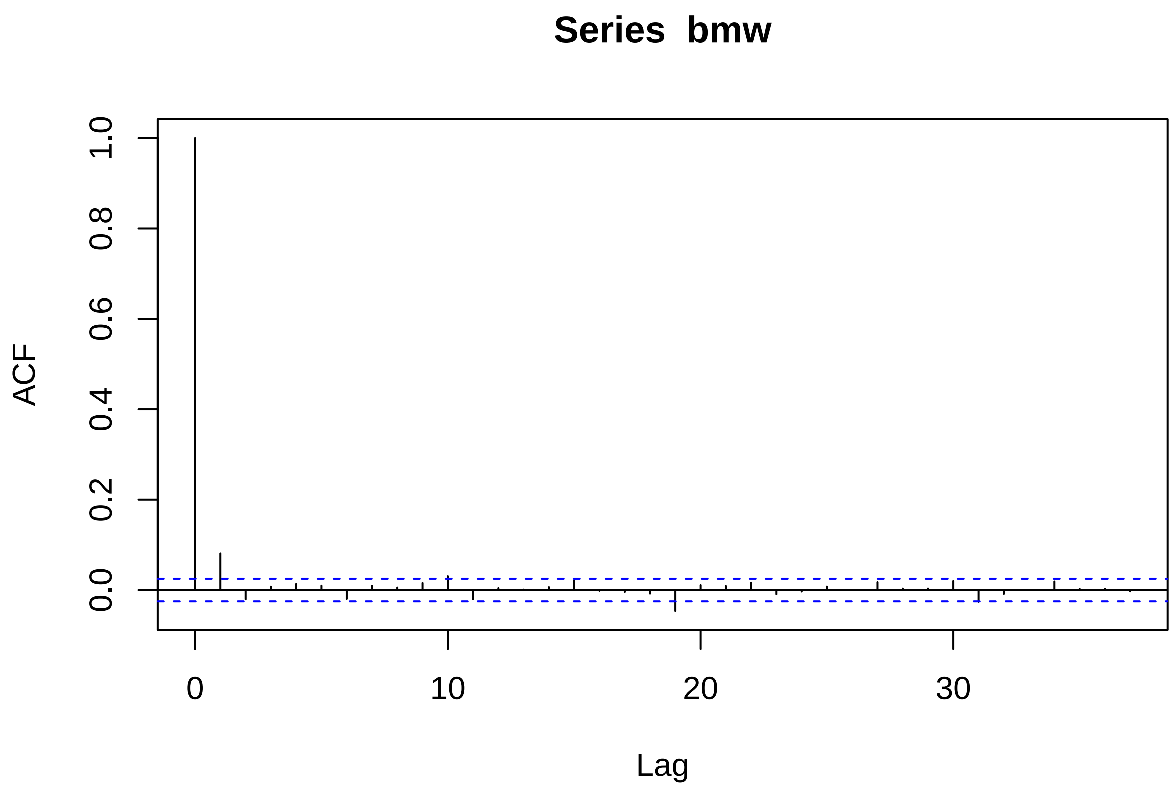
```r
length(bmw)
```

```
## [1] 6146
```

```r
plot(bmw)
```




```r
acf(bmw)
```



The data seems to be stationary in the plot diagram. The autocorrelation function plot shows a well-behaved AR(1) or possibly AR(2) (here all changes lie within the dotted lines).

```
Box.test(x=bmw, lag=5, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  bmw
## X-squared = 44.987, df = 5, p-value = 1.46e-08
```

The test gives a very small p-value, so I could reject the null hypothesis that the data is indipendenly distributed. There is serial correlation within the data.

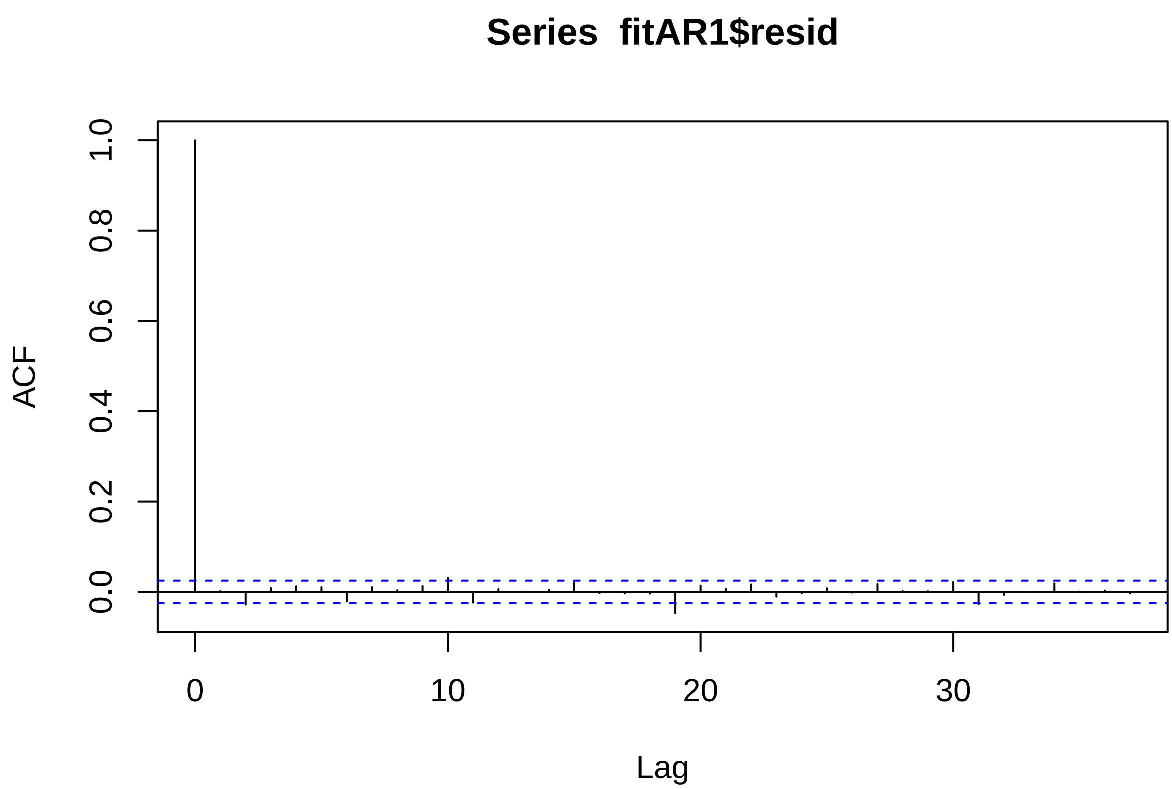
So now

```
fitAR1 = arima(bmw, order = c(1,0,0))
# here I fit AR(1)
fitAR1
```

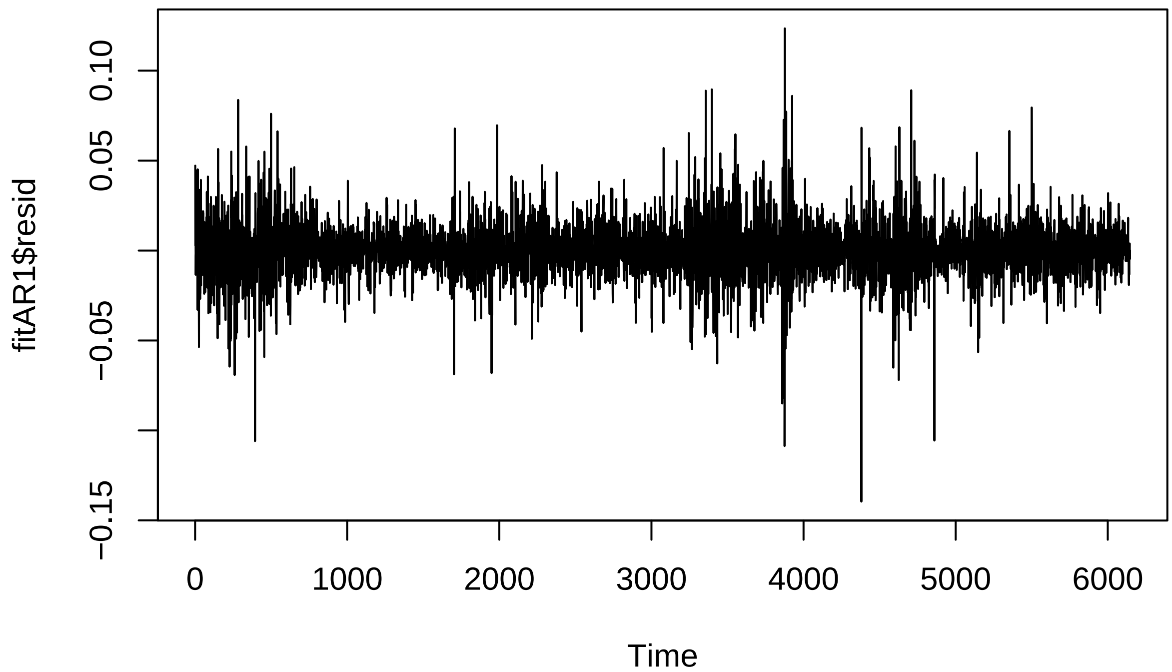
```
##
## Call:
## arima(x = bmw, order = c(1, 0, 0))
##
## Coefficients:
##          ar1  intercept
##       0.0811      3e-04
## s.e.  0.0127      2e-04
##
```

```
## sigma^2 estimated as 0.0002163:  log likelihood = 17212.34,  aic = -34418.68
```

```r
acf(fitAR1$resid)
```




```r
plot(fitAR1$resid)
```




```
Box.test(fitAR1$resid,lag=5,type="Ljung-Box",fitdf=1)
```

```
##
##  Box-Ljung test
##
## data:  fitAR1$resid
## X-squared = 6.8669, df = 4, p-value = 0.1431
```

ACF graph goes quickly to zero, it should indicate stationarity. The Box test has a large p-value, which leads to fail to reject the null hypothesis that the residulals are uncorrelated, at least small lags. So, AR(1) provides a quite adequate fit here. Notice that we use AR(1)'s residuals to test to see if there is any autocorrelation left.

```
data(Mishkin)
summary(Mishkin)
```

```
##      pai1             pai3              tb1               tb3
##  Min.   :-7.565   Min.   :-3.794   Min.   : 0.3215   Min.   : 0.5748
##  1st Qu.: 1.364   1st Qu.: 1.660   1st Qu.: 2.5395   1st Qu.: 2.8821
##  Median : 3.589   Median : 3.679   Median : 4.5711   Median : 5.0470
##  Mean   : 4.006   Mean   : 4.018   Mean   : 4.9983   Mean   : 5.4098
##  3rd Qu.: 6.118   3rd Qu.: 5.609   3rd Qu.: 6.8348   3rd Qu.: 7.4577
##  Max.   :19.570   Max.   :16.431   Max.   :15.7906   Max.   :16.0278
##      cpi
##  Min.   : 23.50
##  1st Qu.: 29.50
##  Median : 39.00
##  Mean   : 55.95
```

```
##   3rd Qu.: 84.40
##   Max.   :133.80
```

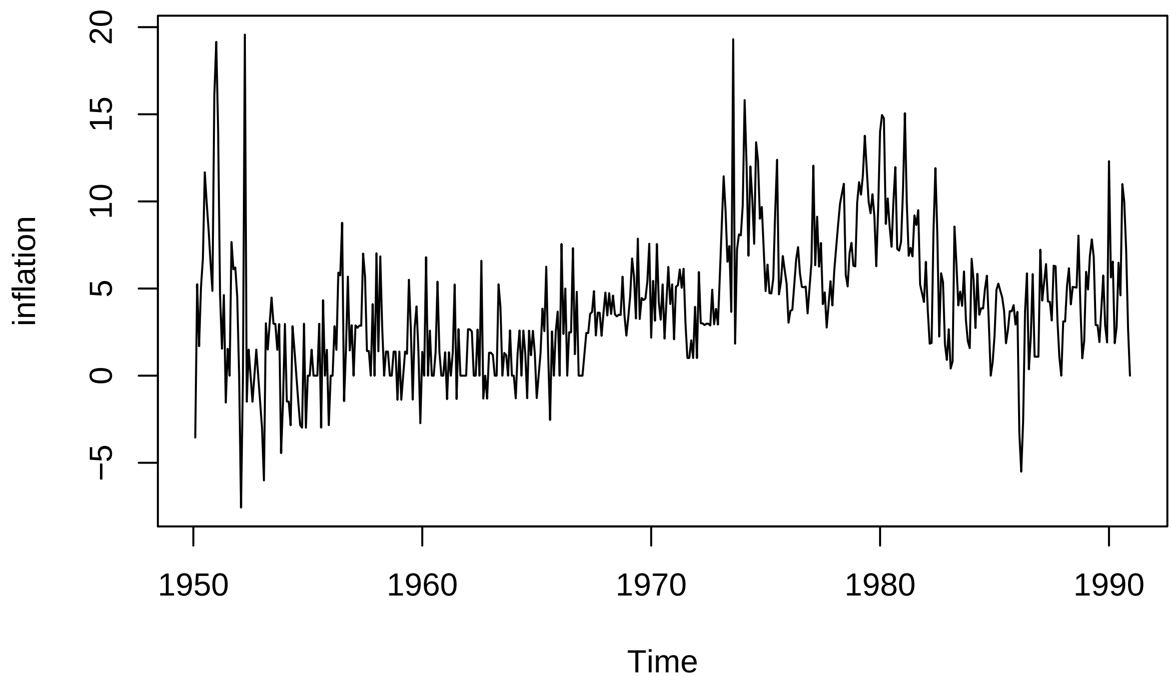
```
?Mishkin
```

```
## starting httpd help server ...
```

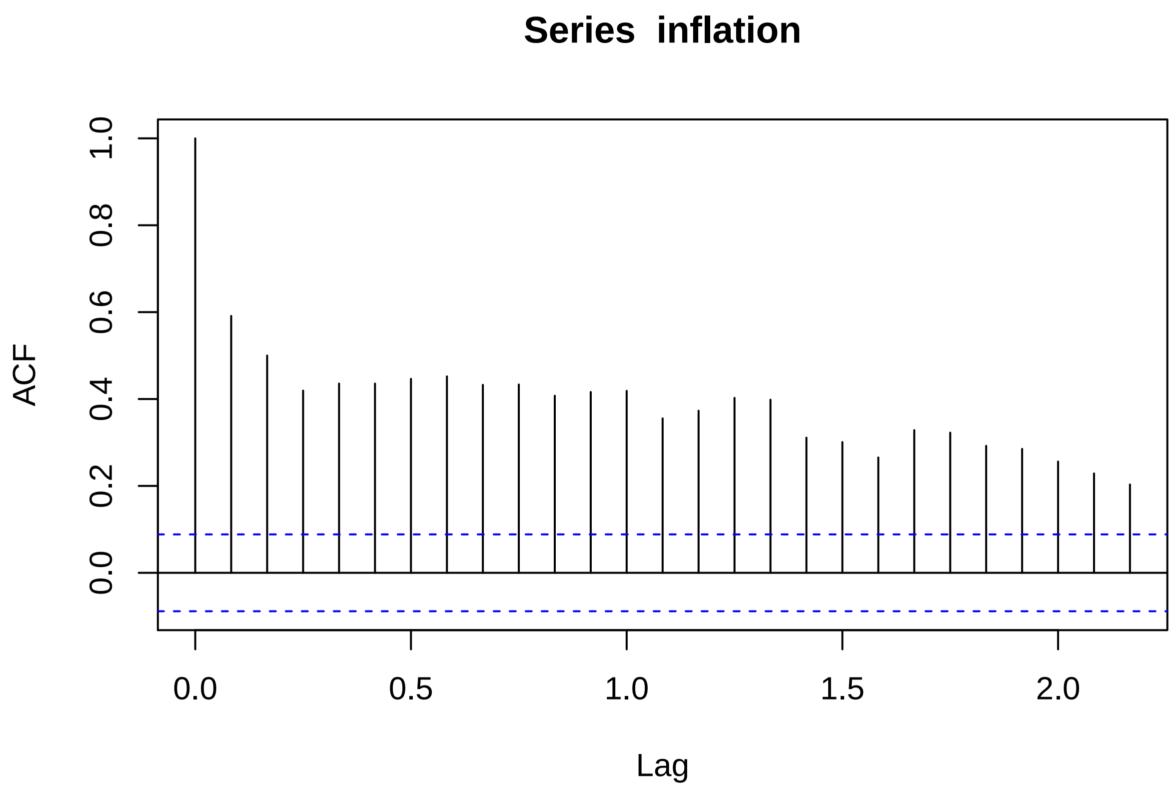
```
##   done
```

The first one is one-month inflation rate (in percent, annual rate).

```
inflation = Mishkin[,1]
plot(inflation)
```

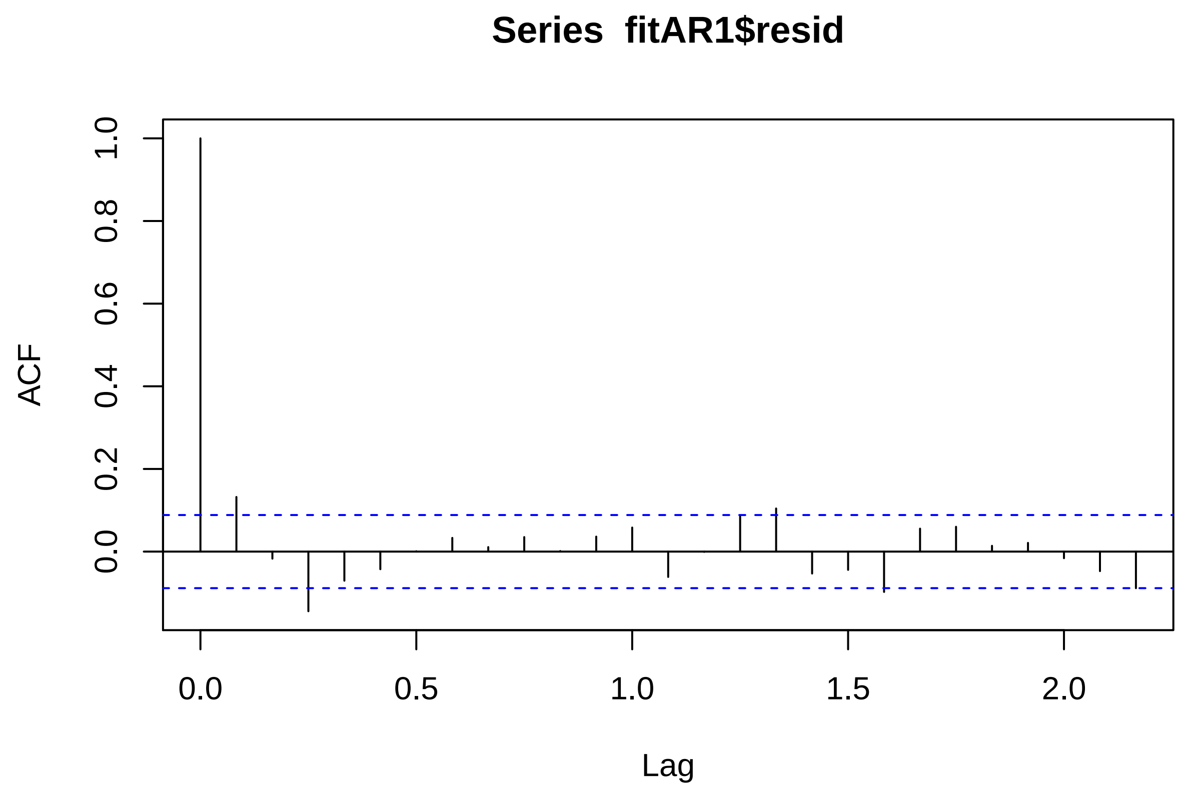



```
# it doesn't seem stationary
acf(inflation)
```




```
# The ACF decays to zero slowly, showing that this is a sign of either nonstationarity or possibly of s
```

```
fitAR1= arima(inflation, order=c(1,0,1))
acf(fitAR1$resid)
```

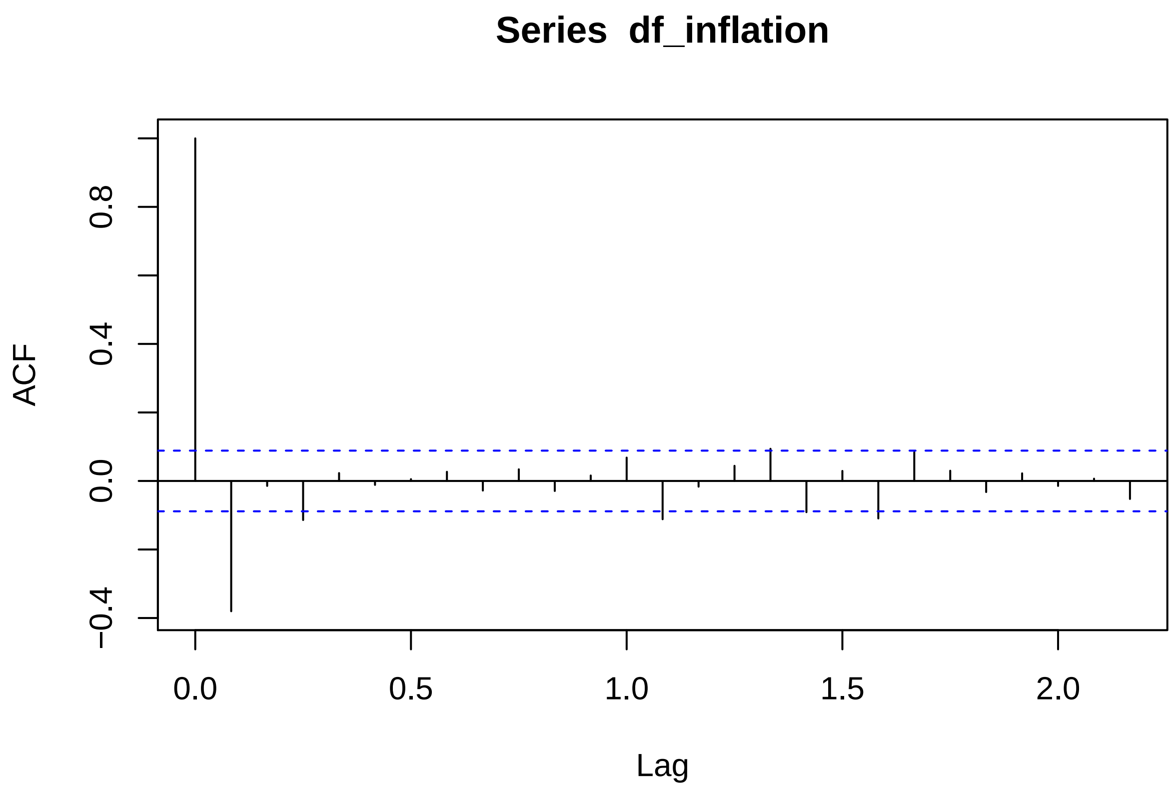



```r
Box.test(fitAR1$resid, lag=5, type="Ljung-Box", fitdf=1)
```

```
##
##  Box-Ljung test
##
## data:  fitAR1$resid
## X-squared = 22.491, df = 4, p-value = 0.00016
```

Low p-value show a significant autocorrelation within the past 4 lags.

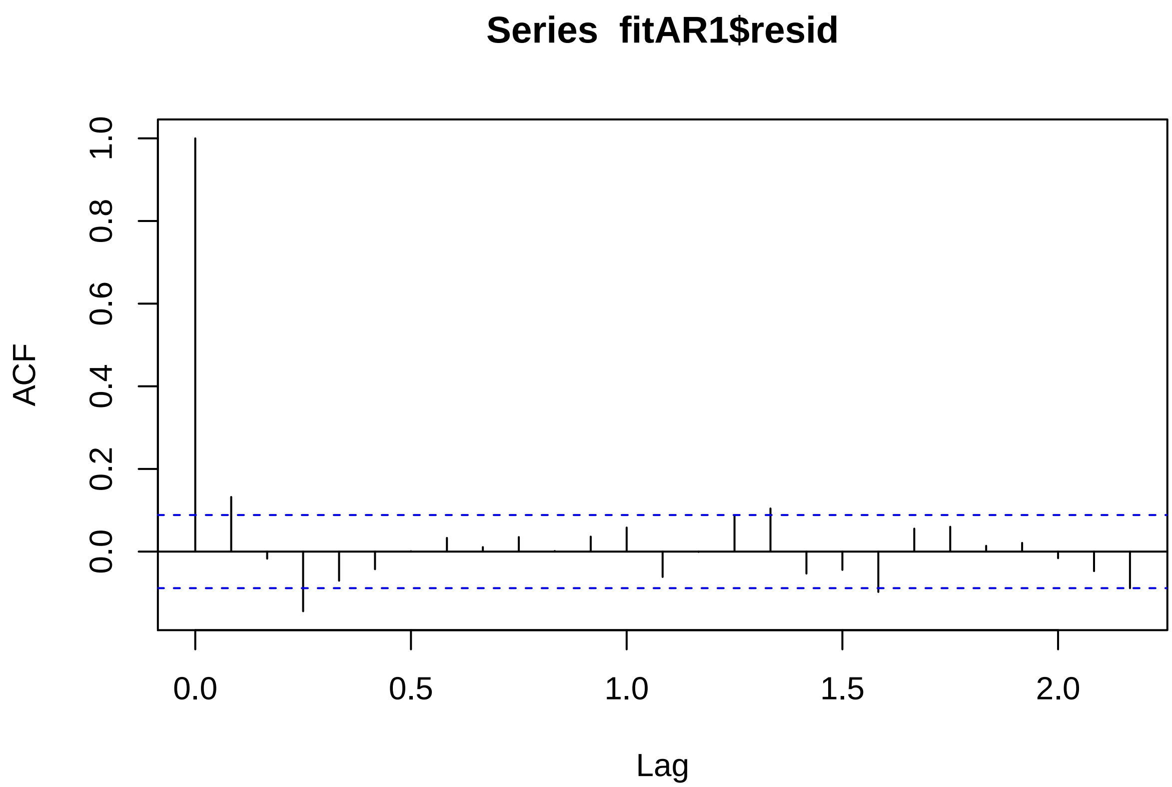
```r
library(timeSeries)
df_inflation = diff(inflation)
?diff
acf(df_inflation) # quicky decaying to zero, quite stationary I assume
```




```r
fitART=arima(df_inflation, order=c(1,0,0))
fitAR1
```

```
##
## Call:
## arima(x = inflation, order = c(1, 0, 1))
##
## Coefficients:
##          ar1      ma1  intercept
##       0.9628  -0.7402     4.0090
## s.e.  0.0182   0.0575     0.8936
##
## sigma^2 estimated as 8.741:  log likelihood = -1229.41,  aic = 2466.83
```

```r
acf(fitAR1$resid)
```




```
Box.test(fitAR1$residuals, lag=5, type="Ljung-Box", fitdf=1)
```

```
##
##  Box-Ljung test
##
## data:  fitAR1$residuals
## X-squared = 22.491, df = 4, p-value = 0.00016
```

```
# there is some autocorrelation in the residuals for that small p-value as we can reject the null hypot
```

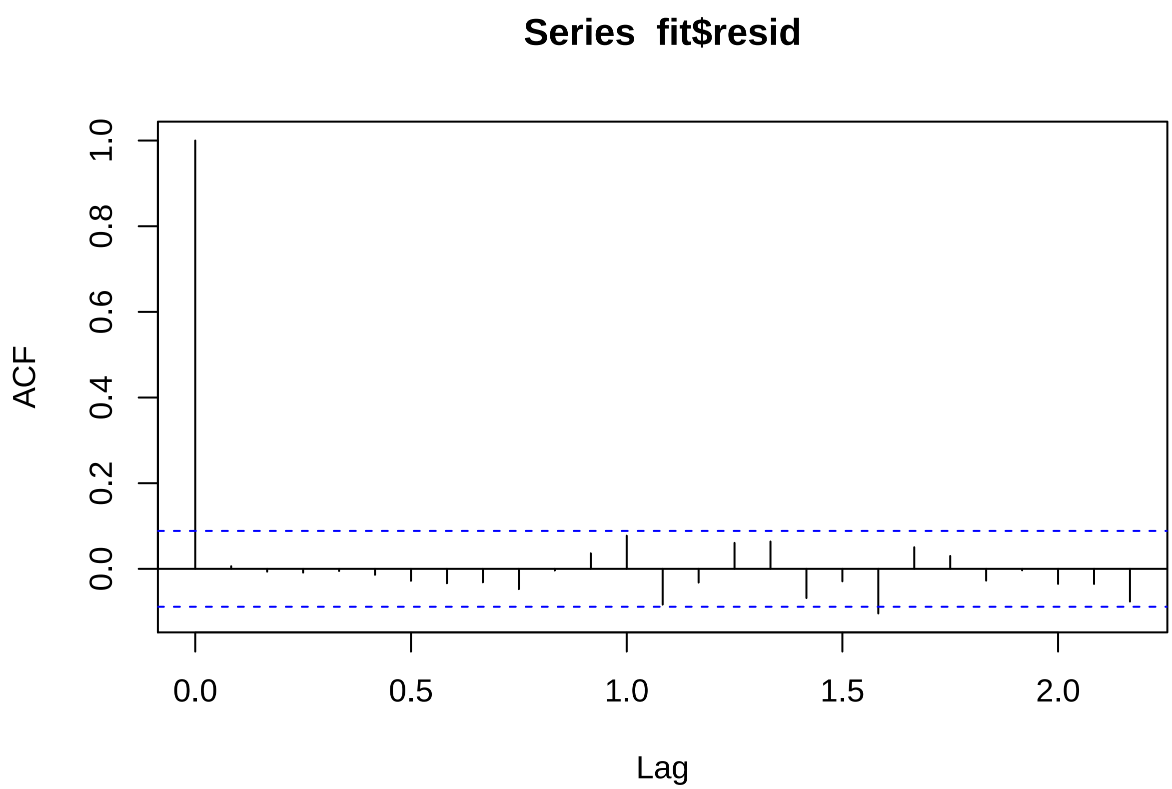
So now we are fitting AR(p) process to df_inflation. With auto.arima, R could find the optimal p, here max.p = 20

```
fit=auto.arima(df_inflation,max.p=20,max.q=0,max.d=0,max.P=0,max.D=0,max.Q=0,ic="aic")
fit
```

```
## Series: df_inflation
## ARIMA(8,0,0) with zero mean
##
## Coefficients:
##           ar1      ar2      ar3      ar4      ar5      ar6      ar7
##       -0.6274  -0.4977  -0.5158  -0.4155  -0.3443  -0.2560  -0.1557
## s.e.   0.0456   0.0536   0.0576   0.0606   0.0610   0.0581   0.0543
##           ar8
##       -0.1051
## s.e.   0.0459
##
## sigma^2 estimated as 8.681:  log likelihood=-1221.2
```

```
## AIC=2460.4   AICc=2460.78   BIC=2498.15
```

```r
acf(fit$resid)
```




```r
Box.test(fit$residuals, lag=5, type="Ljung-Box", fitdf=1)
```

```
##
##  Box-Ljung test
##
## data:  fit$residuals
## X-squared = 0.18324, df = 4, p-value = 0.9961
```

This fit seems to work so well that I cannot reject the non-autocorrelation test within the residuals. So df_inflation has AR(8) process.

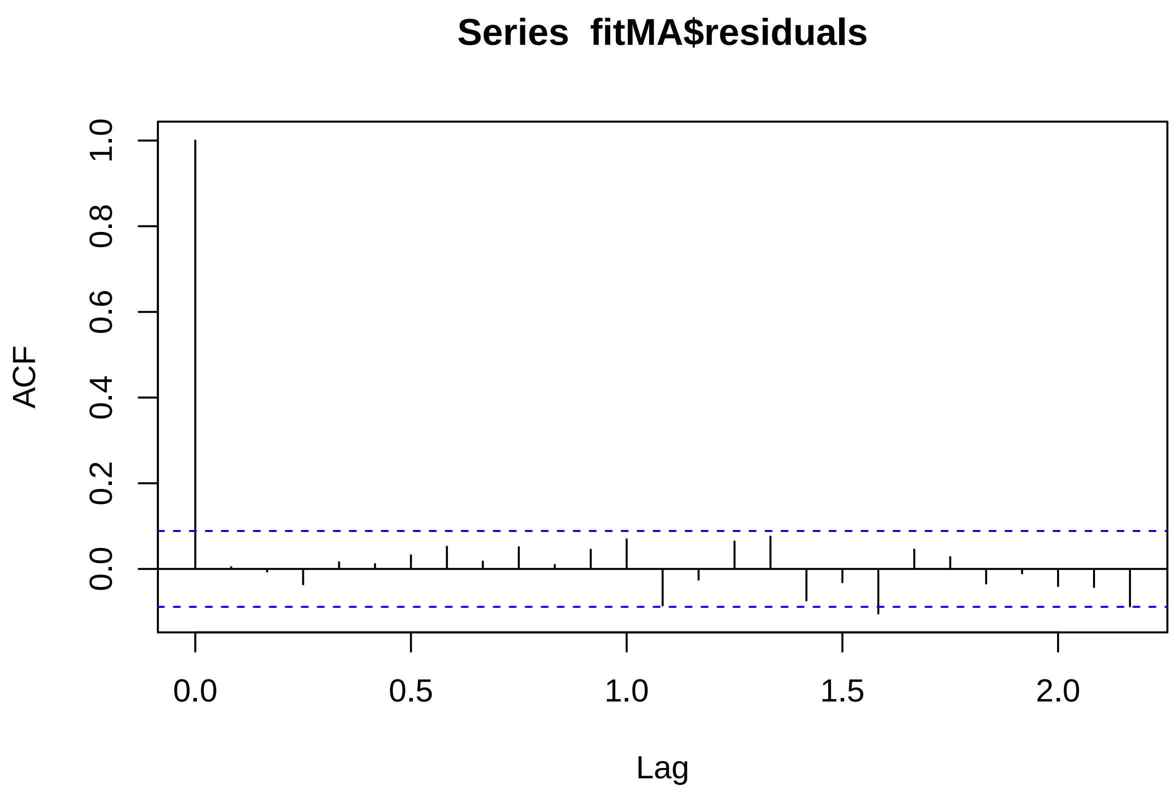
Next, we will fit MA (moving average process)

```r
fitMA=arima(df_inflation, order = c(0,0,3))
fitMA
```

```
##
## Call:
## arima(x = df_inflation, order = c(0, 0, 3))
##
## Coefficients:
##          ma1      ma2      ma3  intercept
##       -0.633  -0.1027  -0.1082    -0.0002
## s.e.   0.046   0.0514   0.0470     0.0209
##
```

```
## sigma^2 estimated as 8.505:  log likelihood = -1220.26,  aic = 2450.52
```

```r
acf(fitMA$residuals)
```




```r
Box.test(fitMA$residuals, lag=5, type="Ljung-Box", fitdf=1)
```

```
##
##  Box-Ljung test
##
## data:  fitMA$residuals
## X-squared = 0.86202, df = 4, p-value = 0.9299
```

Yup, the MA(3) fits well.

Now, we test with ARMA(p,q) fitting

```r
fitARMA=auto.arima(inflation,max.p=5,max.q=5,max.d=0,max.P=0,max.Q=0,max.D=0,ic="aic")
fitARMA
```

```
## Series: inflation
## ARIMA(2,0,1) with non-zero mean
##
## Coefficients:
##          ar1      ar2      ma1  intercept
##       1.2074  -0.2237  -0.8573     4.1068
## s.e.  0.0587   0.0551   0.0337     1.0692
##
## sigma^2 estimated as 8.533:  log likelihood=-1221.52
## AIC=2453.03   AICc=2453.16   BIC=2474.02
```

From above, we have 1.2074>1, we might want to test for stationary process. Now we try to find root for 1-ax-bx^2 to see if the process is stationary. Note that if there exists a non-unit root, the process is stationary.

```
polyroot(c(1,-1.2074, 0.2237))
```

```
## [1] 1.021584+0i 4.375823-0i
```

Here we have two non-unit roots; thus, the process is stationary. We also have different tests for stationary process such as followings:

```
library("tseries")
adf.test(inflation)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  inflation
## Dickey-Fuller = -3.8651, Lag order = 7, p-value = 0.01576
## alternative hypothesis: stationary
```

```
pp.test(inflation)
```

```
## Warning in pp.test(inflation): p-value smaller than printed p-value
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  inflation
## Dickey-Fuller Z(alpha) = -248.75, Truncation lag parameter = 5,
## p-value = 0.01
## alternative hypothesis: stationary
```

```
kpss.test(inflation)
```

```
## Warning in kpss.test(inflation): p-value smaller than printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  inflation
## KPSS Level = 2.51, Truncation lag parameter = 5, p-value = 0.01
```

Just stationary like we stated.

Now we fit ARIMA process to the inflation data

```
fitARIMA=auto.arima(inflation,max.P=0,max.Q=0,max.D=0,ic="aic")
fitARIMA
```

```
## Series: inflation
## ARIMA(1,1,1)
##
## Coefficients:
##          ar1      ma1
##       0.2383  -0.8772
## s.e.  0.0550   0.0269
##
## sigma^2 estimated as 8.587:  log likelihood=-1221.62
## AIC=2449.25   AICc=2449.29   BIC=2461.83
```

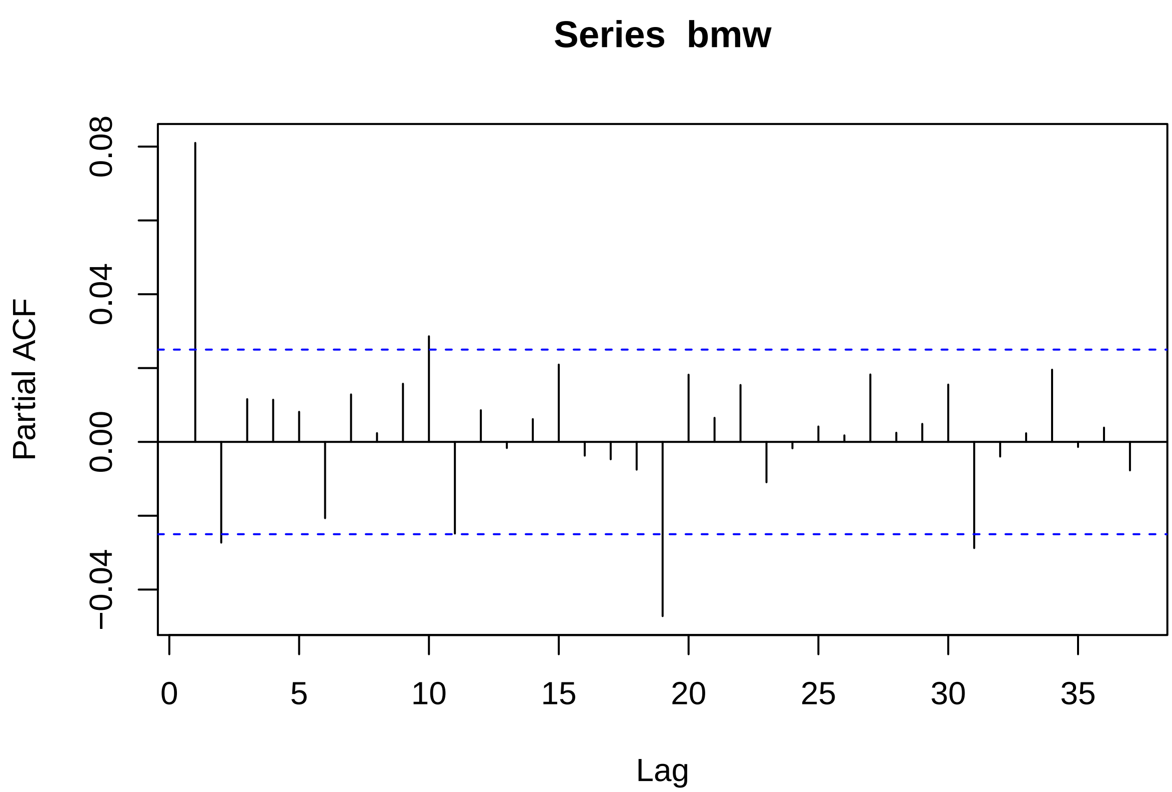
```r
predict(fitARIMA,n.ahead=10)
```

```
## $pred
##           Jan      Feb      Mar      Apr      May      Jun      Jul
## 1991 3.706101 4.589298 4.799773 4.849930 4.861884 4.864732 4.865411
##           Aug      Sep      Oct
## 1991 4.865573 4.865611 4.865620
##
## $se
##           Jan      Feb      Mar      Apr      May      Jun      Jul
## 1991 2.930391 3.115623 3.175183 3.215216 3.250915 3.285350 3.319222
##           Aug      Sep      Oct
## 1991 3.352703 3.385842 3.418657
```

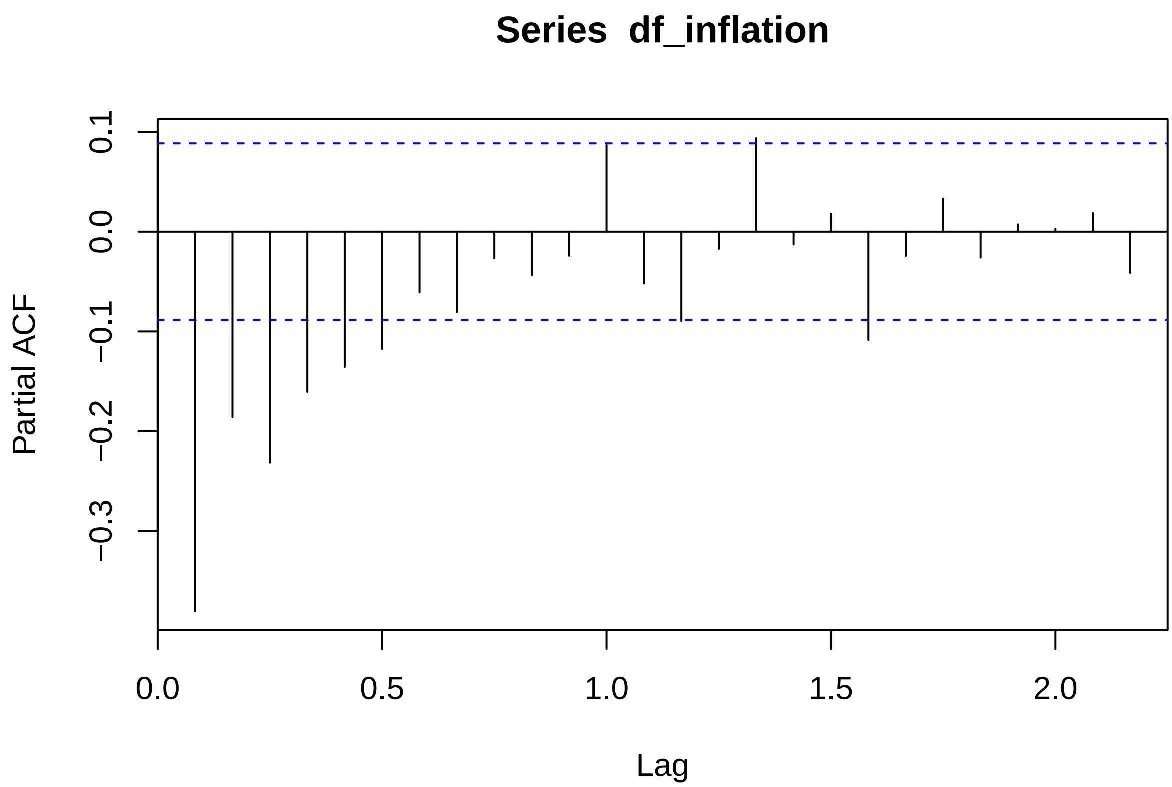
This process is parsimonious since we only need 3 parameters, reducing the problem of overfitting the model. We also predict the next 10 months based on this ARIMA(1,1,1) process. Notice that the predicted value is increasing with the also increasing standard deviation.

For this last part, we fit the inflation data with sample partial autocorrelation function.

```r
pacf(bmw)
```




```r
pacf(df_inflation)
```



The PACF is useful to identify the order of the AR process. The bmw log returns can be modeled as AR(1), while the df_inflation should try MA process instead, which is consistent with the analysis above.