Statistical Methods with Application to Finance

R. Pappadà

a.a. 2023-2024

Lab 2

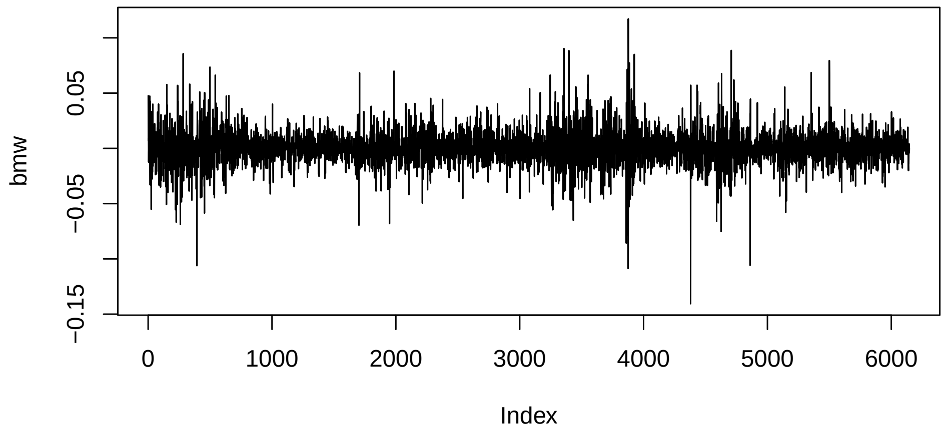
Fitting AR Models: BMW stock data

We start loading the require packages:

```
library(evir)
library(Ecdat)
library(tseries)
library(lmtest)
```

The data set `bmw` in **evir** contains 6146 daily returns on BMW stock from January 3, 1973 to July 23, 1996. Run the following code to obtain the data.

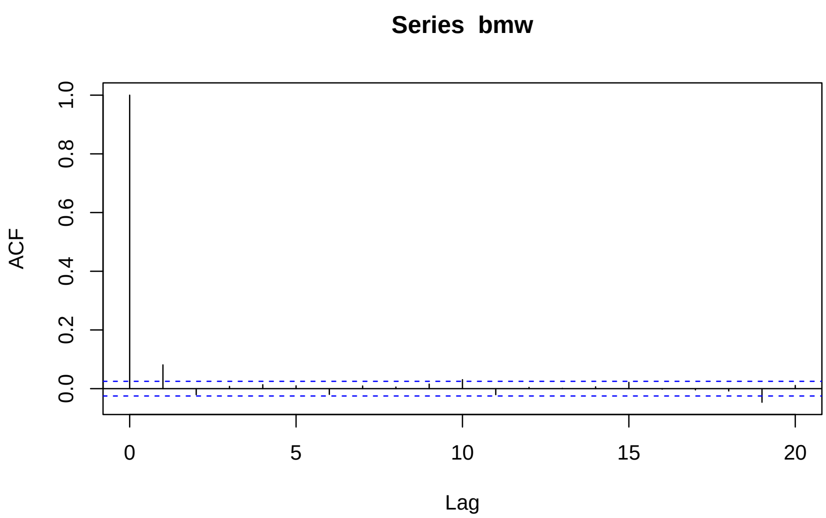
```
#Obtain BMW daily log returns
data(bmw)
plot(bmw, type="l")          #plot of BMW daily log-returns
```



ACF and AR fit.

The sample ACF is obtained via the `acf()` function and displayed below.

```
acf(bmw,lag.max=20)
```



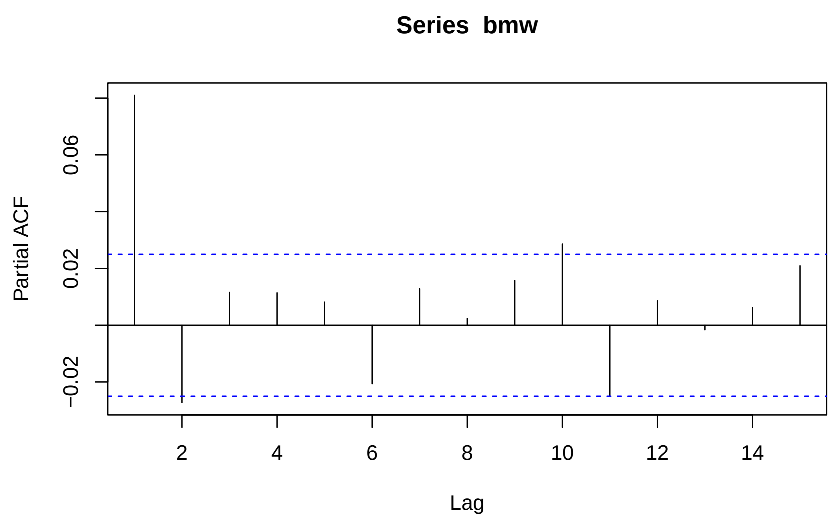
The autocorrelation coefficient at lag 1 is well outside the confidence bounds, so the series has some dependence. The Ljung-Box test can be implemented via the `Box.test()` function; here, the number of autocorrelation coefficients to test is set equal to 5 (other choices give similar results).

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

##
##  Box-Ljung test
##
## data:  bmw
## X-squared = 44.98701, df = 5, p-value = 1.45972e-08
```

The p-value is very small, indicating that that we can reject the null hypothesis that the first lag autocorrelations are 0.

The sample PACF for the BMW log returns is estimated using the **R** function `pacf()`:

```r
pacf(bmw, lag.max =15)
```



The lag-1 PACF is significantly different from zero and large with respect to values at higher lags. Thus, an AR(1) model is fit to data via the **R** function `arima()`:

```r
#fit an AR model of order 1
fitAR1 <- arima(bmw, order = c(1,0,0))
fitAR1

##
## Call:
## arima(x = bmw, order = c(1, 0, 0))
##
## Coefficients:
```

```
##            ar1  intercept
##       0.081116   0.000340
## s.e.  0.012722   0.000205
##
## sigma^2 estimated as 0.00021626:  log likelihood = 17212.34,  aic = -34418.68
```

The AR(1) coefficient estimate is $\hat{\phi}_1 = 0.0811$, with standard error 0.013. The t-ratio gives $0.0811/0.013 = 6.2$, thus it is statistically significant at 1% level. The `intercept` in the output is actually the estimated mean $\hat{\mu} = 0.00034$, so that the intercept is computed as

$$\hat{\phi}_0 = \hat{\mu}(1 - \hat{\phi}_1)$$

```
0.00034*(1-0.0811)    # compute phi(0)

## [1] 0.000312426

sqrt(fitAR1$sigma2)   # compute residual standard error

## [1] 0.0147057692

# compute t-ratio
fitAR1$coef/sqrt(diag(fitAR1$var.coef))

##        ar1  intercept
## 6.37590342 1.66458115

coeftest(fitAR1)

##
## z test of coefficients:
##
##                Estimate   Std. Error z value   Pr(>|z|)
## ar1       0.0811162792 0.0127223193 6.37590 1.8189e-10 ***
## intercept 0.0003404677 0.0002045365 1.66458   0.095996 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The fitted model is

$$y_t = 0.00031 + 0.08112\, y_{t-1} + \hat{a}_t, \quad \hat{\sigma}_a = 0.0147$$

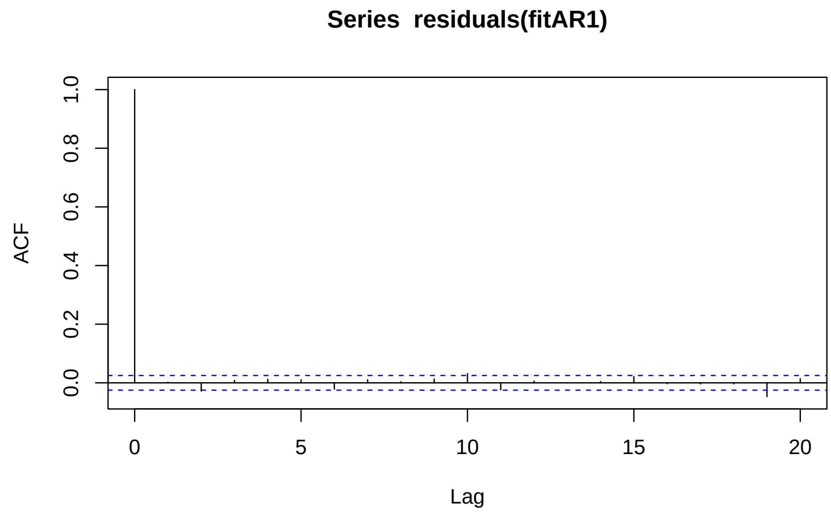
where $\hat{a}_t$ are the *residuals* from the model.

Residual analysis and model checking.

The ACF and the Ljung-Box test on the residuals can be used to check the closeness of $\hat{a}_t$ to a white noise. If there is still evidence of serial correlation in the residual series, then this suggests that a good model has not yet been found.

For the BMW log-returns, the sample ACF of the residuals with 95% confidence bounds from an AR(1) fit is obtained by running the following code.

```
acf(residuals(fitAR1), lag.max=20)
```



None of the autocorrelations at low lags are significant. The Ljung-Box test was applied, with lag equal to 5, and `fitdt=1`, the number of autoregressive coefficient parameters that were estimated. Such adjustment is done in order to account for the use of $\hat{\phi}_1$ in place of the unknown ϕ_1 when the test is applied to residuals.

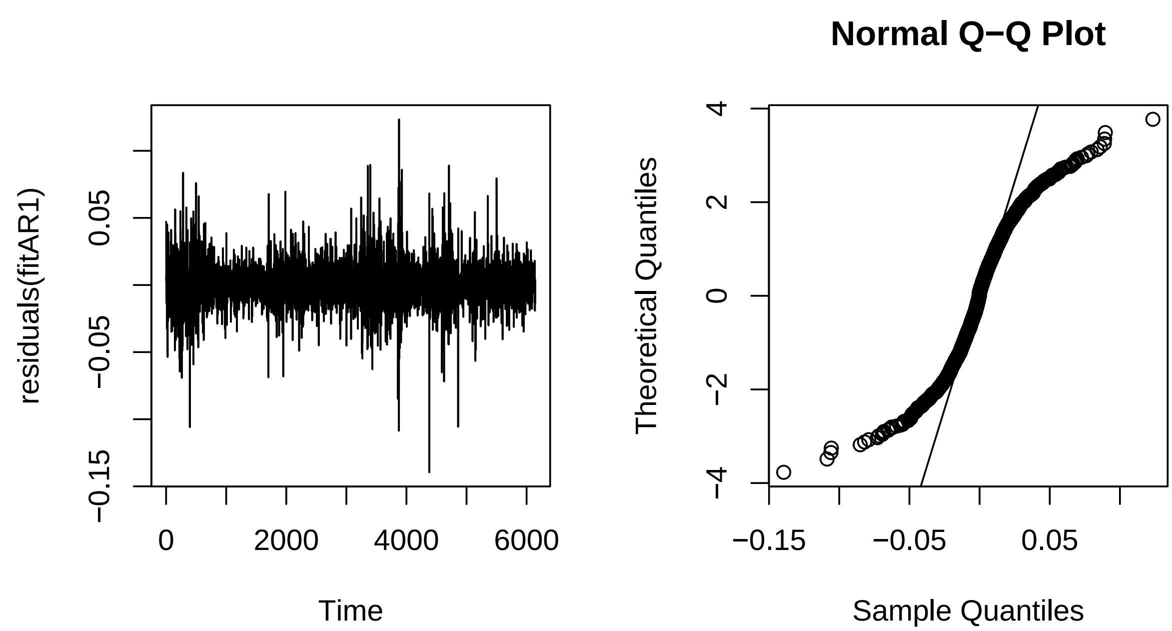
```
Box.test(residuals(fitAR1), lag = 5, type = "Ljung-Box", fitdf = 1)

##
##  Box-Ljung test
##
## data:  residuals(fitAR1)
## X-squared = 6.866933, df = 4, p-value = 0.14309
```

The result suggests that we can accept the null hypothesis that the residuals are uncorrelated, at least at small lags. This may indicate that the $AR(1)$ model provides an adequate fit. However, if the Ljung-Box test is performed at different lags (for instance 10, 15, or 20) then the resulting p-values are statistically significant using the level 0.05.

```r
Box.test(residuals(fitAR1), lag = 15, type = "Ljung-Box", fitdf = 1)

##
##  Box-Ljung test
##
## data:  residuals(fitAR1)
## X-squared = 24.09947, df = 14, p-value = 0.0445707
```

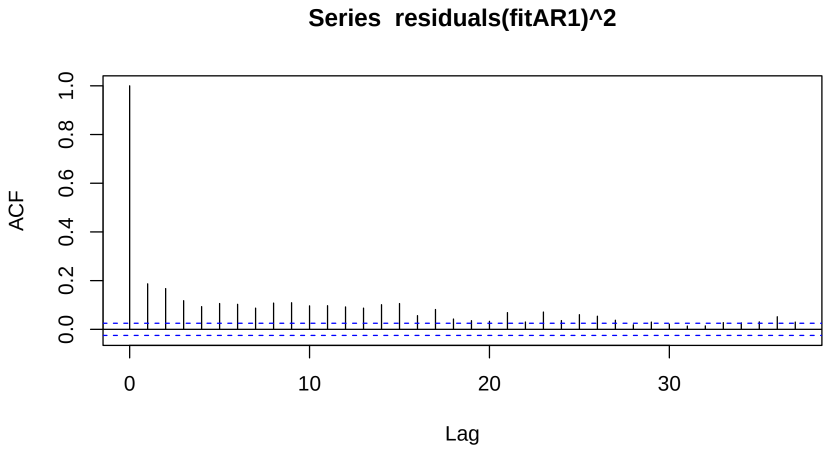
Moreover, the time series plot and normal QQ-plot plot of the $AR(1)$ residuals displayed below still show volatility clustering and heavy tails:

```r
plot(residuals(fitAR1))
qqnorm(residuals(fitAR1), datax=TRUE)
qqline(residuals(fitAR1), datax=TRUE)
```



Finally, we can test for residual heteroscedasticity by examining the ACF of the residuals from the fitted model: if the residuals are heteroscedastic, the squared residuals are autocorrelated:

```r
acf(residuals(fitAR1)^2)
```



EXERCISE. Consider the `bmw` log returns data and compare the fitted AR(1) model with alternative models, e.g., an AR(2) model:

1. check whether all parameters are statistically significant at the 5% level
2. plot the ACF and use the Ljung-Box test for examining the null hypothesis of serial uncorrelation at $m = 10$ and $m = 15$ lags;
3. Do you see significant differences between these models? What does the AIC suggest?

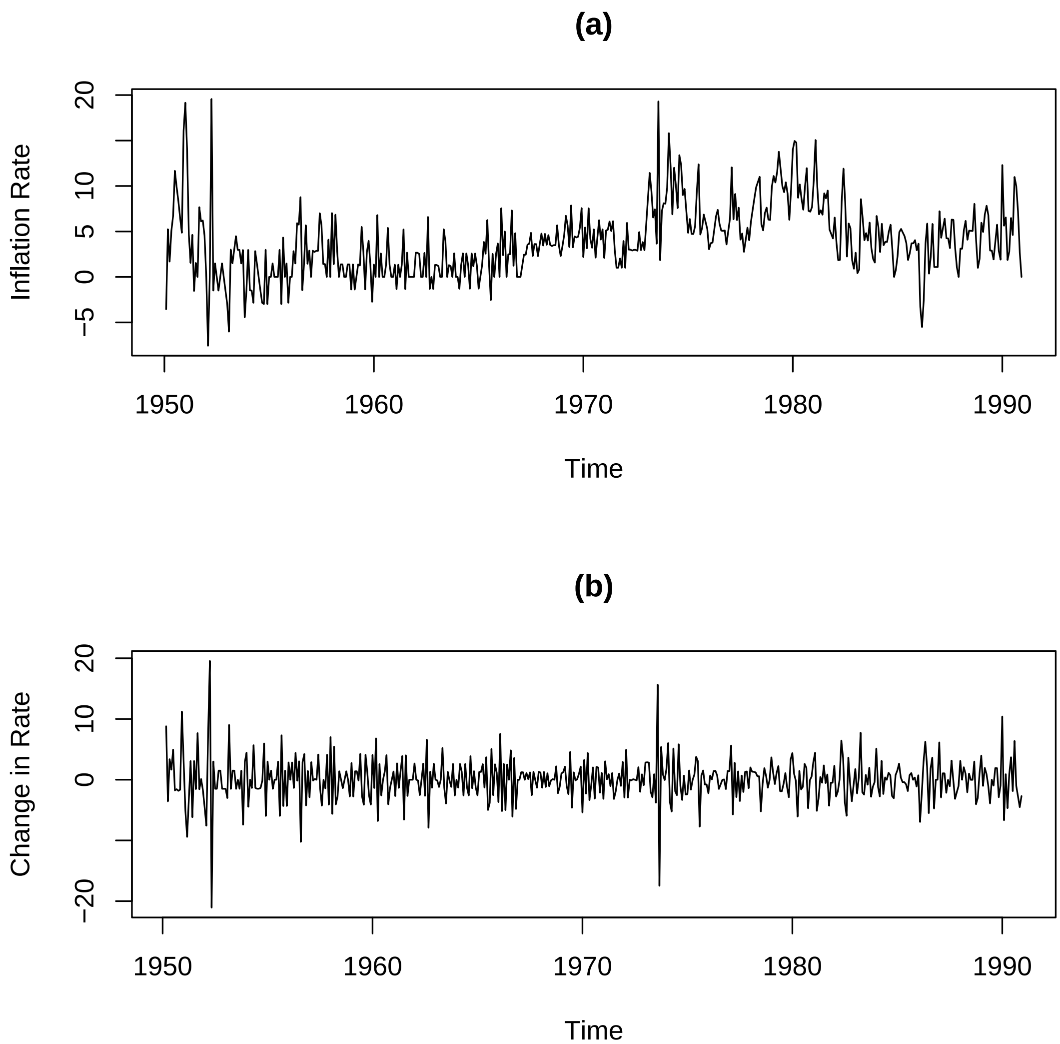
ARIMA modelling: US one-month inflation rate

In what follows we consider the one-month inflation rate (in percent, annual rate) data from 1950 to 1990. The data are contained in the first column of the `Mishkin` data set in the **Ecdat** package (491 observations):

```
data(Mishkin)
y <- as.ts(Mishkin[,1], start=1950, frequency=12)
```

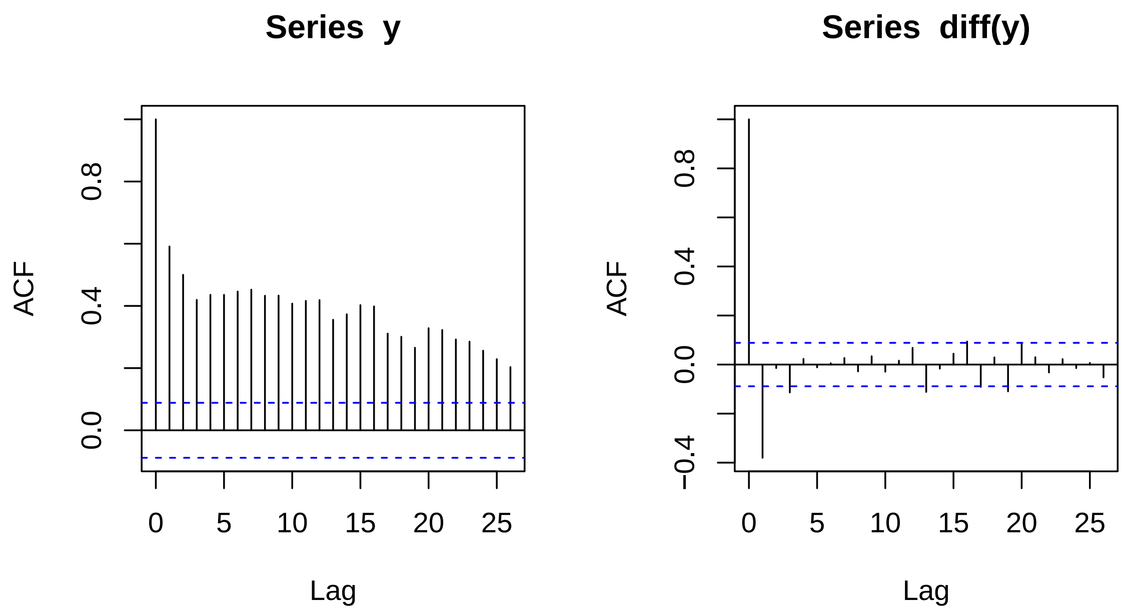
Obtain the time series plots of one-month inflation rate (annual rate, in percent) and first differences (changes) in the one-month inflation rate.

```
plot(y, ylab="Inflation Rate", type="l", main="(a)")
plot(diff(y), ylab="Change in Rate", type="l",  main="(b)")
```



Model identification and estimation

```
y <- as.vector(Mishkin[,1])
acf(y)
acf(diff(y))
Box.test(diff(y), lag=10, type="Ljung-Box")

##
##  Box-Ljung test
##
## data:  diff(y)
## X-squared = 79.91513, df = 10, p-value = 5.21694e-13
```



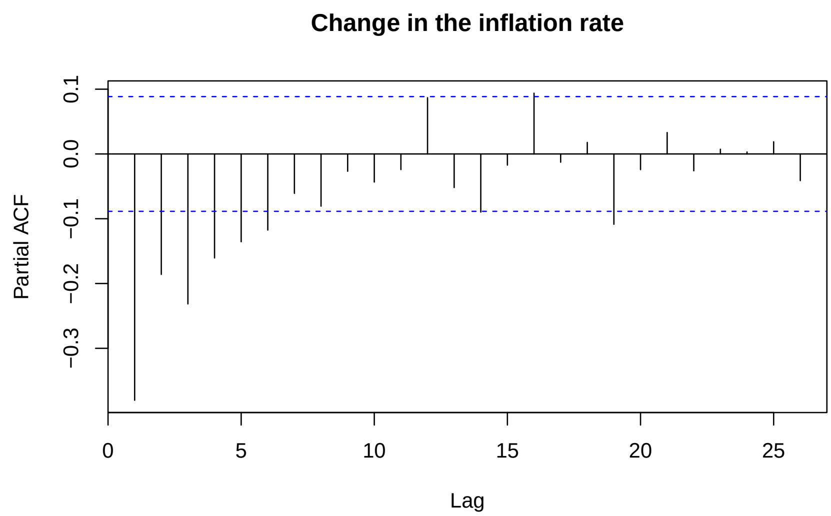
We see that the sample ACF plots of the one-month inflation rate decays to zero slowly. This is a sign of either nonstationarity or possibly of stationarity with long-memory dependence. In contrast, the sample ACF of the differenced series decays to zero quickly with significant ACF at lags 1, 3, and some marginally significant ACF at higher lags. Moreover, we perform the KPSS Test for stationarity.

```
kpss.test(y)

##
##  KPSS Test for Level Stationarity
##
## data:  y
## KPSS Level = 2.510004, Truncation lag parameter = 5, p-value = 0.01
```

The p-value is smaller than 0.01 and the null hypothesis of stationarity can be rejected. This confirms that the inflation rate time series is not stationary.

The sample PACF for the changes in the inflation rate is obtained via the following code:

```r
pacf(diff(y), main = "Change in the inflation rate")
```



The sample PACF decays slowly to zero, rather than showing a cut off as for an AR process. This is an indication that this time series should not be fit by a pure AR process. Other ARMA models (or pure MA models) that could provide a parsimonious fit for the first order differences of the inflation rates ($d = 1$).

We start considering an ARIMA(0,1,3) to the inflation rate time series (or ARIMA(0,0,3) model for the differenced series). The function `arima()` is used for fitting ARIMA models by specifying the orders p, d, q.

```r
fitMA3 <- arima(y, order=c(0,1,3))
fitMA3

##
## Call:
## arima(x = y, order = c(0, 1, 3))
##
## Coefficients:
##             ma1        ma2        ma3
##       -0.632949  -0.102734  -0.108171
## s.e.   0.046017   0.051399   0.046985
##
## sigma^2 estimated as 8.5046:  log likelihood = -1220.26,  aic = 2448.52
```

The t-ratio for the the first MA coefficient $-0.633/0.046 = -13.76$ leads to rejection of the null hypothesis that the coefficient is zero. Similarly, the other MA coefficients are significant at level 0.05.

We can use model selection criteria to compare different MA models

```
fitMA1 <- arima(y, order=c(0,1,1))
fitMA2 <- arima(y, order=c(0,1,2))
```

```
fitMA3$aic

## [1] 2448.52435

fitMA2$aic

## [1] 2451.69507

fitMA1$aic

## [1] 2465.70467
```

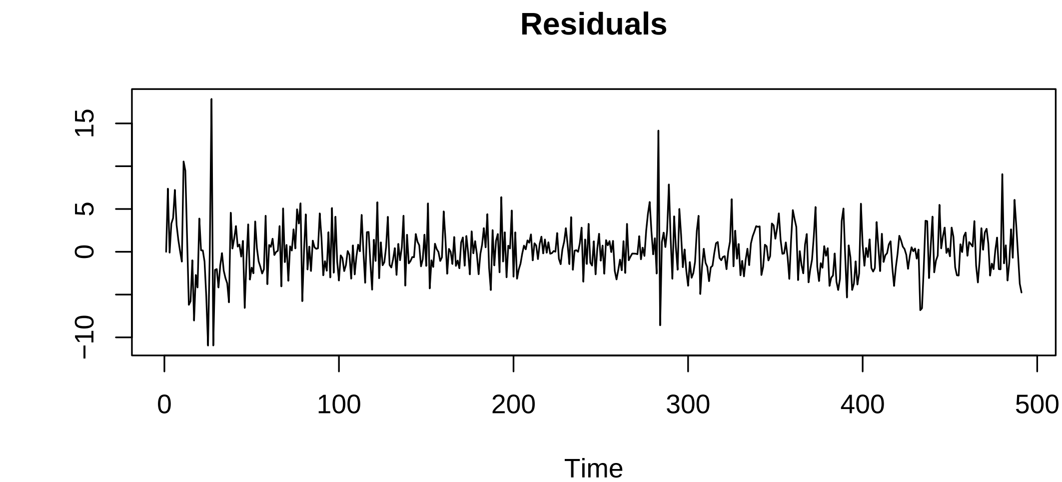
According to the AIC the ARIMA(0,0,3) model provides the best-fit to `diff(y)`.

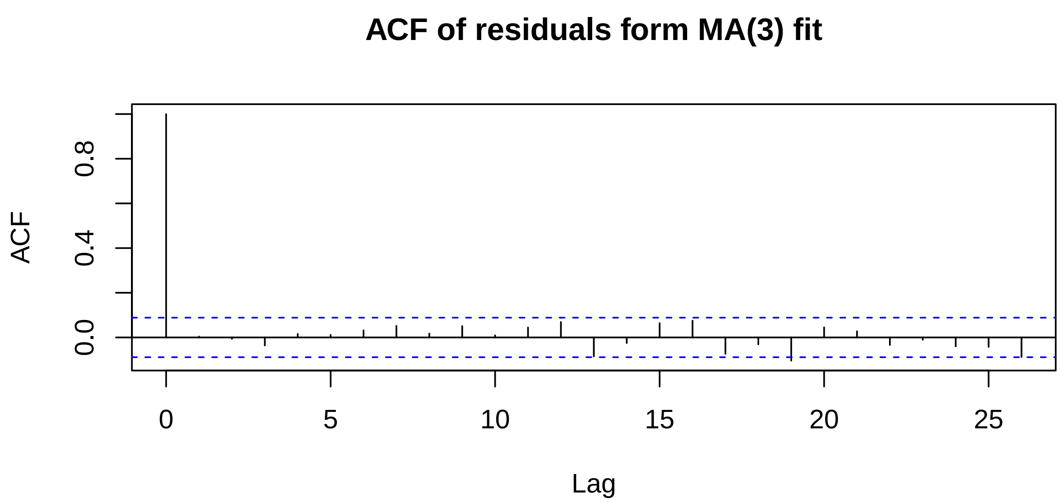
Model checking

If the model fits well, the residuals should approximately behave as white noise. In practice, we look at the time plot, the correlogram, and the results from the Ljung-Box test on residuals. The degrees of freedom of this statistic take into account the number of estimated parameters so the statistic test under H_0 follows approximately a $\chi^2_{(m-g)}$ distribution with $g = p + q$ (in `Box.test` set `fitdf = 3`).

```
Box.test(residuals(fitMA3), type = "Ljung", lag = 24, fitdf = 3)

##
##  Box-Ljung test
##
## data:  residuals(fitMA3)
## X-squared = 28.37414, df = 21, p-value = 0.129856
```

```
#useful plots for residual analysis
par(mfrow=c(2,1))
plot(residuals(fitMA3), type="l", ylab=" ", xlab="Time", main="Residuals")
acf(residuals(fitMA3), main="ACF of residuals form MA(3) fit")
```





The plot shows the standardized residuals and their ACF. Inspection of the time plot of the residuals shows no obvious patterns, but the series exhibits some outliers. The ACF of the residuals shows no apparent departure from the model assumptions in that the residual autocorrelations lie within the approximate normal limits (dotted blue lines).

Alternative ARIMA models

An ARIMA (1,1,1) model could also be tried as an alternative model for the inflation rate series. Model comparison can be carried out by using either AIC (or BIC) and residual analysis.

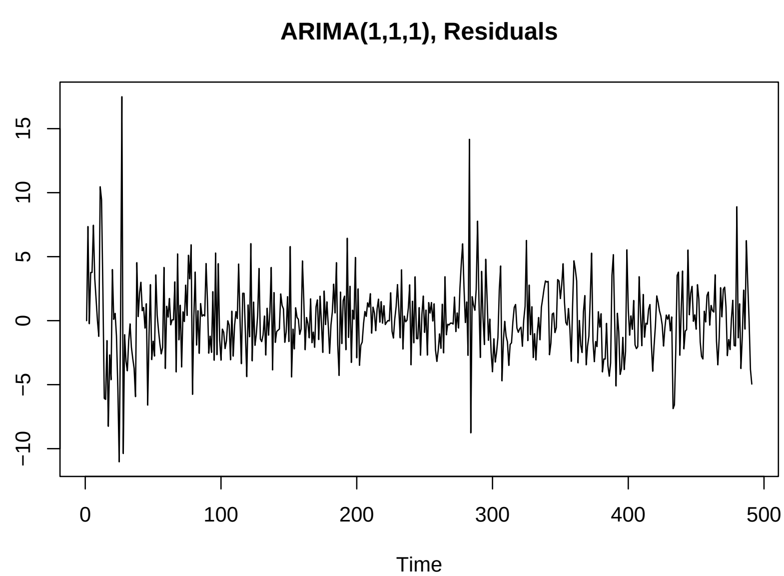
```
fit_arima <- arima(y, order=c(1,1,1))
fit_arima
```

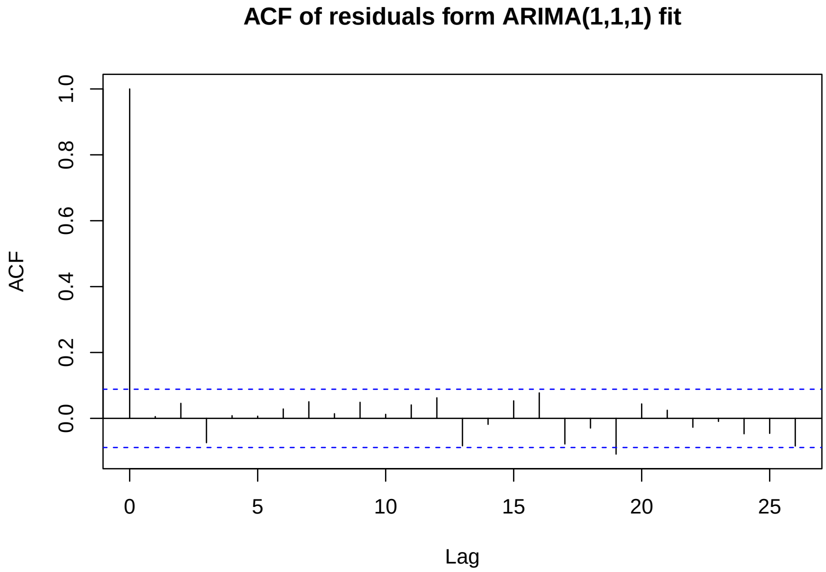
```
##
## Call:
## arima(x = y, order = c(1, 1, 1))
##
## Coefficients:
##              ar1        ma1
##         0.238309  -0.877177
## s.e.    0.055043   0.026929
##
## sigma^2 estimated as 8.55214:  log likelihood = -1221.62,  aic = 2449.25
```

The parameter estimates are $\hat{\phi}_1 = 0.2383$ and $\hat{\theta}_1 = -0.8772$, with standard errors 0.0550 and 0.0269, respectively. Hence, both coefficients are significant. If x_t denotes changes in the one-month inflation rate, ∇y_t, then the fitted model is

$$x_t = 0.2383\, x_{t-1} - 0.8772\, a_{t-1} + \hat{a}_t, \quad \hat{\sigma}_a^2 = 8.55$$

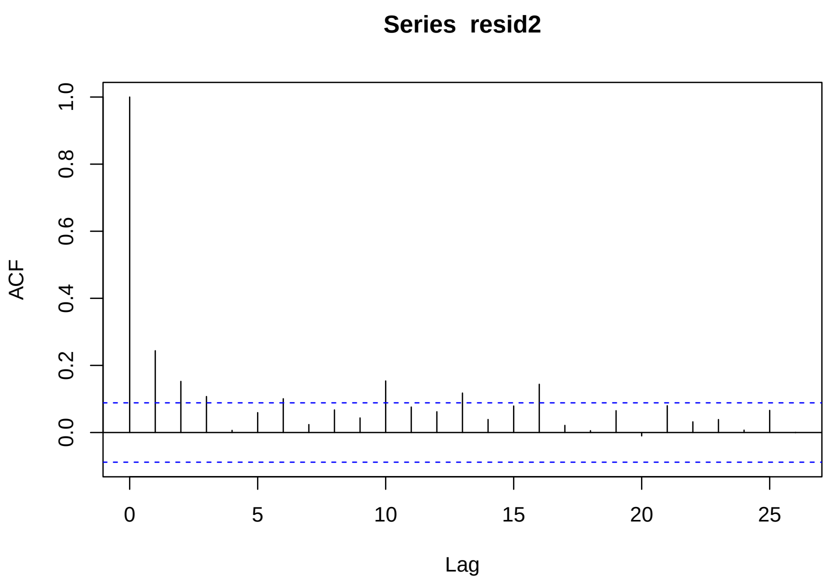
```
Box.test(residuals(fit_arima), lag=20, fitdf=2)

##
##  Box-Pierce test
##
## data:  residuals(fit_arima)
## X-squared = 27.49744, df = 18, p-value = 0.0701265
```





This is a very parsimonious model and residual diagnostics show that it fits well. Still, the plot of standardized residuals indicates the presence of big residuals in the data. By executing the following code, we can examine the autocorrelation in the mean-centered squared residuals:

```
## Testing for ARCH effect
resid2 <- (resid(fit_arima) - mean(resid(fit_arima)))^2
acf(resid2)
```



```
Box.test(resid2, lag = 12, type="Ljung-Box")

##
##  Box-Ljung test
##
## data:  resid2
## X-squared = 73.62932, df = 12, p-value = 6.66671e-11
```

Although there is no serial correlation in the residual series, the series of squared residuals shows profound serial correlation, that is referred to as *ARCH effect*. This implies that the residuals do not behave like a realization of a strict white noise process and that the models for non constant volatility may be required.

EXERCISE. Consider the Stock Price series of Dow Jones Industrial Average (DJIA) in the period 2018–2020; to obtain the data, run the code

```
library(quantmod)
getSymbols("^DJI", from=as.Date("2018-02-01"),
           to=as.Date("2020-12-31"))
```

1. Build log-returns and examine the transformed data (describe the data using a histogram and a normal QQ-plot);
2. Plot the ACF and use the Ljung-Box test to check for the presence of serial correlation;
3. Consider different ARIMA models for the log-returns and compare them via AIC and BIC.
4. Select a model for the log-returns and perform residual analysis. Finally, plot the ACF of squared residuals and comment the result.

Forecasting the one-month inflation rate

We saw that an $ARIMA(1,1,1)$ model provided a good fit to one-month inflation rates. This implies that an $ARMA(1,1)$ model can be appropriate for the changes in the inflation rates, x_t. We derive point forecasts and prediction intervals (P.I.s) for x_t using the model equation.

First, refit the chosen model with the following code

```
fit_diff<-arima(diff(y), c(1,0,1), include.mean = F)
```

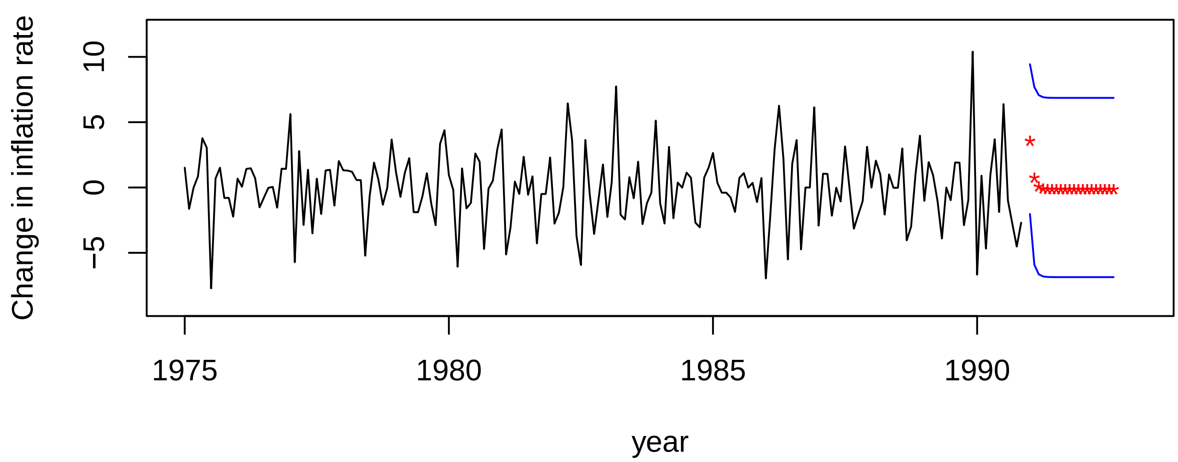
The $ARIMA(1,1,1)$ model is also estimated using the original data.

```r
fit<-arima(y, c(1,1,1))
```

The two models are equivalent, but they lead to different forecasts. Then, we can use function `predict()` to forecast from the fitted models.

The function `predict()` returns the time series of predictions, and the estimated standard errors. The plot below displays point forecasts and 95% P.I.s from the $ARMA(1,1)$ and $ARIMA(1,1,1)$ fit, out to 20 steps ahead. In the latter case, as expected for a non-stationary process, the forecast limits diverge.

```r
t1 <- 300:491
t2 <- 492:511
# monthly observations from 1950-2 to 1990-12
year <- seq(1950 + 1/12, 2000, by=1/12)

pred.infl_diff <- predict(fit_diff, n.ahead = 20)

plot(year[t1], diff(y)[t1], xlim=c(1975, 1993), ylim=c(-9,12),
     type="l", xlab="year", ylab="Change in inflation rate")
points(year[t2], pred.infl_diff$pred, type="p", pch="*", col="red")
lines(year[t2], pred.infl_diff$pred - 1.96*pred.infl_diff$se, col="blue")
lines(year[t2], pred.infl_diff$pred + 1.96*pred.infl_diff$se, col="blue")
```




```r
pred.infl <- predict(fit, n.ahead = 20, se.fit = TRUE)

plot(year[t1],y[t1], xlim=c(1975,1993), ylim=c(-10,18),
```

```
     type="l", xlab="year",ylab="Inflation rate")
points(year[t2], pred.infl$pred,type="p", pch="*", col="red")
lines(year[t2], pred.infl$pred - 1.96*pred.infl$se, col="blue")
lines(year[t2], pred.infl$pred + 1.96*pred.infl$se, col="blue")
```