Project I

TMS088, Financial Time Series

Erik Jansson Jacob Lindbäck

March 2018

Introduction

Forecasting and time series modelling is of major importance in risk assessment and decision making. In this paper, we will test how *best linear predictors* perform on exchange rate data. A crucial assumption that will be utilized is that the underlying time series is stationary. This assumption will be addressed for different transformations of the time series by using graphical methods. The given time series are measurements of the Australian weighted trade index, which is a weighted sum of exchange rates between the Australian dollar and several other currencies. The data set includes observations at 205 subsequent months from January 1978, which will be denoted $(X_t, t = 1, 2, 3..., 205)$ where t is number of months from January 1978.

Data Exploration (Task 1)

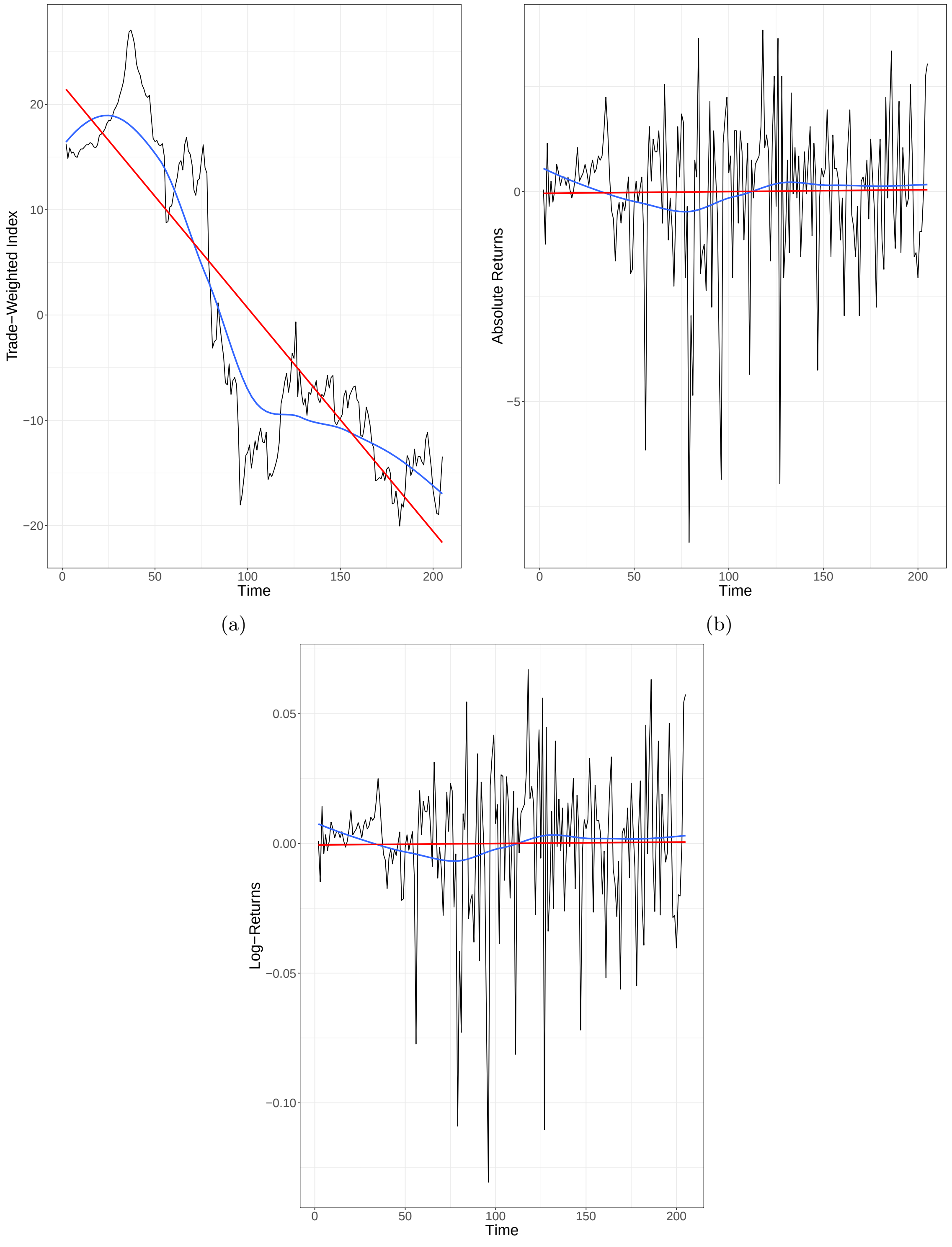
We begin by performing some data transformation. The first thing to do is to mean-correct, that is to say, subtract the sample mean from the time series. The reason for doing this is mainly to obtain nicer properties which simplifies the work, for instance will a stationary time series be centered around zero. One may then form the absolute returns, that is,

$$Y_t = X_t - X_{t-1}.$$

The absolute returns are also mean-corrected. The reason for this data transformation is that while X may exhibit a lot of dependence on earlier values, it could be so that the returns could be more accurate to describe as stationary, which simplifies modelling We then calculate the the logarithmic returns given by (hereafter log-returns).

$$Z_t = \log X_t - \log X_{t-1}$$

and Z is mean-corrected as well. We form the log-returns because it is customary to treat financial data as lognormal. This would then imply that log-returns are Gaussian, and Gaussian time series are generally easy to work with. One should note, however, that as we later will see, the lognormal assumption not always fitting.



(a)

(b)

(c)

Figure 1: The time series considered (Trade-Weighted Index) together with two different data transformations. The superimposed red lines are least squares estimates, and the blue curve is estimated with a local regression method. All time series have been mean-corrected.

In order to address the issue of stationarity, we recall that a time series $(X_t, t \in \mathbb{Z})$ is *(wide sense) stationary* if:

1. $\text{Var}(X_t) < +\infty, \ \forall t \in \mathbb{Z}$, that is to say, the variance is finite for all times.
2. $\exists \mu \in \mathbb{R} \ , s.t, \mu_X(t) = \mu_X, \ \forall t \in \mathbb{Z}$, or constant mean.
3. $\forall s, t, h \in \gamma_X(s,t) = \text{Cov}(X_s, X_t) = \gamma_X(s+h, t+h)$, meaning that the autocovariance function only is a function of the lag.

Note that from figure 1a that is a clear decreasing tendency for the trade weighted index. Thus, the assumption of constant mean is severely violated in this case. Assuming constant mean for the transformed series seems reasonable, since estimated trends are almost constant. In particular, it will be shown in section 5 that if the underlying trend of the trade-weighted index is linear, the mean of absolute returns is constant.

It is difficult to definitely conclude that the variance is finite in each case using only the figures, however, we might be fairly confident in assuming that such is the case, since we do not see any fantastically large deviations and given the real-world nature of the data, infinite variance is unlikely.

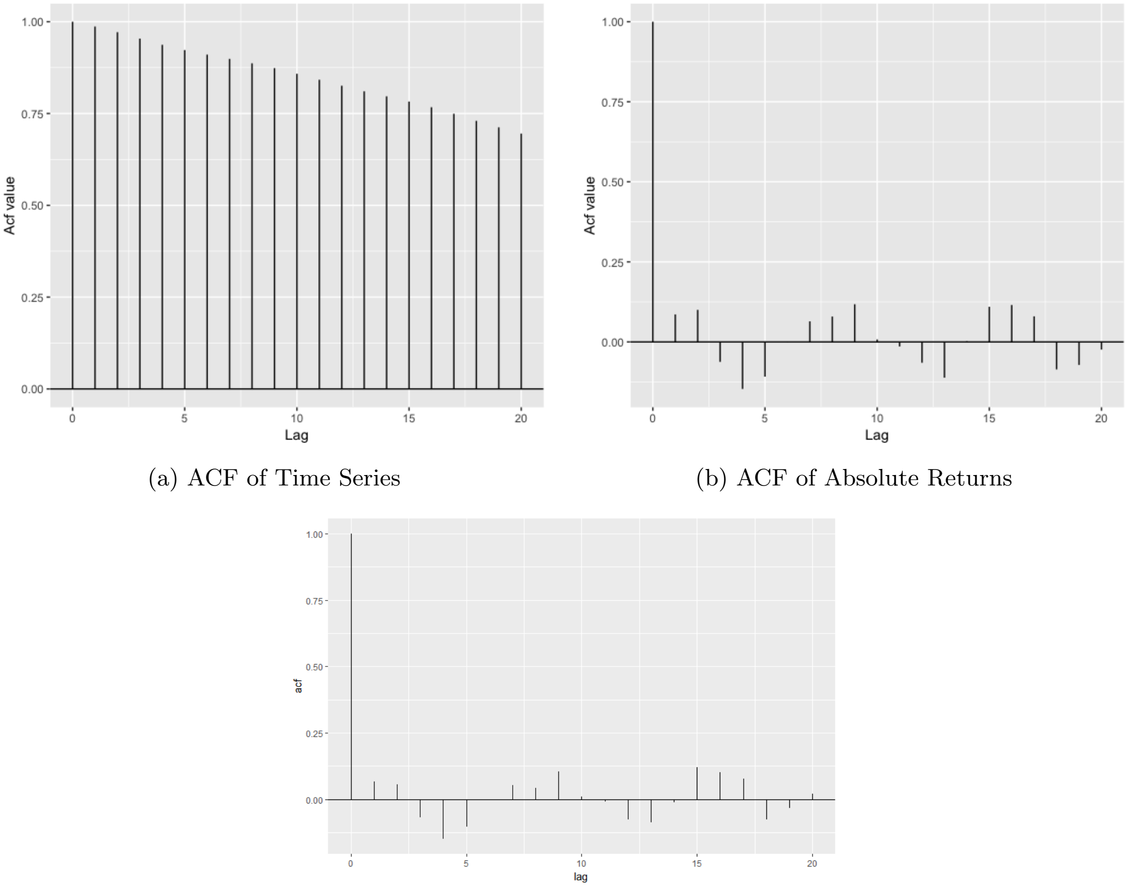
By simply inspecting figures 1b and 1c we may guess that the covariance only depends on the lag, but it difficult to infer that such is the case using only the figures. While there is no clear indication that such is not the case, in order to confidently conclude that stationarity is a reasonable assumption some other test should be performed.

Ljung-Box tests and autocovariance functions (Task 2)

We now turn our attention to the autocovariance function of our time series. The ACF of a time series X_t is given by

$$\rho_X(h) = \frac{\gamma_X(h)}{\gamma_X(0)}.$$

We plot these for the three time series in figure



(a) ACF of Time Series

(b) ACF of Absolute Returns

(c) ACF of Logarithmic Returns

Figure 2: The ACVFs for the different time series, plotted against time.

If we consider figure 2 we see that the returns seem to be fairly uncorrelated, whereas there is much evidence for correlation in the exhange rate time series, seeing as the ACF is large for all lags. In this section we want to use the Ljung-Box portmanteau test in order to test if:

- H_0: $\rho(1) = \rho(2) = ... = \rho(h) = 0$.
- H_1: $\rho(i) \neq 0, \text{ for some } i \in \{1, 2, \ldots h\}$.

In other words, we test for serial correlation. The Ljung-Box test statistic is given by

$$T = n(n+2)\sum_{j=1}^{h} \frac{\hat{\rho}^2(j)}{n-j}.$$

where $\hat{\rho}$ is the sample ACF, given by

$$\hat{\rho}_X(h) = \frac{\hat{\gamma}_X(h)}{\hat{\gamma}_X(0)}, \quad \hat{\gamma}_X(h) = \frac{1}{n}\sum_{i=1}^{n-h}(X_{i+h} - \bar{X}_n)(X_i - \bar{X}_n)$$

T is χ_h^2 under H_0. We perform the Ljung-Box test on each of the three time series. The results are reported in 1

	Intrinsic values	Log-returns	Absolute returns
Value of test statistic	3136.273	24.24897	30.39973
P-value	~ 0	0.232	0.064
Reject at $\alpha = 0.05$	Yes	No	No

Table 1: Results from performing Ljung-Box tests

The results of the Ljung-Box tests are in line with what we expect given the plots of the ACF:s. We see that there is overwhelming evidence that the intrinsic values exhibit serial correlation, and while we do not reject the null hypothesis for the absolute returns the p-value is only 0.064 compared with a p-value of 0.232 for the log-returns and hence we can be more confident when we fail to reject the null hypothesis for the log-returns.

Finding a predictive model (Task 3)

Among all linear predictors that estimate X_{t+h} on the following form:

$$\ell_n(X_{t+h}) = a_0 + \sum_{i=1}^{n} a_i X_{t+1-i}$$

the one that minimizes the the mean square error: $\mathbb{E}(X_{t+h} - \ell_n(X_{t+h}))^2 | X_t, \ldots X_{t+1-n})$ is called the best linear predictor. When the underlying time series model is stationary, the coefficients of the best linear predictor is given by the solution of the following linear system:

$$\Gamma_n(a_1, a_2 \ldots, a_n)' = \gamma_n, \quad a_0 = \mu(1 - \sum_{i=1}^{n} a_i)$$

where

$$\Gamma_n = [\gamma_X(i-j)]_{i,j=1}^{n-1}, \quad \gamma_n = [\gamma_X(h+i)]_{i=0}^{n-1}.$$

We will now test the performance of this linear predictor for $h = 1$ on the log-returns by partitioning the given observations into two equally sized sets,which will be used as a training data set and a test data set respectively. The training data set will be used to estimate the sample ACVF, which will substitute the theoretical ACVF used above. The 20 last observations will be used as explanatory variables in the predictive model. To test the performance of the predictive model, we will use the mean squared difference between the predictions and the observations given by:

$$\frac{1}{104}\sum_{i=103}^{204}(X_i - \hat{a}_1 X_{i-1} - \hat{a}_2 X_{i-2} - \cdots - \hat{a}_{20} X_{i-20})^2. \tag{1}$$

This quantity measures the performance of the model, where $\hat{a}_i$ denotes the estimated coefficients. Note that since the series has been mean corrected, a_0 can be omitted (since $\mu = 0$). Note that the best constant predictor is given by the mean, and since the series has been mean corrected, the following quantity can be used for comparison

$$\frac{1}{104}\sum_{i=103}^{204} X_i^2. \tag{2}$$

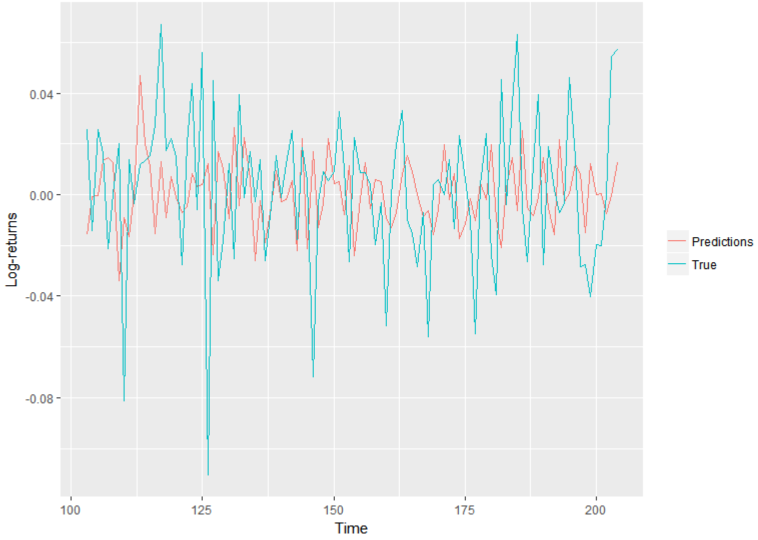


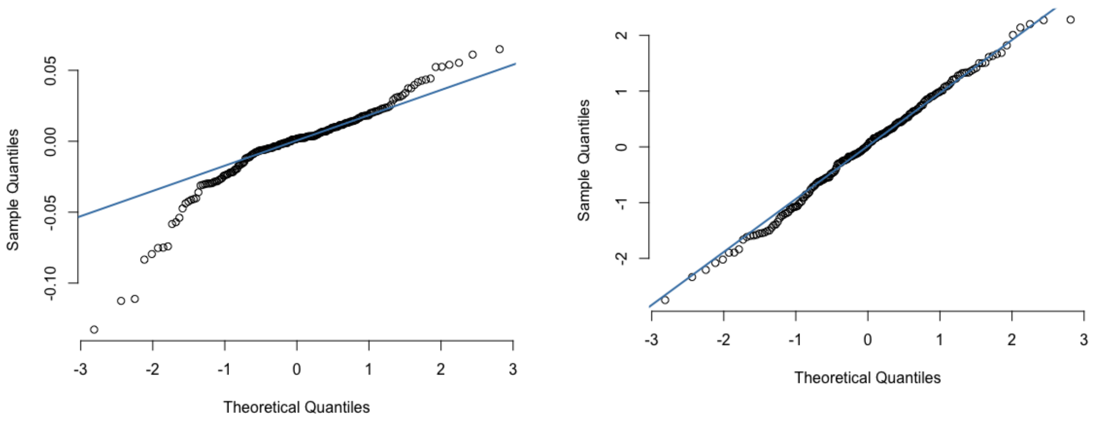
Figure 3: The predicted time series and the observed log returns.

Figure 3 illustrates the estimated predicted series and the observations in the test data. Note that estimated trend departures considerably from the true trend. Moreover, the estimated mean square error given by (1) is 0.00114, which is higher than that of the best constant predictor, which is : 0.00089. An explanation of why the best linear predictor performs worse is because the Ljung-Box portmanteau in section two accepts the hypothesis that the log returns are independent. In other words, the reason why sample ACVF is non-zero for positive lags might as well be because of statistical error. Since it cannot be rejected that $\gamma(h) = \gamma(0)I_{\{0\}}(h)$, it is more justified to model the stationary process as IID$(0, \sigma^2)$ rather than a series with the estimated ACVF. By using this model, the system from which the coefficients are obtained is diagonal with the zero vector on the right hand side, which implies that all coefficients equals to zero. So by assuming IID, the best linear predictor is indeed the constant predictor. Hence, it is reasonable that the best constant predictor outperforms the one based on the sample ACF.

Are Logarithmic returns Gaussian? (Task 4)

Should the log-returns be modeled as a Gaussian time series? This is something that is often used, since it is common to think of financial time series as being log-normal time series. In order to investigate this claim we may use a graphical approach by the method of QQ-plot.

A QQ-plot means that we plot the sorted data against the theoretical quantiles of the normal distribution, and if then the data is fairly normal it should be a fairly straight line. This is easily done using the base R command **qqnorm**. In order to get a better understanding and reason more accurately, we will make a qq-plot of some pseudorandom normal numbers. In figure **??** we see that the logarithmic returns have much heavier tails than the pseudorandom normal numbers, which indicates that we cannot say that the logreturns are normal.



(a) Normal QQ-plot of log-returns

(b) Normal QQ-plot of pseudorandom normal numbers

Figure 4

We now form the absolute values of the log-returns (which we also mean-correct) and plot the sample ACF as well as perform the Ljung-box test. As we see in table 2, there is overwhelming evidence to reject the null hypothesis. This means that we are confident that the absolute value of the log returns exhibit serial correlation. If one considers figure 5, we see that the ACF is slowly decreasing positive. This is an example of what is called volatility clustering, the fact that large positive deviations are followed by large negative ones, which serves as motivation for more advanced (e.g. (G)ARCH) models.

Table 2: Ljung-Box for the Absolute value of the Log-returns.

	Absolute-value of Log-returns
Value of test statistic	83.81742
P-value	8.747795e-10
Reject at $\alpha = 0.05$	Yes

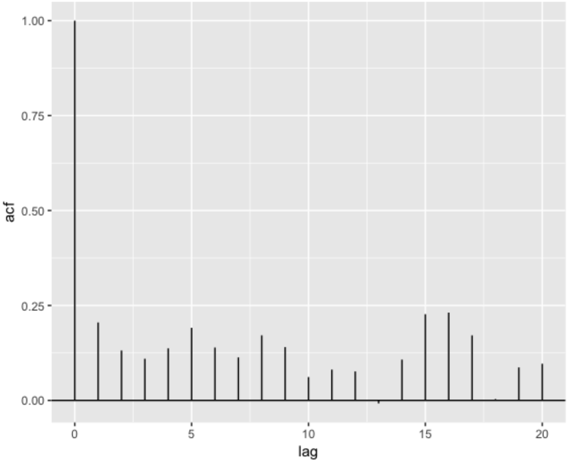


Figure 5: ACF of Absolute value of logreturns

Differentiation of polynomial trends (task 5)

Consider the time series model $X = (X_t,\, t \in \mathbb{Z})$ given by:

$$X_t = m_t + Y_t$$

with a polynomial trend component: $m_t = \sum_{j=0}^{q} a_j t^j$ and an additive stationary noise term Y_t with mean 0. We will now prove the following three properties of this time series model:

1. $\nabla^q m_t = q!\, a_q$ for $q = 1, 2 \dots$.
2. $\nabla^q Y_t$ is stationary.
3. $\nabla^q X_t$ is stationary with mean $q! a_q$.

The first claim will be proven by induction. For $q = 1$ we have that:

$$\begin{aligned} \nabla m_t = m_t - m_{t-1} &= (a_1 t + a_0) - (a_1(t-1) - a_0) \\ &= a_1 = 1!\, a_1 \end{aligned}$$

Thus, property 1 does indeed hold for $q = 1$. Assume that it is true some number $q = p - 1 \geq 1$, i.e.

$$\nabla^{p-1} m_t = \nabla^{p-1} \left(\sum_{j=0}^{p-1} a_j t^j \right) = (p-1)!\, a_{p-1}$$

If this implies that the claim is also true for $q = p$, it must hold for all $q \geq 1$.

$$
\begin{aligned}
\nabla^p m_t = \nabla^p \Bigg(\sum_{j=0}^{p} a_j t^j \Bigg) &= \nabla^p \Bigg(\sum_{j=0}^{p-1} a_j t^j + a_p t^p \Bigg) \\
&= \nabla \Bigg(\nabla^{p-1} \sum_{j=0}^{p-1} a_j t^j \Bigg) + \nabla^p a_p t^p \\
&= \nabla (p-1)!\, a_{p-1} + a_p \nabla^p t^p \\
&= a_p \nabla^p t^p .
\end{aligned}
$$

If $\nabla^p t^p = p!$ for $p = 1, 2, \ldots,$ we are done. This will also be shown by induction. For $p = 1$, the left hand side becomes: $\nabla t = t - (t-1) = 1 = 1!$. Now, we will assume that this is true for all $p \leq a-1$, i.e.

$$
\begin{aligned}
\nabla t =& 1! \\
\nabla^2 t =& 2! \\
& \vdots \\
\nabla^{a-1} t^{a-1} =& (a-1)!.
\end{aligned}
$$

For $p = a$ we have that:

$$
\begin{aligned}
\nabla^a t^a = \nabla^{a-1} \nabla t^a &= \nabla^{a-1} \Bigg(t^a - (t-1)^a \Bigg) \\
&\overset{(*)}{=} \nabla^{a-1} \Bigg(t^a - \sum_{i=0}^{a} \binom{a}{i} t^i \, (-1)^{a-i} \Bigg) \\
&= \nabla^{a-1} \Bigg(\sum_{i=0}^{a-1} \binom{a}{i} t^i \, (-1)^{a-i+1} \Bigg) \\
&\overset{(**)}{=} \sum_{i=0}^{a-2} \binom{a}{i} (-1)^{a-i+1} \underbrace{\nabla^{a-i-1} \underbrace{\nabla^i t^i}_{=i!}}_{=0} + \binom{a}{a-1} \underbrace{\nabla^{a-1} t^{a-1}}_{=(a-1)!} \\
&= a(a-1)! \\
&= a!
\end{aligned}
$$

where $(*)$ is due to the binomial theorem, and $(**)$ by the induction assumption. This establish the first property. To prove the second property, we utilize that $\nabla = (1 - B)$ where B is the backshift operator. In particular, by the binomial theorem:

$$
\nabla^q = (1-B)^q = \sum_{i=0}^{q} \binom{q}{i} (-1)^i B^i
$$

where B^0 should be interpreted as the identity operator. Hence, the time series given in the second property can be expressed as a linear combination of Y at different points in time:

$$\nabla^q Y_t = \sum_{i=0}^{q} \binom{q}{i} (-1)^i B^i Y_t = \sum_{i=0}^{q} \binom{q}{i} (-1)^i Y_{t-i}$$

Thus, the variance of the time series is given by:

$$\begin{aligned}
\mathrm{Var}(\nabla^q Y_t) &= \mathrm{Var}\Bigg(\sum_{i=0}^{q} \binom{q}{i} (-1)^i Y_{t-i} \Bigg) \\
&= \sum_{i=0}^{q} \sum_{j=0}^{q} \binom{q}{i} \binom{q}{j} (-1)^{i+j} \underbrace{\mathrm{Cov}(Y_{t-i}, Y_{t-j})}_{=\gamma_Y(i-j)} \\
&\leq \sum_{i=0}^{q} \sum_{j=0}^{q} \binom{q}{i} \binom{q}{j} \underbrace{|\gamma_Y(i-j)|}_{\leq \gamma_Y(0)} \\
&\leq \gamma_Y(0) \sum_{i=0}^{q} \sum_{j=0}^{q} \binom{q}{i} \binom{q}{j} \\
&< +\infty.
\end{aligned}$$

It follows quickly that the mean is constant and zero, since

$$\mathbb{E}(\nabla^q Y_t) = \sum_{i=0}^{q} \binom{q}{i} (-1)^i \underbrace{\mathbb{E}(Y_{t-i})}_{=0} = 0.$$

Its ACVF is given by:

$$\begin{aligned}
\gamma_{\nabla^q Y_s}(s,t) = \mathbb{E}\Bigg((\nabla^q Y_s)(\nabla^q Y_t) \Bigg) &= \sum_{i=0}^{q} \sum_{j=0}^{q} \binom{q}{i} \binom{q}{j} (-1)^{i+j} \mathbb{E}(Y_{s-i} Y_{t-j}) \\
&= \sum_{i=0}^{q} \sum_{j=0}^{q} \binom{q}{i} \binom{q}{j} (-1)^{i+j} \gamma_Y((s-t)-(i-j))
\end{aligned}$$

Hence, the ACVF is only a function of the lag $h = s - t$. This, together with the finite variance property and the constant mean makes all the conditions in the fulfilled, which establish that $\nabla^q Y_t$ is stationary.

The third property follows quickly from the first two, because:

$$\mathbb{E}(\nabla^q (X_t)) = \mathbb{E}(\nabla^q m_t) + \mathbb{E}(\nabla^q Y_t) = \nabla^q m_t = q! a_q$$

The variance structure is completely inherited by $\nabla^q Y_t$, since:

$$\begin{aligned}
\gamma_{\nabla^q X}(s,t) = \mathrm{Cov}(m_s + \nabla^q Y_s,\, m_t + \nabla^q Y_t) &= \underbrace{\mathrm{Cov}(m_s,\, m_t)}_{=0} + \underbrace{\mathrm{Cov}(\nabla^q Y_s,\, \nabla^q Y_t)}_{=\gamma(s,t)} \\
&+ \underbrace{\mathrm{Cov}(m_s,\, \nabla^q Y_t,)}_{=0} + \underbrace{\mathrm{Cov}(m_t,\, \nabla^q Y_s)}_{=0} \\
&= \gamma(s,t)
\end{aligned}$$

where the first equality is due to bilinearity of the covariance, and $\gamma(s,t)$ denotes the ACVF of $\nabla^q Y_s$. Since we have already proven that γ is only dependent on the lag, $\nabla^q X_t$ is also stationary.

A Code

```
#Install required packages
install.packages('R.matlab')
install.package('ggplot2')
install.packages('Matrix')

#Load required packages
library('R.matlab')
library('gridExtra')
library('Matrix')
library('ggplot2')

setwd('C:/Users/Jacob Lindb?ck/Documents/GitHub/tidsserier/Project 1')
###################################################################
###################### Functions ##################################
###################################################################
sampMean<-function(data){
  output<-sum(data)/length(data) #by formula in book
}

sampVar<-function(data){
  n<-length(data) #by well known formula
  output<-(1/(n-1))*sum((data-sampMean(data))^2)
}

sampAutoCov<-function(data,lag){
  n<-length(data)
  mu<-sampMean(data)
  stDv<-sqrt(sampVar(data))
  lag<-abs(lag)
  output<-0
  for(t in 1:(n-lag) ){ #by summation formula in book
    output<-output+(data[t+lag]-mu)*(data[t]-mu)
  }
  output<-output/n
}

sampAutoCorr<-function(data,lag){
  output<-sampAutoCov(data,lag)/sampAutoCov(data,0) #by formula gamm(t)/gamma(0)
}


ljungBox<-function(data,lagMax,alpha){
  n<-length(data)
  testStat<-0
  lagMax<-abs(lagMax) #unsure of this, but needed for summation.
  for(j in 1:lagMax){
    testStat<-testStat+sampAutoCorr(data,j)^2/(n-j) #build sum
  }
  testStat<-testStat*n*(n+2) #final teststat
  pval<-pchisq(testStat,df=lagMax,lower.tail = FALSE) #test stat is chi2
  res<-testStat>qchisq(1-alpha,df=lagMax) #bool of reject/don't reject
  output<-data.frame(testStat,pval,res) #order output
  print(output) #print nice output
}
acfPlotter<-function(data,lagMax,shouldPlot=TRUE,plotName=""){ #to plot lag
  lag<-seq(0,lagMax) #preallocate space
  subFunc<-function(lag){
    output<-sampAutoCorr(data,lag)
  }
  acf<-sapply(lag,subFunc) #get vector of acf
  if(shouldPlot){ #plot it using ggplot2
    toPlotDf<-data.frame(lag,acf)

     ↪  q<-ggplot(data=toPlotDf,aes(x=lag,y=acf))+geom_hline(aes(yintercept=0))+geom_segment(aes(x
    q
  }
}

my_acf_matrix <- function(gamma){
  max.lag <- length(gamma) - 1
  Gamma = matrix(rep(0,(max.lag+1)^2), max.lag + 1, max.lag+1)

  for(i in 1:(max.lag+1)){
    for(j in i:(max.lag+1)) {
      Gamma[i,j] = gamma[(j-i)+1]
      Gamma[j,i] = gamma[(j-i)+1]
    }
  }
  return(Gamma)
}


###################################################################
####################### Loads and store the examined data
####################### in a data frame.
###################################################################
data <- readMat('exchangerate.mat') #read in data

data <- data$data
```

```
intr_value = data
abs_returns = data[2:205,] - data[1:204] #form absrets

log_returns = log(data[2:205,]) - log(data[1:204]) #form logrets

abs_returns = abs_returns - mean(abs_returns) #mean correct
log_returns = log_returns - mean(log_returns) #mean correct
intr_value = intr_value - mean(intr_value) #mean correct.

exchange_data = data.frame(time = seq(2,205), absolute.return = abs_returns,
 ↪  log.returns = log_returns, intrinsic.value = intr_value[2:205])
#plot data nicley using ggplot2
p1 <- ggplot(data = exchange_data, aes(x = time, y = abs_returns))+ geom_line() +
 ↪  geom_smooth(method = 'loess', se = FALSE) + theme_bw() + xlab('Time') +
 ↪  ylab('Absolute Returns')
p1 <- p1 + geom_smooth(method = 'lm', se = FALSE, color = 'Red') + theme(text =
 ↪  element_text(size=20))
p1

p2 <- ggplot(data = exchange_data, aes(x = time, y = log_returns))+ geom_line() +
 ↪  geom_smooth(method = 'loess',se = FALSE) + theme_bw() + xlab('Time') +
 ↪  ylab('Log-Returns')
p2 <- p2 + geom_smooth(method = 'lm', se = FALSE, color = 'Red') + theme(text =
 ↪  element_text(size=20))
p2

p3 <- ggplot(data = exchange_data, aes(x = time, y = intrinsic.value))+
 ↪  geom_line() + geom_smooth(method = 'loess', se = FALSE)
p3 <- p3 + geom_smooth(method = 'lm', se = FALSE, color = 'red')+ theme_bw() +
 ↪  xlab('Time') + ylab('Trade-Weighted Index') + theme(text =
 ↪  element_text(size=20))
p3

acf(log_returns)
#########################
##### Problem 2 #########
#########################

#### Now do actual task.

alpha=0.05 #signif. level
lag_max=20 #maximal lag
acfPlotter(intr_value,lag_max) #plot acfs
acfPlotter(abs_returns,lag_max)
acfPlotter(log_returns,lag_max)
ljungBox(intr_value,lagMax,alpha)
ljungBox(abs_returns,lagMax,alpha) # run all ljung box
ljungBox(log_returns,lagMax,alpha)
```

```
#########################
##### Problem 3 #########
#########################

#Partions the series into a training data set, and a test set.
num_samples = nrow(exchange_data)
num_train_samples = 102
num_test_samples = num_samples - num_train_samples
train_data = exchange_data[1:num_train_samples,]
test_data = exchange_data[(num_train_samples+1):num_samples,]

#Extracts the last [num_predictors] train_data points and store them in a row
↪   matrix
num_predictors = 20
predictors =
↪   (train_data$log.returns)[num_train_samples:(num_train_samples-num_predictors+1)]
predictors = t(matrix(predictors))

#Computes the acf, and imputes the values corresponding to lag greater or equal to
↪   [num_train_samples] by zero.
train_acf = sapply(0:(num_train_samples-1),
↪   function(x){return(sampAutoCov(train_data$log.returns, lag = x))})
train_acf = c(train_acf, rep(0, 30))

Gamma = my_acf_matrix(train_acf[1:num_predictors])
coefficients <- matrix(rep(0,num_predictors*num_test_samples), num_predictors,
↪   num_test_samples) #Allocate memory for



LU = lu(Gamma)  #Since matrix of the linear system that is to be solved in each
↪   iteration is fix. It's appropriate
                #LU-decompose the matrix in order to save computations.
LU = expand(LU)
L = LU$L
U = LU$U
P = LU$P

for(h in 1:num_test_samples){
  ii = seq(h+1, h+num_predictors)
  b = train_acf[ii]
  b1 = solve(P,b)
  b2 = forwardsolve(L,b1)
  b3 = backsolve(U, b2)
  coefficients[,h] = as.matrix(b3)
}
```

```
predictions = predictors%*%coefficients

df1 = data.frame(time = rep(seq(num_train_samples+1,num_samples),2), log.returns =
 ↪  c(test_data$log.returns, predictions[1,]), type = c(rep('True',
 ↪  num_test_samples), rep('Predictions', num_test_samples)))

#Plots the predicted log-return against time, superimposed with the predicted time
 ↪  series.
p1 <- ggplot(data = df1, aes(x = time, y = log.returns, group = type)) +
 ↪  geom_line(aes(color = type))
p1 <- p1 + xlab('Time') + ylab('Log-returns') +
 ↪  theme(legend.title=element_blank())

#Distribution of the residuals of the naive estimates, and the predicted time
 ↪  series.
df2 = data.frame(Error1 = test_data$log.returns, Error2 =
 ↪  (test_data$log.returns)-predictions[1,])

p2 <- ggplot(data = df2) + geom_histogram(aes(Error1), bins = 15, fill = 'green',
 ↪  alpha = 0.5)
p2 <- p2 + geom_histogram(aes(Error2), bins = 15, fill = 'blue', alpha = 0.5) +
 ↪  xlab('Errors') +ylab('')
grid.arrange(p1, p2, nrow = 2)

MSE1 = mean((test_data$log.returns)^2) #MSE of the "naive estimate"
MSE2 = mean((test_data$log.returns-predictions[1,])^2) #MSE of the predictive
 ↪  model


#################################################################
###### Computes the predictions based on previous observations########
############################ (Task 3) ################################
#################################################################
upd_predictors2 <- predictors[1,]
coeff1 <- coefficients[which(coefficients[,1] != 0),1]
new_predictions2 <- rep(0,num_test_samples)

for(i in 1:num_test_samples){
  ith_prediction <- sum(coeff1*upd_predictors2)
  new_predictions2[i] = ith_prediction
  upd_predictors2 <- c(test_data$log.returns[i],
   ↪  upd_predictors2[1:num_predictors-1])
}

df5 = data.frame(time = rep(seq(num_train_samples+1,num_samples),2), log.returns =
 ↪  c(test_data$log.returns, new_predictions2), type = c(rep('True',
 ↪  num_test_samples), rep('Predictions', num_test_samples)))
```

```
#Plots the predicted log-return against time, superimposed with the predicted time
↪  series.
p5 <- ggplot(data = df5, aes(x = time, y = log.returns, group = type)) +
↪  geom_line(aes(color = type))
p5 <- p5 + xlab('Time') + ylab('Log-returns') +
↪  theme(legend.title=element_blank())

df6 = data.frame(Error1 = test_data$log.returns, Error2 =
↪  (test_data$log.returns)-new_predictions2)

p6 <- ggplot(data = df6) + geom_histogram(aes(Error1), bins = 15, fill = 'green',
↪  alpha = 0.5)
p6 <- p6 + geom_histogram(aes(Error2), bins = 15, fill = 'blue', alpha = 0.5) +
↪  xlab('Errors') +ylab('')
grid.arrange(p5, p6, nrow = 2)

MSE3 = mean((test_data$log.returns-new_predictions2)^2)

#########################
##### Problem 4 #########
#########################
qqnorm(log_returns, pch = 1, frame = FALSE,main="") #use qqnormline command to get
↪  qqplot
qqline(log_returns, col = "steelblue", lwd = 2)

X<-rnorm(204)
qqnorm(X, pch = 1, frame = FALSE,main="") #do same for pseudoranom normal data
qqline(X, col = "steelblue", lwd = 2)

#To do: abs(log.return, Box Ljung)
abslog_returns<-abs(log_returns)  #form abs returns
abslog_returns<-abslog_returns-sampMean(abslog_returns); #mean correct
ljungBox(abslog_returns,lag_max,alpha)
acfPlotter(abslog_returns,lag_max) #plot acf
```