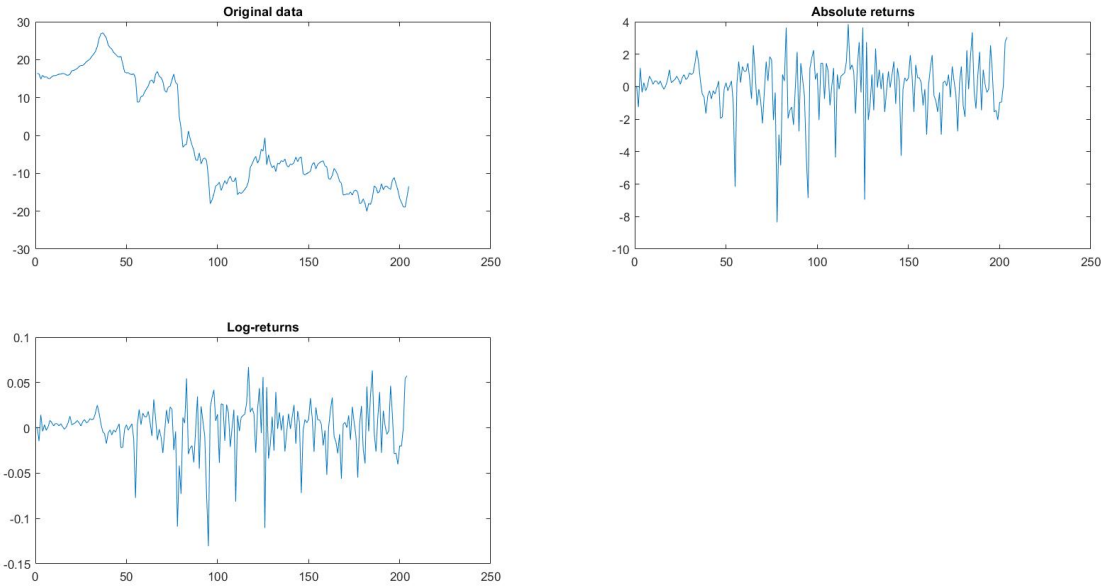
Financial Time Series project 1

Abraham Deniz and Christos Saltapidas

3 maj 2018

Task 1

Here is a plot of the three different time-series after mean-correcting them:

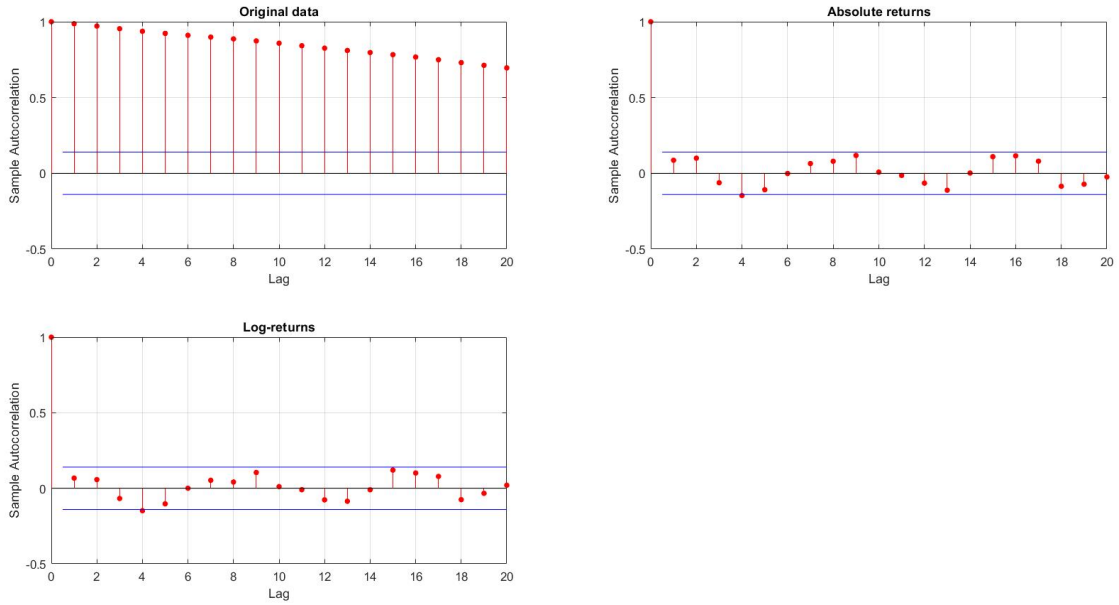


Figur 1: A plot of the Original data, absolute returns and log-returns after mean-correcting them.

Since the series are mean-corrected, we would expect the three series to have values near zero if they were stationary. The Absolute returns seems to have many values near zero, but some values seems to deviate a lot from zero as well (the minimum is approximately -8.35 and the maximum is approximately 3.85) so it is questionable whether it can be modeled as a stationary time series. The original data is clearly not a stationary time series since it has many values that deviates a lot from zero, which indicates that the expected value is not zero. The Log-returns have all values near zero (the minimum is approximately -0.13 and the maximum is approximately 0.07). Thus, if any of the series could be modeled as a stationary time series, then it would be the log-returns, and maybe the absolute returns. But we can't conclude that just by looking at these plots.

Task 2

Here is a plot of the sample ACF for the original data, absolute returns and log-returns:



Figur 2: A plot of the sample ACF for $h = 0, 1, \ldots, 20$ of the original data, absolute returns and log-returns, as produced by the inbuilt matlab function *autocorr*.

The inbuilt matlab function *autocorr* calculates, and plots, the sample ACF (as defined in definition 2.2.7 in the lecture notes) for all lags $h \in \{0, 1, \ldots, K\}$ where K is specified by the user. It also plots the bounds $\pm 1.96/\sqrt{n}$, where n is the length of the data, as can be seen in figure 2. We can clearly see that for the original data, for every lag $h \in \{0, 1, \ldots, 20\}$, the sample ACF exceeds the bounds $\pm 1.96/\sqrt{205}$. This indicates that the time series is not stationary. For the absolute returns and log-returns, some values exceed the bounds $\pm 1.96/\sqrt{204}$ while some values doesn't. This means that we cannot reject the null hypothesis of stationarity just by looking at the sample ACF plot.

The result from the Ljung-Box test is given in the following table:

Tabell 1: The Ljung-Box test results with $h = 20$ and significance level $\alpha = 0.05$.

	Rejected H_0	p-value	test-statistic	
Original data	Yes	0	3136.3	
Absolute returns	No	0.0636	30.4	
Log-returns	No	0.2317	24.25	

If T_0 denotes the observed value of the test-statistic then the p-value is calculated as

$$P(\lambda > T_0) = 1 - P(\lambda \leq T_0) = 1 - F_\lambda(T_0)$$

where $F_\lambda(t)$ is the cdf (cummulative distribution function) for the χ^2-distribution with h degrees of freedom and λ is the test statistic for the Ljung-Box test, as in the lecture notes. The inbuilt matlab function *chi2cdf* is used to calculate the cdf for the χ^2-distribution with $h = 20$ degrees of freedom. The null hypothesis H_0 in the Ljung-box test states that

$$\rho_Y(1) = \rho_Y(2) = \ldots = \rho_Y(h) = 0$$

The fact that the p-value for the Original data is zero is not so surprising, considering that the sample ACF exceeds the bounds $\pm 1.96/\sqrt{205}$ for all lags $h \in \{0, 1, \ldots, 20\}$ as seen in the sample ACF plot in figure 2. The p-value for the absolute returns was expected to be lower than the p-value for the

log-returns since the sample ACF at lags $h \in \{0, 1, \ldots, 20\}$ is lower for the log-returns compared to for the absolute returns, as seen in the sample ACF plot in figure 2. Not rejecting the null hypothesis for the absolute-returns and log-returns means that we could model them as white noise, and we recall that iid random variables is white noise but the converse isn't true in general. But in the case when the white noise is normally distributed then it is iid.

Task 3

After splitting the given data in a training set and a test set, our main goal is to obtain the coefficients $a_0, \ldots, a_{20}$ in order to calculate the Best Linear Predictor $b_n^l(z_{n-1}, \ldots, z_{n-20})$ for $n = 103, \ldots, 204$. We recall that our data is mean corrected, making $a_0 = 0$. The remaining coefficients $a_1, \ldots, a_{20}$ are obtained by solving the equation

$$A \cdot x = b$$

with respect to x, where

$$A = \begin{bmatrix} \hat{\gamma}(0) & \hat{\gamma}(1) & \ldots & \hat{\gamma}(19) \\ \hat{\gamma}(1) & \hat{\gamma}(0) & \ldots & \hat{\gamma}(18) \\ \vdots & \vdots & \ddots & \vdots \\ \hat{\gamma}(19) & \hat{\gamma}(18) & \ldots & \hat{\gamma}(0) \end{bmatrix}$$

and

$$x = \begin{bmatrix} a_1, \ldots, a_{20} \end{bmatrix}^T$$

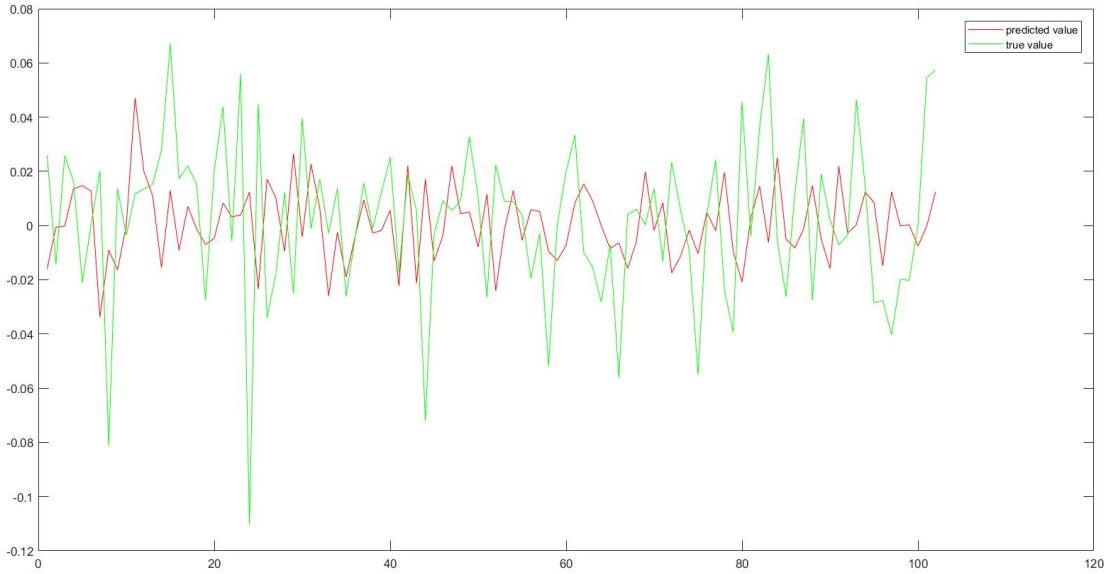
and

$$b = \begin{bmatrix} \hat{\gamma}(1), \hat{\gamma}(2), \ldots, \hat{\gamma}(20) \end{bmatrix}^T$$

and $\hat{\gamma}(h)$ is the sample ACVF (as defined in definition 2.2.7 in the lecture notes). To obtain the matrix A in matlab, we first create a function, which we call *acov*, that calculates the ACVF where the input variables are the lag h and the data. Then we calculate

$$\hat{b} = \begin{bmatrix} \hat{\gamma}(0), \hat{\gamma}(1), \ldots, \hat{\gamma}(19) \end{bmatrix}$$

using *acov* and then uses the inbuilt matlab function *toeplitz* with $\hat{b}$ as input to obtain the matrix A. Plotting the predicted values together with the true values in the same figure yields this plot:

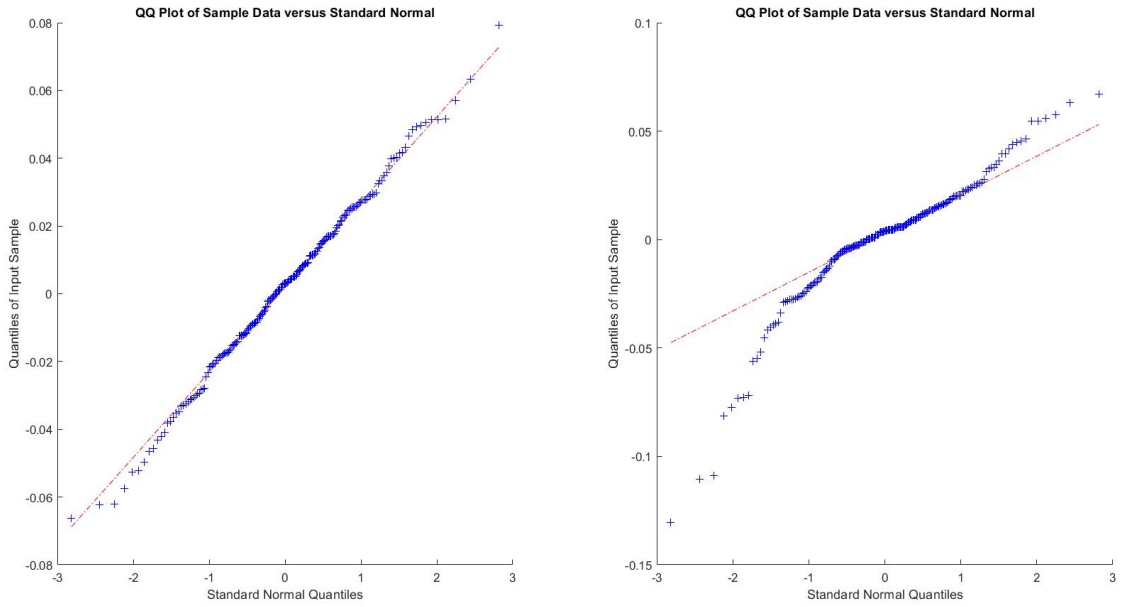


Figur 3: Predicted values and the true values. The red line is the predicted values and the green line is the true values.

The error from using the best linear prediction was calculated to $11 \cdot 10^{-4}$ and the error from the naive prediction of using the mean, which is zero, was calculated to $8.9 \cdot 10^{-4}$. This was expected, since we estimated the ACVF γ with the sample ACVF $\hat{\gamma}$, which means that the best linear predictor is not an optimal predictor. Also, the best linear predictor can be viewed as an AR(20) process and we know that there is no optimal way of finding the best AR(20) process in general. Also, since we didn't reject the null hypothesis of zero correlation for the log-returns in task 2 with the Ljung-Box test, predicted future values will just be a pure guess (since it could be modeled as white noise), making the prediction worse than the naive prediction of using the mean. If we want to make a better prediction, we should use all of our previous available data and not just the former 20. Also, the sample size of 102 in the training set may not be high enough to obtain a good estimate of the true ACVF γ which affects the coefficients a_i.

Task 4

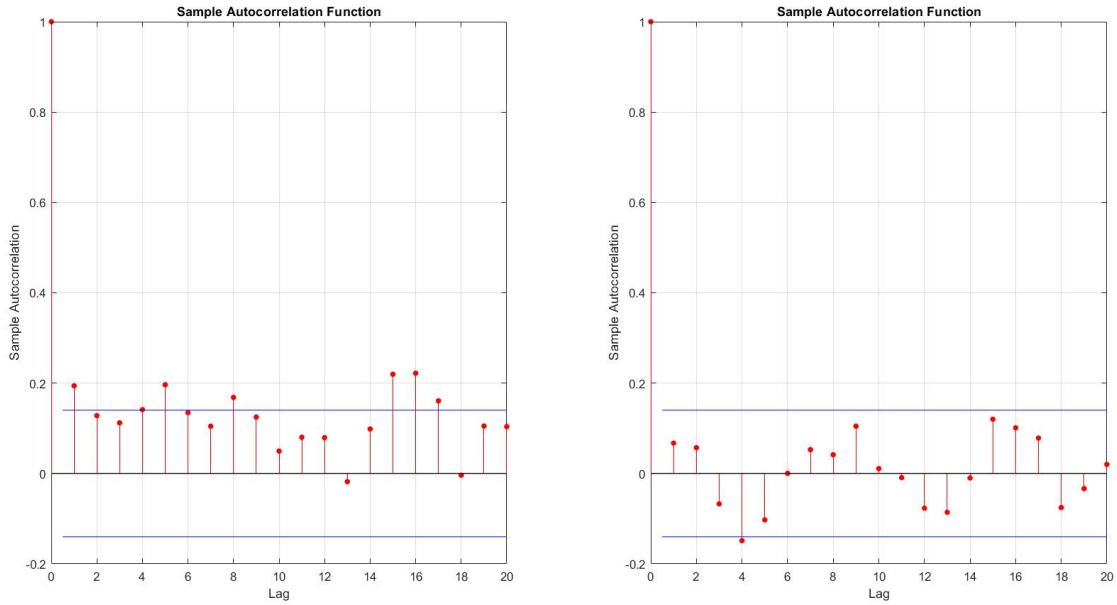
Considering the result from task 2, we wish to test whether the log-returns are normally distributed. We test for normality using a Q-Q (Quantile-Quantile) plot. Here is a qq-plot for the log-returns and for some simulated gaussian data to compare:



Figur 4: A plot of the simulated normal distributed data comparing with our data. The left plot is for the simulated data and the right plot is for our data.

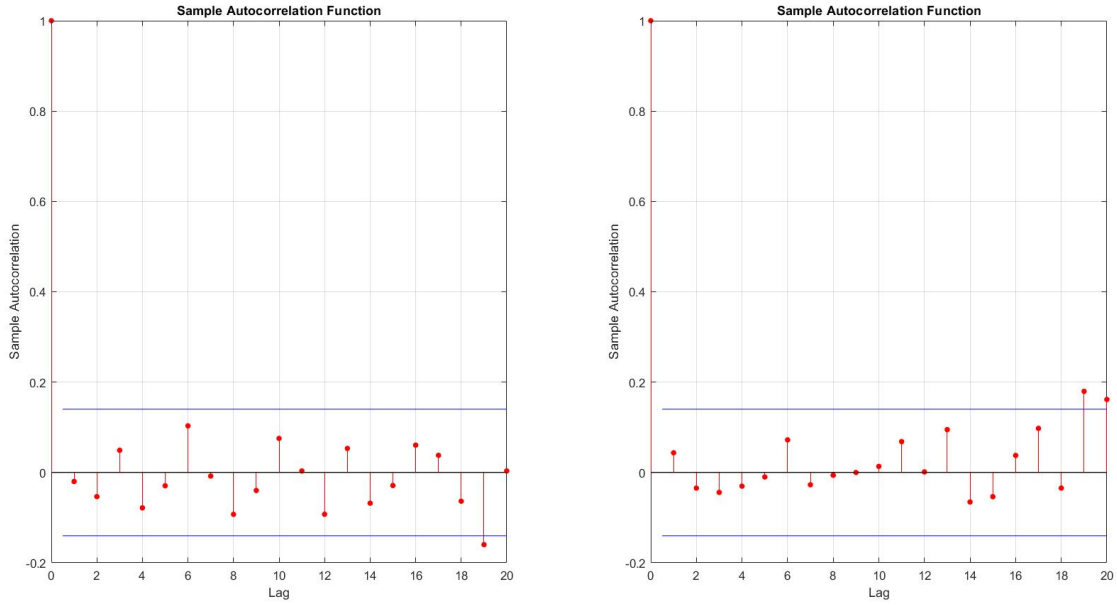
The qq-plots have been produced by the inbuilt matlab function *qqplot*. It displays the quantiles from the sample versus the quantiles from a normal distribution. If the distribution is normal then the plot will appear to be linear. However, we can clearly see that the right plot is non-linear (compared to the left plot) which indicates that the log-returns are not normally distributed.

We also take a look at the sample ACF for |Z|:



Figur 5: Sample ACF for |Z| and Z. The left plot is for |Z| and the right plot is for Z.

We can see that more values of the sample ACF for |Z| exceeds the bound $1.96/\sqrt{204}$ compared to the sample ACF of Z. Moreover, we can also see that most of the values of the sample ACF for |Z| lies closer to the boundary $1.96/\sqrt{204}$ than zero. For some normally-distributed data U, the sample ACF for U and |U| looks like this:



Figur 6: Sample ACF for |U| and U. The left plot is for |U| and the right plot is for U. U is some normally distributed data with mean zero and standard-deviation=sample standard-deviation of the log-returns.

These plots also indicates that the log-returns are not normally-distributed, due to |Z| not having the same shape as |U| if you compare with Z and U.

The result from the Ljung-Box test for the absolute value of the log-returns is given in this table:

Tabell 2: The Ljung-Box test results for |Z|

Rejected H_0	p-value	test-statistic
Yes	$4.204 \cdot 10^{-9}$	79.82

The p-value is very low because most of the values of the ACF lies close to the boundary $1.96/\sqrt{204}$ as seen in the sample ACF plot in figure 6.

Task 5

a)

We use induction to prove that

$$\nabla^q m_t(q) = q! \cdot a_q$$

For $q = 1$, we have

$$\nabla m_t(1) = m_t(1) - m_{t-1}(1) = a_0 + a_1 \cdot t - a_0 - a_1 \cdot (t-1) = a_1 \cdot t - a_1 \cdot (t-1) = a_1 = 1! \cdot a_1$$

For the induction step, we assume that

$$\nabla^q m_t(q) = q! \cdot a_q$$

and prove that

$$\nabla^{q+1} m_t(q+1) = (q+1)! \cdot a_{q+1}$$

Re-writing $m_t(q+1)$ as

$$m_t(q+1) = m_t(q) + a_{q+1} \cdot t^{q+1}$$

we obtain

$$\nabla^{q+1} m_t(q+1) = \nabla^{q+1}(m_t(q) + a_{q+1} \cdot t^{q+1}) = \nabla(\nabla^q(m_t(q) + a_{q+1} \cdot t^{q+1})) = \Big[\nabla \text{ is a linear operator}\Big] =$$

$$\nabla(\nabla^q m_t(q) + a_{q+1}\nabla^q(t^{q+1})) = \Big[\text{Induction Assumption}\Big] = \nabla(q! \cdot a_q + a_{q+1}\nabla^q(t^{q+1})) = \Big[\nabla c = 0 \text{ if } c \in \mathbb{R}\Big] =$$

$$= \nabla(a_{q+1} \cdot \nabla^q(t^{q+1})) = \Big[\nabla \text{ is a linear operator}\Big] = a_{q+1}\nabla^{q+1}(t^{q+1})$$

Thus, it remains to prove that

$$\nabla^{q+1}(t^{q+1}) = (q+1)!$$

We have that

$$\nabla^{q+1}(t^{q+1}) = \nabla^q(\nabla(t^{q+1})) = \nabla^q(t^{q+1} - (t-1)^{q+1}) = \Big[\text{Binomial Theorem and } \nabla \text{ is a linear operator}\Big] =$$

$$= \nabla^q(t^{q+1}) - \nabla^q\left(\sum_{j=0}^{q+1} \binom{q+1}{j} \cdot (-1)^j \cdot t^{q+1-j}\right) = \nabla^q(t^{q+1}) - \nabla^q\left(t^{q+1} + \sum_{j=1}^{q+1} \binom{q+1}{j} \cdot (-1)^j \cdot t^{q+1-j}\right) =$$

$$= \Big[\nabla \text{ is a linear operator}\Big] = -\nabla^q\left(\sum_{j=1}^{q+1} \binom{q+1}{j} \cdot (-1)^j \cdot t^{q+1-j}\right) = \Big[i = j - 1\Big] =$$

$$= -\nabla^q \left(\sum_{i=0}^{q} \binom{q+1}{i+1} \cdot (-1)^{i+1} \cdot t^{q-i} \right) = \left[\text{Induction Assumption} \right] = -(q!) \cdot (-1)^1 \cdot \binom{q+1}{1} =$$

$$= q! \cdot (q+1) = (q+1)!$$

Thus, we conclude that

$$\nabla^q m_t(q) = q! \cdot a_q$$

by the induction principle.

b)

First, we note that

$$\nabla^q(Y_t) = \nabla^{q-1}((1-B)Y_t) = \nabla^{q-2}(\nabla((1-B)Y_t)) = \nabla^{q-2}((1-B)^2 Y_t) = \ldots = (1-B)^q Y_t =$$

$$= \left[\text{Binomial Theorem} \right] = \sum_{j=0}^{q} \binom{q}{j} (-B)^j Y_t = \sum_{j=0}^{q} \binom{q}{j} \cdot (-1)^j B^j Y_t = \sum_{j=0}^{q} \binom{q}{j} \cdot (-1)^j \cdot Y_{t-j} =$$

$$= \sum_{j=0}^{q} c_j \cdot Y_{t-j}$$

where

$$c_j = \binom{q}{j} \cdot (-1)^j$$

Then

$$\mathbb{E}(\nabla^q(Y_t)) = \mathbb{E}\left(\sum_{j=0}^{q} c_j \cdot Y_{t-j} \right) = \left[\text{Expectation is a linear operator} \right] =$$

$$= \sum_{j=0}^{q} c_j \cdot \mathbb{E}(Y_{t-j}) = \sum_{j=0}^{q} c_j \cdot 0 = 0$$

where the last equality holds since

$$\sum_{j=0}^{q} c_j = \sum_{j=0}^{q} \binom{q}{j} \cdot (-1)^j \cdot 1^{q-j} = \left[\text{Binomial Theorem} \right] = (1-1)^q = 0 < +\infty.$$

We also have that

$$\text{Var}(\nabla^q(Y_t)) = \text{Var}\left(\sum_{j=0}^{q} c_j \cdot Y_{t-j} \right) = \left[\text{zero mean} \right] = \mathbb{E}\left[\left(\sum_{j=0}^{q} c_j \cdot Y_{t-j} \right) \cdot \left(\sum_{j=0}^{q} c_j \cdot Y_{t-j} \right) \right] =$$

$$= \mathbb{E}\left[\sum_{i=0}^{q} \sum_{j=0}^{q} c_i \cdot c_j \cdot Y_{t-j} \cdot Y_{t-i} \right] = \left[\text{Expectation is a linear operator} \right] = \sum_{i=0}^{q} \sum_{j=0}^{q} c_i \cdot c_j \cdot \mathbb{E}(Y_{t-j} \cdot Y_{t-i}) =$$

$$= \left[\text{zero mean} \right] = \sum_{i=0}^{q} \sum_{j=0}^{q} c_i \cdot c_j \cdot Cov(Y_{t-j}, Y_{t-i})$$

Since Y is stationary, it must have finite covariance (otherwise it wouldn't make any sense). Since a finite sum (where the summands are finite) cannot be infinite, we obtain that

$$\text{Var}(\nabla^q(Y_t)) = \sum_{i=0}^{q} \sum_{j=0}^{q} c_i \cdot c_j \cdot Cov(Y_{t-j}, Y_{t-i}) < +\infty$$

Finally, we have

$$\gamma_{\nabla^q(Y)}(r+h, s+h) = \Big[\text{zero mean}\Big] = \mathbb{E}\Big[\nabla^q(Y_{r+h})\cdot\nabla^q(Y_{s+h})\Big] = \mathbb{E}\Big[\Big(\sum_{j=0}^{q} c_j\cdot Y_{r+h-j}\Big)\cdot\Big(\sum_{j=0}^{q} c_j\cdot Y_{s+h-j}\Big)\Big] =$$

$$= \mathbb{E}\Big[\sum_{i=0}^{q}\sum_{j=0}^{q} c_i\cdot c_j\cdot Y_{s+h-j}\cdot Y_{r+h-i}\Big] = \Big[\text{Expectation is a linear operator}\Big] = \sum_{i=0}^{q}\sum_{j=0}^{q} c_i\cdot c_j\cdot\mathbb{E}(Y_{s+h-j}\cdot Y_{r+h-i}) =$$

$$= \Big[\text{zero mean}\Big] = \sum_{i=0}^{q}\sum_{j=0}^{q} c_i\cdot c_j\cdot\gamma_Y(s+h-j, r+h-i) = \Big[Y \text{ is stationary}\Big] = \sum_{i=0}^{q}\sum_{j=0}^{q} c_i\cdot c_j\cdot\gamma_Y(s-j, r-i)$$

Using the same calculations backwards, we conclude that

$$\gamma_{\nabla^q(Y)}(r+h, s+h) = \gamma_{\nabla^q(Y)}(r, s)$$

Thus, we conclude that $\nabla^q Y$ is a stationary process with mean zero.

c)

We have

$$\mathbb{E}(\nabla^q(X_t)) = \mathbb{E}\Big[\nabla^q(Y_t) + \nabla^q(m_t(q))\Big] = \Big[\text{part a) and expectation is a linear operator}\Big] =$$

$$= q!\cdot a_q + \mathbb{E}(\nabla^q(Y_t)) = \Big[\text{zero mean by part b)}\Big] = q!\cdot a_q$$

We also have that

$$Var(\nabla^q(X_t)) = Var(q!\cdot a_q + \nabla^q(Y_t)) = \Big[\text{Variance of a constant is zero}\Big] = Var(\nabla^q(Y_t)) < +\infty$$

where the last part follows from part b).

We also have that:

$$Cov(\nabla^q X_t, \nabla^q X_{t+h}) = Cov(\nabla^q Y_t + \nabla^q m_t(q), \nabla^q Y_{t+h} + \nabla^q m_{t+h}(q)) = \Big[\text{Covariance property}\Big] =$$

$$= Cov(\nabla^q Y_t, \nabla^q Y_{t+h}) + Cov(\nabla^q Y_t, \nabla^q m_{t+h}(q) + Cov((\nabla^q m_t(q), (\nabla^q Y_{t+h})$$

$$+Cov(\nabla^q m_t(q), \nabla^q m_{t+h}(q)) = \Big[\text{part a)}\Big] =$$

$$= Cov(\nabla^q Y_t, \nabla^q Y_{t+h}) + Cov(\nabla^q Y_t, q!a_q) + Cov(q!a_q, \nabla^q Y_{t+h}) + Cov(q!a_q, q!a_q)$$

$$= Cov(\nabla^q Y_t, \nabla^q Y_{t+h})$$

where the last equality follows from the fact that $Cov(q!\cdot a_q, q!\cdot a_q) = 0$ since $q!a_q$ is a constant, and second and third terms of the sum are also zero, since the covariances include that constant. The last term is independent of t due to the stationarity of $\nabla^q Y$ and thus

$$Cov(\nabla^q X_t, \nabla^q X_{t+h})$$

is independent of t and thus, $\nabla^q X$ is a stationary process with mean $q!\cdot a_q$.

Appendix MATLAB code

Function for the Ljung-Box test

```
function [H, T, pvalue]=LBT(h, data, alpha)
n=length(data);
A=autocorr(data, h); % The autocorrelation of the data at lag 0,1,...,h.
A=A(2:end); % Removing the first element since we don't need it.
T=0;
for i=1:h
    T=T+(A(i)^2)/(n-i);
end
T=n*(n+2)*T; % The test statistic.
pvalue=1-chi2cdf(T,h); % The p-value.
H=(alpha>=pvalue); % Reject if the p-value is less than the significance
% level. If H=1 then reject the null hypothesis. Otherwise, H=0 and we
% don't reject the null hypothesis.
end
```

Function to calculate the Sample ACVF

```
function y=acov(h, data)
n=length(data);
y=0;
for i=1:n-h
    y=y+(data(i+h)-mean(data))*(data(i)-mean(data));
end
y=y/n;
end
```

Function to calculate the coefficients in proposition 2.3.5

```
function a=prop235(data)
lags=0:19;
A=zeros(length(lags),1);
for i=1:length(lags)
   A(i)=acov(lags(i),data); % computing gammahat(0),...,gammahat(19).
end
gammahat=toeplitz(A); % the gammahat-matrix.
vec=zeros(length(lags),1); % the vector for the right-hand-side.
for i=1:length(lags)
    vec(i)=acov(i, data); % computing gammahat(1),...,gammahat(20).
end
a=inv(gammahat)*vec; % solution of the linear equation.
end
```

Code for task 1

```
data = importdata('exchangerate.mat'); % importing the data.
Y=diff(data); % Absolute returns.
Z=diff(log(data)); % log-returns.

data1=data-mean(data); % mean-corrected data.
Y1=Y-mean(Y); % mean-corrected absolute returns.
```

```
Z1=Z-mean(Z); % mean-corrected log-returns.

subplot(1,3,1) % to get the plots in the same window.
plot(data1) % plotting the mean-corrected data.
title('Original data')
subplot(1,3,2) % to get the plots in the same window.
plot(Y1) % plotting the mean-corrected absolute returns.
title('Absolute returns')
subplot(1,3,3) % to get the plots in the same window.
plot(Z1) % plotting the mean-corrected log-returns.
title('Log-returns')
```

Code for task 2

```
subplot(2,2,1) % to get the plots in the same window.
autocorr(data1, 20) % Plotting the sample ACF for the mean-corrected
    data.
title('Original data')
subplot(2,2,2) % to get the plots in the same window.
autocorr(Y1, 20) % Plotting the sample ACF for the mean-corrected
    absolute returns.
title('Absolute returns')
subplot(2,2,3) % to get the plots in the same window.
autocorr(Z1, 20) % Plotting the sample ACF for the mean-corrected log-
    returns.
title('Log-returns')

h=20;
[H1, T1, p1]=LBT(h, data1, 0.05); % The Ljung-Box test for the mean-
    corrected data.
[H2, T2, p2]=LBT(h, Y1, 0.05); % The Ljung-Box test for the mean-
    corrected absolute returns.
[H3, T3, p3]=LBT(h, Z1, 0.05); % The Ljung-Box test for the mean-
    corrected log-returns.
```

Code for task 3

```
N=102;
ind1=1:N; % index for the training data.
ind2=N+1:2*N; % index for the testing data.
Ztrain=Z1(ind1); % the training data.
Ztest=Z1(ind2); % the testing data.
blp=zeros(102,1); % vector for the best linear predictor.
a=prop235(Ztrain); % the coefficients from proposition 2.3.5.
n=103:204;
for i=1:length(blp)
    blp(i)=sum(a.*Z1(n(i)-1:-1:n(i)-20)); % the best linear predictor.
end

plot(blp,'red') % plotting the best linear predictor.
hold on % to get the plots on the same figure.
plot(Ztest,'green') % plotting the test data.
legend('predicted value', 'true value')
```

```
17
18 mse1=(1/102)*sum((blp-Ztest).^2); % Error from the prediction using the
       best-linear predictor.
19 mse2=(1/102)*sum(Ztest.^2); % Error from the prediction using the mean.
```

Code for task 4

```
1  %% Task 2 on the absolute value of Z.
2  absZ=abs(Z1); % The absolute value of the mean-corrected log-returns.
3  [H4, T4, p4]=LBT(h, absZ, 0.05); % Ljung-Box test for absZ.
4  subplot(1,2,1) % to get the plots in the same window
5  autocorr(absZ) % plotting the sample ACF for the absolute value of the
       mean-corrected log-returns.
6  subplot(1,2,2) % to get the plots in the same window.
7  autocorr(Z1) % plotting the sample ACF for the mean-corrected log-
       returns.
8  %% testing for normality of mean-corrected log-returns using qqplot.
9  U=std(Z1).*randn(length(Z1),1); % Some Normally distributed data.
10 subplot(1,2,1) % to get the plots in the same window.
11 qqplot(U) % Q-Q plot of the normally distributed data
12 subplot(1,2,2) % to get the plots in the same window.
13 qqplot(Z1) % Q-Q plot of the mean-corrected log-returns
14 %% Sample ACF plot for the normally-distributed data.
15 subplot(1,2,1)
16 autocorr(abs(U))
17 subplot(1,2,2)
18 autocorr(U)
```