Quantitative Risk Management

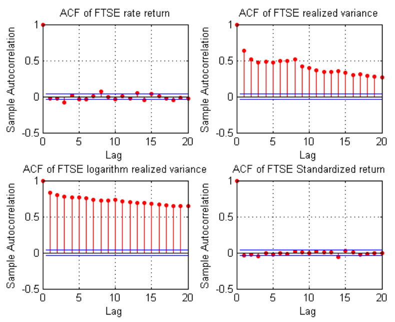
Coursework One

Chan Pruksapha (00999432)
Jiefeng Tan (00654522)
Jialiang Luo (00982125)
Jieshun Luo (00650690)

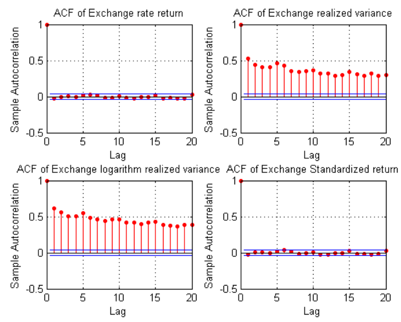
Contents

1 Problem 1 - Descriptive statistics

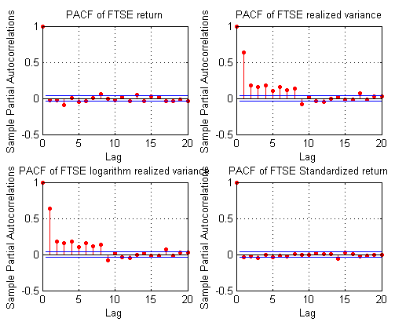
(i) Plot the ACF and PACF for daily returns, daily realized variance, logarithmic realized variance and daily returns standardized by realized volatility. Discuss the time-series properties of the these time series.



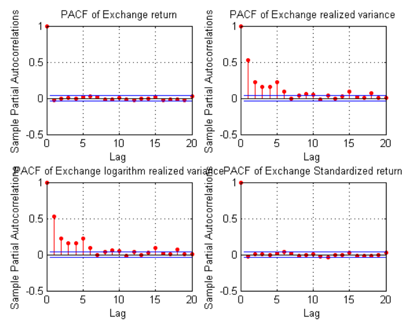
(a) FTSE ACF



(b) Exchange rate ACF



(c) FTSE pACF



(d) Exchange rate PACF

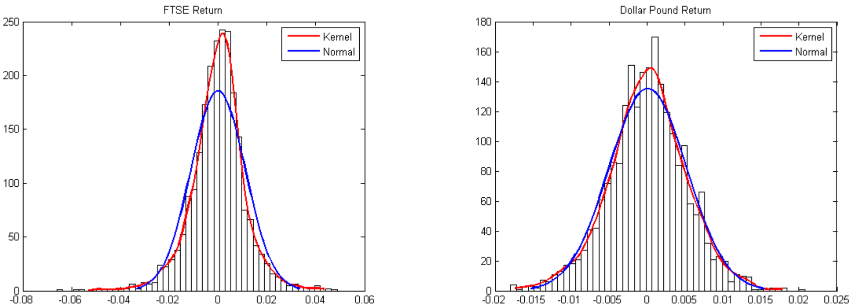
Figure 1: ACF and PACF plots

From the ACF plots of the 4 variables based on the FTSE 100 returns and realized variance in Figure 1, it is clear that the return itself has no autocorrelation. The absence of autocorrelation still applies when the we adjust the returns to the standardized returns. As to the ACF of the realized variance and logarithm realized variance, one can see strong correlations among the data with high degrees of persistence. Therefore, it is possible that the long memory might exist in the data. It is worth pointing out that these results agree with the literature. Please see Cont(2001) [3]

In Figure 1, we can observe similar patterns happening in exhange rate returns as in the FTSE 100 case. The strong correlation among realized variance and logarithm realized variance still exists, though at a lower level.

According to the partial autocorrelation plots of both financial assets, Figure 1 and Figure **??**, we can see that there are still, although smaller, correlations between the realized variance and its lag values. The same situation also applies to the logarithm of realized variance. Combining with the ACF plots, one might suggest to use an ARMA or ARIMA model to model the realized variance and logarithm of realized variance. Further, the PACF plots of the returns and standardized returns are similar to the corresponding ACF plots.

(ii) Plot the empirical densities, calculate descriptive statistics and test for serial correlation and normality of returns, squared returns, RV, $logRV$, $r/RV^{1/2}$. Discuss the distributional properties of these variables.



(a) FTSE Return (b) Exchange rate Return

Figure 2: Histogram for FTSE and Exchange rate returns

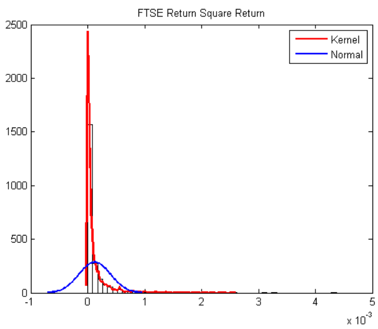
According to the histograms of financial returns of FTSE 100 and USD/GBP spot exchange rate in Figure 2, it is easy to observe from the kernel density that returns should follow some distributions with heavier tails than normal distribution such as the Student t distribution. However, it is worth pointing out that the distribution of USD/GBP spot exchange rate is closer to a normal distribution than the FTSE 100 does. Further, the plots also agree with the sylized fact that finanical returns are asymmetric (Please see Cont(2001) [3]).

Turing to the square returns and realized variance, Figure 3, it is obviously wrong to suggest that these variables are normally distributed. Instead, to find a better fit for the huge spikes in the histograms, the Gamma distribution might be a better alternative , as can be seen from the fitted curves.

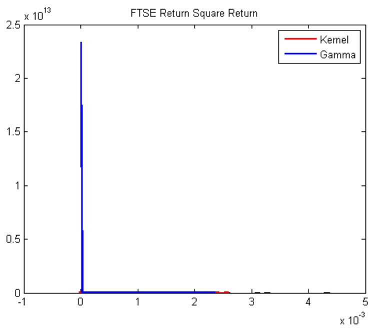
For the logarithm of realized variance and standardized returns, Figure 4, the kernel densities of both financial returns suggest that these variables are still not following a normal distribution. However, the density curves for the standardized returns are getting closer to the normal distribution compared to returns themselves.

The extra histograms in Figure 5 compare the kernel densities of different returns with a student t distribution. The well fitness of the two curves further confirm that a heavier distribution is needed for the financial returns, although we still need to deal with the asymmetry problem.

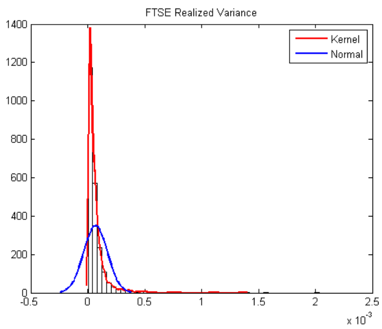
We will confirm our observations from the histograms with the descriptive statis-



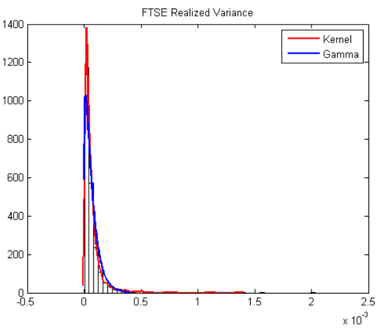
(a)FTSE Return Square Return



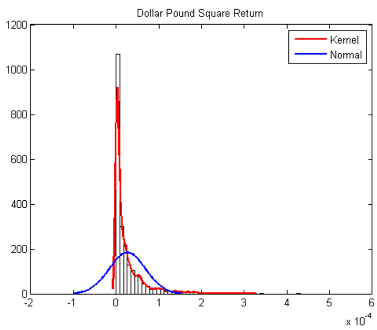
(b) FTSE Return Square Return Gamma



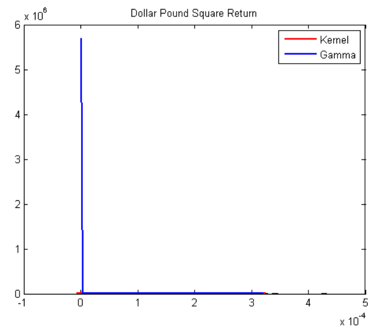
(c) FTSE Realized Variance



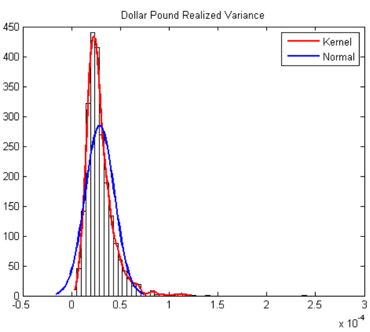
(d) FTSE Realized Variance Gamma



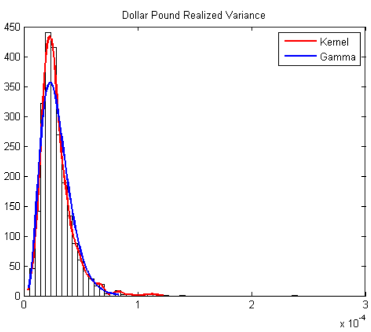
(e) Exchange Rate square return



(f) Exchange Rate square return Gamma

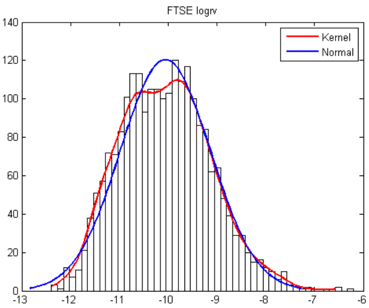


(g) Exchange Rate Realized Variance

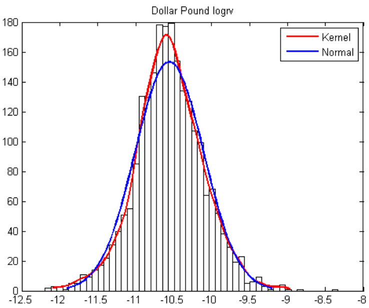


(h) Exchange Rate Realized Variance Gamma

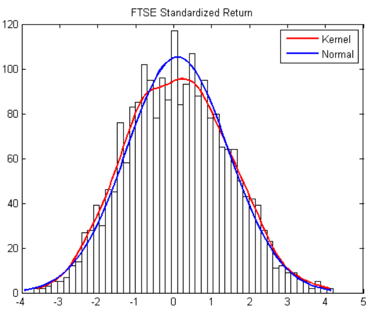
Figure 3: Histograms for square returns and realized variance



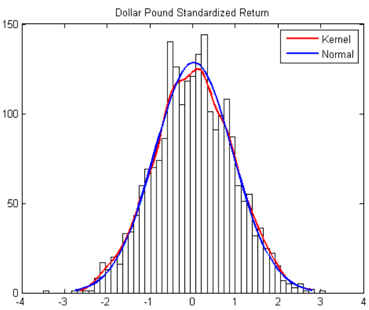
(a) FTSE log RV



(b) Exchange rate log RV

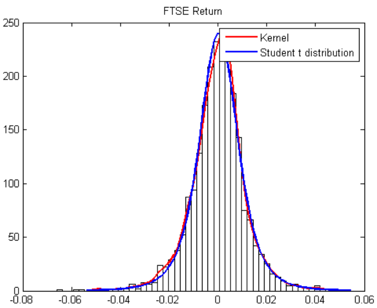


(c) FTSE Standardized returns

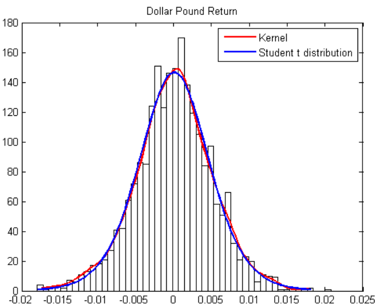


(d) Exchange rate Standardized returns

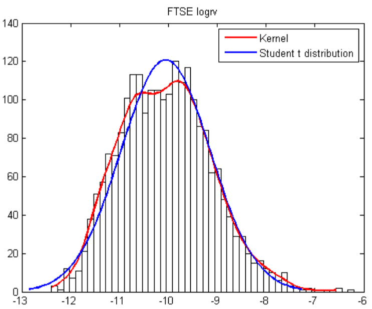
Figure 4: Histogram for log RV and Standardized returns



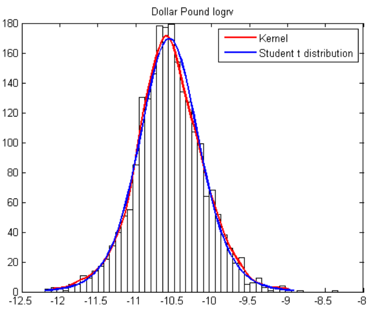
(a) FTSE log RV



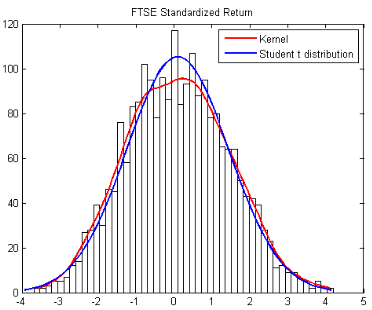
(b) Exchange rate log RV



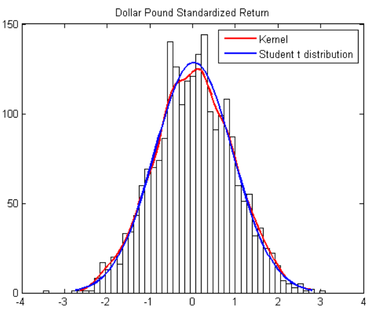
(c) FTSE Standardized returns



(d) Exchange rate Standardized returns



(e) FTSE Standardized returns



(f) Exchange rate Standardized returns

Figure 5: Histogram for log RV and Standardized returns with Student t distribution

tics and the results of serial correlation tests and normality tests. The following two tables contain the descriptive statistics of 5 different variables.

Foreign Exchange	Returns	Squared Returns	RV	logRV	Standardized Returns
Mean	0.00002	0.00006	0.0001	-10.057	0.09758
Variance	0.0001	1.08×10^{-8}	7.83×10^{-8}	0.8527	1.8059
Skewness	-0.3171	8.4236	6.1426	0.3252	0.0889
Kurtosis	5.9632	118.0953	59.5202	3.0348	2.6789

Table 1: Descriptive Statistics for Exchange rate return

Foreign Exchange	Returns	Squared Returns	RV	logRV	Standardized Returns
Mean	0.000088	0.000028	0.000026	-10.5558	0.0262
Variance	0.000026	1.78×10^{-9}	2.19×10^{-10}	0.2021	0.8469
Skewness	-0.0648	3.0651	3.3495	0.0635	-0.0048
Kurtosis	3.6002	27.5275	18.6791	3.8752	2.8828

Table 2: Descriptive Statistics for FTSE return

From Table 1 and Table 2, just as what we expected, the relatively high kurtosis of the finanical returns themselves suggests that the returns should follow a heavier-tailed distribution. Further, the logarithm of RV and standardized returns do give a lower kurtosis than the returns themselves do and they are very close to 3, which is the kurtosis of a normal distribution. An additional evidence supporting that the distribution of standardized returns and logarithm of realized variance are closed to normal is that the skewness of these two variables are getting closer to 0, which means that they are getting more symmetric.

Finally, we look at the test results of serial correlation test and normality test.

To test the serial correlation of the financial returns, we propose the Ljung-box test whose null hypothesis is that the data are independently distributed.

The statistic

$$Q = T\,(T+2)\sum_{i=1}^{h} \frac{\hat{\rho}_i^2}{T-i},$$

where T is the sample size, $\hat{\rho}_i$ is the sample autocorrelation at lag i, and h is the number of lags being tested.

From Table 3 and Table 4, we can see that most of the Ljung-Box tests are rejected, suggesting that the data might exhibit some extent of correlation. Not surprisingly, the test results for both standardized returns and the return data for USD/GBP spot exchange rate data suggest not to reject the null hypothesis, giving evidence to the fact that standardized returns and exchange rate returns might follow a normal distribution.

To test the normality of the financial returns, we propose the Jarque-Bera test whose null hypothesis is that the data are independently distributed.

Ljung–Box test FTSE	Results
Returns	Reject
Squared Returns	Reject
RV	Reject
logRV	Reject
Standardized Returns	Not Reject

Table 3: Ljung-Box test for FTSE return

Ljung–Box test Exchange	Results
Returns	Not Reject
Squared Returns	Reject
RV	Reject
logRV	Reject
Standardized Returns	Not Reject

Table 4: Ljung-Box test for exchange rate return

The statistic

$$JB = \frac{T}{6}\left(b^2 + \frac{1}{4}(k-3)^2\right),$$

where T is the number of observations (or degrees of freedom in general); b is the sample skewness, and k is the sample kurtosis:

$$b = \frac{\hat{\mu}_3}{\hat{\sigma}^3} = \frac{\frac{1}{T}\sum_{t=1}^{T}(r_t-\bar{r})^3}{\left(\frac{1}{T}\sum_{t=1}^{T}(r_t-\bar{r})^2\right)^{3/2}},$$

$$k = \frac{\hat{\mu}_4}{\hat{\sigma}^4} = \frac{\frac{1}{T}\sum_{t=1}^{T}(r_t-\bar{r})^4}{\left(\frac{1}{T}\sum_{t=1}^{T}(r_t-\bar{r})^2\right)^{2}},$$

where $\hat{\mu}_3$ and $\hat{\mu}_4$ are the estimates of third and fourth central moments, respectively, $\bar{r}$ is the sample mean, and $\hat{\sigma}^2$ is the estimate of the second central moment, the variance.

Jarque-Bera test FTSE	Results
Returns	Reject
Squared Returns	Reject
RV	Reject
logRV	Reject
Standardized Returns	Reject

Table 5: Jarque-Bera test for FTSE return

From Table 5 and Table 6, we can see that most of the normality assumption made to the 5 variables are rejected, except the one for standardized USD/GBP spot exchange rate returns. This suggests that the USD/GBP spot exchange rate return might follow a normal distribution. In fact, this result has been confirmed by Anderson, Bollerslev, Diebold and Labys(1999) [1](who suggested that exchange rate return standardized by realized volatility are (nearly) Gaussian.

Jarque-Bera test Exchange	Results
Returns	Reject
Squared Returns	Reject
RV	Reject
logRV	Reject
Standardized Returns	Not Reject

Table 6: Jarque-Bera test for exchange rate return

(iii) In this part, we will repeat the descriptive statistics and normality tests for the weekly, monthly and quarterly returns.

Before the results are presented, we will first explain the methodology we used to construct the weekly, monthly and quarterly data. The origin file we have is the daily FTSE and USD/GBP spot exchange rate returns. The difficulty here is that as there exists hoildays which make the number of returns each week are not exactly 5 and therefore some weeks might contain 3 or 4 numbers only. Therefore, we use our own strategy to tackle this problem.

For the weekly return data, we fisrt generate all the days including holidays and weekends between to and store them in a new vector. After this step, we construct another new return vector which is of the same size as the new date vector. We fill in all the returns in the corresponding dates from the daily returns and leave the returns of those hoilday and weekend cells as zeros. Then, we have a new return vector which consists of the original returns and zeros. Hence, we can now sum up the returns in every 7 days and store them into a new data. Notice that the last day in our original data set, , is a monday. We have discarded the return on this day when dealing with the weekly return data as an one-day return is not enough to construct a one-week return. Therefore, there are 468 weeks in our weekly return data instead of 469.

For the monthly data and weekly data, we have used a much easier way. Taking the monthly return as an example, we first construct a vector which contains all the months of each return. We then sum up returns which have the same month number and store them into a new vector. We apply the same trick to quarterly return data.

Having explained the methodology, we now look at the results.

FTSE 100	Daily	Weekly	Monthly	Quarterly
Mean	-0.00004	0.0001	0.0004	0.0013
Variance	0.00012	0.0005	0.0015	0.0051
Skewness	-0.3991	-0.3293	-0.8725	-1.0079
Kurtosis	6.1104	5.2178	4.0027	4.5802

Table 7: Descriptive Statistics for FTSE daily, weekly, monthly and quarterly return

According to Table 7 and Table 8, the absolute mean and variance tend to increase as the time scale increases.This is expected as we expect to earn more

Foreign Exchange	Daily	Weekly	Monthly	Quarterly
Mean	0.00007	0.0004	0.0018	0.0055
Variance	0.00003	0.0001	0.0004	0.0011
Skewness	-0.0848	-0.1104	0.0741	0.2912
Kurtosis	3.6029	3.1473	2.5652	2.2378

Table 8: Descriptive Statistics for Exchange rate daily, weekly, monthly and quarterly return

and the returns will become more volatile as the investment period becomes longer. However, there exhibits opposite trends in skewness between FTSE 100 and the USD/GBP spot exchange rate. While the skewness in FTSE 100 becomes more negative as the time scale increases, that of USD/GBP spot exchange rate increases and even beccomes positive when considering monthly and quarterly returns. The skewness indicates that there is less asymmetry embedded in the returns of foreign exchange rate compared to that of FTSE 100. Further, one can easily see that the kurtosis is decreasing in both returns as the time scale increases from daily to quarterly, except that there is an outlier of 4.5802 in the FTSE 100 returns. However, one explanation to this outlier is that there are only 36 returns that we can used to calculate the quarterly kurtosis, which might lead to an inaccurate result. Therefore, the results suggest that, as the time scale increases, the distribution of the financial returns tends to follow a normal distribution, giving evidence to the stylized fact of Aggregational Gaussianity (Please see Cont(2001) [3]).

To test the normality of returns in different time scales, we still use the Jarque–Bera test with the null hypothesis that returns follow a normal distribution when the time scale changes from daily to others.

Time Scale	Test Results for FTSE 100
Weekly	Reject
Monthly	Reject
Quarterly	Reject

Table 9: Jarque-Bera test for FTSE weekly, monthly and quarterly return

Time Scale	Test Results for Exchange rate
Weekly	Not Reject
Monthly	Not Reject
Quarterly	Not Reject

Table 10: Jarque-Bera test for Exchange rate weekly, monthly and quarterly return

From Table 9 and Table 10, it is suggested that the FTSE 100 returns do not follow a normal distribution even when the time scale increases to quarterly. In contrast, the Jarque-Bera tests for the USD/GBP spot exchange rate are all not rejected, suggesting that the USD/GBP spot exchange rate might follow a normal distribution.

2 Problem 2 - GARCH for daily returns

Problem 2.1 , By using **ugarchfit** function in **R** We can fit the garch(1,1) with constant mean model to FTSE 's return data. We obtain the mean of around 3%. The parameters obtained is according to equations

$$r = \mu$$
$$\sigma_t^2 = \omega + \alpha_{t-1}\epsilon^2 + \beta_{t-1}\sigma_{t-1}^2.$$

From the program output in listing 1, it can be seen that estimation error is small compared to the estimates and hence all of the parameters significantly differ from zero.

Also present in the listing is the Ljung-box test obtained from **ugarchfit** object. It can be seen that the hypothesis of no serial correlation in the standardized residual $epsilon_t/sigma_t$ survives the test for 3 lags is not rejected, namely one two and five, because the *p-value* obtained all is much higher than 0.05. Moreover, the acceptance of null hypothesis also hold for square of standardized residuals and this contrasts with the Ljung-box test for square return in problem 1, where the no-serial correlation assumption has been rejected. The no-serial correlation is confirmed again by the acf plot in figure 6

```

Conditional Variance Dynamics
-----------------------------------
GARCH Model  : sGARCH(1,1)
Mean Model   : ARFIMA(0,0,0)
Distribution  : norm

Optimal Parameters
------------------------------------
        Estimate  Std. Error  t value Pr(>|t|)
mu      0.033464    0.016286   2.0547 0.039909
omega   0.013871    0.004071   3.4071 0.000657
alpha1  0.113074    0.014353   7.8781 0.000000
beta1   0.876941    0.014971  58.5751 0.000000

LogLikelihood : -3027.514

Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic p-value
Lag[1]                     0.5861  0.4439
Lag[2*(p+q)+(p+q)-1][2]    0.8087  0.5647
Lag[4*(p+q)+(p+q)-1][5]    3.3909  0.3403
d.o.f=0
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                     0.5995  0.4388
Lag[2*(p+q)+(p+q)-1][5]    1.3486  0.7770
Lag[4*(p+q)+(p+q)-1][9]    2.9399  0.7689
d.o.f=2
H0 : No serial correlation
```

Listing 1: FTSE return's garch

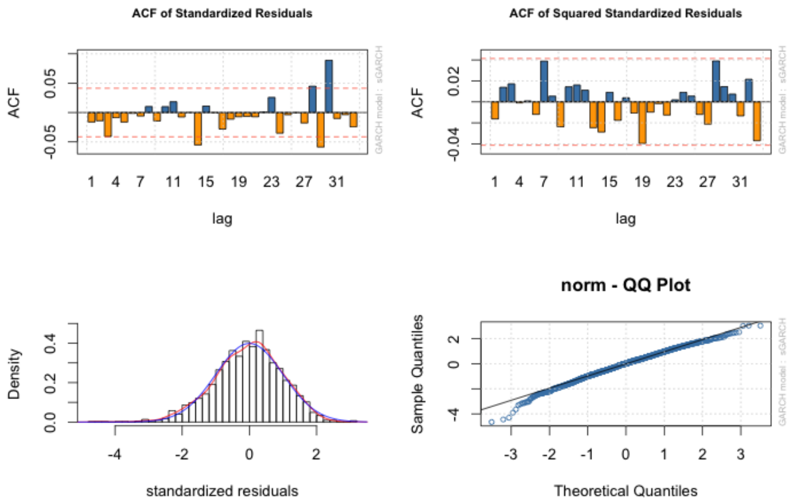


Figure 6: Residuals and standardized residual auto-correlations for FTSE return

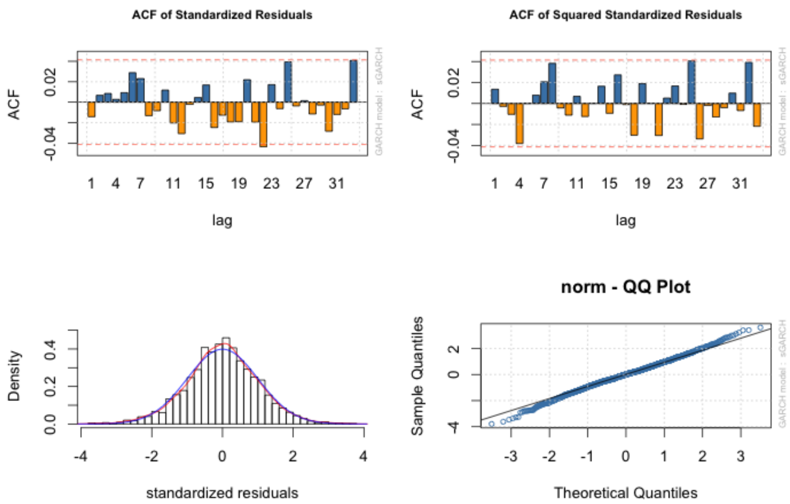


Figure 7: Residuals and standardized residual auto-correlations for GBP return

```
> print(dStats(as.numeric(residuals(fit)/sigma(fit)), filename))
              mean        sd         skew     kurt     jb.pvalue
ftse  -0.03624139 1.001062  -0.3507315 3.429448 1.765255e-14
> print(dStats(as.numeric(residuals(fit)/sigma(fit)), filename))
              mean        sd           skew     kurt     jb.pvalue
gbp  -0.003552507 0.9997188  -0.05486186 3.462702 2.599595e-05
```

Listing 2: Descriptive Statistics

In term of normality test, of the standardized residuals, we find that its tail is heavier than normal, as indicated y `kutorsis` = 3.43, and it is negatively skewed as `skewness` = -0.35. Not surprisingly, jarque test rejects the normality with p-value closed to zero.

Next, we look at the garch fit for GBP returns data. In much the same way, the standardized residuals pass the serial correlation test. However, it doesn't pass the jarque test due to negative skewness and a greter-than-3 kurtosis.

```
Conditional Variance Dynamics
-----------------------------------
GARCH Model  : sGARCH(1,1)
Mean Model   : ARFIMA(0,0,0)
Distribution  : norm

Optimal Parameters
------------------------------------
        Estimate  Std. Error  t value Pr(>|t|)
mu      0.012042    0.010306   1.1685 0.242610
omega   0.003750    0.001137   3.2972 0.000977
alpha1  0.032960    0.005061   6.5120 0.000000
beta1   0.953065    0.005686 167.6286 0.000000

LogLikelihood : -1639.311

Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic p-value
Lag[1]                     0.4728  0.4917
Lag[2*(p+q)+(p+q)-1][2]    0.5227  0.6831
Lag[4*(p+q)+(p+q)-1][5]    0.6879  0.9252
d.o.f=0
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                     0.3253  0.5684
Lag[2*(p+q)+(p+q)-1][5]    1.9145  0.6388
Lag[4*(p+q)+(p+q)-1][9]    3.9615  0.5954
d.o.f=2
H0 : No serial correlation
```

Listing 3: GBP return's garch

Problem 2.2 Listing 4 show tests for leverage effects for FTSE return, we first test the sample correlation between square return and one lagged return. It results in negative correlation so we proceed by doing *bias* and *sign bias test*. It happen to be that the average of squared return following shock of negative sign is around 10% higher than the average of squared return after positive shock, as indicated by the coefficient of $S^-_{t-1} := 1_{\{\epsilon_{t-1}<0\}}$

Also, it can be seen that the magnitude of coefficient for $S^-_{t-1}\epsilon_{t-1}$ is larger than that of $S^+_{t-1}\epsilon_{t-1}$. This adds evidence to asymmetric leverage effects.

```
======= Correlation X^2(t) and X(t-1) =======

 -0.059759

============= Sign Bias test ============

Call:
lm(formula = res_t^2 ~ sminus)


Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.162e-04  8.179e-06  14.208   <2e-16 ***
sminus      1.437e-05  1.163e-05   1.235    0.217
---

============ Negative and Positive Sign Bias ============

Call:
lm(formula = res_t^2 ~ sminus + I(sminus * res_t_1) + I(splus *
    res_t_1))

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)           6.611e-05  1.170e-05   5.653 1.78e-08 ***
sminus               -9.880e-06  1.648e-05  -0.600    0.549
I(sminus * res_t_1)  -8.806e-03  9.880e-04  -8.913  < 2e-16 ***
I(splus * res_t_1)    6.566e-03  1.120e-03   5.861 5.29e-09 ***
---
```

Listing 4: FTSE return's asymmetric test

```
GARCH Model  : gjrGARCH(1,1)
Mean Model   : ARFIMA(0,0,0)
Distribution : norm

Optimal Parameters
------------------------------------
        Estimate  Std. Error  t value Pr(>|t|)
mu      0.000051    0.000165  0.31086  0.75591
omega   0.000001    0.000001  1.44114  0.14954
alpha1  0.012064    0.013898  0.86805  0.38537
beta1   0.905538    0.015189 59.61712  0.00000
gamma1  0.132726    0.020412  6.50233  0.00000

LogLikelihood : 7342.244
```

Listing 5: GJR-GARCH parameters

Above show the parameters we get from fitting gjr-garch to the FTSE data. Notice that estimation for γ_1, which capture the effects of negative schock, is of high significance. By looking at the acf of square standardized residuals in figure 8, we found no-serial correlation. Also, the overlay of standardized residuals on *standard normal distribution* suggests that it is negatively skewed and may have heavier tails. The most important thing to look at is the bottom most part of figure 8, here we find that news impact curve of the model is obviously asymmetric and place higher value on the event after negative shock ϵ_{t-1}.

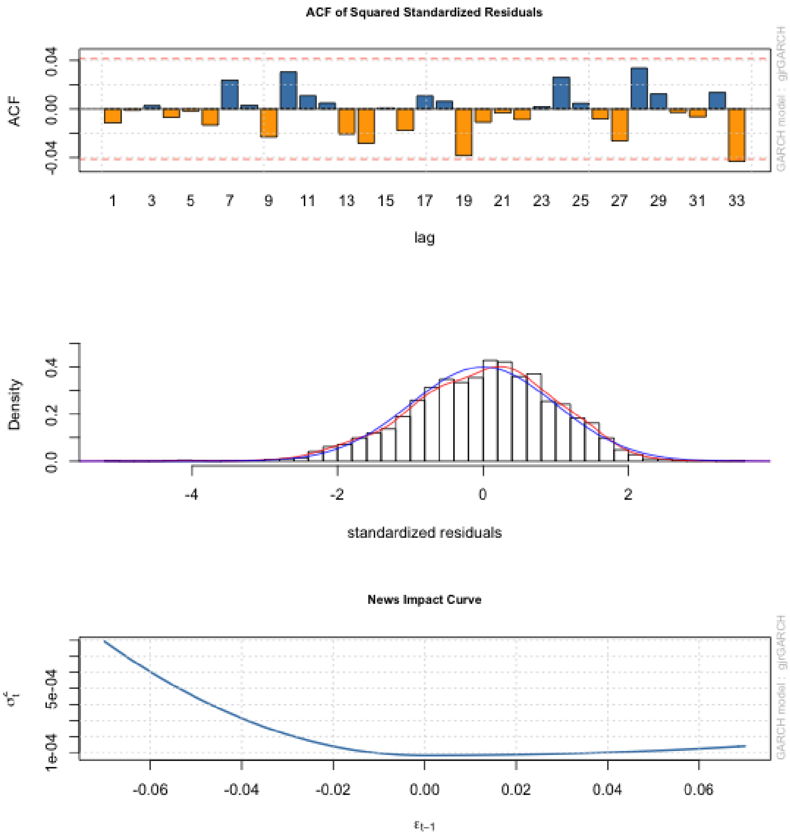


Figure 8: GJR-GARCH plot for GBP return

In Listing 6 we perform similar analysis on GBP, but do not found significant leveraging effects. Although the correlation between between square return and one lagged return results in negative value, we found no evidence of bias and sign bias test. the slope term on bias test is very small compared to interception. This means that negative shock almost never increase the size of square returns on avarage. For sign bias test, we found that coefficent for $S^-_{t-1}\epsilon_{t-1}$ is about two time larger in magnitude than that of $S^+_{t-1}\epsilon_{t-1}$, but the estimate is not highly significant in terms of p-value.

```
======== Correlation X^2(t) and X(t-1) ========

 -0.016473

============== Sign Bias test =============

Call:
lm(formula = res_t^2 ~ sminus)

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 2.593e-05  1.260e-06  20.579   <2e-16 ***
sminus      4.933e-07  1.786e-06   0.276    0.782
---

============= Negative and Positive Sign Bias =============

Call:
lm(formula = res_t^2 ~ sminus + I(sminus * res_t_1) + I(splus *
    res_t_1))

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)          2.454e-05  1.988e-06  12.343   <2e-16 ***
sminus              -1.201e-06  2.807e-06  -0.428   0.6687
I(sminus * res_t_1) -7.694e-04  3.810e-04  -2.019   0.0436 *
I(splus * res_t_1)   3.551e-04  3.950e-04   0.899   0.3688
---
```

Listing 6: GBP return's asymmetric test

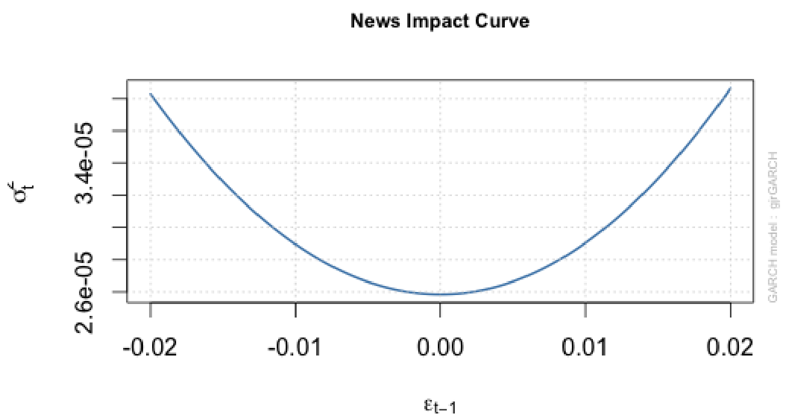


Figure 9: News impact curve obtained from fitting GJR-GARCH to GBP return. It is provided here to show that the model does not find any asymmetric effects.

Problem 2.3 In **rugarch** package, one can use the function *ugarchroll* to fit garch parameter in a rolling scheme. Specifically, by setting *n.start=T, refit.every = 1, refit.window='moving'*, **R** will 1) use the data from index 1 up to index T as data set for fitting garch model 2) fit the garch and keep resulting parameters in the output table 3) move data window to *[2, T+1], [3, T+2], [4, T+3]* and so on. As a result the parameters estimated, i.e., conditional mean and variance, can be used to estimate Value-at-Risk and Expected Shortfall starting from index *T+1, T+2, T+3,...* until the final index. Although this function is capable for calculating 1-step ahead *VaR* we choose to calculate it ourself.

```

spec = ugarchspec(variance.model=list(model="sGARCH", garchOrder=c
    (1,1)), mean.model=list(armaOrder=c(0,0), include.mean=TRUE),
    distribution.model="norm")

roll = ugarchroll(spec, timeserie, n.start = N - 500, refit.every =
     1, refit.window = "moving", solver = "hybrid", keep.coef =
    TRUE)
```

Listing 7: Code example for running rolling estimate for garch parameters for the preserved 500 out-of-sample data. Denote by N the total lenght of time serie data.

```
                          Mu          Sigma  Skew  Shape  Shape(GIG)
                Realized
2006-01-10  0.0002872084  0.005286539     0     0          0
    -0.005144188
2006-01-11  0.0002840919  0.005378281     0     0          0
    0.002680380
2006-01-12  0.0002843674  0.005229109     0     0          0
    0.003301169
2006-01-13  0.0003032699  0.005120828     0     0          0
    -0.006227171
2006-01-16  0.0002929959  0.005362310     0     0          0
    0.005808579
2006-01-17  0.0002997335  0.005452128     0     0          0
    -0.004931632
2006-01-18  0.0002782108  0.005509557     0     0          0
    -0.005608165
2006-01-19  0.0002742416  0.005620229     0     0          0
    0.004643481
2006-01-20  0.0002810091  0.005575741     0     0          0
    -0.003128463
2006-01-23  0.0002766398  0.005459188     0     0          0
    -0.004392820
```

Listing 8: FTSE, standard normal innovation

```
                              Mu          Sigma  Skew      Shape  Shape(GIG)
                                Realized
2006-01-10  0.0003671518  0.005193912     0  19.19419           0
    -0.005144188
2006-01-11  0.0003602072  0.005293027     0  19.25214           0
    0.002680380
2006-01-12  0.0003656678  0.005150682     0  19.40932           0
    0.003301169
2006-01-13  0.0003773793  0.005041203     0  19.98371           0
    -0.006227171
2006-01-16  0.0003644001  0.005283412     0  20.38688           0
    0.005808579
2006-01-17  0.0003693730  0.005361838     0  20.31616           0
    -0.004931632
2006-01-18  0.0003539342  0.005421097     0  20.45726           0
    -0.005608165
2006-01-19  0.0003474083  0.005538289     0  20.62465           0
    0.004643481
2006-01-20  0.0003513347  0.005493350     0  20.65085           0
    -0.003128463
2006-01-23  0.0003491576  0.005395644     0  20.65312           0
    -0.004392820
```

Listing 9: FTSE, std innovation

Listing 8, 9 shows the results of rolling estimation for FTSE return.

We follow as in lecture note 3 the formula for calculating *VaR* and *ES* under *standard normal*:

$$
\begin{aligned}
VaR_\alpha &= \mu_L + \sigma_L q_\alpha^Z \\
ES_\alpha &= \mu_L + \sigma_L \frac{\phi(q_\alpha^Z)}{1-\alpha}
\end{aligned}
$$

And *standardized t-distribution* :

$$
\begin{aligned}
VaR_\alpha &= \mu_L + \sigma_L q_\alpha^{t_\nu} \\
ES_\alpha &= \mu_L + \sigma_L \frac{g_\nu(q_\alpha^{t_\nu})}{1-\alpha} \left(\frac{\nu + (q_\alpha^{t_\nu})^2}{\nu - 1} \right)
\end{aligned}
$$

While computing quantile and density function for normal distribution is straightforward in **R**, one need to be careful in t-distribution case. This is because function provided in standard library is for t-distribution without standardization. This can be done by recognizing that a random variable with the standardized t-distribution can be written as scaling version of t-distribution case:

$$T_{std} = \sqrt{\frac{\nu - 2}{\nu}} T_t$$
$$q^{std}(\alpha) = \sqrt{\frac{\nu - 2}{\nu}} q^t(\alpha)$$
$$f_{std}(z) = \sqrt{\frac{\nu - 2}{\nu}} f_t(t)$$

Here we denote by $T_{std}, q^{std}, f_{std}$: random variable, quantile function and *pdf* of standardized t-distribtuion. So we can calculate *VaR* using all these facts as follows

```

# Standard normal
VaR95 <- mu + sigma*qnorm(0.95)
ES95 <- mu + sigma*dnorm(qnorm(0.95))/.05

# Standardized t
quantile <- sqrt((nu-2)/nu)*qt(0.95, nu)
density <- sqrt(nu/(nu-2))*dt(qt(0.95, nu), nu)
VaR95 <- mu + sigma*quantile
ES95 <- mu + sigma*(density/0.05)*((nu + quantile^2)/(nu - 1))
```

After that we calculate calculate number of days that $Loss_t > VaR_{\alpha}, t$ as shown below. We also show the plot VaR for all type for FTSE return in figure 10 and 11.

The result for GBP return can be found on Listing 12, 13 , 14 and 15 below.

```
> print(count(VaR95, 0.95))
  exceeds expected percent
       29       25     5.8

> print(count(VaR99, 0.99))
  exceeds expected percent
        9        5     1.8
```

Listing 10: FTSE, standard normal innovation

```
>   print(count(VaR99, 0.99))
  exceeds expected percent
        7        5     1.4
>   print(count(VaR95, 0.95))
  exceeds expected percent
       29       25     5.8
```

Listing 11: FTSE, standardized t innovation

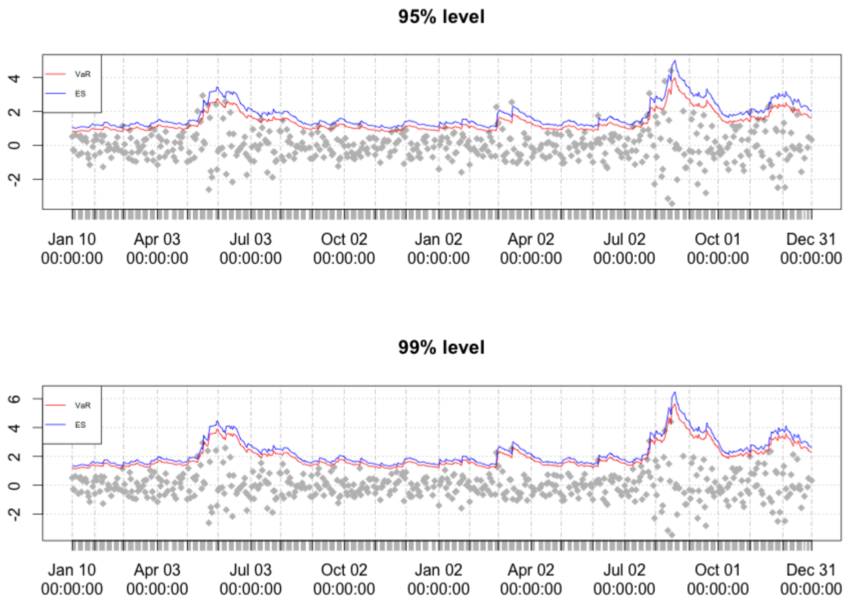


Figure 10: Rolling estimate of VaR and ES for FTSE's return for 500 reserved data points - standard normal innovation

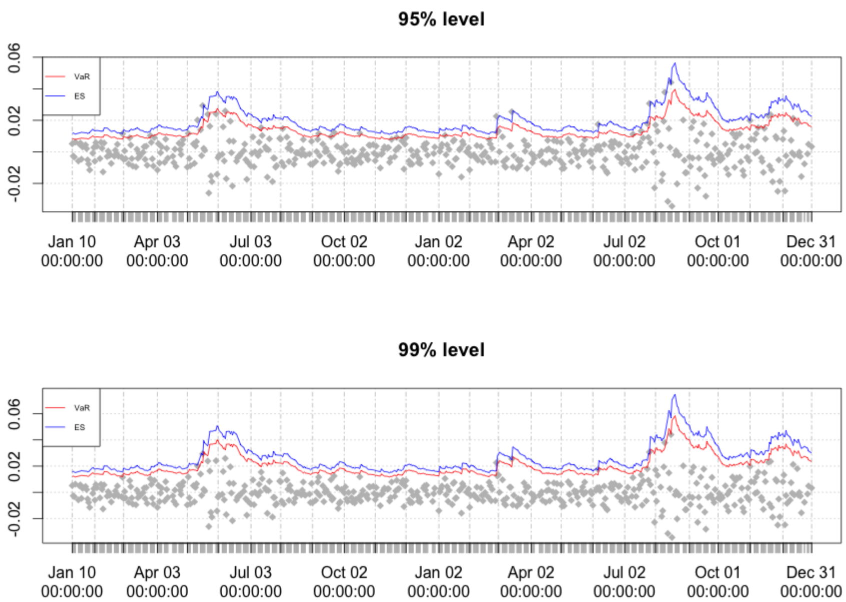


Figure 11: Rolling estimate of VaR and ES for FTSE's return for 500 reserved data points - standardized t innovation

```
                                Mu           Sigma  Skew  Shape  Shape(GIG)
        Realized
2006-01-10  5.377537e-05  0.005530654    0     0          0
    -0.000311593
2006-01-11  8.587309e-05  0.005416843    0     0          0
    -0.000226674
2006-01-12  8.306270e-05  0.005357912    0     0          0
    -0.002439994
2006-01-13  4.979335e-05  0.005340899    0     0          0
    0.009696326
2006-01-16  6.942394e-05  0.005488586    0     0          0
    -0.004991902
2006-01-17  3.234202e-05  0.005448734    0     0          0
    -0.000395905
2006-01-18  8.697987e-05  0.005400405    0     0          0
    -0.002236979
2006-01-19  7.181448e-05  0.005339182    0     0          0
    -0.002440895
2006-01-20  7.704425e-05  0.005296509    0     0          0
    0.006345271
2006-01-23  3.260003e-05  0.005281065    0     0          0
    0.009415075
```

Listing 12: GBP, standard normal innovation

```
                          Mu          Sigma  Skew      Shape  Shape(GIG)
                              Realized
2006-01-10  9.182012e-05  0.005569487     0  12.73618           0
    -0.000311593
2006-01-11  9.595834e-05  0.005484682     0  12.58017           0
    -0.000226674
2006-01-12  9.441641e-05  0.005401238     0  12.55417           0
    -0.002439994
2006-01-13  8.725837e-05  0.005341658     0  12.44230           0
    0.009696326
2006-01-16  8.886080e-05  0.005514624     0  12.50095           0
    -0.004991902
2006-01-17  8.872799e-05  0.005500260     0  12.52684           0
    -0.000395905
2006-01-18  8.787962e-05  0.005416453     0  12.51621           0
    -0.002236979
2006-01-19  8.413410e-05  0.005350316     0  12.48026           0
    -0.002440895
2006-01-20  8.738584e-05  0.005291331     0  12.34297           0
    0.006345271
2006-01-23  8.859192e-05  0.005322533     0  12.41480           0
    0.009415075
```

Listing 13: GBP, std innovation

```
>   print(count(VaR99, 0.99))
  exceeds expected percent
        6        5     1.2
>   print(count(VaR95, 0.95))
  exceeds expected percent
       23       25     4.6
```

Listing 14: GBP, standard normal innovation

```
>   print(count(VaR99, 0.99))
  exceeds expected percent
        5        5       1
>   print(count(VaR95, 0.95))
  exceeds expected percent
       23       25     4.6
```

Listing 15: GBP, standardized t innovation

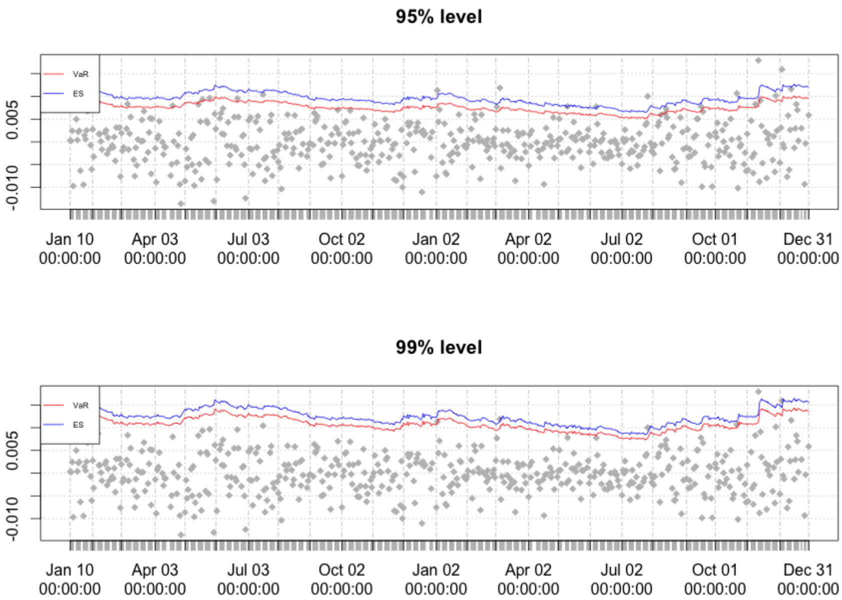


Figure 12: Rolling estimate of VaR and ES for FTSE's return for 500 reserved data points - standard normal innovation

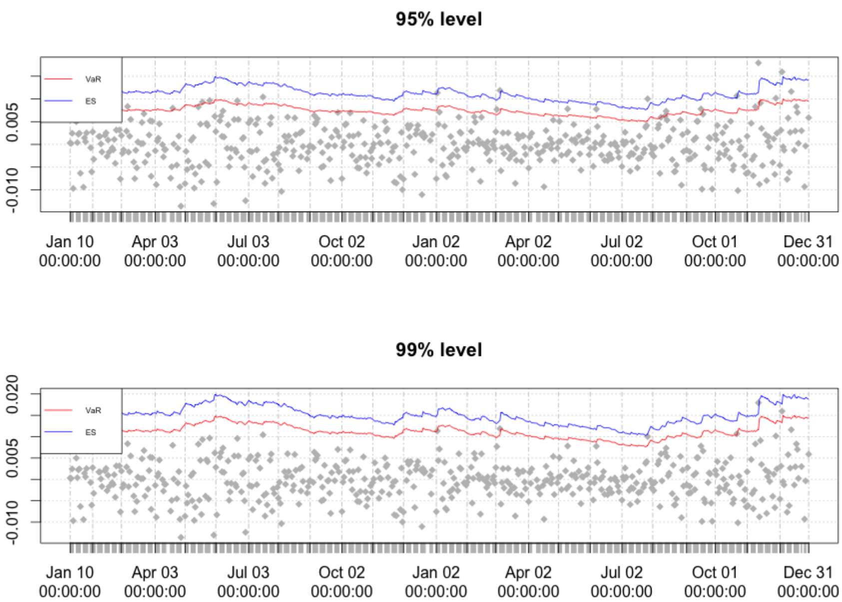


Figure 13: Rolling estimate of VaR and ES for FTSE's return for 500 reserved data points - standardized t innovation

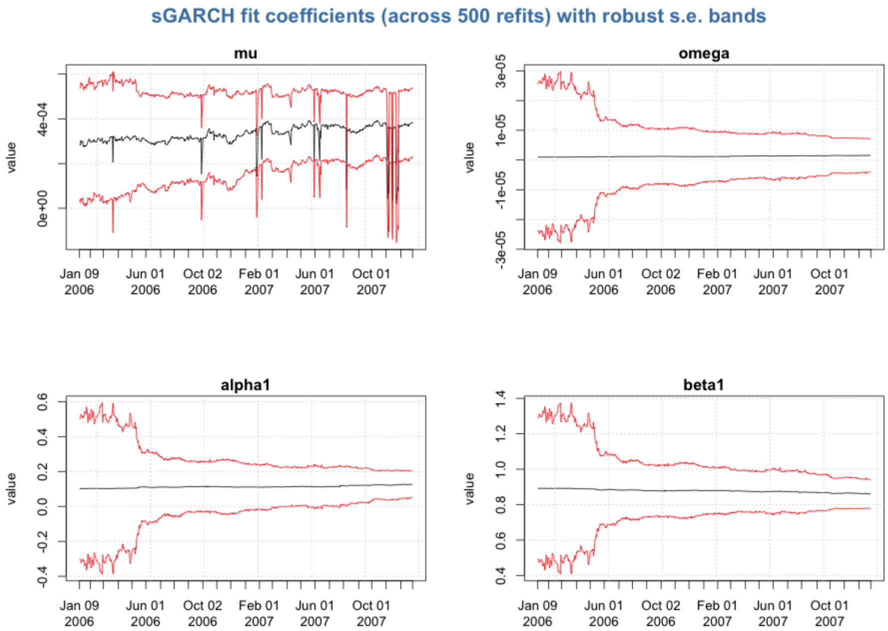


Figure 14: Rolling estimate of FTSE - standard normal innovation

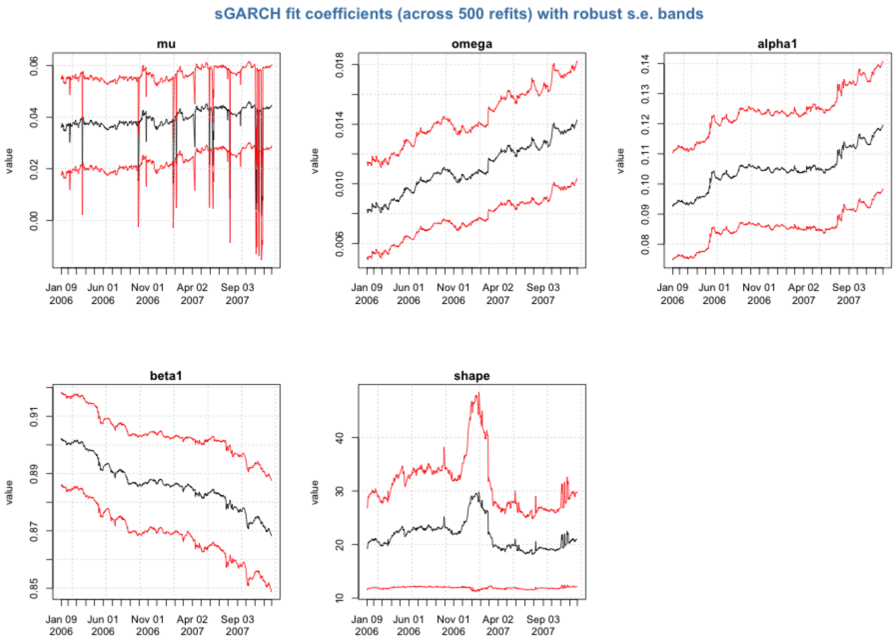


Figure 15: Rolling estimate of FTSE - standardized t innovation

Problem 2.4

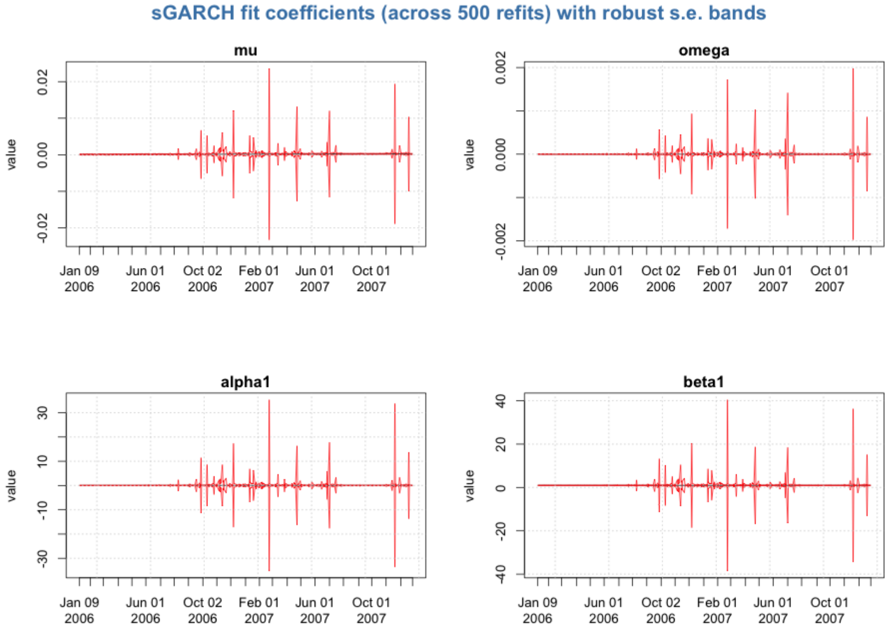


Figure 16: Rolling estimate of GBP - standard normal innovation

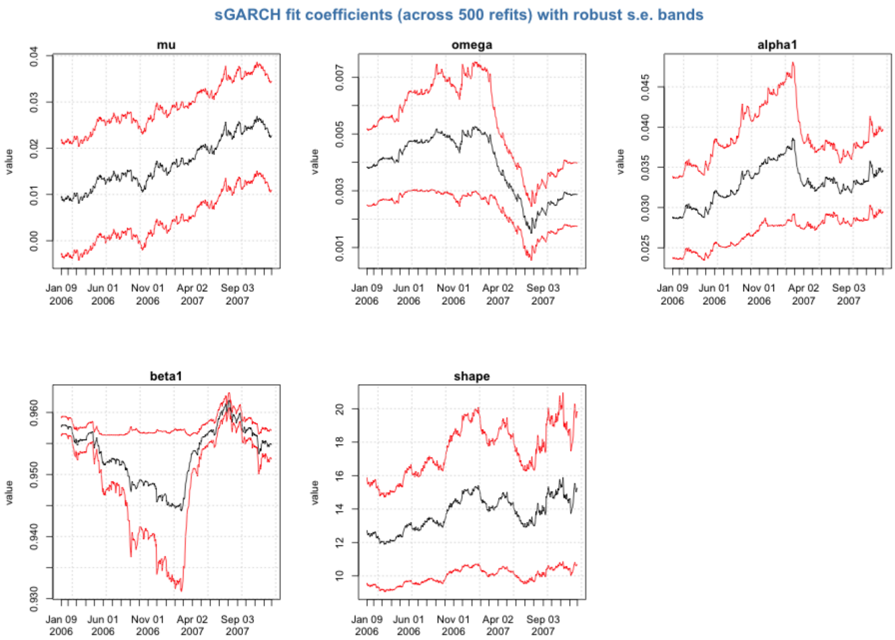


Figure 17: Rolling estimate of GBP - standardized t innovation

Problem 2.5 Three tests with 95% confident interval for likelihood ratio test

```
> load("~/Desktop/qrm/p23/roll_norm_ftse.Rda")
> prob25.var95(roll)
  test          LR critical    p.value
  unc 0.6421395 3.841459 0.42293713
   cc 3.5809977 3.841459 0.05844404
joint 4.2231371 5.991465 0.12104795
> load("~/Desktop/qrm/p23/roll_norm_gbp.Rda")
> prob25.var95(roll)
  test          LR critical    p.value
  unc 0.1728552 3.841459 0.67758663
   cc 3.5809977 3.841459 0.05844404
joint 4.2231371 5.991465 0.12104795

> load("~/Desktop/qrm/p23/roll_tdist_ftse.Rda")
> prob25.var95(roll)
  test          LR critical    p.value
  unc 0.6421395 3.841459 0.42293713
   cc 3.5809977 3.841459 0.05844404
joint 4.2231371 5.991465 0.12104795
> load("~/Desktop/qrm/p23/roll_tdist_gbp.Rda")
> prob25.var95(roll)
  test          LR critical    p.value
  unc 0.3942393 3.841459 0.53007944
  ind 3.5809977 3.841459 0.05844404
   cc 4.2231371 5.991465 0.12104795
```

```
> load("~/Desktop/qrm/p23/roll_norm_ftse.Rda")
> prob25.var99(roll)
  test        LR critical    p.value
  unc 2.612571 3.841459 0.10601978
  ind 3.580998 3.841459 0.05844404
   cc 4.223137 5.991465 0.12104795
> load("~/Desktop/qrm/p23/roll_norm_gbp.Rda")
> prob25.var99(roll)
  test          LR critical    p.value
  unc 0.1898802 3.841459 0.66301631
  ind 3.5809977 3.841459 0.05844404
   cc 4.2231371 5.991465 0.12104795
> load("~/Desktop/qrm/p23/roll_tdist_ftse.Rda")
> prob25.var99(roll)
  test        LR critical    p.value
  unc 2.612571 3.841459 0.10601978
  ind 3.580998 3.841459 0.05844404
   cc 4.223137 5.991465 0.12104795
> load("~/Desktop/qrm/p23/roll_tdist_gbp.Rda")
> prob25.var99(roll)
  test          LR critical    p.value
  unc 0.1898802 3.841459 0.66301631
  ind 3.5809977 3.841459 0.05844404
   cc 4.2231371 5.991465 0.12104795
```

3 ARMA and HAR for daily realized volatility

(i) Realized Variance and Logarithmic Realized Variance AR(1), ARMA(1,1), HAR models

1 Unit root tests

In this part, we first test whether the two datasets are stationary under the AR(1), ARMA(1,1), HAR models. We use the Dickey–Fuller test to see whether a unit root is present in our models, in which case that the data is non-stationary. It is meaningless to use our models to fit a non-stationary data. Hence we check the whether they are stationary before we fit our models. The below shows the testing results.

Table 11: Dickey–Fuller test

Dickey–Fuller test	FTS100 RV	USD/GBP RV
AR(1)	Reject	Reject
ARMA(1,1)	Reject	Reject
HAR	Reject	Reject
AR(1) for log RV	Reject	Reject
ARMA(1,1) for log RV	Reject	Reject
HAR for log RV	Reject	Reject

Clearly, the null hypothesis that a unit root is present is rejected in every models we want to apply. Hence our models could be applied to these data.

2 Model parameters

In this part, we look into the parameters of our fitted AR(1), ARMA(1,1), HAR models for the realized variance and logarithmic realized variance of FTS100 and USD/GBP. We use the MFE tool box in Matlab to fit our models.
The table 12 to table14 below show the parameters for AR(1), ARMA(1,1), HAR models fitting realized variance of FTS100 and USD/GBP respectively.

AR(1)	FTS100 RV	USD/GBP RV
a_0	2.48e-05	1.36e-05
a_1	0.6395	0.5291

Table 12: AR(1) model Parameters

ARMA(1,1)	FTS100 RV	USD/GBP RV
a_0	5.47e-06	8.19e-06
a_1	0.92	0.9563
b_1	-0.555	-0.7161

Table 13: ARMA(1,1) model Parameters

HAR	FTS100 RV	USD/GBP RV
a_0	8.19e-06	3.62e-06
a_1	0.3527	0.2029
a_2	0.3173	0.3955
a_3	0.2088	0.2765

Table 14: HAR model Parameters

The table 15 to table17 below show the parameters for AR(1), ARMA(1,1), HAR models fitting log realized variance of FTS100 and USD/GBP respectively.

AR(1)	FTS100 log(RV)	USD/GBP log(RV)
a_0	-1.6557	-4.067
a_1	0.8355	0.6147

Table 15: AR(1) Model Parameters for Log Dat

3 Model Diagnostics

In this part, we want to evaluate our models. We would test the t-statistics of our parameters in each model first, which suggest whether it is fitted appropriately. Then We would aim at testing the normality, serial correlation of the residuals of our models.

ARMA(1,1)	FTS100 log(RV)	USD/GBP log(RV)
a_0	-0.1745	-0.3477
a_1	0.9827	0.9671
b_1	-0.6566	-0.7252

Table 16: ARMA(1,1) model Parameters for log data

HAR	FTS100 log(RV)	USD/GBP log(RV)
a_0	-0.4089	-0.9972
a_1	0.2972	0.2152
a_2	0.4947	0.4417
a_3	0.1677	0.2486

Table 17: HAR model Parameters for log data

First, we calculate the t-statistics for the parameters in each model. As the MFE toolbox does not give the t-statistics directly, while merely offering the variance of fitted parameters. we would have to calculate the t-statistics by the formula below:

$$t = \frac{parameter}{standard\, variance}$$

The following table 18 to table 23 is the t-statistics for the parameters in each models.

AR(1)	FTS100 RV	USD/GBP RV
a_0	12.256	23.3746
a_1	39.4031	29.4902

Table 18: t-value for AR(1)

ARMA(1,1)	FTS100 RV	USD/GBP RV
a_0	4.4521	5.3093
a_1	197.5168	153.4031
b_1	74.2414	58.0892

Table 19: t-value for ARMA(1,1)

HAR	FTS100 RV	USD/GBP RV
a_0	3.6876	4.589
a_1	14.3843	8.2194
a_2	7.6273	8.6108
a_3	5.4053	6.084

Table 20: t-value for HAR

From the results above, we could see that all the t-statistics results are bigger than the 95% quantiles of t distribution, which suggests that our parameters

AR(1)	FTS100 log(RV)	USD/GBP log(RV)
a_0	14.091	23.079
a_1	71.8011	36.8555

Table 21: t-value for log AR(1)

ARMA(1,1)	FTS100 log(RV)	USD/GBP log(RV)
a_0	4.2163	5.1494
a_1	239.9402	151.3779
b_1	46.4253	43.83

Table 22: t-value for log ARMA(1,1)

HAR	FTS100 log(RV)	USD/GBP log(RV)
a_0	3.3856	4.4088
a_1	11.74	8.639
a_2	12.0538	9.9631
a_3	5.106	6.0195

Table 23: t-value for log HAR

are all significant. We could safely use the parameters to make our forcast.

Next, we would like to test the normality of residuals, we use QQplot to plot the quantiles of our residuals versus the standard normal quantiles.

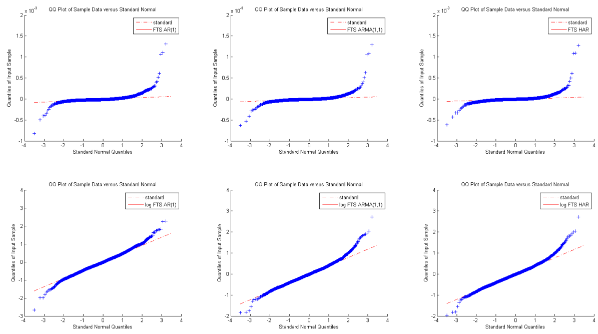


Figure 18: FTS100 models residuals QQplot

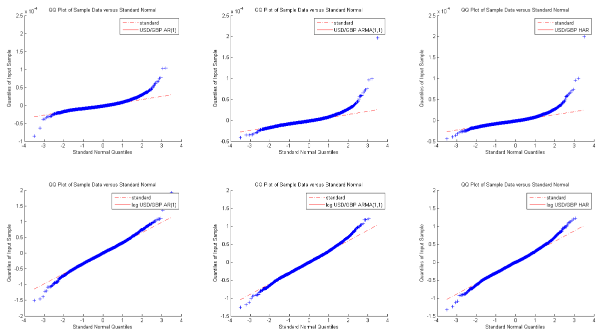


Figure 19: USD/GBP models residuals QQplot

Generally speaking, We could see that the qqplot of models in table 18 and 19 for both data are nearly linear, which means the residuals are nearly normal, suggesting the fitting is good. Specifically, the central part of the quantiles for residuals are quite closed to the standard quantiles. However, we could see that the tail part of the residuals quantiles shift away from the standard normal quantiles. The right tails of residuals tend to be bigger that the standard normal quantiles, vice versa. It shows that the residuals have fat tail, not strictly fitted into standard normal distribution.

Second, to test the serial correlation of residuals, we use Ljung–Box test.

Results in table 24 show that the null hypothesis that residuals have no serial correlation are all rejected in every models. It suggests there should be serial correlation in the residuals and the models above fail to capture the serial correlation of the data very well. The data shows a long memory feature.

Ljung–Box test	FTS100 RV	USD/GBP RV
AR(1)	Reject	Reject
ARMA(1,1)	Reject	Reject
HAR	Reject	Reject
AR(1) for log RV	Reject	Reject
ARMA(1,1) for log RV	Reject	Reject
HAR for log RV	Reject	Reject

Table 24: Ljung–Box test For Residuals

Third, to decide which is the most suitable model, we use the AIC tests.
For the realized variance, we have:

model AIC	FTS100	GBP
AR(1)	-18.8679	-22.5678
ARMA(1,1)	-18.9506	-22.6974
HAR	-18.9861	-22.7288

Table 25: AIC for Realized Variance AR(1), ARMA(1,1), HAR models

In table 25 We could see that AR(1) has a better AIC result, thanks to the simplicity and efficiency of the model.

For the log realized variance, we have:

log model AIC	FTS100	GBP
AR(1)	-1.3511	-2.072
ARMA(1,1)	-1.5715	-2.2403
HAR	-1.5931	-2.2688

Table 26: AIC for log Realized Variance AR(1), ARMA(1,1), HAR models

Similarly, in table 26 AR(1) turns out to be the more effective model to use.

(ii) AR(1), ARMA(1,1), HAR GARCH models Forecast

1 Construct 1-step ahead realized variance forecasts

In this part, we reserve the last 500 observations for out-of-sample forecasting exercise and use the rolling scheme to generate volatility forecasts for the last 500 days in the sample. For each models and each data set, we perform the forecast under both the recursive and rolling scheme.

First, let's see the 500 days realized variance forecast for FTS100 under a recursive scheme.

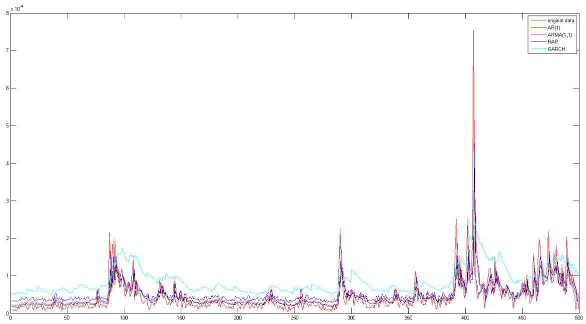


Figure 20: FTS100 500 days realized variance forecast under recursive scheme

Then, we could also see the 500 days realized variance forecast for FTS100 under a rolling scheme.

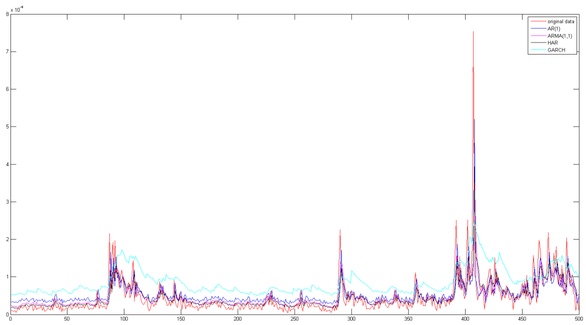


Figure 21: FTS100 500 days realized variance forecast under rolling scheme

We could find that the results under recursive scheme and rolling scheme are quite similar, while the performance of models under recursive scheme is slightly better.

Second, let's see the 500 days realized variance forecast for USD/GBP under a recursive scheme.

Then, we could also see the 500 days realized variance forecast for FTS100 under a rolling scheme.

In both data sets, the models performance are similar.
Generally speaking, we could see from the graph that the HAR model outperforms the others. The forecast 500 days realized variance black curve given by the HAR model seem to be fairly closed to the original FTS 100 index realized variance red curve. The ARMA(1,1) model also performs very well. In

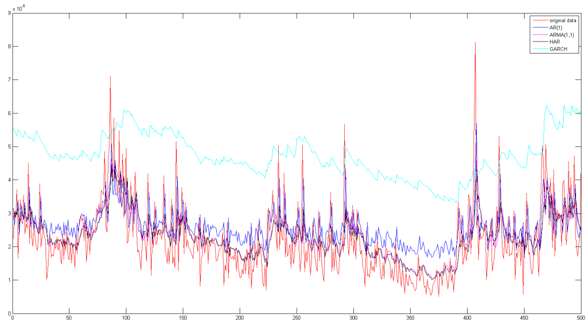


Figure 22: USD/GBP 500 days realized variance forecast under recursive scheme

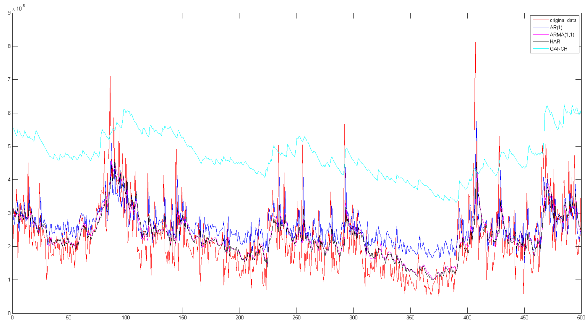


Figure 23: USD/GBP 500 days realized variance forecast under rolling scheme

both the rolling and recursive scheme, the forecast 500 days realized variance purple curve given by ARMA(1,1) model is quite near the HAR curve as well as the original curve. Compared to the HAR and ARMA(1,1) models, AR(1) performs not very well. We could see from the graphs that the curve given by AR(1) model is quite away from the original FTS 100 index realized variance curve. However, the worst model tend to be the GARCH model. We use the predicted FTS 100 standard variance for the 500 days in problem 2. It turns out that there is a huge error between the curve given by GARCH model and the original realized variance curve, especially for the USD/GBP data set.
Moreover, in both data sets, we could find from the graphs that the predicted realized variances generally tend to be bigger than the actual realized variances.

2 Forecasts evaluations—errors

We calculate the mean-square errors and the mean absolute errors to test the forecast accuracy of our models. We also test our results under both recursive scheme and rolling scheme.

Mean Square Error	FTS100		USD/GBP	
	recursive	rolling	recursive	rolling
AR(1)	2.04E-09	2.05E-09	9.35E-11	9.41E-11
ARMA(1,1)	1.82E-09	1.83E-09	7.03E-11	7.02E-11
HAR	1.75E-09	1.76E-09	6.90E-11	6.90E-11
GARCH	3.55E-09	3.55E-09	7.27E-10	7.27E-10

Table 27: Mean-Square Errors

Mean Absolute Error	FTS100		USD/GBP	
	recursive	Rolling	Recursive	Rolling
AR(1)	2.49E-05	2.47E-05	7.81E-06	7.83E-06
ARMA(1,1)	2.10E-05	2.10E-05	6.21E-06	6.18E-06
HAR	1.96E-05	1.96E-05	6.10E-06	6.09E-06
GARCH	4.99E-05	4.99E-05	2.58E-05	2.58E-05

Table 28: Mean Absolute Errors

Agian, we could find out that the results in rolling and recursive scheme have little difference. The performance of each model is also similar in different data sets.
We use the error between the predicted realized variance and the actual one to judge the accuracy of our models.The GARCH model has the largest error in all the cases, ranging from roughly 1.7 to 4 times of HAR model error. AR(1) model has better mean square error and mean absolute error in both data set, which is basically 1.2 to 1.3 times of the HAR model error. The ARMA(1,1) is better than AR(1) model in terms of both mean square error and mean absolute error. Consistent to our analysis above, the HAR predicted curve is closest to the original data, giving out the smallest mean square error and mean absolute error.

3 Forecasts evaluations—Mincer-Zarnowitz regression

For each model forecast, we perform a Mincer-Zarnowitz regression to evaluate the focast accuracy. Under our setting, intercept b should nearly equals 0, while slope a should nearly equals to 1.
The model of the Mincer-Zarnowitz regression is below:

$$\sigma_{t+h}^2 = \alpha + \beta * E(\sigma_{T+h}^2) + e_{i,T+h}$$

Under the FTS100, we have the Mincer-Zarnowitz regression results under both recursive and rolling scheme:

Similiarly, for the USD/GBP data, we have the Mincer-Zarnowitz regression results under both recursive and rolling scheme:

In this part, we could derive the same conclusion as above. HAR is the best

Recursive	α	β	R^2
AR(1)	0	0.8167	0.276
ARMA(1,1)	0	0.89	0.3267
HAR	0	0.9141	0.3463
GARCH	0	0.8465	0.3413

Table 29: FTS100 Mincer-Zarnowitz regression results under recursive scheme

Rolling	α	β	R^2
AR(1)	0	0.8073	0.2734
ARMA(1,1)	0	0.8852	0.3249
HAR	0	0.9073	0.3454
GARCH	0	0.8465	0.3413

Table 30: FTS100 Mincer-Zarnowitz regression results under rolling scheme

Recursive	α	β	R^2
AR(1)	0	1.0259	0.2813
ARMA(1,1)	0	0.9846	0.3751
HAR	0	0.9987	0.3816
GARCH	0	0.8077	0.2512

Table 31: USD/GBP Mincer-Zarnowitz regression results under recursive scheme

Rolling	α	β	R^2
AR(1)	0	0.9947	0.274
ARMA(1,1)	0	0.9791	0.3743
HAR	0	0.9872	0.3802
GARCH	0	0.8077	0.2512

Table 32: USD/GBP Mincer-Zarnowitz regression results under rolling scheme

model in both data sets, while ARMA(1,1) is better than AR(1). The intercept α in the test of each model tend to be 0, while the slope β is all fairly closed to 1. The slope β of the test for HAR model forecast is closest to 1 compared with other models, which is 0.9073 for FTS100 realized variance and 0.9872 for USD/GBP realized variance. HAR also has the largest R^2. On the other hand, the results of Mincer-Zarnowitz regression have different features for two data sets. Under the FTS100 realized variance, AR(1) model tends to be the worst model, with slope $\beta = 0.8073$ and $R^2 = 0.2734$. But under the USD/GBP realized variance dataset, the GARCH model turns out to perform the worst, with slope $\beta = 0.8077$ and $R^2 = 0.276$ smaller than other models.

4 Conclusion

From the graphs and error analysis above, we could draw a conclusion that

HAR model tends to have the best performance. The curve of forcast data of HAR is the closest one to the original data curve. It also has the least mean square error and absolute error. On the other hand, GARCH model, compared to HAR, ARMA(1,1) and AR(1) models which utilize the realized variance to predict volatility, has the worst performance.
The accuracy of HAR model could be attributed to the efficiency and simplicity of the models. HAR take advantage of fully using the 22 legs of data while only needing to estimate 3 parameters. As the realized variance in these two data sets both show a strong time correlation, HAR serves to capture the long-memory feature of the data and prove to perform very successfully in the practice. It suggests that it is more accurate to use intra-day realized variance to predict volatility than the traditional way to use the GARCH models.
On the other hand, basically ARMA(1,1) model has a satisfying forcast result too. It is the model that perform the second best. It has the second closest curve, with merely very slightly shift from the original data curve. However, the AR(1) model perform not so well. though still better than the GARCH model. GARCH model turns out to have the worst performance. It could be due to that GARCH merely use the daily retun standard variance data to model the volatility, while other models take advantage of using the intra-day realized variance data to model volatility.
Therefore, realized variance is a good estimation of volatility, while the daily variance is not.

(iii) 1-day ahead out-of-sample VaR and ES at the 95 and 99 level

We assume that the daily returns standardized by realized volatility are iid standard norma. Therefore we could derive the daily return value at risk VaR at α level by://

$$VaR_\alpha = \sigma * quantiles_\alpha$$

Similiarly, we could derive the daily return expected shortfall ES at α level by://

$$ES_\alpha = \sigma * \Phi(quantiles_\alpha)$$

Hence, we got the results below.

1 VaR95

The belows are the FTS100 VaR at 95 percent under both recursive and rolling scheme.

The belows are the USD/GBP VaR at 95 percent under both recursive and rolling scheme.

We could see that the original return data is bounded by the VaR95 and VaR5 curve in both data sets. It suggests that VaR95 could be a satisfying risk measure.

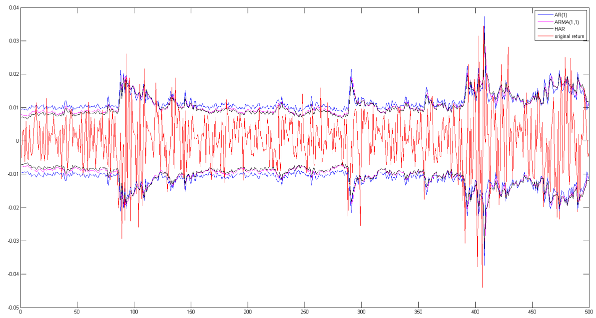


Figure 24: FTS100 500 days VaR at 95 percent under recursive scheme

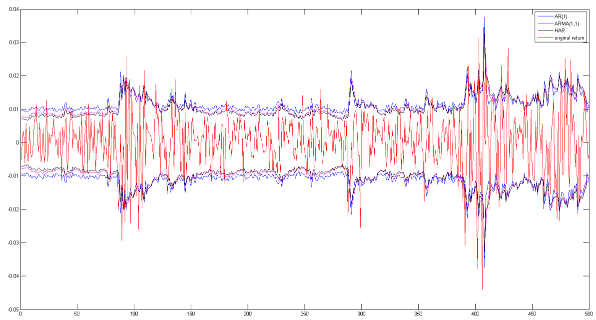


Figure 25: FTS100 500 days VaR at 95 percent under rolling scheme

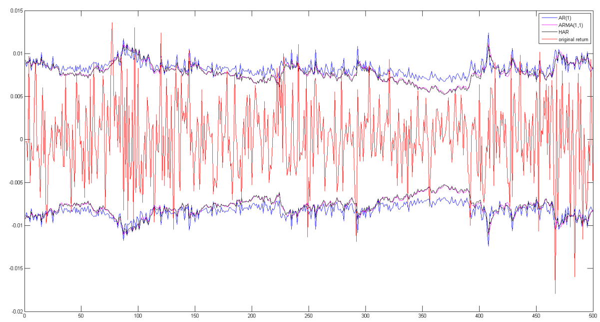


Figure 26: USD/GBP 500 days VaR at 95 percent under recursive scheme

2 VaR99

The belows are the FTS100 VaR at 99 percent under both recursive and rolling scheme.

The belows are the USD/GBP VaR at 99 percent under both recursive and

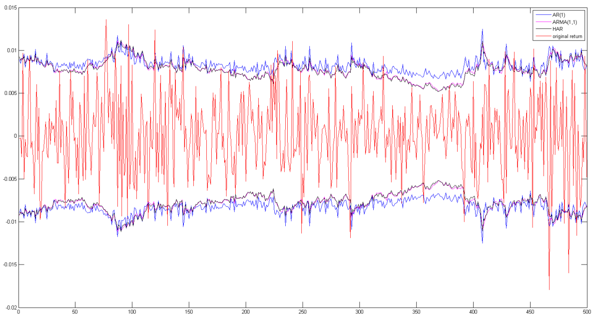


Figure 27: USD/GBP 500 days VaR at 95 percent under rolling scheme

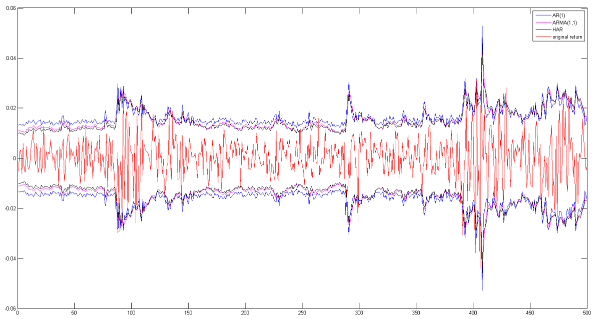


Figure 28: FTS100 500 days VaR at 99 percent under recursive scheme

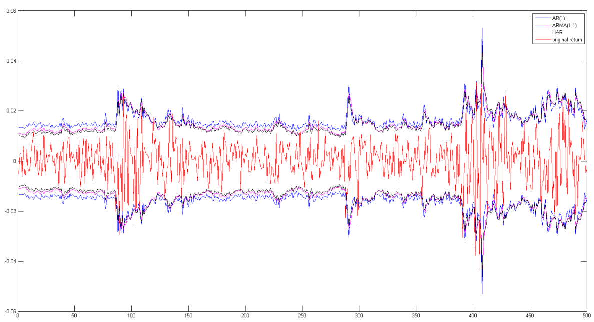


Figure 29: FTS100 500 days VaR at 95 percent under rolling scheme

rolling scheme.

We could see that the original return data is bounded by the VaR99 and VaR1 curve in both data sets. It suggests that VaR95 could be a satisfying risk measure.

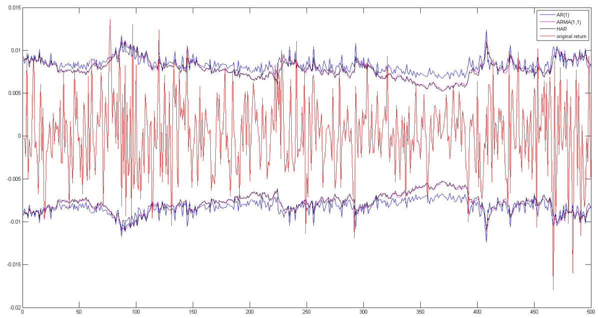


Figure 30: USD/GBP 500 days VaR at 99 percent under recursive scheme

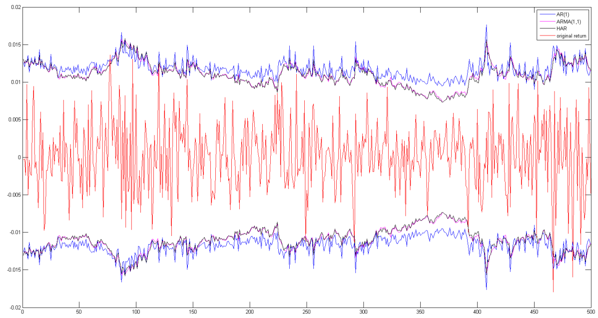


Figure 31: USD/GBP 500 days VaR at 99 percent under rolling scheme

3 ES95

The belows are the FTS100 expected shortfall at 95 percent under both recursive and rolling scheme.

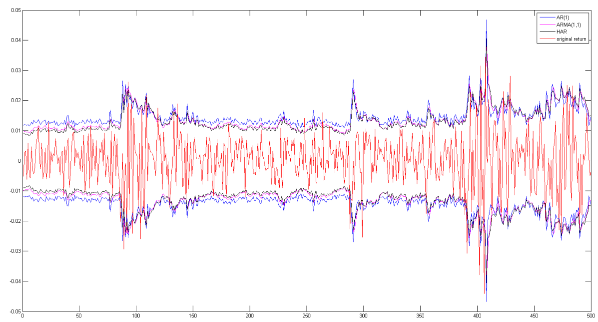


Figure 32: FTS100 500 days expected shortfall at 95 percent under recursive scheme

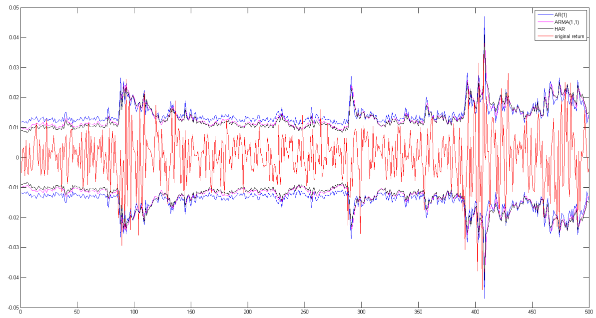


Figure 33: FTS100 500 days expected shortfall at 95 percent under rolling scheme

The belows are the USD/GBP expected shortfall at 95 percent under both recursive and rolling scheme.

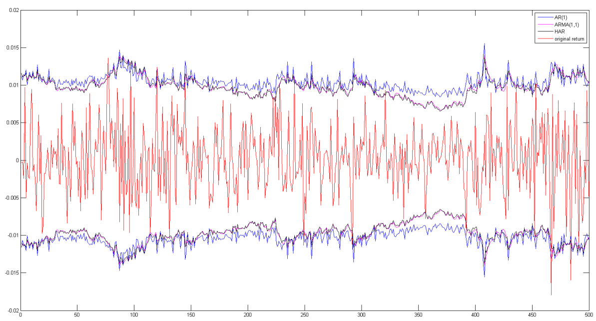


Figure 34: USD/GBP 500 days expected shortfall at 95 percent under recursive scheme

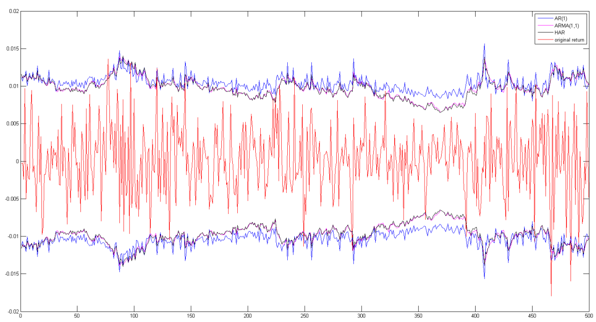


Figure 35: USD/GBP 500 days expected shortfall at 95 percent under rolling scheme

We could see that the original return data is also bounded by the ES95 and ES5 curve in both data sets. It suggests that ESR95 could be a satisfying risk measure.

4 ES99

The belows are the FTS100 expected shortfall at 99 percent under both recursive and rolling scheme.

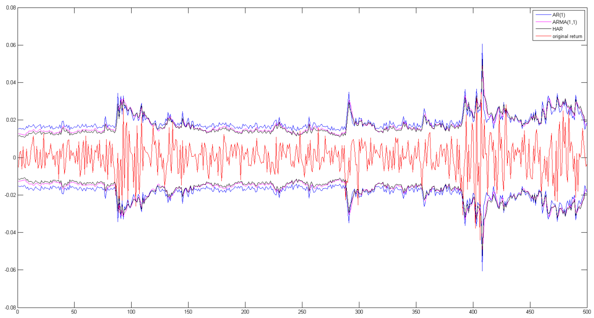


Figure 36: FTS100 500 days expected shortfall at 99 percent under recursive scheme

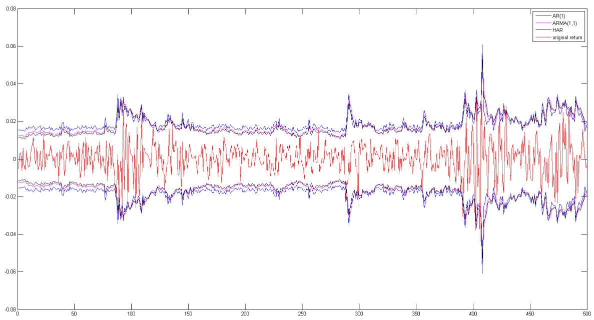


Figure 37: FTS100 500 days expected shortfall at 99 percent under rolling scheme

The belows are the USD/GBP expected shortfall at 99 percent under both recursive and rolling scheme.

We could see that the original return data is also bounded by the ES99 and ES1 curve in both data sets. It suggests that ESR95 could be a satisfying risk measure.

5 VaR discuss

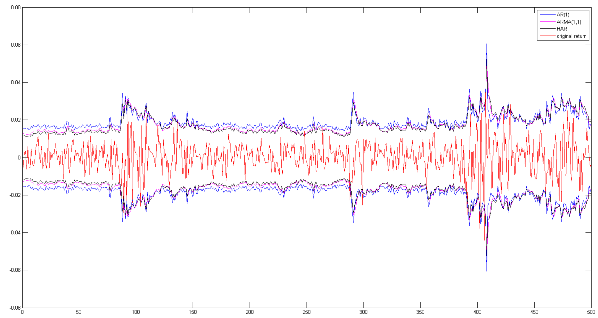


Figure 38: USD/GBP 500 days expected shortfall at 99 percent under recursive scheme

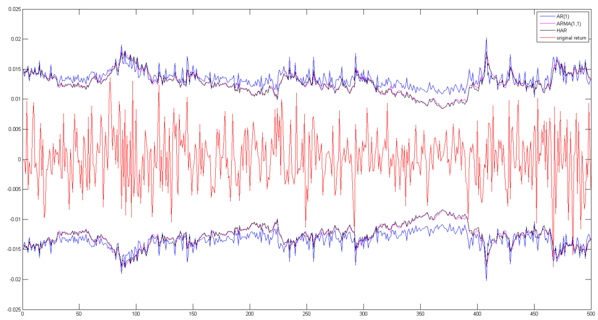


Figure 39: USD/GBP 500 days expected shortfall at 95 percent under rolling scheme

In this part, we calculate the number and frequency that the original return data of FTS100 and USD/GBP go above the VaR95 and VaR99

Across VaR	FTS100	USD/GBP		
	across VaR95	across VaR99	across VaR95	across VaR99
AR(1)	34	8	17	1
ARMA(1,1)	40	8	25	2
HAR	43	9	27	1

Table 33: Number of original return above the VaR

Across VaR	FTS100		USD/GBP	
	across VaR95	across VaR99	across VaR95	across VaR99
AR(1)	0.068	0.016	0.034	0.002
ARMA(1,1)	0.08	0.016	0.05	0.004
HAR	0.086	0.018	0.054	0.002

Table 34: Frequency of original return above the VaR

We could see the frequency that original return data of FTS100 and USD/GBP go above the VaR95 in each model is bigger than the 0.05, which under the null hypothesis of normality that the frequency should be near 0.05. The same feature appears at the VaR99 case too. It suggests the fat tails of both data sets.

4 Problem 4 - VaR and ES by Monte Carlo

The model specified in the question has the form as follow.

$$
\begin{aligned}
r_t &= c + \rho r_{t-1} + \gamma \sigma_t + \epsilon_t \\
\epsilon_t &= \sigma_t Z_t, Z_t \overset{\text{iid}}{\sim} t_\nu \\
\sigma_t^2 &= a_0 + a_1 \epsilon_{t-1}^2 + b_1 \sigma_{t-1}^2
\end{aligned}
$$

It has the form of GARCH-in-Mean(GARCH-M) in Lecture 6, where $g(\sigma_t) = \sigma_t$ in this case. Even though the problem has not explicitly specified, we have assumed that the innovation distribution has been standardised i.e. variance equal to 1, scaled by $\sqrt{\frac{\nu}{\nu-2}}$. By this assumption, we can achieve stationarity with the given parameters.

In order to find the 5-day VaR and ES at 95% and 99% level, We first need to simulate multiple paths of 5 day returns. The risk measure is the sum $\sum_{j-1}^{5} r_{T+j}$.

The VaR and ES are then calculated empirically with quantiles of the corresponding confidence levels, 95% and 99%.

> VaR is calculated by finding the 5% (=1-95%) and 1%(=1-99%) quantiles in the empirical distribution.
>
> ES is calculated by finding the mean of the lowest 5% (=1-95%) and 1% (=1-99%) values in the empirical distribution

With 2000000 trails, the VaR and ES converge to the values in table 35.

	VaR	ES
95%	-0.328	-1.287
99%	-1.830	-2.900

Table 35: VaR and ES

In the Figure 40 the values of VaR and ES become stable when the sample size is larger than 400000 and the error inferred from the plot is about ± 0.02.

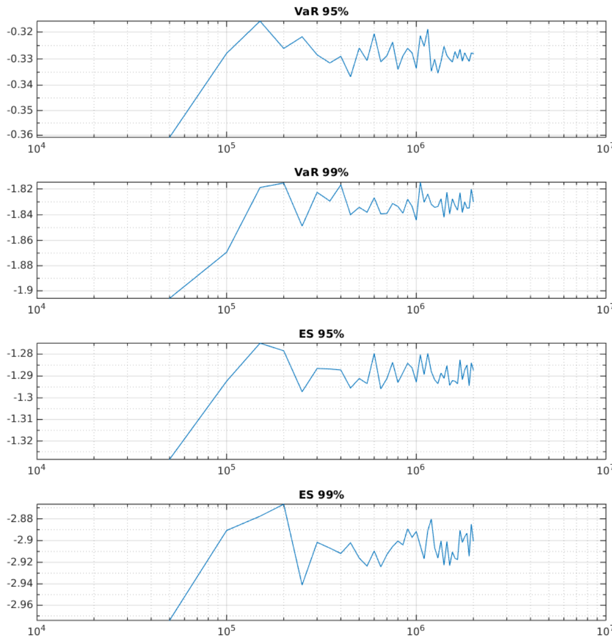


Figure 40: VaR and ES against Number of Sample Paths

5 Problem 5 - Constructing realized measures

Notes:

> The regular time grid is filled with tick data. The missing values are filled using the "last tick" method introduced in the lecture. For the days with first transaction happening later then 7:20, the price of first transaction is filled backward within in that day.
>
> The main script of this problem is `q5.m` each sub-problem has its own section inside the script.
>
> Two function from outside in included for compatibility reasons. For example `zoh` for last tick data filling and `stdtrnd`.

(i) Construct 1-second intraday log returns.

The function to calculate log-return is `get_log_rtn`. The utility function `get_time_grid` is used to create regular time gird for the market open periods. For details please refer to the source code section.

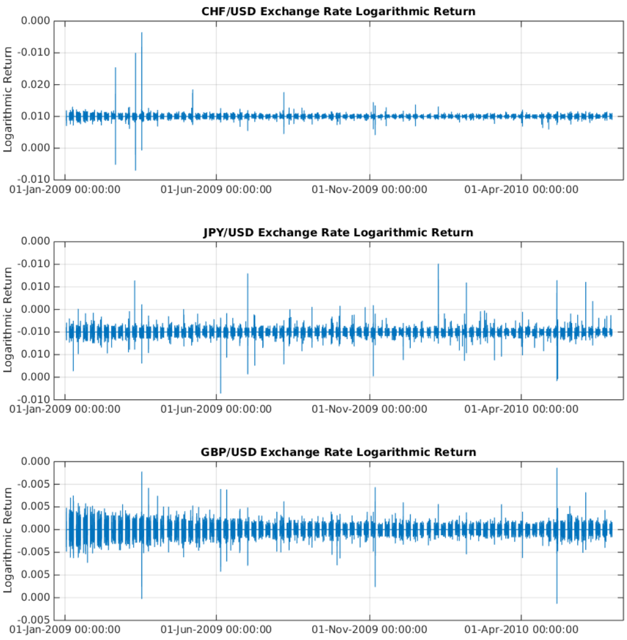


Figure 41: Log Returns

In plot Figure 41 log-returns of the exchange rates time series over the one year period. One can clear see that during the weekends there is no returns as the exchange is closed.

In addition, the general trend of the change of return is reducing during the period from 2009 to 2010. The suggest the volatility is decreasing over the time frame consider in this problem. This observation can be verified by the result from the next part.

(ii) Construct the time-series of daily average 30-minute $RV^{avg}_{t,30mins}$ using the 1-minute grid. Construct the time-series of daily range-based estimates RP_t.

The function to calculate RV estimators is `get_rv_estimators`. It computes all three RV estimators, RV,RP and average RV all in one go. This function is also used in the latter part of this question.

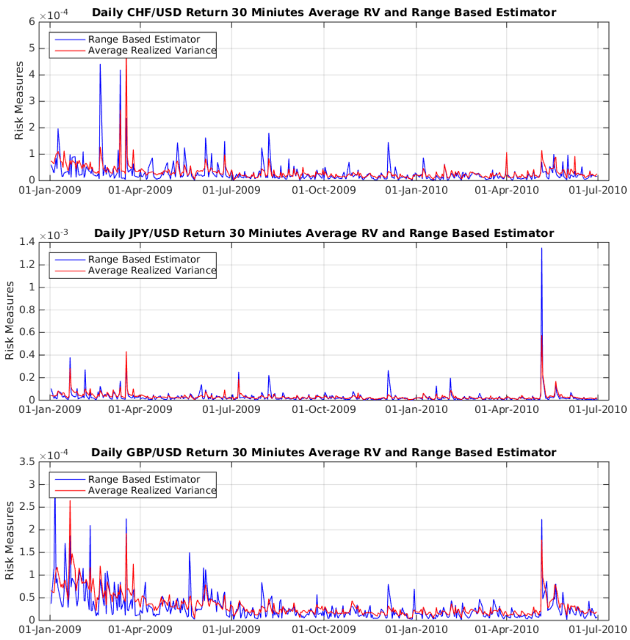


Figure 42: Average RV versus Range-based Estimates

As suggested from the previous plot of returns. Both of the 30 minutes average realised variance and the range based estimator have a decreasing trend. However there is anomaly in the plot during the second quarter of 2010.

The way to construct the 30 minutes RV estimator is based on the following formula,

$$RV^{Avg}_{t,30min} = \frac{1}{30}\sum_{s=1}^{30} RV^{s}_{t,30min}$$

The original time series is multiplied with two matrices and one scaling factor in Matlab to achieve the calculations specified above. Define the vector $\mathbf{V}$ as a vector of regular one minute data vector with regular interval. $\mathbf{W}$, the moving windows matrix. $\mathbf{S}$, the summing matrix together with $1/n = 1/30$, the scaling factor to perform averaging. $\mathbf{V^{30min}_{RV}} = \mathbf{\frac{1}{n}SWV^{1min}}$

$$\mathbf{W} = \begin{pmatrix} \underbrace{1\ 1\ \cdots\ 1}_{30} & & & & & & \\ & \underbrace{1\ 1\ \cdots\ 1}_{30} & & & & & \\ & & \underbrace{1\ 1\ \cdots\ 1}_{30} & & & & \\ & & & \ddots & & & \\ & & & & \ddots & & \\ & & & & 1 & 1\ 1\ 1 & \\ & & & & & 1\ 1\ 1 & \\ & & & & & \ 1\ 1 & \\ & & & & & \ 1 & \end{pmatrix} \quad \mathbf{S} = \begin{pmatrix} \underbrace{1\ 1\ \cdots\ 1}_{30} & & & \\ & \underbrace{1\ 1\ \cdots\ 1}_{30} & & \\ & & \ddots & \\ & & & \underbrace{1\ 1\ \cdots\ 1}_{30} \end{pmatrix}$$

Notice that the moving window matrix is not completely regular. This is due the daily time interval 7:20 to 14:00 does not the fits 30 min intervals required completely. For the last 30 minutes average RV we have to assume there is zero square return for the part of the sliding window which is outside the trading period. The justification is that we should only consider the volatility within each trading periods. Hence any scaling or padding at the end would artificially introduce extra volatility that does not created during the time when the market is open. Therefore padding zeros at the end, i.e. not adding anything extra would be the solution to this problem.

For the range based estimator,

$$RP_t = \frac{1}{4\log(2)} D_t^2$$

where $D_t = p_t^{high} - p_t^{low}$ and p is the log-prices time series.

The general shapes of $RV^{Avg}_{t,30min}$ and RP does not always coincide as one would expected since they are estimating the same underlying quantity, the realised variance. On some particular days the difference is quite large. This might correspond to a sudden jump in the price on some days or very liquid trading on particular day while the bid-ask spread is tight but not much change in price level.

According to Christoffersen [2], RP is a more suitable measure for less liquid market as it is less noise. While RV is more reliable in a highly liquid market.

(iii) Construct the volatility signature plot (use sampling frequencies ranging from 30 sec- onds to 30 minutes). Include the average RV and range-based RP from the previous step for comparison.

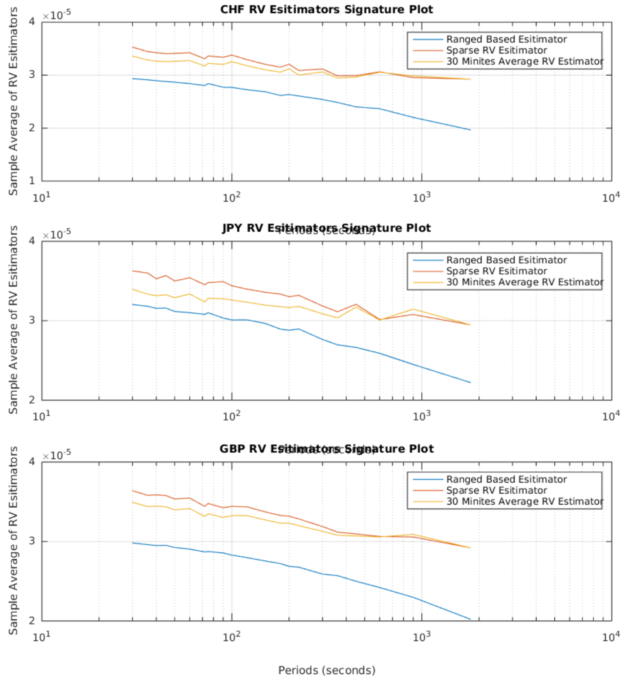


Figure 43: Signature Plots

The volatility signature plot clearly show that the realized variance suffer the most from micro-structure noise. There two observations:

> The average realised variance performs relative more stable then the realised variance.
>
> There is stabilised region from 30 minutes to 2 minutes interval for the sparse realised variance and averaged realised variance.
>
> Ranged Based estimator always has lower estimated then the other two risk measures. In particular, for at the lower end of the sampling frequencies. This is due to the fact that the more sparse the sampling the gird is the more likely we miss some of the true maximum and minimum values.

(iv) Plot the ACF of 1-minute, 5-minute and 30-minute RV_t , $RV^{avg}_{t,30mins}$ and RP_t . Comment on the relative persistence of these series.

Figure 44, 45 and 46 show the autocorrelation of daily realised variances estimators for different intra-day sampling frequencies for 1 to 100 days.

One can clearly see that all RV estimators from different sampling frequencies exhibit positive autocorrelation for many lags. The autocorrlation is most significant in the GBP/USB pair. There are significant autocorrelation up to

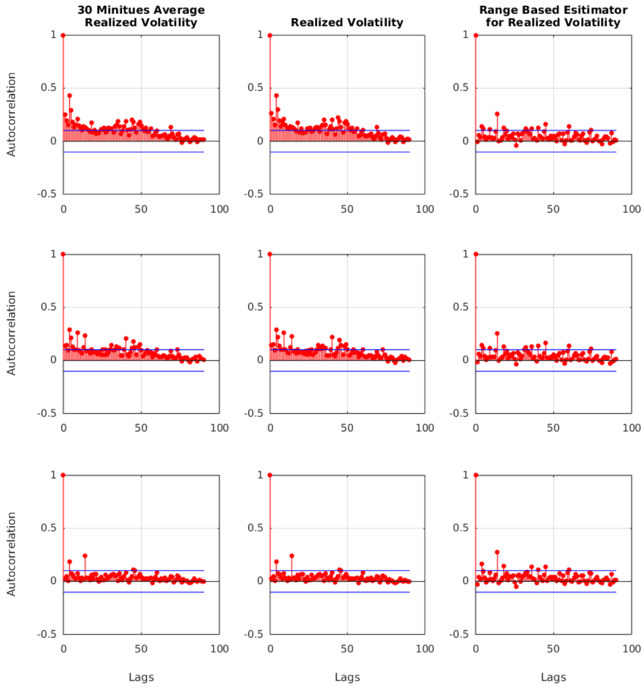


Figure 44: ACF for CHF/USD Log-Return Different RV Estimators

almost three months.

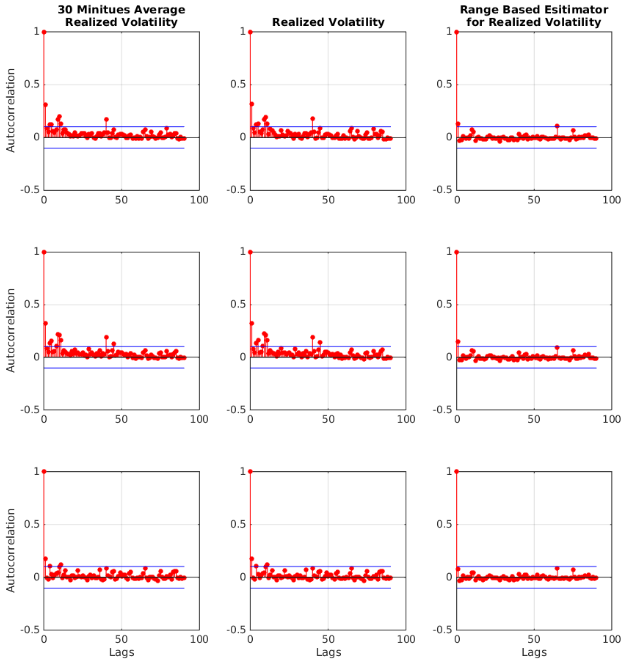


Figure 45: ACF for JPY/USD Log-Return Different RV Estimators

Across all currencies pairs there are several more observations, which are summarised in the following list.

The autocorrelation of average RV estimator and sparse RV estimator is almost identical while range based estimator has the lowest autocorrelation.

The autocorrelation of all estimator reduces as the sampling frequencies decrease. The implication is that we have more intra-day data then we can predict further into the future.

Autocorrelation drops significantly after one month, which might suggests that there is some calendar effect.

Between currencies pairs, while all of the three currencies pairs shows significant level of autocorrelation, their relative levels of autocorrelation are quite different. For example the autocovariance of GBP/USD is almost double of the autocorrelation of CHF/USD and JPY/USD.

iv The measures constructed so far are open-close measures of variance. Construct the corresponding 24-hour measures based on the 5-minute RV and the two different approaches for handling the overnight period discussed in the lecture.

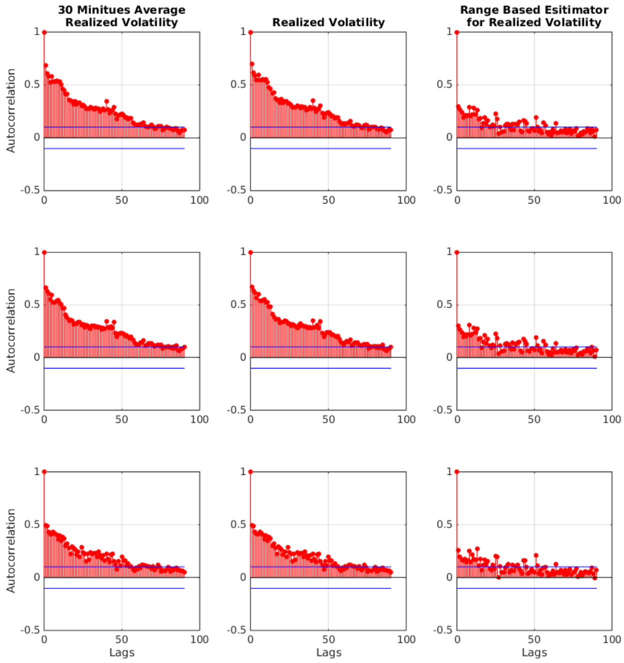
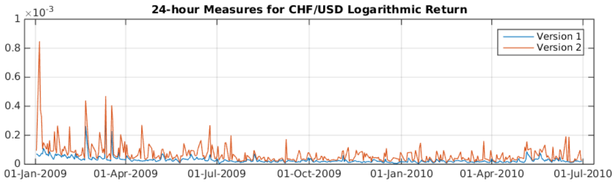
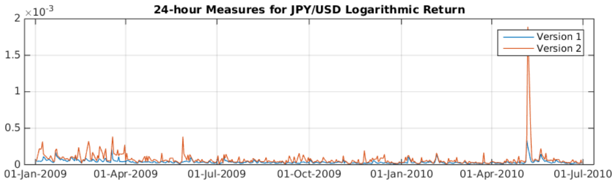
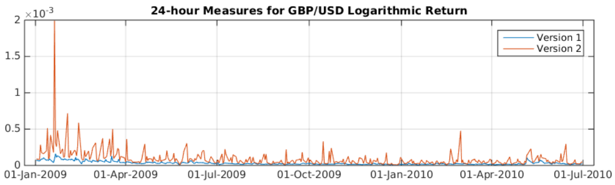


Figure 46: ACF for GBP/USD Log-Return Different RV Estimators

There are two methods used to construct the 24-hours RV estimators. The first one is by scaling and the second one is by adding.







Method 1 (Scaling),

$$RV_{t+1}^{24H} = \left(\frac{\Sigma_{t=1}^{T} R_t^2}{\Sigma_{t=1}^{T} RV_t^{Open}} \right) RV_{t+1}^{Open}$$

Method 2 (Adding)

$$RV_{t+1}^{24H} = \ln \left(S_{t+1}^{Open} / S_t^{Close} \right)^2 + RV_{t+1}^{Open}$$

In the book by Christoffersen [2] page 109, the optimal weights of the two terms which minimises the 24 hours variance give very little weight to the first term. Which suggests that the scaling approach might be a term method to adjust for overnight volatility.

6 Source Code

6.1 Problem 1

```
%% Question 1 of Quantitative Risk Management
% Housekeeping
clc;
clear all;

[num, txt ] = xlsread('daily.xlsx');

ftse_rtn = num(1:end,2);
ftse_rv = num(1:end,3);
dol_p_rtn = num(1:end,4);
dol_p_rv = num(1:end,5);
a = isnan(dol_p_rtn);                          % removing the NaNs
dol_p_rtn(a) = [];
dol_p_rv(a) = [];

%return square
ftse_sq_rtn=ftse_rtn.^2;
dol_p_sq_rtn = dol_p_rtn.^2;
% log realized vaiance
ftse_log_rv = log(ftse_rv);
dol_p_log_rv = log(dol_p_rv);
% r/RV^0.5
ftse_standardized = ftse_rtn./sqrt(ftse_rv);
dol_p_standardized = dol_p_rtn./sqrt(dol_p_rv);

% vector
ftse = [ftse_rtn,ftse_rv,ftse_sq_rtn,ftse_log_rv,ftse_standardized];
dol_p = [dol_p_rtn,dol_p_rv,dol_p_sq_rtn,dol_p_log_rv,dol_p_standardized];


%% i)
% the inputs for FTSE data are ftse_rtn, ftse_rv, ftse_log_rv,
% ftse_standardized
figure                                         % ACF Plots
subplot(2,2,1)
autocorr(dol_p_rtn)
title('ACF of Exchange rate return')

subplot(2,2,2)
autocorr(dol_p_rv)
title('ACF of Exchange realized variance')

subplot(2,2,3)
autocorr(dol_p_log_rv)
title('ACF of Exchange logarithm realized variance')

subplot(2,2,4)
autocorr(dol_p_standardized)
title('ACF of Exchange Standardized return')

figure                                         % PACF Plots
subplot(2,2,1)
parcorr(dol_p_rtn)
title('PACF of Exchange return')

subplot(2,2,2)
```

```
parcorr(dol_p_rv)
title('PACF of Exchange realized variance')

subplot(2,2,3)
parcorr(dol_p_rv)
title('PACF of Exchange logarithm realized variance')

subplot(2,2,4)
parcorr(dol_p_standardized)
title('PACF of Exchange Standardized return')

%% ii)
% Histogram
% distfit - normal
% distfit1- Gamma
nbins = 50 ;
names_ftse = {'FTSE Return' 'FTSE Realized Variance' ...
     'FTSE Return Square Return' 'FTSE logrv' ...
    'FTSE Standardized Return'};
for i = 1:5
distfit( ftse(:,i), nbins, names_ftse(i) )
end

names_dol_p = {'Dollar Pound Return' 'Dollar Pound Realized Variance'
    ...
    'Dollar Pound Square Return' 'Dollar Pound logrv' ...
    'Dollar Pound Standardized Return'};
for i = 1:5
distfit( dol_p(:,i), nbins, names_dol_p(i) )
end

names_ftse1={ 'FTSE Realized Variance'...
    'FTSE Return Square Return'};
for i = 1:2
distfit1( ftse(:,i+1), nbins,names_ftse1(i) );
end

names_dol_p1={'Dollar Pound Realized Variance' ...
    'Dollar Pound Square Return'};
for i = 1:2
distfit1( dol_p(:,i+1), nbins,names_dol_p1(i) );
end
% Descriptive Statistics
% Moments
ftse_mean = mean(ftse);
dol_p_mean = mean(dol_p);
ftse_var = var(ftse);
dol_p_var = var(dol_p);
ftse_skew = skewness(ftse);
dol_p_skew = skewness(dol_p);
ftse_kurt = kurtosis(ftse);
dol_p_kurt = kurtosis(dol_p);

% Testing for normality and serial correlation
for i = 1:5
ftse_jb_result(i) = jbtest(ftse(:,i));
end

for i = 1:5
dol_p_jb_result(i) = jbtest(dol_p(:,i));
end

```

```
for i = 1:5
ftse_lb_result(i) = lbqtest(ftse(:,i));
end

for i = 1:5
dol_p_lb_result(i) = lbqtest(dol_p(:,i));
end

%% iii)
%Housekeeping
old_date = num(:,1);

% old_date(a) = [];                %use when calculating dol_p_rtn
                                   %comment this line if you are calculating
                                   %ftse_rtn

% constructing weekly, monthly and quarterly returns
% please see the explanation in the reports
start_date= datenum('11-Jan-1999');
end_date = datenum('31-Dec-2007');
date = start_date:1:end_date;
date_vector = datevec(date);
new_date = 10000*date_vector(:,1)+100*date_vector(:,2)+date_vector(:,3);

[intersection,ia, ib] = intersect(old_date,new_date);   %ib index of overlapping
new_rtn =  zeros(length(new_date),1);

% return data
new_rtn(ib) = ftse_rtn;  %dol_p_rtn;

% for weekly
weekly_rtn = zeros(floor(length(new_rtn)/7),1);
for i=1:length(weekly_rtn)
    weekly_rtn(i) = sum(new_rtn(i*7-6:i*7));
end


% for monthly
month_indicator = date_vector(:,2);
monthly_rtn = count(month_indicator,new_rtn);
monthly_rtn = monthly_rtn';

%for quaterly
quater_indicator = zeros(length(month_indicator),1);
for i=1:length(month_indicator)
    if month_indicator(i)<=3
    quater_indicator(i) = 1;
    elseif month_indicator(i)>3 && month_indicator(i)<=6
    quater_indicator(i) = 2;
    elseif month_indicator(i)>6 && month_indicator(i)<=9
    quater_indicator(i) = 3;
    elseif month_indicator(i)>9 && month_indicator(i)<=12
    quater_indicator(i) = 4;

    end
end

quaterly_rtn = count(quater_indicator,new_rtn);
quaterly_rtn=quaterly_rtn';
```

```
% Desciptive Questions and normality testing
data_length = input('plz enter weekly, monthly or quaterly ', 's');

switch data_length
    case 'weekly'
        weekly_mean = mean(weekly_rtn);
        weekly_var = var(weekly_rtn);
        weekly_skewness = skewness(weekly_rtn);
        weekly_kurtosis = kurtosis(weekly_rtn);
        weekly_JB = jbtest(weekly_rtn);
        disp([weekly_mean,weekly_var,weekly_skewness,weekly_kurtosis,
            weekly_JB])
    case 'monthly'
        monthly_mean = mean(monthly_rtn);
        monthly_var = var(monthly_rtn);
        monthly_skewness = skewness(monthly_rtn);
        monthly_kurtosis = kurtosis(monthly_rtn);
        monthly_JB = jbtest(monthly_rtn);
        disp([monthly_mean,monthly_var,monthly_skewness,monthly_
            kurtosis,monthly_JB])
    case 'quaterly'
        quaterly_mean = mean(quaterly_rtn);
        quaterly_var = var(quaterly_rtn);
        quaterly_skewness = skewness(quaterly_rtn);
        quaterly_kurtosis = kurtosis(quaterly_rtn);
        quaterly_JB = jbtest(quaterly_rtn);
        disp([quaterly_mean,quaterly_var,quaterly_skewness,quaterly_
            kurtosis,quaterly_JB])

end
```

code/p1/RiskCoursework_Q1.m

```
function [ partial_sum ] = count( indicator,rtn )
%  A function to sum up monthly and quaterly data
%  indicator,  either month_indicator or quarter_indicator
%  rtn,    original returns
%  the way of summing up is to sum up the returns
%  which have the same group of indicator
sum1= rtn(1);
partial_sum = [];       %partial sum for the
                        %returns with the same indicator
j = 1 ;
for i=1:length(indicator)-1
    if(indicator(i)==indicator(i+1))
        sum1 = sum1 + rtn(i+1);
    else
        partial_sum(j) =  sum1 ;
        j = j+1;
        sum1 = rtn(i+1);
    end
end
partial_sum(j) =sum1;


end
```

code/p1/count.m

```
function [ h ] = distfit( data, nbins, name )
% DISTFIT  a function to plot histogram with fitted
```

```
% kernel density and normal density

figure
h1 = histfit(data,nbins,'kernel');
hold on
h2 = histfit(data,nbins,'normal');    % change to 't' for student t
set(h1(1),'FaceColor',[1 1 1]);
set(h2(1),'Visible','off');
set(h2(2),'Color','b');
legend([h1(2) h2(2)],'Kernel','Normal'); % 'Student t distribution'

title(name)

end
```

code/p1/distfit.m

```
function [ h ] = distfit1( data, nbins, name )
% DISTFIT  a function to plot histogram with fitted
% kernel density and gamma density
figure
h1 = histfit(data,nbins,'kernel');
hold on
h2 = histfit(data,nbins,'gamma');
set(h1(1),'FaceColor',[1 1 1]);
set(h2(1),'Visible','off');
set(h2(2),'Color','b');
legend([h1(2) h2(2)],'Kernel','Gamma');

title(name)

end
```

code/p1/distfit1.m

6.2 Problem 2

```
source('util.r')
source('graphic.r')

loadDaily()


require('xts')

FTSE_r_ts <- xts(ftse$FTSE_r,ftse$Dates)
GBP_r_ts <- xts(gbp$GBP_r,gbp$Dates)

if(!require('rugarch')) {
  install.packages('rugarch')
}

library('rugarch')

#### prob 2.1 =======


prob21 <- function(timeserie, filename) {
  spec = ugarchspec(variance.model=list(model="sGARCH", garchOrder=c
      (1,1)),
                    mean.model=list(armaOrder=c(0,0), include.mean=TRUE
                        ),
```

```
                    distribution.model="norm")
  fit=ugarchfit(spec=spec,data=timeserie)
  show(fit)

  print(dStats(as.numeric(residuals(fit)/sigma(fit)), filename))

  plot(fit,which=10)
  plot(fit,which=11)
  emprPlot(residuals(fit)/sigma(fit),'standardized residuals')
  plot(fit,which=9)
}

prob21(FTSE_r,'ftse')

prob21(GBP_r_ts,'gbp')


#### prob 2.2 =======

prob22.asym <- function(timeserie) {

  N = length(timeserie)

  writeLines(sprintf('\n =======  Correlation X^2(t) and X(t-1) =======\n\n %f',
                     cor((timeserie^2)[-1], timeserie[-N])))

  spec = ugarchspec(variance.model=list(model="sGARCH", garchOrder=c(1,1)),
                    mean.model=list(armaOrder=c(0,0), include.mean=TRUE),
                    distribution.model="norm")
  m=ugarchfit(spec=spec,data=timeserie)

  res_t <- residuals(m)[-1]
  res_t_1 <- residuals(m)[-N]

  sminus_boo = res_t_1 < 0
  splus_boo = !sminus_boo
  sminus = as.integer(sminus_boo)
  splus = as.integer(splus_boo)


  signBiasTest <- lm(res_t^2 ~ sminus)
  writeLines('\n============= Sign Bias test ============')
  print(summary(signBiasTest))

  writeLines(sprintf('\n============ Difference in means ============\n\n%f',
                     mean((res_t^2)[sminus_boo]) - mean((res_t^2)[splus_boo])))

  writeLines('\n============ Negative and Positive Sign Bias ============')
  negSignBiasTest <- lm(res_t^2 ~ sminus + I(sminus*res_t_1) + I(splus*res_t_1))
  print(summary(negSignBiasTest))

}

prob22.asym(FTSE_r_ts)
prob22.asym(GBP_r_ts)
```

```r

prob22.gjr <- function(timeserie,filename) {
  spec = ugarchspec(variance.model=list(model="gjrGARCH", garchOrder=c
      (1,1)),
                    mean.model=list(armaOrder=c(0,0), include.mean=TRUE
                        ),
                    distribution.model="norm")
  m=ugarchfit(spec=spec,data=timeserie)
  print(show(m))

  #png(paste('p22/',filename,'.png',sep=''))
  par(mfrow=c(1,1))
  #plot(m,which=11)
  #emprPlot(residuals(m)/sigma(m),'standardized residuals')
  plot(m,which=12)
  #invisible(dev.off())
}

prob22.gjr(FTSE_r,'ftse_gjr')
prob22.gjr(GBP_r,'gbp_gjr')

#### prob 2.3 =======

prob23.norm.roll <- function(timeserie,filename){
  N <- length(timeserie)
  spec = ugarchspec(variance.model=list(model="sGARCH", garchOrder=c
      (1,1)),
                    mean.model=list(armaOrder=c(0,0), include.mean=TRUE
                        ),
                    distribution.model="norm")

  roll = ugarchroll(spec, timeserie, n.start = N - 500, refit.every =
      1,
                    refit.window = "moving", solver = "hybrid", keep.
                        coef = TRUE)
  save(roll,file=paste('p23/roll_norm_',filename,'.Rda',sep=''))
}

prob23.std.roll <- function(timeserie,filename){
  N <- length(timeserie)
  spec = ugarchspec(variance.model=list(model="sGARCH", garchOrder=c
      (1,1)),
                    mean.model=list(armaOrder=c(0,0), include.mean=TRUE
                        ),
                    distribution.model="std")

  roll = ugarchroll(spec, timeserie, n.start = N - 500, refit.every =
      1,
                    refit.window = "moving", solver = "hybrid",
                        calculate.VaR = TRUE,
                    VaR.alpha = c(0.01, 0.05), keep.coef = TRUE)

  save(roll,file=paste('p23/roll_tdist_',filename,'.Rda',sep=''))
}

prob23.norm.roll(FTSE_r_ts,'ftse')
prob23.norm.roll(GBP_r_ts,'gbp')
prob23.std.roll(FTSE_r_ts * 100,'ftse')
prob23.std.roll(GBP_r_ts *100,'gbp')

prob23.norm.risk <- function(roll, filename) {
  #report(roll, type = "VaR", VaR.alpha = 0.05, conf.level = 0.95)
```

```
  #report(roll, type = "VaR", VaR.alpha = 0.01, conf.level = 0.95)
  par(mfrow=c(3,1))

  # plot(roll, which=4,VaR.alpha = 0.05)
  # plot(roll, which=4,VaR.alpha = 0.01)

  df <- as.data.frame(roll)
  df_ts <- xts(df[,c('Mu','Sigma','Realized')],as.Date(row.names(df)))
  ret <- df_ts$Realized
  # mu_L = -mu_r, sigma_L = sigma_r
  VaR95 <- -df_ts$Mu + df_ts$Sigma*qnorm(0.95)
  VaR99 <- -df_ts$Mu + df_ts$Sigma*qnorm(0.99)
  ES95 <- -df_ts$Mu + df_ts$Sigma*dnorm(qnorm(0.95))/.05
  ES99 <- -df_ts$Mu + df_ts$Sigma*dnorm(qnorm(0.99))/.01

  par(mfrow=c(2,1))
  plot(c(VaR95,ES95,-ret), type='n',main='95% level')
  points(-ret,bg='grey',pch=18,col='grey')
  lines(VaR95, col='red')
  lines(ES95, col='blue')
  legend('topleft',c('VaR','ES'), col=c('red','blue'),lty = 'solid',pch
      =NA, cex=0.5)

  plot(c(VaR99,ES99,-df_ts$Realized),type='n',main='99% level')
  points(-ret,bg='grey',pch=18,col='grey')
  lines(VaR99, col='red')
  lines(ES99, col='blue')
  legend('topleft',c('VaR','ES'), col=c('red','blue'),lty = 'solid',pch
      =NA, cex=0.5)

  count <- function(VaR,alpha) {
    exceeds <- sum(VaR < -ret)
    expected <- (1 - alpha)*500
    percent <- exceeds * 100 / 500
    data.frame(exceeds=exceeds,expected=expected,percent=percent)
  }

  print(count(VaR99, 0.99))
  print(count(VaR95, 0.95))
}

prob23.std.risk <- function(roll, filename) {
  report(roll, type = "VaR", VaR.alpha = 0.05, conf.level = 0.95)
  report(roll, type = "VaR", VaR.alpha = 0.01, conf.level = 0.95)

  df <- as.data.frame(roll)
  df[,c('Mu','Sigma','Realized')] <- df[,c('Mu','Sigma','Realized')] /
      100
  df_ts <- xts(df[,c('Mu','Sigma','Realized','Shape')],as.Date(row.
      names(df)))

  ret <- df_ts$Realized
  mu <- -df_ts$Mu
  sigma <- df_ts$Sigma
  nu <- df_ts$Shape

  quantile <- sqrt((nu-2)/nu)*qt(0.95, nu)
  density <- dt(qt(0.95, nu), nu)*sqrt(nu/(nu-2))
  VaR95 <- mu + sigma*quantile
  ES95 <- mu + sigma*(density/0.05)*((nu + quantile^2)/(nu - 1))

  quantile <- sqrt((nu-2)/nu)*qt(0.99, nu)
```

```
  density <- dt(qt(0.99, nu), nu)*sqrt(nu/(nu-2))
  VaR99 <- mu + sigma*quantile
  ES99 <- mu + sigma*(density/0.01)*((nu + quantile^2)/(nu - 1))

  par(mfrow=c(2,1))
  plot(c(VaR95,ES95,-ret), type='n',main='95% level')
  points(-ret,bg='grey',pch=18,col='grey')
  lines(VaR95, col='red')
  lines(ES95, col='blue')
  legend('topleft',c('VaR','ES'), col=c('red','blue'),lty = 'solid',pch
      =NA, cex=0.5)

  plot(c(VaR99,ES99,-df_ts$Realized),type='n',main='99% level')
  points(-ret,bg='grey',pch=18,col='grey')
  lines(VaR99, col='red')
  lines(ES99, col='blue')
  legend('topleft',c('VaR','ES'), col=c('red','blue'),lty = 'solid',pch
      =NA, cex=0.5)

  count <- function(VaR,alpha) {
    exceeds <- sum(VaR < -ret)
    expected <- (1 - alpha)*500
    percent <- exceeds * 100 / 500
    data.frame(exceeds=exceeds,expected=expected,percent=percent)
  }

  print(count(VaR99, 0.99))
  print(count(VaR95, 0.95))
}

prob23.std.risk(roll, 'ftse')


#### prob 2.4 =======
par(mfrow=c(1,1))
prob24 <- function(filename){
  load(paste('p23/roll_',filename,'.Rda',sep=''))

  png(paste('p24/param', filename, '.png',sep=''))
  plot(roll,which=5)
  invisible(dev.off())
}

prob24('norm_ftse')
prob24('norm_gbp')

#### prob 2.5 ==========
prob25 <- function(roll) {

  df <- as.data.frame(roll)
  df_ts <- xts(df[,c('Mu','Sigma','Realized')],as.Date(row.names(df)))

  # Loss = minus log return
  Loss = -df_ts$Real
  # mu_L = -mu_r, sigma_L = sigma_r
  VaR95 <- -df_ts$Mu + df_ts$Sigma*qnorm(0.95)
  hit <- as.numeric(Loss > VaR95)

  unc = unc.test(.05, hit, alpha.level=.05)
  cond  = ind.test( hit, alpha.level=.05)
  joint = cc.test(unc$LR, ind$LR, alpha.level=.05)

```

```
  ans = rbind(rbind(unc,ind),cc)
  print(ans)

}


#### prob 2.5 ==========
prob25.var95 <- function(roll) {

  df <- as.data.frame(roll)
  df_ts <- xts(df[,c('Mu','Sigma','Realized')],as.Date(row.names(df)))

  # Loss = minus log return
  Loss = -df_ts$Real
  # mu_L = -mu_r, sigma_L = sigma_r
  VaR <- -df_ts$Mu + df_ts$Sigma*qnorm(0.95)
  hit <- as.numeric(Loss > VaR)

  unc = unc.test(.05, hit, alpha.level=.05)
  cond  = ind.test( hit, alpha.level=.05)
  joint = cc.test(unc$LR, ind$LR, alpha.level=.05)

  ans = rbind(rbind(unc,ind),cc)
  print(ans)

}


prob25.var99 <- function(roll) {

  df <- as.data.frame(roll)
  df_ts <- xts(df[,c('Mu','Sigma','Realized')],as.Date(row.names(df)))

  # Loss = minus log return
  Loss = -df_ts$Real
  # mu_L = -mu_r, sigma_L = sigma_r
  VaR <- -df_ts$Mu + df_ts$Sigma*qnorm(0.99)
  hit <- as.numeric(Loss > VaR)

  unc = unc.test(.01, hit, alpha.level=.05)
  cond  = ind.test( hit, alpha.level=.05)
  joint = cc.test(unc$LR, ind$LR, alpha.level=.05)

  ans = rbind(rbind(unc,ind),cc)
  print(ans)

}

unc.test <- function(p, hit, alpha.level = 0.05) {
  T1 = sum(hit)
  T0 = length(hit) - T1
  est <- T1 / (T1 + T0)

  null.lh <- p ^ T1 * (1-p) ^ T0
  alt.lh <- est ^ T1 * (1 - est) ^ T0

  LR = -2 * log( null.lh / alt.lh )
  critical = qchisq( alpha.level, 1, lower.tail=FALSE)
  p.value = pchisq(LR, 1, lower.tail=FALSE)

  data.frame(test='unc', LR=LR,critical=critical,p.value=p.value)
}
```

```r

ind.test <- function(hit, alpha.level = 0.05) {

  hit.today <- hit[c(-length(hit))]
  hit.tomorrow <- hit[c(-1)]

  T00 <- sum(!hit.today & !hit.tomorrow)
  T01 <- sum(!hit.today & hit.tomorrow)
  T10 <- sum(hit.today & !hit.tomorrow)
  T11 <- sum(hit.today & hit.tomorrow)


  null.est <- (T01 + T11) / (T00 + T01 + T10 + T11)
  null.lh <- (1 - null.est)^(T00 + T10) * null.est^(T01 + T11)

  alt.est01 <- T01 / (T00 + T01)
  alt.est11 <- T11 / (T10 + T11)
  alt.lh <- (1-alt.est01)^T00 * alt.est01^T01 * (1-alt.est11)^T10 * alt
      .est11^T11

  LR = -2 * log( null.lh / alt.lh )
  critical = qchisq( alpha.level, 1, lower.tail=FALSE)
  p.value = pchisq(LR, 1, lower.tail=FALSE)

  data.frame(test='ind',LR=LR,critical=critical,p.value=p.value)
}

cc.test <- function(uc.LR, ind.LR, alpha.level) {
  LR = uc.LR + ind.LR
  critical = qchisq( alpha.level, 2, lower.tail=FALSE)
  p.value = pchisq(LR, 2, lower.tail=FALSE)

  data.frame(test='cc',LR=LR,critical=critical,p.value=p.value)
}
```

code/p2/report.r

```r

corrPlot <- function(timeserie) {
  timeserie.acf <- acf(timeserie,lag=100,main='')
  timeserie.pacf <- pacf(timeserie,lag=100,main='')
}

# Bandwith selection follow Scott (1992)
# See R-manual for option 'bw.nrd'
# https://stat.ethz.ch/R-manual/R-devel/library/stats/html/bandwidth.
    html

emprPlot <- function(timeserie,xlab) {
  dent <- density(timeserie,bw="nrd")
  maxDent <- max(dent$y)


  hist(timeserie, prob=TRUE, breaks=50, main='', xlab=xlab, ylim=c(0,
      maxDent*1.2))
  # empirical density
  lines(dent,col='red')
  # normal density
  xrange <- seq(par('usr')[1],par('usr')[2],len=100)
  lines(xrange,dnorm(xrange,mean=0,sd=1),col='blue')
}

boxTestPlot <- function(timeserie, minLag=1, maxLag=30) {
```

```
  pvalues <- sapply(minLag:maxLag,function(lag){Box.test(timeserie, lag
      =lag, type="Ljung")$p.value})
  plot(minLag:maxLag,pvalues,xlab='lag', ylab='p-values',type='b')
  abline(h=.05, col='red')
}
```

code/p2/graphic.r

```
if(!require('gdata')) {
  install.packages('gdata')
}

loadDaily <- function() {
  require(gdata)
  df = read.xls ("daily.xlsx", sheet = 1, header = TRUE,skip=1)
  head(df)
  names(df)[1] <- 'Dates'
  head(df)
  # http://www.statmethods.net/input/dates.html
  df$Dates <- as.Date(as.character(df$Dates),'%Y%m%d')

  ftse <<- df[,c("Dates","FTSE_r","FTSE_rv")]

  missingValue <- which(!complete.cases(df[c("GBP_r","GBP_rv")]))
  gbp <<- data.frame(df[-missingValue,c("Dates","GBP_r","GBP_rv")])

  attach(ftse)
  attach(gbp)
}

dStats <- function(timeserie, row.names) {
  require(moments)
  data.frame(
    mean=mean(timeserie),
    sd=sd(timeserie),
    skew=skewness(timeserie),
    kurt=kurtosis(timeserie),
    jb.pvalue=jarque.test(timeserie)$p.value,
    row.names=row.names
  )
}
```

code/p2/util.r

6.3 Problem 3

```
clc;
close all;
clear all;

tic

load('FTSE100RealizedVariance.mat');
load('FTSE100Returns.mat');
load('USDBritishPoundRealizedVa.mat');
load('USDBritishPoundReturns.mat');
load('Date.mat');

% Data cleaning
USDBritishPoundRealizedVa=USDBritishPoundRealizedVa(~isnan(
    USDBritishPoundRealizedVa));
```

```
USDBritishPoundReturns=USDBritishPoundReturns(~isnan(
    USDBritishPoundReturns));
USBdate=Date(~isnan(USDBritishPoundReturns));

FTSlength=length(FTSE100RealizedVariance);
UBlength=length(USDBritishPoundRealizedVa);

% unit root test
[FTSar1ADFSTAT,FTSar1ADFPVAL,FTSar1ADFCRITVAL]=augdf(
    FTSE100RealizedVariance,1,1);
[UBar1ADFSTAT,UBar1ADFPVAL,UBar1ADFCRITVAL]=augdf(
    USDBritishPoundRealizedVa,1,1);
[logFTSar1ADFSTAT,logFTSar1ADFPVAL,logFTSar1ADFCRITVAL]=augdf(log(
    FTSE100RealizedVariance),1,1);
[logUBar1ADFSTAT,logUBar1ADFPVAL,logUBar1ADFCRITVAL]=augdf(log(
    USDBritishPoundRealizedVa),1,1);

[FTSharADFSTAT,FTSharADFPVAL,FTSharADFCRITVAL]=augdf(
    FTSE100RealizedVariance,1,22);
[UBharADFSTAT,UBharADFPVAL,UBharADFCRITVAL]=augdf(
    USDBritishPoundRealizedVa,1,22);
[logFTSharADFSTAT,logFTSharADFPVAL,logFTSharADFCRITVAL]=augdf(log(
    FTSE100RealizedVariance),1,22);
[logUBharADFSTAT,logUBharADFPVAL,logUBharADFCRITVAL]=augdf(log(
    USDBritishPoundRealizedVa),1,22);

% prpblem 1
% AR(1) estimator
[FTSar1cof, FTSar1LL, FTSar1ERRORS, FTSar1SEREGRESSION,
    FTSar1DIAGNOSTICS, FTSar1VCVROBUST,FTSar1VCV, FTSar1LIKELIHOODS]=
    armaxfilter(FTSE100RealizedVariance,1,1);
[UBar1cof, UBar1LL, UBar1ERRORS, UBar1SEREGRESSION, UBar1DIAGNOSTICS,
    UBar1VCVROBUST,UBar1VCV, UBar1LIKELIHOODS]=armaxfilter(
    USDBritishPoundRealizedVa,1,1);
FTSar1coftvalue=abs(FTSar1cof)./sqrt(diag(FTSar1VCV));
                        %t value
UBar1coftvalue=abs(UBar1cof)./sqrt(diag(UBar1VCV));

% ARMA(1,1) estimator
[FTSarmacof, FTSarmaLL, FTSarmaERRORS, FTSarmaSEREGRESSION,
    FTSarmaDIAGNOSTICS, FTSarmaVCVROBUST,FTSarmaVCV, FTSarmaLIKELIHOODS
    ]=armaxfilter(FTSE100RealizedVariance,1,1,1);
[UBarmacof, UBarmaLL, UBarmaERRORS, UBarmaSEREGRESSION,
    UBarmaDIAGNOSTICS, UBarmaVCVROBUST,UBarmaVCV, UBarmaLIKELIHOODS]=
    armaxfilter(USDBritishPoundRealizedVa,1,1,1);
FTSarmacoftvalue=abs(FTSarmacof)./sqrt(diag(FTSarmaVCV));
                     %t value
UBarmacoftvalue=abs(UBarmacof)./sqrt(diag(UBarmaVCV));

% HAR estimator
[FTSharcof, FTSharERRORS, FTSharSEREGRESSION, FTSharDIAGNOSTICS,
    FTSharVCVROBUST, FTSharVCV]=heterogeneousar(FTSE100RealizedVariance
    ,1,[1; 5; 22]);
[UBharcof, UBharERRORS, UBharSEREGRESSION, UBharDIAGNOSTICS,
    UBharVCVROBUST, UBharVCV]=heterogeneousar(USDBritishPoundRealizedVa
    ,1,[1; 5; 22]);
FTSharcoftvalue=abs(FTSharcof)./sqrt(diag(FTSharVCV));
                        %t value
UBharcoftvalue=abs(UBharcof)./sqrt(diag(UBharVCV));

% log AR(1) estimator
[logFTSar1cof, logFTSar1LL, logFTSar1ERRORS, logFTSar1SEREGRESSION,
```

```
logFTSar1DIAGNOSTICS, logFTSar1VCVROBUST,logFTSar1VCV, logFTSar1LIKELIHOODS]=armaxfilter(log(FTSE100RealizedVariance),1,1);
[logUBar1cof, logUBar1LL, logUBar1ERRORS, logUBar1SEREGRESSION, logUBar1DIAGNOSTICS, logUBar1VCVROBUST,logUBar1VCV, logUBar1LIKELIHOODS]=armaxfilter(log(USDBritishPoundRealizedVa),1,1);
logFTSar1coftvalue=abs(logFTSar1cof)./sqrt(diag(logFTSar1VCV));                %t value
logUBar1coftvalue=abs(logUBar1cof)./sqrt(diag(logUBar1VCV));

% log ARMA(1,1) estimator
[logFTSarmacof, logFTSarmaLL, logFTSarmaERRORS, logFTSarmaSEREGRESSION, logFTSarmaDIAGNOSTICS, logFTSarmaVCVROBUST,logFTSarmaVCV, logFTSarmaLIKELIHOODS]=armaxfilter(log(FTSE100RealizedVariance),1,1,1);
[logUBarmacof, logUBarmaLL, logUBarmaERRORS, logUBarmaSEREGRESSION, logUBarmaDIAGNOSTICS, logUBarmaVCVROBUST,logUBarmaVCV, logUBarmaLIKELIHOODS]=armaxfilter(log(USDBritishPoundRealizedVa),1,1,1);
logFTSarmacoftvalue=abs(logFTSarmacof)./sqrt(diag(logFTSarmaVCV));            %t value
logUBarmacoftvalue=abs(logUBarmacof)./sqrt(diag(logUBarmaVCV));

% log HAR estimator
[logFTSharcof, logFTSharERRORS, logFTSharSEREGRESSION, logFTSharDIAGNOSTICS, logFTSharVCVROBUST, logFTSharVCV]=heterogeneousar(log(FTSE100RealizedVariance),1,[1; 5; 22]);
[logUBharcof, logUBharERRORS, logUBharSEREGRESSION, logUBharDIAGNOSTICS, logUBharVCVROBUST, logUBharVCV]=heterogeneousar(log(USDBritishPoundRealizedVa),1,[1; 5; 22]);
logFTSharcoftvalue=abs(logFTSharcof)./sqrt(diag(logFTSharVCV));                %t value
logUBharcoftvalue=abs(logUBharcof)./sqrt(diag(logUBharVCV));


% model diagose

% residuals lbq test
FTSar1lbqtest=lbqtest(FTSar1ERRORS);
UBar1lbqtest=lbqtest(UBar1ERRORS);
FTSarmalbqtest=lbqtest(FTSarmaERRORS);
UBarmalbqtest=lbqtest(UBarmaERRORS);
FTSharlbqtest=lbqtest(FTSharERRORS);
UBharlbqtest=lbqtest(UBharERRORS);
logFTSar1lbqtest=lbqtest(logFTSar1ERRORS);
logUBar1lbqtest=lbqtest(logUBar1ERRORS);
logFTSarmalbqtest=lbqtest(logFTSarmaERRORS);
logUBarmalbqtest=lbqtest(logUBarmaERRORS);
logFTSharlbqtest=lbqtest(logFTSharERRORS);
logUBharlbqtest=lbqtest(logUBharERRORS);

% residuals qqplot test
figure('Name','FTS model diagnotics qqplot');
subplot(2,3,1);
qqplot(FTSar1ERRORS);
legend('standard','FTS AR(1)');
subplot(2,3,4);
qqplot(logFTSar1ERRORS);
legend('standard','log FTS AR(1)');
subplot(2,3,2);
qqplot(FTSarmaERRORS);
```

```
legend('standard','FTS ARMA(1,1)');
subplot(2,3,5);
qqplot(logFTSarmaERRORS);
legend('standard','log FTS ARMA(1,1)');
subplot(2,3,3);
qqplot(FTSharERRORS);
legend('standard','FTS HAR');
subplot(2,3,6);
qqplot(logFTSharERRORS);
legend('standard','log FTS HAR');

figure('Name','USD/British Pound model diagnotics qqplot');
subplot(2,3,1);
qqplot(UBar1ERRORS);
legend('standard','USD/GBP AR(1)');
subplot(2,3,4);
qqplot(logUBar1ERRORS);
legend('standard','log USD/GBP AR(1)');
subplot(2,3,2);
qqplot(UBarmaERRORS);
legend('standard','USD/GBP ARMA(1,1)');
subplot(2,3,5);
qqplot(logUBarmaERRORS);
legend('standard','log USD/GBP ARMA(1,1)');
subplot(2,3,3);
qqplot(UBharERRORS);
legend('standard','USD/GBP HAR');
subplot(2,3,6);
qqplot(logUBharERRORS);
legend('standard','log USD/GBP HAR');


% prpblem 2
windowslength=500;

% ar1 moving windows estimation

% recursive version
FTSar1_recursitmulation=zeros(windowslength,1);
UBar1_recursitmulation=zeros(windowslength,1);

for i=1:windowslength
    FTSar1cof1=armaxfilter(FTSE100RealizedVariance(1:(FTSlength-windowslength+i-1),1),1,1);
    [b,~,~,~]=arma_forecaster(FTSE100RealizedVariance(1:(FTSlength-windowslength+i-1)),FTSar1cof1,1,1,[],FTSlength-windowslength+i-1,1);
    FTSar1_recursitmulation(i)=b(~isnan(b));
    UBar1cof1=armaxfilter(USDBritishPoundRealizedVa(1:(UBlength-windowslength+i-1),1),1,1);
    [b,~,~,~]=arma_forecaster(USDBritishPoundRealizedVa(1:(UBlength-windowslength+i-1)),UBar1cof1,1,1,[],UBlength-windowslength+i-1,1);
    UBar1_recursitmulation(i)=b(~isnan(b));
end

% moving windows version
FTSar1_movwstimulation=zeros(windowslength,1);
UBar1_movwsitmulation=zeros(windowslength,1);

for i=1:windowslength
    FTSar1cof1=armaxfilter(FTSE100RealizedVariance(i:(FTSlength-
```

```
        windowslength+i-1),1),1,1);
    [b,~,~,~]=arma_forecaster(FTSE100RealizedVariance(i:(FTSlength-
        windowslength+i-1)),[FTSar1cof1(1);FTSar1cof1(2)],1,1,[],
        FTSlength-windowslength,1);
    FTSar1_movwstimulation(i)=b(~isnan(b));
    UBar1cof1=armaxfilter(USDBritishPoundRealizedVa(i:(UBlength-
        windowslength+i-1),1),1,1);
    [b,~,~,~]=arma_forecaster(USDBritishPoundRealizedVa(i:(UBlength-
        windowslength+i-1)),[UBar1cof1(1);UBar1cof1(2)],1,1,[],UBlength
        -windowslength,1);
    UBar1_movwsitmulation(i)=b(~isnan(b));
end

% error test
FTSar1_recurerror=(FTSar1_recursitmulation(1:windowslength,1)-
    FTSE100RealizedVariance((FTSlength-windowslength+1):FTSlength));
meansquare_FTSar1_recurerror=FTSar1_recurerror'*FTSar1_recurerror/
    windowslength; %mean square error
abs_FTSar1_recurerror=sum(abs(FTSar1_recurerror))/windowslength;
                    %absolute square error
FTSar1_movwerror=(FTSar1_movwstimulation(1:windowslength,1)-
    FTSE100RealizedVariance((FTSlength-windowslength+1):FTSlength));
meansquare_FTSar1_movwerror=FTSar1_movwerror'*FTSar1_movwerror/
    windowslength;    %mean square error
abs_FTSar1_movwerror=sum(abs(FTSar1_movwerror))/windowslength;
                      %absolute square error


UBar1_recurerror=(UBar1_recursitmulation(1:windowslength,1)-
    USDBritishPoundRealizedVa((UBlength-windowslength+1):UBlength));
meansquare_UBar1_recurerror=UBar1_recurerror'*UBar1_recurerror/
    windowslength;    %mean square error
abs_UBar1_recurerror=sum(abs(UBar1_recurerror))/windowslength;
                      %absolute square error
UBar1_movwerror=(UBar1_movwsitmulation(1:windowslength,1)-
    USDBritishPoundRealizedVa((UBlength-windowslength+1):UBlength));
meansquare_UBar1_movwerror=UBar1_movwerror'*UBar1_movwerror/
    windowslength;       %mean square error
abs_UBar1_movwerror=sum(abs(UBar1_movwerror))/windowslength;
                        %absolute square error

FTSar1_recurerrorlbqtest2=lbqtest(FTSar1_recurerror);
                               %lbq test
FTSar1_movwerrorlbqtest2=lbqtest(FTSar1_movwerror);
UBar1_recurerrorlbqtest2=lbqtest(UBar1_recurerror);
UBar1_movwerrorlbqtest2=lbqtest(UBar1_movwerror);

% Mincer-Zarnowitz regression for AR(1)
[FTSar1_recur_B,FTSar1_recur_TSTAT,FTSar1_recur_S2,FTSar1_recur_VCV,
    FTSar1_recur_VCVWHITE,FTSar1_recur_R2]=ols(FTSE100RealizedVariance
    ((FTSlength-windowslength+1):FTSlength),FTSar1_recursitmulation(1:
    windowslength,1));
[FTSar1_movw_B,FTSar1_movw_TSTAT,FTSar1_movw_S2,FTSar1_movw_VCV,FTSar1_
    movw_VCVWHITE,FTSar1_movw_R2]=ols(FTSE100RealizedVariance((
    FTSlength-windowslength+1):FTSlength),FTSar1_movwstimulation(1:
    windowslength,1));
[UBar1_recur_B,UBar1_recur_TSTAT,UBar1_recur_S2,UBar1_recur_VCV,UBar1_
    recur_VCVWHITE,UBar1_recur_R2]=ols(USDBritishPoundRealizedVa((
    UBlength-windowslength+1):UBlength),UBar1_recursitmulation(1:
    windowslength,1));
[UBar1_movw_B,UBar1_movw_TSTAT,UBar1_movw_S2,UBar1_movw_VCV,UBar1_movw_
    VCVWHITE,UBar1_movw_R2]=ols(USDBritishPoundRealizedVa((UBlength-
```

```
    windowslength+1):UBlength),UBar1_movwsitmulation(1:windowslength,1)
    );


% arma moving windows estimation

% recursive version
FTSarma_recursitmulation=zeros(windowslength,1);
UBarma_recursitmulation=zeros(windowslength,1);

for i=1:windowslength
    FTSarmacof1=armaxfilter(FTSE100RealizedVariance(1:(FTSlength-
        windowslength+i-1),1),1,1,1);
    [b,~,~,~]=arma_forecaster(FTSE100RealizedVariance(1:(FTSlength-
        windowslength+i-1)),FTSarmacof1,1,1,1,FTSlength-windowslength+i
        -1,1);
    FTSarma_recursitmulation(i)=b(~isnan(b));
    UBarmacof1=armaxfilter(USDBritishPoundRealizedVa(1:(UBlength-
        windowslength+i-1),1),1,1,1);
    [b,~,~,~]=arma_forecaster(USDBritishPoundRealizedVa(1:(UBlength-
        windowslength+i-1)),UBarmacof1,1,1,1,UBlength-windowslength+i
        -1,1);
    UBarma_recursitmulation(i)=b(~isnan(b));
end

% moving windows version
FTSarma_movwstimulation=zeros(windowslength,1);
UBarma_movwsitmulation=zeros(windowslength,1);

for i=1:windowslength
    FTSarmacof1=armaxfilter(FTSE100RealizedVariance(i:(FTSlength-
        windowslength+i-1),1),1,1,1);
    [b,~,~,~]=arma_forecaster(FTSE100RealizedVariance(i:(FTSlength-
        windowslength+i-1)),FTSarmacof1,1,1,1,FTSlength-windowslength
        ,1);
    FTSarma_movwstimulation(i)=b(~isnan(b));
    UBarmacof1=armaxfilter(USDBritishPoundRealizedVa(i:(UBlength-
        windowslength+i-1),1),1,1,1);
    [b,~,~,~]=arma_forecaster(USDBritishPoundRealizedVa(i:(UBlength-
        windowslength+i-1)),UBarmacof1,1,1,1,UBlength-windowslength,1);
    UBarma_movwsitmulation(i)=b(~isnan(b));
end

% error test
FTSarma_recurerror=(FTSarma_recursitmulation(1:windowslength,1)-
    FTSE100RealizedVariance((FTSlength-windowslength+1):FTSlength));
meansquare_FTSarma_recurerror=FTSarma_recurerror'*FTSarma_recurerror/
    windowslength; %mean square error
abs_FTSarma_recurerror=sum(abs(FTSarma_recurerror))/windowslength;     %
    absolute square error
FTSarma_movwerror=(FTSarma_movwstimulation(1:windowslength,1)-
    FTSE100RealizedVariance((FTSlength-windowslength+1):FTSlength));
meansquare_FTSarma_movwerror=FTSarma_movwerror'*FTSarma_movwerror/
    windowslength;  %mean square error
abs_FTSarma_movwerror=sum(abs(FTSarma_movwerror))/windowslength;       %
    absolute square error

UBarma_recurerror=(UBarma_recursitmulation(1:windowslength,1)-
    USDBritishPoundRealizedVa((UBlength-windowslength+1):UBlength));
meansquare_UBarma_recurerror=UBarma_recurerror'*UBarma_recurerror/
    windowslength; %mean square error
abs_UBarma_recurerror=sum(abs(UBarma_recurerror))/windowslength;       %
```

```
absolute square error
UBarma_movwerror=(UBarma_movwsitmulation(1:windowslength,1)-USDBritishPoundRealizedVa((UBlength-windowslength+1):UBlength));
meansquare_UBarma_movwerror=UBarma_movwerror'*UBarma_movwerror/windowslength;     %mean square error
abs_UBarma_movwerror=sum(abs(UBarma_movwerror))/windowslength;        % absolute square error

FTSarma_recurerrorlbqtest2=lbqtest(FTSarma_recurerror);          % lbq test
FTSarma_movwerrorlbqtest2=lbqtest(FTSarma_movwerror);
UBarma_recurerrorlbqtest2=lbqtest(UBarma_recurerror);
UBarma_movwerrorlbqtest2=lbqtest(UBarma_movwerror);

% Mincer-Zarnowitz regression for ARMA(1,1) model
[FTSarma_recur_B,FTSarma_recur_TSTAT,FTSarma_recur_S2,FTSarma_recur_VCV,FTSarma_recur_VCVWHITE,FTSarma_recur_R2]=ols(FTSE100RealizedVariance((FTSlength-windowslength+1):FTSlength),FTSarma_recursitmulation(1:windowslength,1));
[FTSarma_movw_B,FTSarma_movw_TSTAT,FTSarma_movw_S2,FTSarma_movw_VCV,FTSarma_movw_VCVWHITE,FTSarma_movw_R2]=ols(FTSE100RealizedVariance((FTSlength-windowslength+1):FTSlength),FTSarma_movwstimulation(1:windowslength,1));
[UBarma_recur_B,UBarma_recur_TSTAT,UBarma_recur_S2,UBarma_recur_VCV,UBarma_recur_VCVWHITE,UBarma_recur_R2]=ols(USDBritishPoundRealizedVa((UBlength-windowslength+1):UBlength),UBarma_recursitmulation(1:windowslength,1));
[UBarma_movw_B,UBarma_movw_TSTAT,UBarma_movw_S2,UBarma_movw_VCV,UBarma_movw_VCVWHITE,UBarma_movw_R2]=ols(USDBritishPoundRealizedVa((UBlength-windowslength+1):UBlength),UBarma_movwsitmulation(1:windowslength,1));




% Har moving windows estimation

% recursive version
FTShar_recursitmulation=zeros(windowslength,1);
UBhar_recursitmulation=zeros(windowslength,1);

for i=1:windowslength
    FTSharcof1=heterogeneousar(FTSE100RealizedVariance(1:(FTSlength-windowslength+i-1),1),1,[1; 5; 22]);
    [b,~,~,~]=arma_forecaster(FTSE100RealizedVariance(1:(FTSlength-windowslength+i-1)),[FTSharcof1(1);(FTSharcof1(2)+0.2*FTSharcof1(3)+FTSharcof1(4)/22);(0.2*FTSharcof1(3)+FTSharcof1(4)/22)*ones(4,1);(FTSharcof1(4)/22)*ones(17,1)],1,1:22,[],FTSlength-windowslength+i-1,1);
    FTShar_recursitmulation(i)=b(~isnan(b));
    UBharcof1=heterogeneousar(USDBritishPoundRealizedVa(1:(UBlength-windowslength+i-1),1),1,[1; 5; 22]);
    [b,~,~,~]=arma_forecaster(USDBritishPoundRealizedVa(1:(UBlength-windowslength+i-1)),[UBharcof1(1);(UBharcof1(2)+0.2*UBharcof1(3)+UBharcof1(4)/22);(0.2*UBharcof1(3)+UBharcof1(4)/22)*ones(4,1);(UBharcof1(4)/22)*ones(17,1)],1,1:22,[],UBlength-windowslength+i-1,1);
    UBhar_recursitmulation(i)=b(~isnan(b));
end

% moving windows version
FTShar_movwstimulation=zeros(windowslength,1);
```

```
UBhar_movwsitmulation=zeros(windowslength,1);

for i=1:windowslength
    FTSharcof1=heterogeneousar(FTSE100RealizedVariance(i:(FTSlength-windowslength+i-1),1),1,[1; 5; 22]);
    [b,~,~,~]=arma_forecaster(FTSE100RealizedVariance(i:(FTSlength-windowslength+i-1)),[FTSharcof1(1);(FTSharcof1(2)+0.2*FTSharcof1(3)+FTSharcof1(4)/22);(0.2*FTSharcof1(3)+FTSharcof1(4)/22)*ones(4,1);(FTSharcof1(4)/22)*ones(17,1)],1,1:22,[],FTSlength-windowslength,1);
    FTShar_movwstimulation(i)=b(~isnan(b));
    UBharcof1=heterogeneousar(USDBritishPoundRealizedVa(i:(UBlength-windowslength+i-1),1),1,[1; 5; 22]);
    [b,~,~,~]=arma_forecaster(USDBritishPoundRealizedVa(i:(UBlength-windowslength+i-1)),[UBharcof1(1);(UBharcof1(2)+0.2*UBharcof1(3)+UBharcof1(4)/22);(0.2*UBharcof1(3)+UBharcof1(4)/22)*ones(4,1);(UBharcof1(4)/22)*ones(17,1)],1,1:22,[],UBlength-windowslength,1);
    UBhar_movwsitmulation(i)=b(~isnan(b));
end

% error test
FTShar_recurerror=(FTShar_recursitmulation(1:windowslength,1)-FTSE100RealizedVariance((FTSlength-windowslength+1):FTSlength));
meansquare_FTShar_recurerror=FTShar_recurerror'*FTShar_recurerror/windowslength;  %mean square error
abs_FTShar_recurerror=sum(abs(FTShar_recurerror))/windowslength;                     %absolute square error
FTShar_movwerror=(FTShar_movwstimulation(1:windowslength,1)-FTSE100RealizedVariance((FTSlength-windowslength+1):FTSlength));
meansquare_FTShar_movwerror=FTShar_movwerror'*FTShar_movwerror/windowslength;     %mean square error
abs_FTShar_movwerror=sum(abs(FTShar_movwerror))/windowslength;                       %absolute square error

UBhar_recurerror=(UBhar_recursitmulation(1:windowslength,1)-USDBritishPoundRealizedVa((UBlength-windowslength+1):UBlength));
meansquare_UBhar_recurerror=UBhar_recurerror'*UBhar_recurerror/windowslength;     %mean square error
abs_UBhar_recurerror=sum(abs(UBhar_recurerror))/windowslength;                       %absolute square error
UBhar_movwerror=(UBhar_movwsitmulation(1:windowslength,1)-USDBritishPoundRealizedVa((UBlength-windowslength+1):UBlength));
meansquare_UBhar_movwerror=UBhar_movwerror'*UBhar_movwerror/windowslength;        %mean square error
abs_UBhar_movwerror=sum(abs(UBhar_movwerror))/windowslength;                         %absolute square error

%lbq test
FTShar_recurerrorlbqtest2=lbqtest(FTShar_recurerror);
FTShar_movwerrorlbqtest2=lbqtest(FTShar_movwerror);
UBhar_recurerrorlbqtest2=lbqtest(UBhar_recurerror);
UBhar_movwerrorlbqtest2=lbqtest(UBhar_movwerror);

% Mincer-Zarnowitz regression for HAR model
[FTShar_recur_B,FTShar_recur_TSTAT,FTShar_recur_S2,FTShar_recur_VCV,FTShar_recur_VCVWHITE,FTShar_recur_R2]=ols(FTSE100RealizedVariance((FTSlength-windowslength+1):FTSlength),FTShar_recursitmulation(1:windowslength,1));
[FTShar_movw_B,FTShar_movw_TSTAT,FTShar_movw_S2,FTShar_movw_VCV,FTShar_movw_VCVWHITE,FTShar_movw_R2]=ols(FTSE100RealizedVariance((FTSlength-windowslength+1):FTSlength),FTShar_movwstimulation(1:
```

```
windowslength,1));
[UBhar_recur_B,UBhar_recur_TSTAT,UBhar_recur_S2,UBhar_recur_VCV,UBhar_recur_VCVWHITE,UBhar_recur_R2]=ols(USDBritishPoundRealizedVa((UBlength-windowslength+1):UBlength),UBhar_recursitmulation(1:windowslength,1));
[UBhar_movw_B,UBhar_movw_TSTAT,UBhar_movw_S2,UBhar_movw_VCV,UBhar_movw_VCVWHITE,UBhar_movw_R2]=ols(USDBritishPoundRealizedVa((UBlength-windowslength+1):UBlength),UBhar_movwsitmulation(1:windowslength,1));



% Garch

% directly citing the data from the problem 2
load('FTSGARCHstimulation.mat');
FTSGARCHstimulation=FTSGARCHstimulation/10000;
load('UBGARCHstimulation.mat');
UBGARCHstimulation=UBGARCHstimulation/10000;

FTSGARCH_error=FTSGARCHstimulation-FTSE100RealizedVariance((FTSlength-windowslength+1):FTSlength);
UBGARCH_error=UBGARCHstimulation-USDBritishPoundRealizedVa((UBlength-windowslength+1):UBlength);

meansquare_FTSGARCH_error=FTSGARCH_error'*FTSGARCH_error/windowslength;
abs_FTSGARCH_error=sum(abs(FTSGARCH_error))/windowslength;
meansquare_UBGARCH_error=UBGARCH_error'*UBGARCH_error/windowslength;
abs_UBGARCH_error=sum(abs(UBGARCH_error))/windowslength;

[FTSGARCH_B,FTSGARCH_TSTAT,FTSGARCH_S2,FTSGARCH_VCV,FTSGARCH_VCVWHITE,FTSGARCH_R2]=ols(FTSE100RealizedVariance((FTSlength-windowslength+1):FTSlength),FTSGARCHstimulation);
[UBGARCH_B,UBGARCH_TSTAT,UBGARCH_S2,UBGARCH_VCV,UBGARCH_VCVWHITE,UBGARCH_R2]=ols(USDBritishPoundRealizedVa((UBlength-windowslength+1):UBlength),UBGARCHstimulation);



% model compare
% forcast versus actural data
figure('Name','FTS 500 days rv forcast diagnotics recursive');
plot(FTSE100RealizedVariance((FTSlength-windowslength+1):FTSlength,1),'r');
hold on;
plot(FTSar1_recursitmulation,'b');
hold on;
plot(FTSarma_recursitmulation,'m');
hold on;
plot(FTShar_recursitmulation,'k');
hold on;
plot(FTSGARCHstimulation,'c');
hold on;
legend('original data','AR(1)','ARMA(1,1)','HAR','GARCH');

figure('Name','FTS 500 days rv forcast diagnotics moving windows');
plot(FTSE100RealizedVariance((FTSlength-windowslength+1):FTSlength,1),'r');
hold on;
plot(FTSar1_movwstimulation,'b');
hold on;
plot(FTSarma_movwstimulation,'m');
```

```
hold on;
plot(FTShar_movwstimulation,'k');
hold on;
plot(FTSGARCHstimulation,'c');
hold on;
legend('original data','AR(1)','ARMA(1,1)','HAR','GARCH');


figure('Name','USDBritishPound 500 days rv forcast diagnotics recursive');
plot(USDBritishPoundRealizedVa((UBlength-windowslength+1):UBlength,1),'r');
hold on;
plot(UBar1_recursitmulation,'b');
hold on;
plot(UBarma_recursitmulation,'m');
hold on;
plot(UBhar_recursitmulation,'k');
hold on;
plot(UBGARCHstimulation,'c');
hold on;
legend('original data','AR(1)','ARMA(1,1)','HAR','GARCH');

figure('Name','USDBritishPound 500 days rv forcast diagnotics moving windows');
plot(USDBritishPoundRealizedVa((UBlength-windowslength+1):UBlength,1),'r');
hold on;
plot(UBar1_movwsitmulation,'b');
hold on;
plot(UBarma_movwsitmulation,'m');
hold on;
plot(UBhar_movwsitmulation,'k');
hold on;
plot(UBGARCHstimulation,'c');
hold on;
legend('original data','AR(1)','ARMA(1,1)','HAR','GARCH');


% calculate the VaR and ES

q95=norminv(0.95);
q5=norminv(0.05);

FTSar1VAR95_recur=sqrt(FTSar1_recursitmulation)*q95;
FTSar1ES95_recur=sqrt(FTSar1_recursitmulation)*pdf('norm',q95,0,1)/0.05;
FTSar1VAR95_movw=sqrt(FTSar1_movwstimulation)*q95;
FTSar1ES95_movw=sqrt(FTSar1_movwstimulation)*pdf('norm',q95,0,1)/0.05;


FTSarmaVAR95_recur=sqrt(FTSarma_recursitmulation)*q95;
FTSarmaES95_recur=sqrt(FTSarma_recursitmulation)*pdf('norm',q95,0,1)/0.05;
FTSarmaVAR95_movw=sqrt(FTSarma_movwstimulation)*q95;
FTSarmaES95_movw=sqrt(FTSarma_movwstimulation)*pdf('norm',q95,0,1)/0.05;

FTSharVAR95_recur=sqrt(FTShar_recursitmulation)*q95;
FTSharES95_recur=sqrt(FTShar_recursitmulation)*pdf('norm',q95,0,1)/0.05;
FTSharVAR95_movw=sqrt(FTShar_movwstimulation)*q95;
```

```
FTSharES95_movw=sqrt(FTShar_movwstimulation)*pdf('norm',q95,0,1)/0.05;

UBar1VAR95_recur=sqrt(UBar1_recursitmulation)*q95;
UBar1ES95_recur=sqrt(UBar1_recursitmulation)*pdf('norm',q95,0,1)/0.05;
UBar1VAR95_movw=sqrt(UBar1_movwsitmulation)*q95;
UBar1ES95_movw=sqrt(UBar1_movwsitmulation)*pdf('norm',q95,0,1)/0.05;

UBarmaVAR95_recur=sqrt(UBarma_recursitmulation)*q95;
UBarmaES95_recur=sqrt(UBarma_recursitmulation)*pdf('norm',q95,0,1)/0.05;
UBarmaVAR95_movw=sqrt(UBarma_movwsitmulation)*q95;
UBarmaES95_movw=sqrt(UBarma_movwsitmulation)*pdf('norm',q95,0,1)/0.05;

UBharVAR95_recur=sqrt(UBhar_recursitmulation)*q95;
UBharES95_recur=sqrt(UBhar_recursitmulation)*pdf('norm',q95,0,1)/0.05;
UBharVAR95_movw=sqrt(UBhar_movwsitmulation)*q95;
UBharES95_movw=sqrt(UBhar_movwsitmulation)*pdf('norm',q95,0,1)/0.05;

FTSar1VAR5_recur=sqrt(FTSar1_recursitmulation)*q5;
FTSar1ES5_recur=-sqrt(FTSar1_recursitmulation)*pdf('norm',q5,0,1)/0.05;
FTSar1VAR5_movw=sqrt(FTSar1_movwstimulation)*q5;
FTSar1ES5_movw=-sqrt(FTSar1_movwstimulation)*pdf('norm',q5,0,1)/0.05;


FTSarmaVAR5_recur=sqrt(FTSarma_recursitmulation)*q5;
FTSarmaES5_recur=-sqrt(FTSarma_recursitmulation)*pdf('norm',q5,0,1)/0.05;
FTSarmaVAR5_movw=sqrt(FTSarma_movwstimulation)*q5;
FTSarmaES5_movw=-sqrt(FTSarma_movwstimulation)*pdf('norm',q5,0,1)/0.05;

FTSharVAR5_recur=sqrt(FTShar_recursitmulation)*q5;
FTSharES5_recur=-sqrt(FTShar_recursitmulation)*pdf('norm',q5,0,1)/0.05;
FTSharVAR5_movw=sqrt(FTShar_movwstimulation)*q5;
FTSharES5_movw=-sqrt(FTShar_movwstimulation)*pdf('norm',q5,0,1)/0.05;

UBar1VAR5_recur=sqrt(UBar1_recursitmulation)*q5;
UBar1ES5_recur=-sqrt(UBar1_recursitmulation)*pdf('norm',q5,0,1)/0.05;
UBar1VAR5_movw=sqrt(UBar1_movwsitmulation)*q5;
UBar1ES5_movw=-sqrt(UBar1_movwsitmulation)*pdf('norm',q5,0,1)/0.05;

UBarmaVAR5_recur=sqrt(UBarma_recursitmulation)*q5;
UBarmaES5_recur=-sqrt(UBarma_recursitmulation)*pdf('norm',q5,0,1)/0.05;
UBarmaVAR5_movw=sqrt(UBarma_movwsitmulation)*q5;
UBarmaES5_movw=-sqrt(UBarma_movwsitmulation)*pdf('norm',q5,0,1)/0.05;

UBharVAR5_recur=sqrt(UBhar_recursitmulation)*q5;
UBharES5_recur=-sqrt(UBhar_recursitmulation)*pdf('norm',q5,0,1)/0.05;
UBharVAR5_movw=sqrt(UBhar_movwsitmulation)*q5;
UBharES5_movw=-sqrt(UBhar_movwsitmulation)*pdf('norm',q5,0,1)/0.05;

% VaR analysis
FTSar1VAR95_across=length(find((FTSar1VAR95_movw-FTSE100Returns((FTSlength-windowslength+1):FTSlength,1))<0));
FTSarmaVAR95_across=length(find((FTSarmaVAR95_movw-FTSE100Returns((FTSlength-windowslength+1):FTSlength,1))<0));
FTSharVAR95_across=length(find((FTSharVAR95_movw-FTSE100Returns((FTSlength-windowslength+1):FTSlength,1))<0));
UBar1VAR95_across=length(find((UBar1VAR95_movw-USDBritishPoundReturns((UBlength-windowslength+1):UBlength,1))<0));
UBarmaVAR95_across=length(find((UBarmaVAR95_movw-USDBritishPoundReturns((UBlength-windowslength+1):UBlength,1))<0));
UBharVAR95_across=length(find((UBharVAR95_movw-USDBritishPoundReturns((
```

```
    UBlength-windowslength+1):UBlength,1))<0));

%VaR and ES versus return data plot

figure('Name','FTS 500 days VAR95 recursive');
plot(FTSar1VAR95_recur,'b');
hold on;
plot(FTSarmaVAR95_recur,'m');
hold on;
plot(FTSharVAR95_recur,'k');
hold on;
plot(FTSE100Returns((FTSlength-windowslength+1):FTSlength,1),'r');
hold on;
plot(FTSar1VAR5_recur,'b');
hold on;
plot(FTSarmaVAR5_recur,'m');
hold on;
plot(FTSharVAR5_recur,'k');
hold on;
legend('AR(1)','ARMA(1,1)','HAR','original return');

figure('Name','FTS 500 days VAR95 moving windows');
plot(FTSar1VAR95_movw,'b');
hold on;
plot(FTSarmaVAR95_movw,'m');
hold on;
plot(FTSharVAR95_movw,'k');
hold on;
plot(FTSE100Returns((FTSlength-windowslength+1):FTSlength,1),'r');
hold on;
plot(FTSar1VAR5_movw,'b');
hold on;
plot(FTSarmaVAR5_movw,'m');
hold on;
plot(FTSharVAR5_movw,'k');
hold on;
legend('AR(1)','ARMA(1,1)','HAR','original return');

figure('Name','USDBritishPound 500 days VAR95 recursive');
plot(UBar1VAR95_recur,'b');
hold on;
plot(UBarmaVAR95_recur,'m');
hold on;
plot(UBharVAR95_recur,'k');
hold on;
plot(USDBritishPoundReturns((UBlength-windowslength+1):UBlength,1),'r')
    ;
hold on;
plot(UBar1VAR5_recur,'b');
hold on;
plot(UBarmaVAR5_recur,'m');
hold on;
plot(UBharVAR5_recur,'k');
hold on;
legend('AR(1)','ARMA(1,1)','HAR','original return');

figure('Name','USDBritishPound 500 days VAR95 moving windows');
plot(UBar1VAR95_movw,'b');
hold on;
plot(UBarmaVAR95_movw,'m');
hold on;
plot(UBharVAR95_movw,'k');
```

```
hold on;
plot(USDBritishPoundReturns((UBlength-windowslength+1):UBlength,1),'r')
    ;
hold on;
plot(UBar1VAR5_movw,'b');
hold on;
plot(UBarmaVAR5_movw,'m');
hold on;
plot(UBharVAR5_movw,'k');
hold on;
legend('AR(1)','ARMA(1,1)','HAR','original return');

figure('Name','FTS 500 days ES95 recursive');
plot(FTSar1ES95_recur,'b');
hold on;
plot(FTSarmaES95_recur,'m');
hold on;
plot(FTSharES95_recur,'k');
hold on;
plot(FTSE100Returns((FTSlength-windowslength+1):FTSlength,1),'r');
hold on;
plot(FTSar1ES5_recur,'b');
hold on;
plot(FTSarmaES5_recur,'m');
hold on;
plot(FTSharES5_recur,'k');
hold on;
legend('AR(1)','ARMA(1,1)','HAR','original return');

figure('Name','FTS 500 days ES95 moving windows');
plot(FTSar1ES95_movw,'b');
hold on;
plot(FTSarmaES95_movw,'m');
hold on;
plot(FTSharES95_movw,'k');
hold on;
plot(FTSE100Returns((FTSlength-windowslength+1):FTSlength,1),'r');
hold on;
plot(FTSar1ES5_movw,'b');
hold on;
plot(FTSarmaES5_movw,'m');
hold on;
plot(FTSharES5_movw,'k');
hold on;
legend('AR(1)','ARMA(1,1)','HAR','original return');

figure('Name','USDBritishPound 500 days ES95 recursive');
plot(UBar1ES95_recur,'b');
hold on;
plot(UBarmaES95_recur,'m');
hold on;
plot(UBharES95_recur,'k');
hold on;
plot(USDBritishPoundReturns((UBlength-windowslength+1):UBlength,1),'r')
    ;
hold on;
plot(UBar1ES5_recur,'b');
hold on;
plot(UBarmaES5_recur,'m');
hold on;
plot(UBharES5_recur,'k');
hold on;
```

```
legend('AR(1)','ARMA(1,1)','HAR','original return');

figure('Name','USDBritishPound 500 days ES95 moving windows');
plot(UBar1ES95_movw,'b');
hold on;
plot(UBarmaES95_movw,'m');
hold on;
plot(UBharES95_movw,'k');
hold on;
plot(USDBritishPoundReturns((UBlength-windowslength+1):UBlength,1),'r')
    ;
hold on;
plot(UBar1ES5_movw,'b');
hold on;
plot(UBarmaES5_movw,'m');
hold on;
plot(UBharES5_movw,'k');
hold on;
legend('AR(1)','ARMA(1,1)','HAR','original return');

q99=norminv(0.99);
q1=norminv(0.01);
FTSar1VAR99_recur=sqrt(FTSar1_recursitmulation)*q99;
FTSar1ES99_recur=sqrt(FTSar1_recursitmulation)*pdf('norm',q99,0,1)/
    0.01;
FTSar1VAR99_movw=sqrt(FTSar1_movwstimulation)*q99;
FTSar1ES99_movw=sqrt(FTSar1_movwstimulation)*pdf('norm',q99,0,1)/0.01;

FTSarmaVAR99_recur=sqrt(FTSarma_recursitmulation)*q99;
FTSarmaES99_recur=sqrt(FTSarma_recursitmulation)*pdf('norm',q99,0,1)/
    0.01;
FTSarmaVAR99_movw=sqrt(FTSarma_movwstimulation)*q99;
FTSarmaES99_movw=sqrt(FTSarma_movwstimulation)*pdf('norm',q99,0,1)/
    0.01;

FTSharVAR99_recur=sqrt(FTShar_recursitmulation)*q99;
FTSharES99_recur=sqrt(FTShar_recursitmulation)*pdf('norm',q99,0,1)/
    0.01;
FTSharVAR99_movw=sqrt(FTShar_movwstimulation)*q99;
FTSharES99_movw=sqrt(FTShar_movwstimulation)*pdf('norm',q99,0,1)/0.01;

UBar1VAR99_recur=sqrt(UBar1_recursitmulation)*q99;
UBar1ES99_recur=sqrt(UBar1_recursitmulation)*pdf('norm',q99,0,1)/0.01;
UBar1VAR99_movw=sqrt(UBar1_movwsitmulation)*q99;
UBar1ES99_movw=sqrt(UBar1_movwsitmulation)*pdf('norm',q99,0,1)/0.01;

UBarmaVAR99_recur=sqrt(UBarma_recursitmulation)*q99;
UBarmaES99_recur=sqrt(UBarma_recursitmulation)*pdf('norm',q99,0,1)/
    0.01;
UBarmaVAR99_movw=sqrt(UBarma_movwsitmulation)*q99;
UBarmaES99_movw=sqrt(UBarma_movwsitmulation)*pdf('norm',q99,0,1)/0.01;

UBharVAR99_recur=sqrt(UBhar_recursitmulation)*q99;
UBharES99_recur=sqrt(UBhar_recursitmulation)*pdf('norm',q99,0,1)/0.01;
UBharVAR99_movw=sqrt(UBhar_movwsitmulation)*q99;
UBharES99_movw=sqrt(UBhar_movwsitmulation)*pdf('norm',q99,0,1)/0.01;

FTSar1VAR1_recur=sqrt(FTSar1_recursitmulation)*q1;
FTSar1ES1_recur=-sqrt(FTSar1_recursitmulation)*pdf('norm',q1,0,1)/0.01;
FTSar1VAR1_movw=sqrt(FTSar1_movwstimulation)*q1;
FTSar1ES1_movw=-sqrt(FTSar1_movwstimulation)*pdf('norm',q1,0,1)/0.01;
```

```
FTSarmaVAR1_recur=sqrt(FTSarma_recursitmulation)*q1;
FTSarmaES1_recur=-sqrt(FTSarma_recursitmulation)*pdf('norm',q1,0,1)/
    0.01;
FTSarmaVAR1_movw=sqrt(FTSarma_movwstimulation)*q1;
FTSarmaES1_movw=-sqrt(FTSarma_movwstimulation)*pdf('norm',q1,0,1)/0.01;

FTSharVAR1_recur=sqrt(FTShar_recursitmulation)*q1;
FTSharES1_recur=-sqrt(FTShar_recursitmulation)*pdf('norm',q1,0,1)/0.01;
FTSharVAR1_movw=sqrt(FTShar_movwstimulation)*q1;
FTSharES1_movw=-sqrt(FTShar_movwstimulation)*pdf('norm',q1,0,1)/0.01;

UBar1VAR1_recur=sqrt(UBar1_recursitmulation)*q1;
UBar1ES1_recur=-sqrt(UBar1_recursitmulation)*pdf('norm',q1,0,1)/0.01;
UBar1VAR1_movw=sqrt(UBar1_movwsitmulation)*q1;
UBar1ES1_movw=-sqrt(UBar1_movwsitmulation)*pdf('norm',q1,0,1)/0.01;

UBarmaVAR1_recur=sqrt(UBarma_recursitmulation)*q1;
UBarmaES1_recur=-sqrt(UBarma_recursitmulation)*pdf('norm',q1,0,1)/0.01;
UBarmaVAR1_movw=sqrt(UBarma_movwsitmulation)*q1;
UBarmaES1_movw=-sqrt(UBarma_movwsitmulation)*pdf('norm',q1,0,1)/0.01;

UBharVAR1_recur=sqrt(UBhar_recursitmulation)*q1;
UBharES1_recur=-sqrt(UBhar_recursitmulation)*pdf('norm',q1,0,1)/0.01;
UBharVAR1_movw=sqrt(UBhar_movwsitmulation)*q1;
UBharES1_movw=-sqrt(UBhar_movwsitmulation)*pdf('norm',q1,0,1)/0.01;

FTSar1VAR99_across=length(find((FTSar1VAR99_movw-FTSE100Returns((
    FTSlength-windowslength+1):FTSlength,1))<0));
FTSarmaVAR99_across=length(find((FTSarmaVAR99_movw-FTSE100Returns((
    FTSlength-windowslength+1):FTSlength,1))<0));
FTSharVAR99_across=length(find((FTSharVAR99_movw-FTSE100Returns((
    FTSlength-windowslength+1):FTSlength,1))<0));
UBar1VAR99_across=length(find((UBar1VAR99_movw-USDBritishPoundReturns((
    UBlength-windowslength+1):UBlength,1))<0));
UBarmaVAR99_across=length(find((UBarmaVAR99_movw-USDBritishPoundReturns
    ((UBlength-windowslength+1):UBlength,1))<0));
UBharVAR99_across=length(find((UBharVAR99_movw-USDBritishPoundReturns((
    UBlength-windowslength+1):UBlength,1))<0));

figure('Name','FTS 500 days VAR99 recursive');
plot(FTSar1VAR99_recur,'b');
hold on;
plot(FTSarmaVAR99_recur,'m');
hold on;
plot(FTSharVAR99_recur,'k');
hold on;
plot(FTSE100Returns((FTSlength-windowslength+1):FTSlength,1),'r');
hold on;
plot(FTSar1VAR1_recur,'b');
hold on;
plot(FTSarmaVAR1_recur,'m');
hold on;
plot(FTSharVAR1_recur,'k');
hold on;
legend('AR(1)','ARMA(1,1)','HAR','original return');

figure('Name','FTS 500 days VAR99 moving windows');
plot(FTSar1VAR99_movw,'b');
hold on;
plot(FTSarmaVAR99_movw,'m');
hold on;
plot(FTSharVAR99_movw,'k');
```

```
hold on;
plot(FTSE100Returns((FTSlength-windowslength+1):FTSlength,1),'r');
hold on;
plot(FTSar1VAR1_movw,'b');
hold on;
plot(FTSarmaVAR1_movw,'m');
hold on;
plot(FTSharVAR1_movw,'k');
hold on;
legend('AR(1)','ARMA(1,1)','HAR','original return');

figure('Name','USDBritishPound 500 days VAR99 recursive');
plot(UBar1VAR99_recur,'b');
hold on;
plot(UBarmaVAR99_recur,'m');
hold on;
plot(UBharVAR99_recur,'k');
hold on;
plot(USDBritishPoundReturns((UBlength-windowslength+1):UBlength,1),'r')
    ;
hold on;
plot(UBar1VAR1_recur,'b');
hold on;
plot(UBarmaVAR1_recur,'m');
hold on;
plot(UBharVAR1_recur,'k');
hold on;
legend('AR(1)','ARMA(1,1)','HAR','original return');

figure('Name','USDBritishPound 500 days VAR99 moving windows');
plot(UBar1VAR99_movw,'b');
hold on;
plot(UBarmaVAR99_movw,'m');
hold on;
plot(UBharVAR99_movw,'k');
hold on;
plot(USDBritishPoundReturns((UBlength-windowslength+1):UBlength,1),'r')
    ;
hold on;
plot(UBar1VAR1_movw,'b');
hold on;
plot(UBarmaVAR1_movw,'m');
hold on;
plot(UBharVAR1_movw,'k');
legend('AR(1)','ARMA(1,1)','HAR','original return');

figure('Name','FTS 500 days ES99 recursive');
plot(FTSar1ES99_recur,'b');
hold on;
plot(FTSarmaES99_recur,'m');
hold on;
plot(FTSharES99_recur,'k');
hold on;
plot(FTSE100Returns((FTSlength-windowslength+1):FTSlength,1),'r');
hold on;
plot(FTSar1ES1_recur,'b');
hold on;
plot(FTSarmaES1_recur,'m');
hold on;
plot(FTSharES1_recur,'k');
legend('AR(1)','ARMA(1,1)','HAR','original return');

```

```
figure('Name','FTS 500 days ES99 moving windows');
plot(FTSar1ES99_movw,'b');
hold on;
plot(FTSarmaES99_movw,'m');
hold on;
plot(FTSharES99_movw,'k');
hold on;
plot(FTSE100Returns((FTSlength-windowslength+1):FTSlength,1),'r');
hold on;
plot(FTSar1ES1_movw,'b');
hold on;
plot(FTSarmaES1_movw,'m');
hold on;
plot(FTSharES1_movw,'k');
legend('AR(1)','ARMA(1,1)','HAR','original return');

figure('Name','USDBritishPound 500 days ES99 recursive');
plot(UBar1ES99_recur,'b');
hold on;
plot(UBarmaES99_recur,'m');
hold on;
plot(UBharES99_recur,'k');
hold on;
plot(USDBritishPoundReturns((UBlength-windowslength+1):UBlength,1),'r')
    ;
hold on;
plot(UBar1ES1_recur,'b');
hold on;
plot(UBarmaES1_recur,'m');
hold on;
plot(UBharES1_recur,'k');
legend('AR(1)','ARMA(1,1)','HAR','original return');

figure('Name','USDBritishPound 500 days ES99 moving windows');
plot(UBar1ES99_movw,'b');
hold on;
plot(UBarmaES99_movw,'m');
hold on;
plot(UBharES99_movw,'k');
hold on;
plot(USDBritishPoundReturns((UBlength-windowslength+1):UBlength,1),'r')
    ;
hold on;
plot(UBar1ES1_movw,'b');
hold on;
plot(UBarmaES1_movw,'m');
hold on;
plot(UBharES1_movw,'k');
legend('AR(1)','ARMA(1,1)','HAR','original return');


toc;
```

code/p3/Problem3.m

6.4 Problem 4

```
%% Q4
clc;clear;


```

```
% Calcuates VaR and ES for different number of sample paths

n_trials = 50000:50000:2000000;
result = []
for i=n_trials
    disp(i);
    result = [result ;get_var_es(i)];
end


% Check covergence

subplot(411);loglog(n_trials,result(:,1))
grid on;title('VaR 95%')
subplot(412);loglog(n_trials,result(:,2))
grid on;title('VaR 99%')
subplot(413);loglog(n_trials,result(:,3))
grid on;title('ES 95%')
subplot(414);loglog(n_trials,result(:,4))
grid on;title('ES 99%')

set(gcf,'PaperUnits', 'inch', 'PaperPosition', [0.25 0 10 10]);
print(gcf,'-dpng','-r100',sprintf('covgent_plt.png',code{:}));
```

code/p4/q4.m

```
function [ output ] = get_var_es( N )
% Get VaR and ES for a given number of sample paths


% Parametes

rtn_T = -0.03;
sigma_s_T = 0.8;
eps_T = -0.1;

c = 0.1;
rho = -0.1;
gamma = 0.5;
alpha_0 = 0.02;
alpha_1 = 0.08;
beta_1 = 0.9;
nu = 5;
n_days = 5;


% Path storages variables

rtn_ts = [rtn_T*ones(N,1) zeros(N,n_days)];
sigma_s_ts = [sigma_s_T*ones(N,1) zeros(N,n_days)];
eps_ts = [eps_T*ones(N,1) zeros(N,n_days)];


% Main iteration Loop (5 days ahead), Used stdrand from MFE toolbox

for d = 1:n_days
    sigma_s_ts(:,d+1) = alpha_0+alpha_1*(eps_ts(:,d).^2)+beta_1*sigma_s
        _ts(:,d);
    eps_ts(:,d+1) = sqrt(sigma_s_ts(:,d+1)) .* stdtrnd(nu,N,1);
    rtn_ts(:,d+1) =  c + rho*rtn_ts(:,d) + gamma*sqrt(sigma_s_ts(:,d+1)
        )+eps_ts(:,d+1);
end

```

```

% Calculate VaR and ES

rm = sum(rtn_ts(:,2:(n_days+1)),2);
rm = sort(rm);
output = [rm(N*0.05),rm(N*0.01),mean(rm(1:N*0.05)),mean(rm(1:N*0.01))];
end
```

code/p4/get_var_es.m

6.5 Problem 5

```
%% Q5

%% Clean data
clc;clear;
codes = {'CHF','JPY','GBP'};
for k = 1:numel(codes)

    % Reading data and prase timestamp

    code = codes(k);
    data = importdata(strcat(code{:},'.csv'));
    date = datenum(data.textdata(1:end,1),'mm/dd/yyyy');
    time = datenum(data.textdata(1:end,2),'HH:MM:SS');
    % Add and reset time origin
    date_time = date+time-datenum('0:0','HH:MM');
    data = data.data;


    % Construct regular time grid for each trading dates

    start_date = date_time(1);
    trading_dates = unique(date);
    % Absolute time grid in seconds
    date_time_grid_abs_sec = get_time_grid(1/24/60/60,trading_dates);
    % Relative time grid in seconds (to start date)
    date_time_grid_sec = get_time_grid(1/24/60/60,trading_dates)-start_
        date;


    % Joining

    shift_mat = 10.^[10 8 6 4 2 0]; % Convert datenum to integer keys
    date_time_cmp = datevec(date_time_grid_abs_sec)*shift_mat';
    tran_time_cmp = datevec(date_time)*shift_mat';

    [~,ia,ib]=intersect(date_time_cmp,tran_time_cmp); % Joining
    ts_sec_data = NaN*ones(length(date_time_grid_abs_sec),1);
    ts_sec_data(ia,1) = data(ib);


    % Fill the gaps in the time series

    n_days = length(unique(date));
    ts_sec_data = reshape(ts_sec_data,length(ts_sec_data)/n_days,n_days
        );
    for i=1:n_days
        trading_time = (1:size(ts_sec_data,1))';
        nan_idx =  isnan(ts_sec_data(:,i));
        % 2014a (Uses downloaded function zoh for last tick filling)
```

```
        ts_sec_data(nan_idx,i) =  zoh(trading_time(~nan_idx),...
                                      ts_sec_data(~nan_idx,i),...
                                      trading_time(nan_idx));

         first_nan_idx = find(isnan(ts_sec_data(:,i))==0,1,'last')+1;
         ts_sec_data(first_nan_idx:end,i) = ts_sec_data(first_nan_idx-1,i);

         last_nan_idx = find(isnan(ts_sec_data(:,i))==0,1)-1;
         ts_sec_data(1:last_nan_idx,i) = ts_sec_data(last_nan_idx+1,i);
%          % Revert to last tick intra-day
%          if i>1
%             ts_sec_data(1:last_nan_idx,i) = ts_sec_data(end,i-1);
%          end


        % 2014b (Uses built-in function interp1 for last tick filling)
%         ts_sec_data(nan_idx,i) =  interp1(trading_time(~nan_idx),...
%             ts_sec_data(~nan_idx,i),...
%             trading_time(nan_idx),...
%             'previous','extrap'...
%              );
%         nan_idx =  isnan(ts_sec_data(:,i));
%         nnan_idx = find(nan_idx==0,1);
%         ts_sec_data(nan_idx,i) = ts_sec_data(nnan_idx,i);
%         % Revert to last tick intra-day
%         if i>1
%             ts_sec_data(nan_idx,i) = ts_sec_data(end,i-1);
%         end
    end
    ts_sec_data = reshape(ts_sec_data,length(ts_sec_data)*n_days,1);


    % Create time series object

    ts.ts = get_timeseries(ts_sec_data, date_time_grid_sec, start_date);
    ts.ts.Name = sprintf('%s/USD Exchange Futures Contracts',code{:});
    ts.DataInfo.Unit = 'dollar';
    ts.date = unique(date);
    ts.time = get_time_grid(1/24/60/60,0);


    % Save as mat files

    eval(sprintf('%s=ts;',code{:}));
    eval(sprintf('save(''%s.mat'',''%s'')',code{:},code{:}));
    eval(sprintf('clear ''%s''',code{:}));

end
%% (1)
codes = {'CHF','JPY','GBP'};
idx = 1;
for code=codes
    eval(sprintf('load %s;',code{:}));
    eval(sprintf('[ts_sec, date, time, n_sec, n_days, start_date] = get_data(%s);',code{:}));
    ts_log_rtn_sec = get_log_rtn(ts_sec,date);
    subplot(length(codes),1,idx);plot(ts_log_rtn_sec);
    ylabel('Logarithmic Return');
    set(gca, 'YTickLabel', num2str(get(gca,'YTick')','%1.3f'))
    title(ts_log_rtn_sec.Name)
```

```
    grid on
    idx = idx+1;
end
set(gcf,'PaperUnits', 'inch', 'PaperPosition', [0.25 0 10 10]);
print(gcf,'-dpng','-r100','log_return.png')

%% (2)
codes = {'CHF','JPY','GBP'};
idx = 1;
for code=codes
    eval(sprintf('load %s;',code{:}));
    eval(sprintf('[ts_sec, date, time, n_sec, n_days, start_date] = get_data(%s);',code{:}));

    [ rv_ts, rv_avg_30_ts, rp_ts ] = get_rv_estimators(ts_sec, date, time, 60);
    subplot(length(codes),1,idx);
    plot(rp_ts,'b')
    hold on
    plot(rv_avg_30_ts,'r')
    hold off
    grid on
    ylabel('Risk Measures');
    title(sprintf('Daily %s/USD Return 30 Miniutes Average RV and Range Based Estimators',code{:}))
    grid on
    legend('Range Based Estimator','30 Minites Average RV Estimator','Location','northwest')
    idx = idx+1;
end
set(gcf,'PaperUnits', 'inch', 'PaperPosition', [0.25 0 10 10]);
print(gcf,'-dpng','-r100','rv_rp.png');

%% (3)
codes = {'CHF','JPY','GBP'};
idx = 1;
for code=codes
    eval(sprintf('load %s;',code{:}));
    eval(sprintf('[ts_sec, date, time, n_sec, n_days, start_date] = get_data(%s);',code{:}));

    [ rv_ts, rv_avg_30_ts, rp_ts ] = get_rv_estimators(ts_sec, date, time, 60);
    subplot(length(codes),1,idx);

    rp_sig = []; rv_sig = []; avg_rv_sig = [];
    periods = find_all_factor(1800);
    periods = periods(15:end);
    for k = 1:length(periods)
        period=periods(k);
        disp(period);
        [ rv_ts, rv_avg_30_ts, rp_ts ] = get_rv_estimators(ts_sec, date,...
                                                                 time, period);
        rp_sig = [rp_sig mean(rp_ts)];
        rv_sig = [rv_sig mean(rv_ts)];
        avg_rv_sig = [avg_rv_sig mean(rv_avg_30_ts)];
    end
    semilogx(periods,[rp_sig;rv_sig;avg_rv_sig]);
    grid on;
    title(sprintf('%s RV Esitimators Signature Plot',code{:}));
```

```
    xlabel('Periods (seconds)')
    ylabel('Sample Average of RV Esitimators');
    legend('Ranged Based Esitimator',...
           'Sparse RV Esitimator',...
           '30 Minites Average RV Estimator')
    idx = idx+1;
end
set(gcf,'PaperUnits', 'inch', 'PaperPosition', [0.25 0 10 10]);
print(gcf,'-dpng','-r100','sig_plt.png');

%% (4)
codes = {'CHF','JPY','GBP'};
for code=codes
    eval(sprintf('load %s;',code{:}));
    eval(sprintf('[ts_sec, date, time, n_sec, n_days, start_date] = get
        _data(%s);',code{:}));

    idx = 1;
    periods = [60 300 1800];
    for period=periods

    [ rv_ts, rv_avg_30_ts, rp_ts ] = get_rv_estimators(ts_sec, date,...
        time, period);


        subplot(3,3,1+(idx-1)*3);autocorr(rv_avg_30_ts.Data,90);
        set(gca, 'Title',[],'XLabel',[],'YLabel',[])
        if 1+(idx-1)*3 <= 3
            title({'30 Minitues Average RV Esitimator','Realized
                Volatility'})
        end
        if 1+(idx-1)*3 >= 6
            xlabel('Lags')
        end
        ylabel(sprintf('Autocorrelation %d Minitues',period/60));


        subplot(3,3,2+(idx-1)*3);autocorr(rv_ts.Data,90);
        set(gca, 'Title',[],'XLabel',[],'YLabel',[])
        if 2+(idx-1)*3 <= 3
            title('Sparse RV Esitimator')
        end
        if 1+(idx-1)*3 >= 6
            xlabel('Lags')
        end

        subplot(3,3,3+(idx-1)*3);autocorr(rp_ts.Data,90);
        set(gca, 'Title',[],'XLabel',[],'YLabel',[])
        if 3+(idx-1)*3 <= 3
            title({'Range Based RV Esitimator', 'for Realized
                Volatility'})
        end
        if 1+(idx-1)*3 >= 6
            xlabel('Lags')
        end

        idx = idx + 1;
    end
    set(gcf,'PaperUnits', 'inch', 'PaperPosition', [0.25 0 10 10]);
    print(gcf,'-dpng','-r100',sprintf('acf_plt_%s.png',code{:}));
end
```

```
%% (5)
codes = {'CHF','JPY','GBP'};
idx = 1;
for code=codes
    eval(sprintf('load %s;',code{:}));
    eval(sprintf('[ts_sec, date, time, n_sec, n_days, start_date] = get
        _data(%s);',code{:}));

    subplot(length(codes),1,idx);

    % Relative time grid in day
    date_time_grid_day = get_time_grid(1,date)-start_date;

    log_rtn_sec = reshape(log(ts_sec.data),n_sec,n_days);
    daily_on_rtn_s = (log_rtn_sec(end,:)-log_rtn_sec(1,:)).^2;
    daily_on_rtn_s_sum = sum(daily_on_rtn_s);

    date_time_grid_5_sec = get_time_grid((1/24/60/60)*5,date)-start_
        date;
    ts_5_sec = resample(ts_sec, date_time_grid_5_sec, 'zoh');
    ts_log_rtn = get_log_rtn(ts_5_sec,date);
    n_tick = ts_5_sec.Length/n_days;


    % RV 24 (Method 1)
    log_rtn_sec = reshape(log(ts_5_sec.Data),n_tick,n_days);
    daily_log_rtn = diff(log_rtn_sec);
    rv_daily = sum(daily_log_rtn.^2);
    rv_daily_sum = sum(rv_daily);
    scale_factor = daily_on_rtn_s_sum/rv_daily_sum;
    ts_rv_24_ver1 = get_timeseries(scale_factor*rv_daily,...
                               date_time_grid_day,start_date,1);


    % RV 24 (Method 2)
    log_rtn_sec = reshape(log(ts_5_sec.Data),n_tick,n_days);
    daily_log_rtn = diff(log_rtn_sec);
    rv_daily = sum(daily_log_rtn.^2);
    level = [0 (log_rtn_sec(1,2:end)-log_rtn_sec(end,1:end-1))].^2;
    ts_rv_24_ver2 = get_timeseries(level+rv_daily,...
                                date_time_grid_day,start_date,1);
    plot(ts_rv_24_ver1);
    hold on
    plot(ts_rv_24_ver2);
    hold off
    grid on
    legend('Method 1','Method 2');
    title(sprintf('24-hour RV Estimator for %s/USD Logarithmic Return',
        code{:}))
    idx = idx+1;
end
set(gcf,'PaperUnits', 'inch', 'PaperPosition', [0.25 0 10 10]);
print(gcf,'-dpng','-r100',sprintf('24_rv.png',code{:}));
```

code/p5/q5.m

```
function [ factors ] = find_all_factor( a )
% Utility function for find all factor of a number

    pf = factor(a);
    factors = [];
    for i=1:length(pf)
        factors = [factors prod(combnk(pf,i),2)'];
```

```
    end
    factors = unique(factors);

end
```

code/p5/find_all_factor.m

```
function [ ts_sec, date, time, n_sec, n_days, start_date ] = get_data(
    input )
% Utility function for getting the data from save .mat variables

ts_sec = input.ts;
date = input.date;
time = input.time;
n_sec = length(time);
n_days = length(date);
start_date = input.ts.timeInfo.StartDate;

end
```

code/p5/get_data.m

```
function [ ts_log_rtn ] = get_log_rtn( ts, date )
% Caculate log rtn for a given time series of price


% Get timeseries information

start_date = ts.timeInfo.StartDate;
n_days = length(date);
n_tick = ts.Length/n_days;


% Reshape matrix to column per day assume 0 initial return every day

log_rtn = reshape(log(ts.Data),n_tick,n_days);
log_rtn = [zeros(1,n_days);
           diff(log_rtn)];
log_rtn = reshape(log_rtn,(n_tick)*n_days,1);


% Create new timeseries object for output

ts_log_rtn = get_timeseries(log_rtn,ts.Time,start_date);
ts_log_rtn.Name = sprintf('%s Logarithmic Return',ts.Name);

end
```

code/p5/get_log_rtn.m

```
function [ date_time_grid ] = get_time_grid(incr,date)
% Get regular time interval grid


% Create date and time grid independently

date_grid = date;
time_fmt = 'HH:MM:SS';
time_grid = [datenum('07:20:00',time_fmt):incr:datenum('14:00:00',time_
    fmt)]';


```

```
% Join date and time together

date_time_grid = kron(date_grid,ones(length(time_grid),1))+... % date
                      kron(ones(length(date_grid),1),time_grid)... %
                          time
                            -datenum('0:0','HH:MM')*... % offset
                             ones(length(time_grid)*length(date_grid)
                                 ,1);
end
```

code/p5/get_time_grid.m

```
function [ ts ] = get_timeseries(data, time, start, day_only)
% Utility function for setting initial parameters of timeseries object
% for object creation

    ts = timeseries(data,time);
    ts.TimeInfo.StartDate = start;
    ts.TimeInfo.Units = 'days';
    if nargin < 4
        day_only = false;
    end
    if day_only == true
        ts.TimeInfo.StartDate = '01/02/2009';
        ts.TimeInfo.Format = 'dd-mmm-yyyy';
    end
end
```

code/p5/get_timeseries.m

References

[1] Torben G. Andersen, Tim Bollerslev, Francis X. Diebold, and Paul Labys. Exchange rate returns standardized by realized volatility are (nearly) gaussian. Working Paper 7488, National Bureau of Economic Research, January 2000.

[2] P.F. Christoffersen. *Elements of Financial Risk Management*. Number v. 1 in Elements of Financial Risk Management. Academic Press, 2003.

[3] Rama Cont. Empirical properties of asset returns: stylized facts and statistical issues. *Quantitative Finance*, 1:223–236, 2001.