STAT0011 ICA

Group 27

1 Question 1

1.1 Theory

- N(0,σ_1^2) $t \leq \tau$
- N(0,σ_2^2) where $t > \tau$ and σ_1, σ_2, τ are unknown

We seek to estimate τ using the Bayesian approach:

1. Start with a prior $p(\tau)$
2. Specify likelihood $p(\mathbf{y}|\tau$
3. Apply the Bayes' Theorem to obtain the posterior distribution $p(\tau|\mathbf{y})$ which represents our knowledge about the change point after seeing the data.

$$p(\tau|\mathbf{y}) = \frac{p(\tau)p(\mathbf{y}|\tau)}{p(\mathbf{y})}$$

1.1.1 The Prior

If there are n observations, the last point cannot be a change point since we require at least one observation on both sides of τ. As such, there are n−1 possible locations, so the non-informative prior is just a discrete uniform distribution on $1, ..., n-1$:

$$P(\tau = k) = \frac{1}{n-1}, k = 1, ..., n-1$$

1.1.2 The Likelihood

Now we consider the likelihood $p(\mathbf{y}|\tau, \sigma_1, \sigma_2)$, which depends on the unkown parameters σ_1 and σ_2. Hence we use the Inverse Gamma(1,1) distribution as a conjugate prior for σ^2

$$f_z(z; 1, 1) = \frac{1}{\Gamma(1)}(1/z^2)\exp\left(-1/z\right), z > 0$$

We consider the area to the left of the change point τ. We can use the Law of Total Probability to remove σ_1^2.

Therefore,

$$P(X_{t\leq\tau}|\tau) = \int (\prod_{i=0}^{\tau} p(X_i|\sigma_1^2))p(\sigma_1^2)d\sigma_1^2$$

$$= \int \prod_{i=0}^{\tau} \frac{1}{\sigma_1\sqrt{2\pi}} \exp{(-\frac{1}{2}(\frac{x_i}{\sigma_1}^2))} \frac{1}{\Gamma(1)}(1/\sigma_1^2)^2 \exp{(-1/\sigma_1^2)} d\sigma_1^2$$

$$= \int \prod_{i=0}^{\tau} \frac{1}{\sigma_1\sqrt{2\pi}} \exp{(-\frac{1}{2}(\frac{x_i}{\sigma_1}^2))}(1/\sigma_1^2)^2 \exp{(-1/\sigma_1^2)} d\sigma_1^2$$

Similarly for the RHS:

$$P(X_{t>\tau}|\tau) = \int (\prod_{i=\tau+1}^{n} p(X_i|\sigma_2^2))p(\sigma_2^2)d\sigma_2^2$$

$$= \int \prod_{i=\tau+1}^{n} \frac{1}{\sigma_2\sqrt{2\pi}} \exp{(-\frac{1}{2}(\frac{x_i}{\sigma_2}^2))} \frac{1}{\Gamma(1)}(1/\sigma_2^2)^2 \exp{(-1/\sigma_2^2)} d\sigma_2^2$$

$$= \int \prod_{i=\tau+1}^{n} \frac{1}{\sigma_2\sqrt{2\pi}} \exp{(-\frac{1}{2}(\frac{x_i}{\sigma_2}^2))}(1/\sigma_2^2)^2 \exp{(-1/\sigma_2^2)} d\sigma_2^2$$

Hence we can now calculate the full likelihood $P(X_t|\tau)$:

$$P(\mathbf{X}_t|\tau) = \int \prod_{i=0}^{\tau} \frac{1}{\sigma_1\sqrt{2\pi}} \exp{(-\frac{1}{2}(\frac{x_i}{\sigma_1}^2))}(1/\sigma_1^2)^2 \exp{(-1/\sigma_1^2)} d\sigma_1^2$$

$$\times \int \prod_{i=\tau+1}^{n} \frac{1}{\sigma_2\sqrt{2\pi}} \exp{(-\frac{1}{2}(\frac{x_i}{\sigma_2}^2))}(1/\sigma_2^2)^2 \exp{(-1/\sigma_2^2)} d\sigma_2^2 := \phi$$

Finally, combining with the prior $p(\tau)$ gives the posterior distribution for the change point τ:

$$P(\tau|\mathbf{X}_t) = \frac{\frac{1}{n-1}\phi}{\sum_i^n X_t}$$

Where ϕ is defined above.

1.2 R-Program Solution

```
data = read.csv("/Users/harrisonknowles/Desktop/stat11_ica/group27.csv")
x = data[,1]
i = 1:299
prob_tau = 1/298

integranda <- function(sigsq,tau) {
  u <-1
  for(i in 1:tau){
    u = u * 1/(sqrt(2*pi*sigsq)) * exp(-1/2*(x[i]/sqrt(sigsq))^2)
  }
  return(u)
}

integranda2 <- function(sigsq,tau) {
  u <-1
  for(i in (tau:298)){
    u = u * 1/(sqrt(2*pi*sigsq)) * exp(-1/2*(x[i]/sqrt(sigsq))^2)
  }
  return(u)
}

integrandb <- function(sigsq){
  v = (1/(sigsq))^2 * exp(-1/sigsq)
  return(v)
}

integrand1 <- function(sigsq,tau){
  int = integranda(sigsq,tau)*integrandb(sigsq)
  return(int)
}

integrand2 <- function(sigsq,tau){
  int = integranda2(sigsq,tau)*integrandb(sigsq)
  return(int)
}

probs <- rep(0,298)
j <-1
```

```
while(j < length(probs)){
  probs[j]=(integrate(integrand1,0,Inf,tau=j)$value)*(integrate(integrand2,0,Inf,tau=j)$value)
  j = j+1
}
probs = (probs*prob_tau) / sum(probs)
which.max(probs)

plot(i,x)
abline(v=145, col="blue")
text(x = 120, y = -6, label = "Tau=145", srt = 0,cex=1)
```

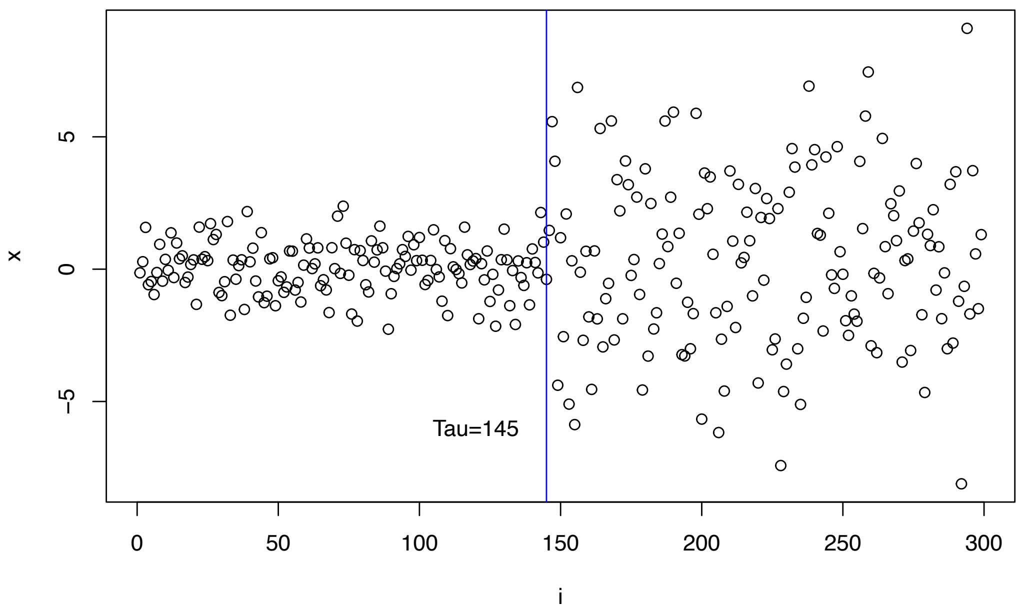


Figure 1: Change Point Estimate

2 Question 2

2.1 Theory

We started by downloading the daily price data for the S&P500 stock index from finance.yahoo.com. At first it seems logical to use closing prices, however there are certain events such as stock splits and stock buybacks which change the prices of equities without altering the value (in most cases). As such, it is best to proceed with the adjusted closing prices that implement calculations into the price to account for such events. Although this doesn't usually effect stock indices such as the

S&P500, it is a technicality that needs to be accounted for when analysing a portfolio with weight in individual equities.

In order to implement the risk measure Value-at-Risk for our equally weighted portfolio, the resulting risk measurement is dependent on the specification of:

- (Probability of losses exceeding VaR) $= p$
- Holding period (interval in which losses may occur)

We then have to estimate the probability distribution of the profit and loss for the portfolio. We will do this by utilising past observations (historical data) and fitting a statistical model.

Probability of losses exceeding VaR is pre-determined i.e. we are considering $p = \{0.01, 0.05\}$ such that we will obtain $\text{VaR}^{99\%}$ and $\text{VaR}^{95\%}$, respectfully. We assume the holding period to be one week.

2.2 Strengths and weaknesses of using the Monte Carlo simulation approach in this context.

The future price movements for the SP500 and FTSE100 are subject to randomness and uncertainty due to the many variables that affect price action. The Monte Carlo simulation is appropriate to use in this context as it aims to reduce uncertainty by providing a spectrum of outcomes. This gives the MCS an advantage when compared to other models. It is particularly important when estimating value at risk since underestimating losses would be undesirable. This combined with a significant amount of prior data means the Monte Carlo simulation can provide accurate predictions in this context. Although, the Monte Carlo simulation works best for perfectly efficient markets and there are many factors the simulation does not account for. Given the strengths of an MCS approach, we will now discuss the drawbacks.

The MCS provides outcomes and the likelihood of events occurring, however, the quality of the results is purely determined by the inputs provided leaving potential room for error. Firstly, as well as the underlying stochastic model, the random number generator is deterministic of the end results. The RNG used must have sufficient periodicity especially for portfolios of many assets (even more so if this includes exotic derivative products). Secondly, the MCS tends to have less accuracy at the 'ends' of the distribution, in our case this is the extreme losses with low probability of happening. More specifically, this is due to the transformation method. Many methods are optimally tuned for the centre of the distribution and utilise linear approximations at the tails. This leads to extreme uniforms being incorrectly transformed. Finally, the number of simulations is also an important consideration. Given the MCS utilises computing power, it is important to have enough simulations to yield accurate results, whilst also being considerate of the available computer memory. Traditional computing proposals of relating the accuracy to the inverse of the simulation size do not apply in this case, making it even more difficult to balance. This is because the MCS method for estimating portfolio VaR uses nonlinear transformations of random variables.

To summarise, the MCS is a good choice in this context as it helps reduce the effects of uncertainty that are inevitable in the stock market. With the drawbacks being that the MCS cannot predict major market crises, and requires careful considerations in its application.

```r
options(warn=-1);
library("tseries");
```

```r
library("zoo");
```

```r
library(gridExtra);
library(ggTimeSeries);
```

```r
library(xts);
library(pracma);
library(VineCopula);
library(fGarch);
```

```r
library(KScorrect);
library(stats);
library(ADGofTest);
rm(list=ls(all=TRUE));
```

Downloading adjusted close price data for the specified time period and chosen stock indices.

```r
start_date<-as.Date("2009-10-01");
end_date<-as.Date("2016-02-02");
priceSP = get.hist.quote(instrument="^gspc", start = start_date, end = end_date, quote="AdjClose",compression =
"w");
```

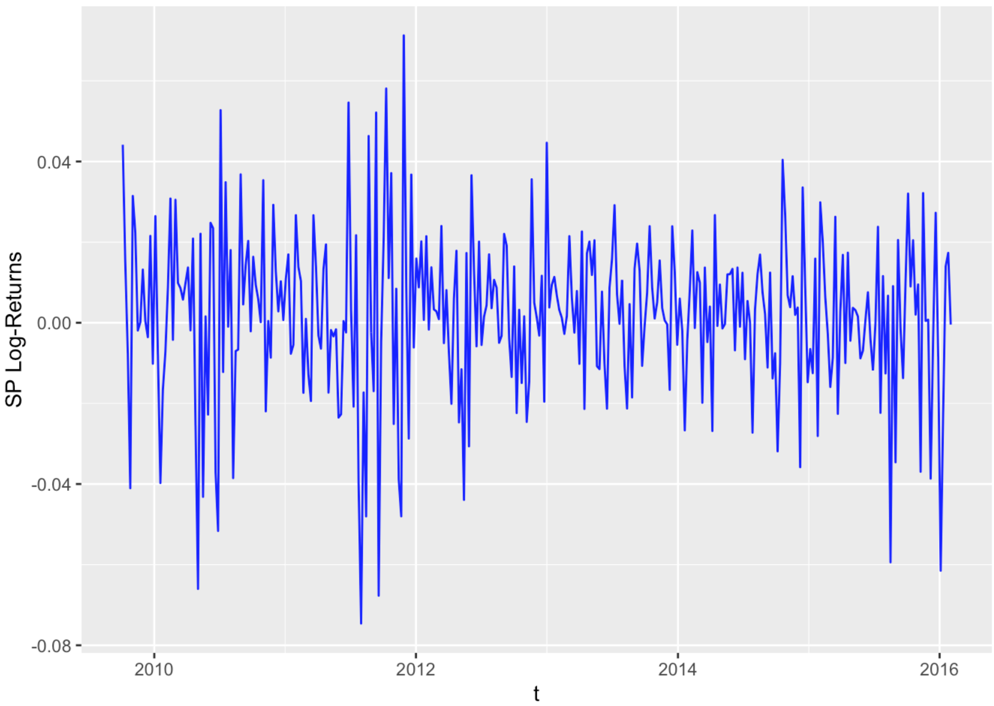
```r
priceFTSE = get.hist.quote(instrument="^ftse", start = start_date, end = end_date, quote="AdjClose",compression =
 "w");
```

y_individual_price and y_individual are zoo objects representing time-series data for the prices of each asset, and the log-returns of each asset respectfully.

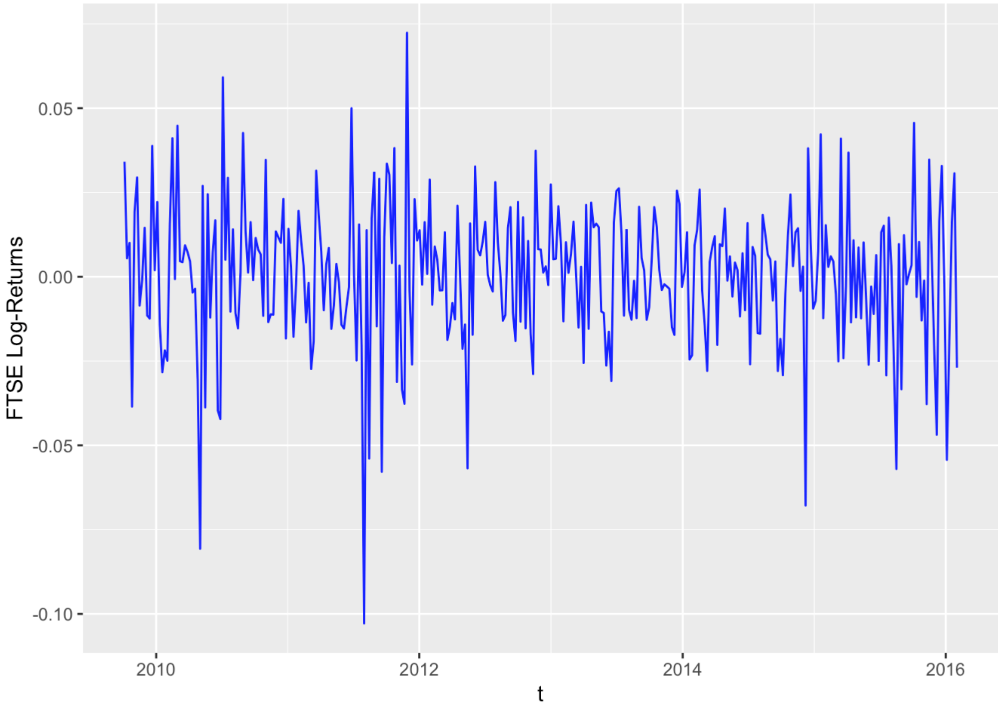
```r
y_individual_price = na.omit(cbind(priceSP, priceFTSE));#Prices of each asset
y_individual = na.omit(diff(log(cbind(priceSP, priceFTSE))));#Log returns of each asset
```

Here we plot each individual asset's log-returns over the time period.

```r
ggplot(data=y_individual,aes(x=Index,y=Adjusted.priceSP))+geom_line(colour="blue")+labs(x ="t", y = "SP Log-Retur
ns")
```



```
ggplot(data=y_individual,aes(x=Index,y=Adjusted.priceFTSE))+geom_line(colour="blue")+labs(x ="t", y = "FTSE Log-R
eturns")
```



Constructing an equally weighted portfolio with prices.

```
SP_price_init =drop(coredata(y_individual_price[1]$Adjusted.priceSP))
FTSE_price_init =drop(coredata(y_individual_price[1]$Adjusted.priceFTSE))
portfolio_weights<- c(FTSE_price_init/SP_price_init,1)
weightsum = sum(portfolio_weights)
portfolio_weights[1] = portfolio_weights[1]/weightsum
portfolio_weights[2] = portfolio_weights[2]/weightsum
```

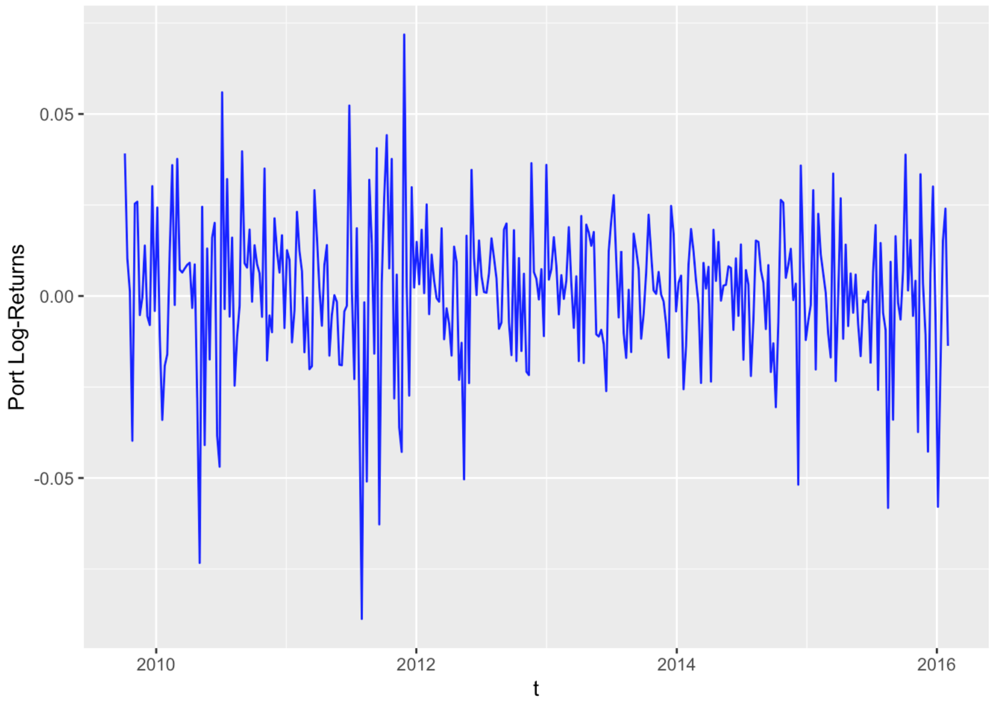
Constructing individual historical log-returns for each asset.

```
ret1 = portfolio_weights[1]*drop(coredata(y_individual[,1]))
ret2 = portfolio_weights[2]*drop(coredata(y_individual[,2]))
```

Constructing historical portfolio log-returns with plot.

```
equally_weighted_LR <- vector(mode = "numeric", length = dim(y_individual)[1])
for(i in 1:dim(y_individual)[1]){
  a = rowSums(y_individual[i])/2
  equally_weighted_LR[i] <- a}
dates = fortify.zoo(y_individual[,0])
equally_weighted_LR_df = data.frame(dates,equally_weighted_LR)
equally_weighted_LR.xts <- xts(x=equally_weighted_LR_df[,2],order.by= (equally_weighted_LR_df$Index))

ggplot(data=equally_weighted_LR.xts,aes(x=Index,y=equally_weighted_LR))+geom_line(colour="blue")+labs(x ="t", y =
 "Port Log-Returns")
```



Test of whether sample data have the skewness and kurtosis matching a normal distribution. Small p-value implies reject Null-Hypothesis i.e. these data do not come from a Normal Distribution.

```
jarqueberaTest(ret1)
```

```
##
## Title:
##  Jarque - Bera Normalality Test
##
## Test Results:
##   STATISTIC:
##     X-squared: 39.4003
##   P VALUE:
##     Asymptotic p Value: 2.782e-09
##
## Description:
##  Mon Mar 21 00:02:23 2022 by user:
```

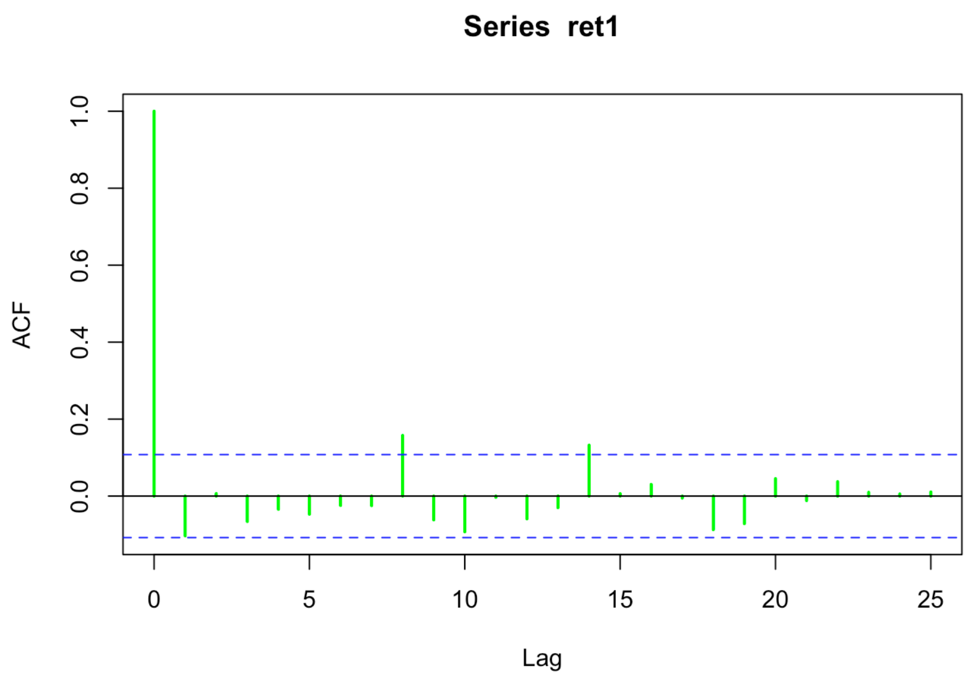
```
jarqueberaTest(ret2)
```

```
##
## Title:
##  Jarque - Bera Normalality Test
##
## Test Results:
##   STATISTIC:
##     X-squared: 82.9833
##   P VALUE:
##     Asymptotic p Value: < 2.2e-16
##
## Description:
##  Mon Mar 21 00:02:23 2022 by user:
```

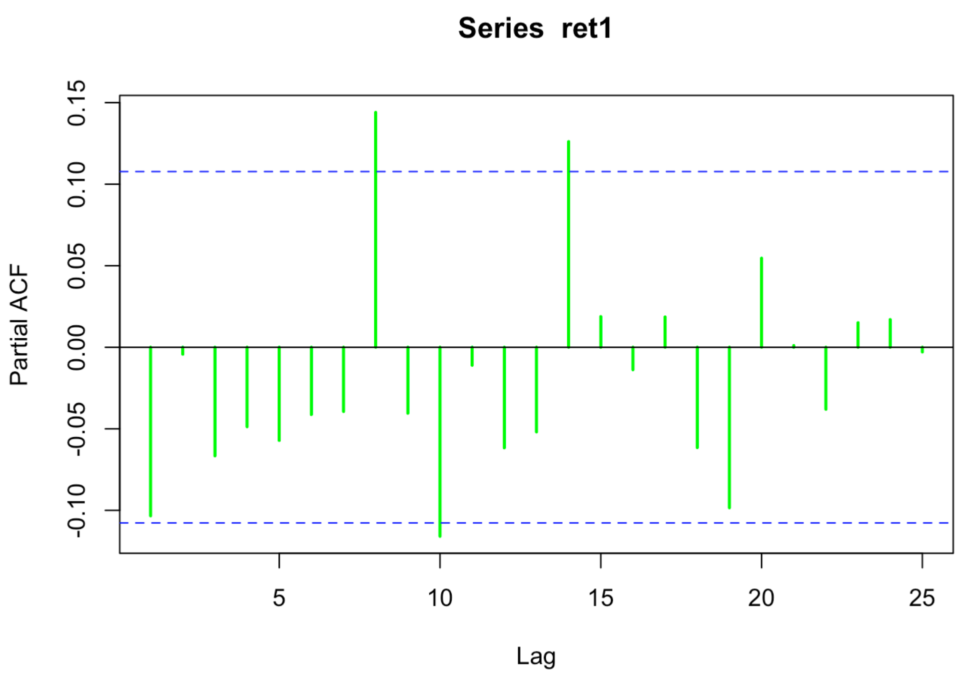
Here we can see that neither the Log-Returns for S&P500 or FTSE100 over the specified time period are distributed normally. The p-values are very low, making it easy to reject the null-hypothesis.

Building AR models: the Box - Jenkins approach Step 1: Identification

```
#S&P500
acf(ret1, col="green", lwd=2)
```

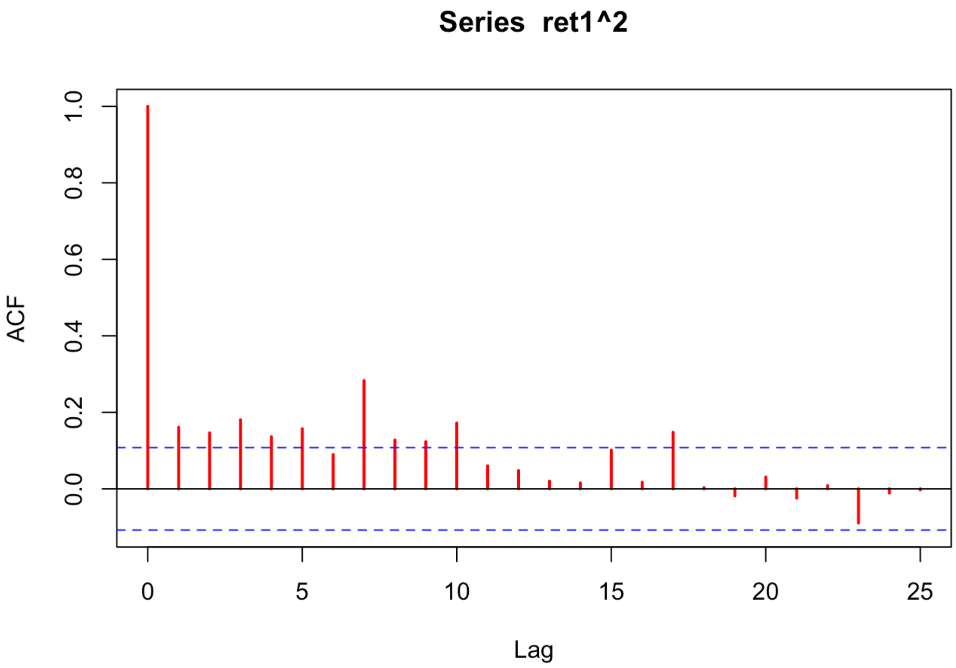



```
#S&P500
pacf(ret1, col="green", lwd=2)
```



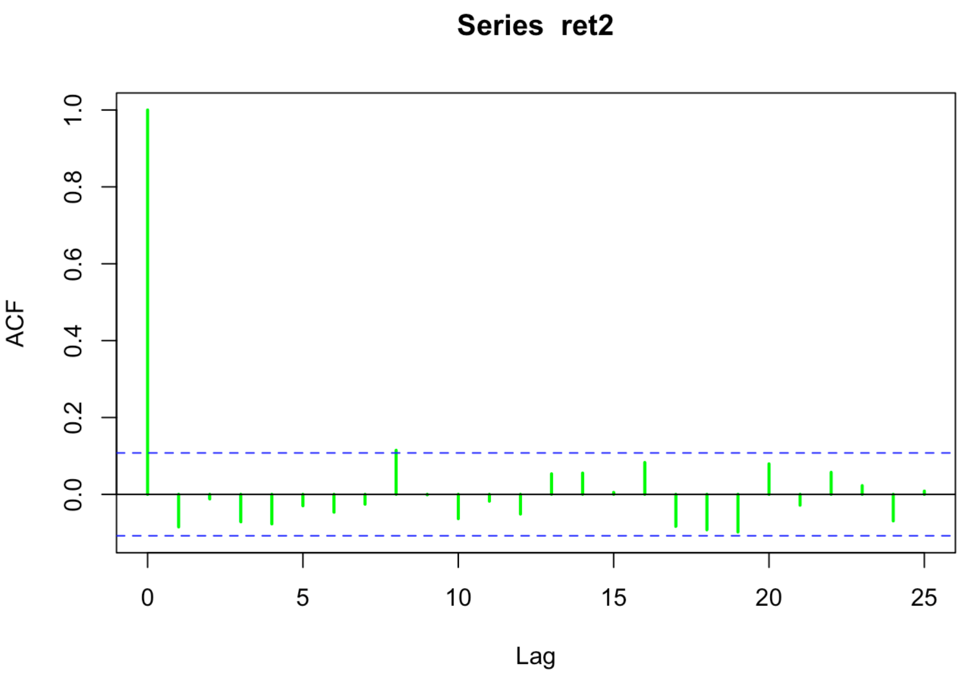
Both “ACF” and “Partial ACF” plots imply there could be significance in a lag of 8 weeks, or 1 week. This gives us motivation to test both AR(8) and AR(1) with various GARCH models to find the model that minimises the AIC.

```r
#S&P500
acf(ret1^2, col="red", lwd=2)
```

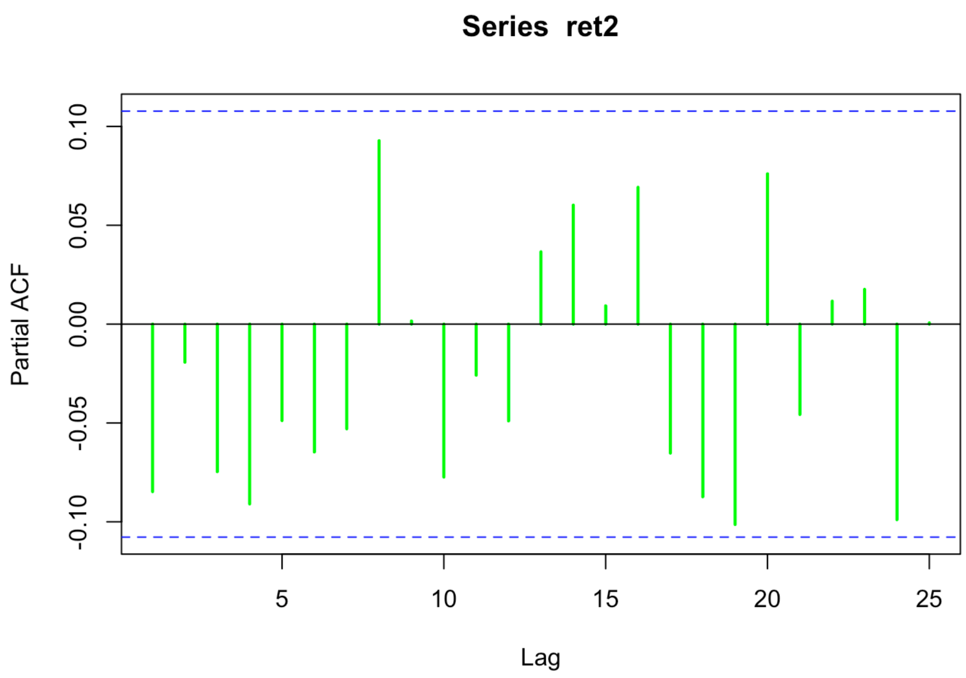


The ACF plot of squared residuals tends to zero significants as lags increase. This provides motivation to explore GARCH models supposed to a simpler MA volatility model. Given in a GARCH model, returns on older days have a lesser impact than more recent days. This allows us to deal with volatility clustering. GARCH models are also preferred to an EWMA model since GARCH models can be parametrised more easily and can become more specific given ACF plots.

```r
#FTSE100
acf(ret2, col="green", lwd=2)
```

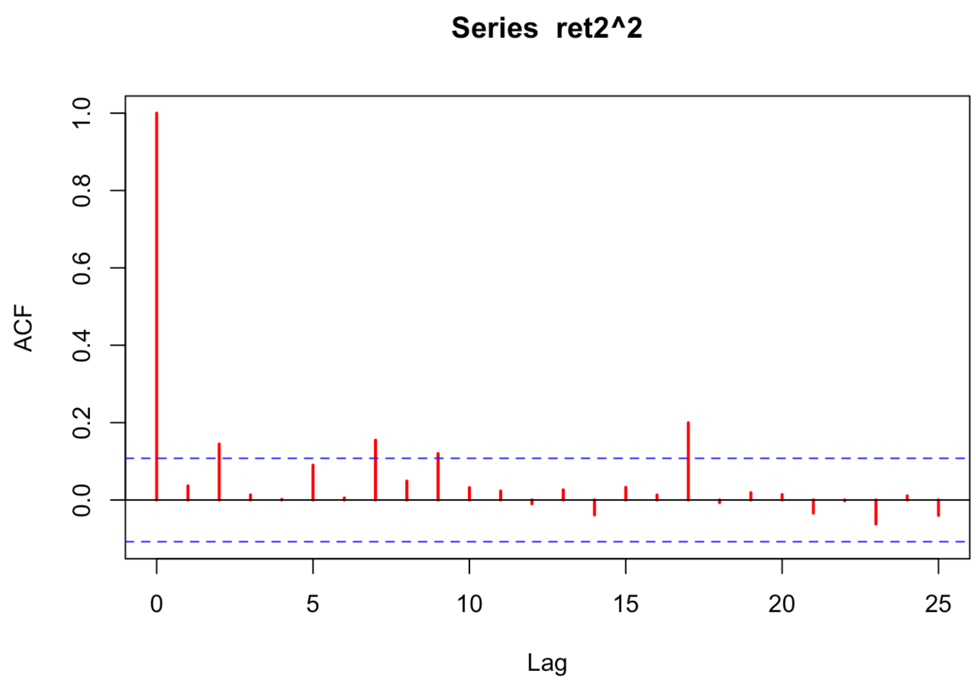


```r
#FTSE100
pacf(ret2, col="green", lwd=2)
```



Similarly for FTSE100, the “ACF” and “Partial ACF” plots imply there could be significance in a lag of 1 weeks, or 0 weeks. This gives us motivation to test both AR(1) and AR(0) with various GARCH models to find the model that minimises the AIC.

```r
#FTSE100
acf(ret2^2, col="red", lwd=2)
```



The ACF plot of squared residuals tends to zero significants as lags increase. This provides motivation to explore GARCH models supposed to a simpler MA volatility model. Given in a GARCH model, returns on older days have a lesser impact than more recent days. This allows us to deal with volatility clustering. GARCH models are also preferred to an EWMA model since GARCH models can be parametrised more easily and can become more specific given ACF plots.

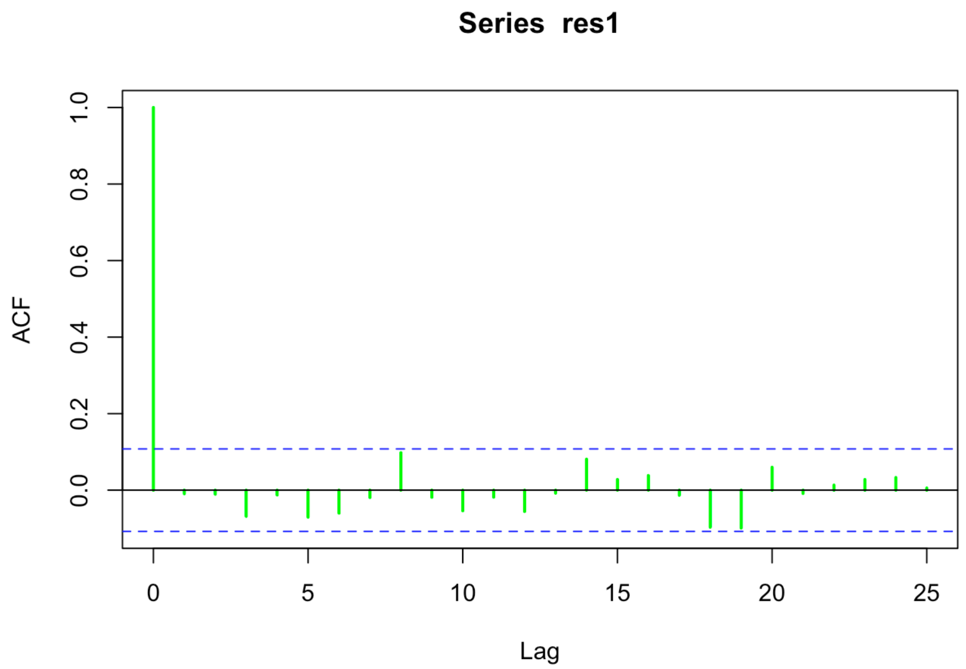
Step 2: Estimation & Model checking Below is an example of models for each asset in our chosen portfolio. This showcases how each model was tested. (All variations & results have been recorded in report table below).

```r
model1=garchFit(formula=~arma(1,0)+garch(1,1),data=ret1,trace=F,cond.dist="ged")
model2=garchFit(formula=~arma(1,0)+garch(1,1),data=ret2,trace=F,cond.dist="std")
```

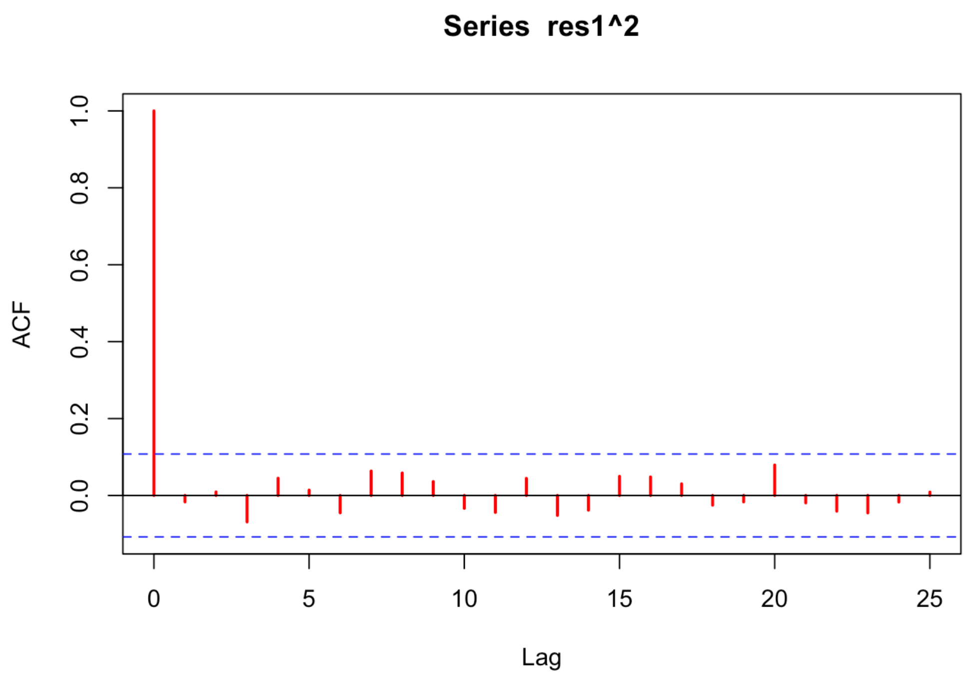
Step 3: Model checking

Model 1 (S&P500)

```
res1 <- residuals(model1, standardize=TRUE)
acf(res1, col="green", lwd=2)
```




```
acf(res1^2, col="red", lwd=2)
```



ACF plots of residuals (and residuals squared) show that there is no significant autocorrelation present.

```
Box.test(res1, lag = 10, type = c("Ljung-Box"), fitdf = 1)
```

```
##
##  Box-Ljung test
##
## data:  res1
## X-squared = 9.108, df = 9, p-value = 0.4274
```

```
Box.test(res1^2, lag = 10, type = c("Ljung-Box"), fitdf = 1)
```

```
##
##  Box-Ljung test
##
## data:  res1^2
## X-squared = 6.5196, df = 9, p-value = 0.687
```

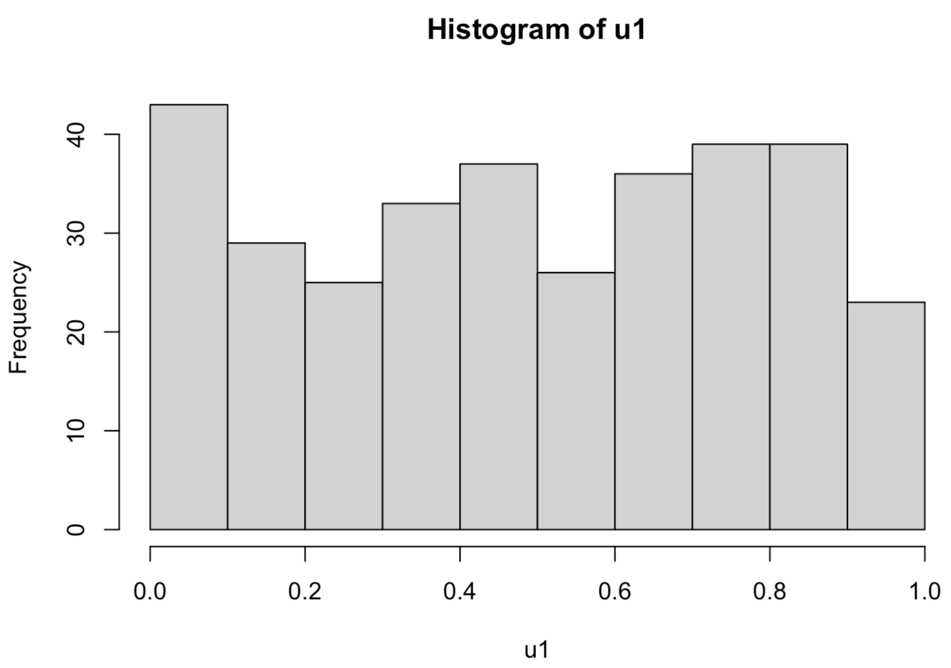
The Box-Ljung tests for residuals (and residuals squared) show there is no evidence that the modelling conclusions introduced thus far are different to that of the true model.

```
model1@fit$ics
```

```
##       AIC       BIC       SIC      HQIC
## -5.425559 -5.356638 -5.426201 -5.398071
```

Various information criteria are quoted that help us determine which model fits the data best. Minimisation of the AIC is the most important. As long as the distributional tests are passed, AIC minimisation determines the model chosen.

```
#u1<-pnorm(res1, mean=0, sd=1)[2:length(ret1)]
#u1<-pstd(res1, mean=0, sd=1,nu=coef(model1)[6])[2:length(ret1)]
#u1<-psstd(res1, mean=0, sd=1,nu=coef(model1)[7])[2:length(ret1)]
u1<-pged(res1, mean=0, sd=1,nu=coef(model1)[6])[2:length(ret1)]
hist(u1)
```



Here we plot the marginal standard uniform transformed returns for asset 1 (S&P500).

```
KStest1<-LcKS(u1, cdf = "punif")
KStest1$p.value
```

```
## [1] 0.632
```

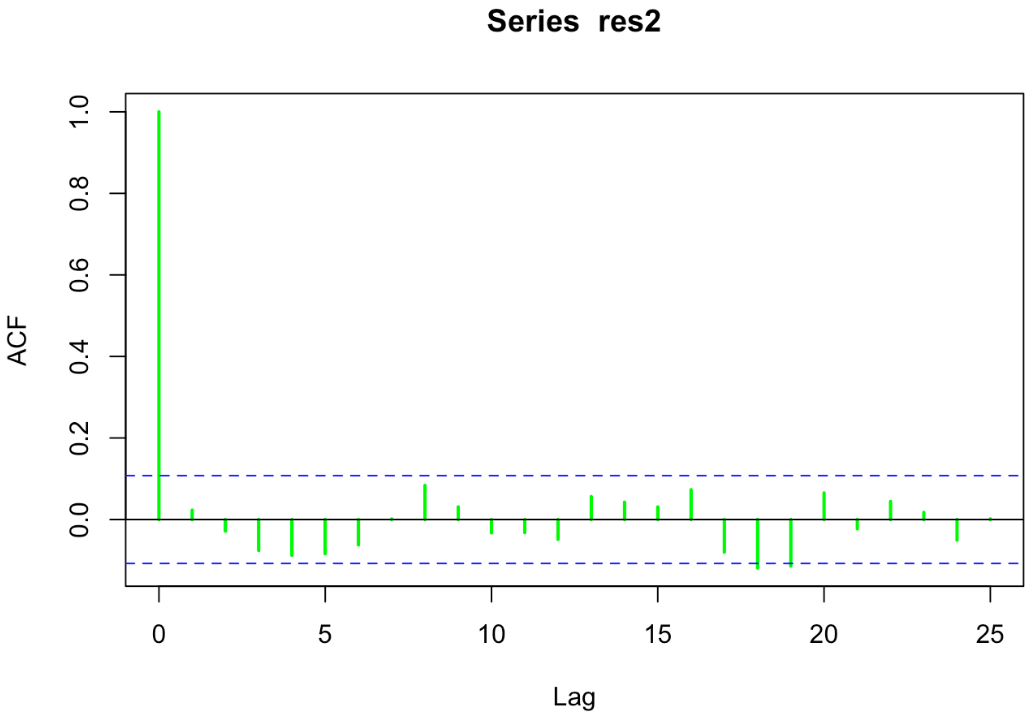
```
ADtest1<-ad.test(u1, null="punif")
ADtest1$p.value
```

```
##        AD
## 0.3336055
```

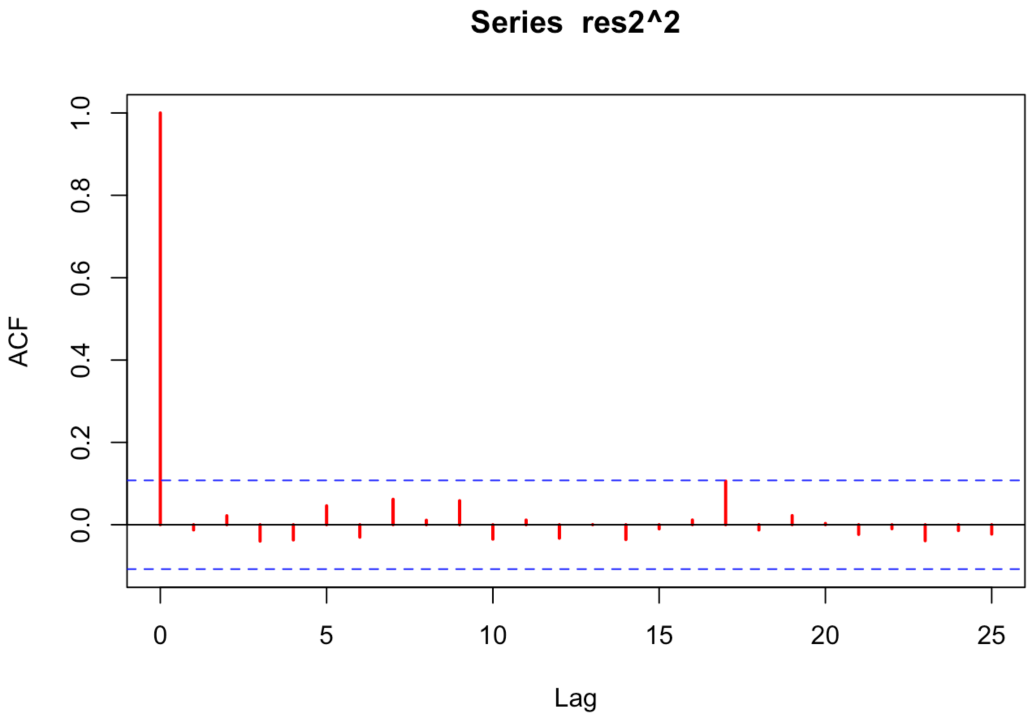
Further distributional checks are in place to ensure the marginal distribution is in fact standard uniform. (For use in copula).

Model 2 (FTSE100)

```
res2 <- residuals(model2, standardize=TRUE)
acf(res2, col="green", lwd=2)
```




```
acf(res2^2, col="red", lwd=2)
```



ACF plots of residuals (and residuals squared) show that there is no significant autocorrelation present.

```
Box.test(res2, lag = 10, type = c("Ljung-Box"), fitdf = 1)
```

```
##
##  Box-Ljung test
##
## data:  res2
## X-squared = 11.78, df = 9, p-value = 0.226
```

```
Box.test(res2^2, lag = 10, type = c("Ljung-Box"), fitdf = 1)
```

```
##
##  Box-Ljung test
##
## data:  res2^2
## X-squared = 5.1502, df = 9, p-value = 0.821
```

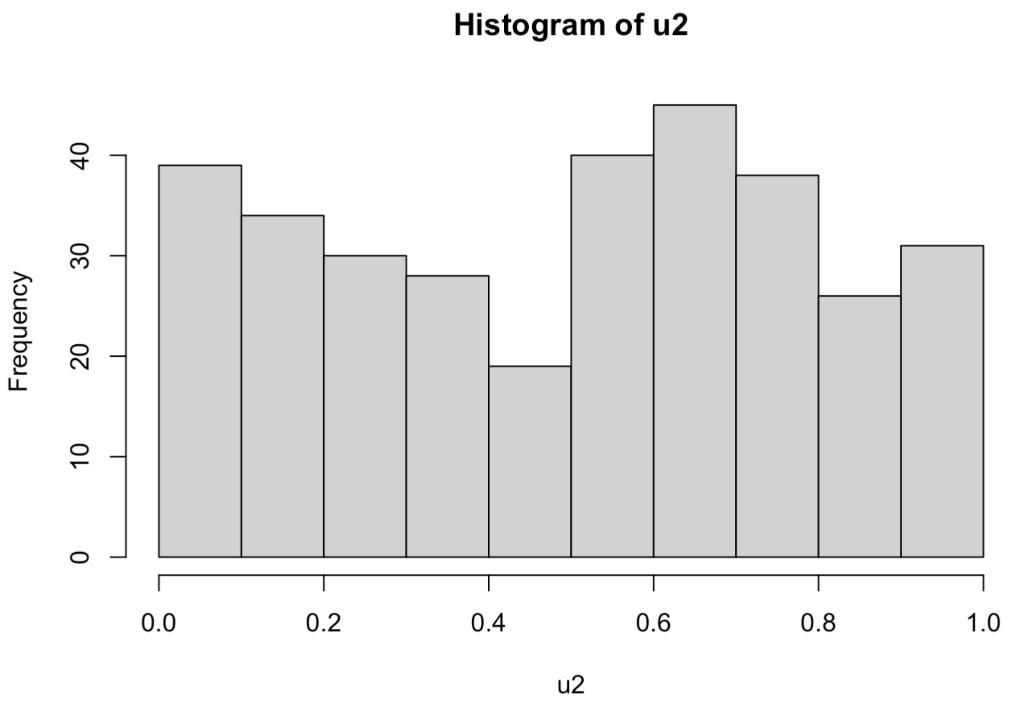
The Box-Ljung tests for residuals (and residuals squared) show there is no evidence that the modelling conclusions introduced thus far are different to that of the true model.

```
model2@fit$ics
```

```
##       AIC       BIC       SIC      HQIC
## -8.454408 -8.385487 -8.455050 -8.426919
```

Various information criteria are quoted that help us determine which model fits the data best. Minimisation of the AIC is the most important. As long as the distributional tests are passed, AIC minimisation determines the model chosen.

```
#u2<-pnorm(res2, mean=0, sd=1)[2:length(ret2)]
u2<-pstd(res2, mean=0, sd=1,nu=coef(model2)[6])[2:length(ret2)]
#u2<-psstd(res2, mean=0, sd=1,nu=coef(model2)[7])[2:length(ret2)]
#u2<-pged(res2, mean=0, sd=1,nu=coef(model2)[6])[2:length(ret2)]
hist(u2)
```



Here we plot the marginal standard uniform transformed returns for asset 1 (S&P500).

```
KStest2<-LcKS(u2, cdf = "punif")
KStest2$p.value
```

```
## [1] 0.2526
```

```
ADtest2<-ad.test(u2, null="punif")
ADtest2$p.value
```

```
##        AD
## 0.3588836
```

Further distributional checks are in place to ensure the marginal distribution is in fact standard uniform. (For use in copula).

Parameter Testing for S&P500					
AR(8)+GARCH(1,1)					
Distribution	Box Test (Res)	Box Test (Res^2)	AIC	KS Test	AD test
Norm	0.168	0.030	-5.425	0.62	0.20
STD	0.231	0.031	-5.446	0.54	0.21
SSTD	0.003	0.035	-5.480	0.001	0.0001
GED	0.257	0.018	-5.451	0.57	0.23
AR(1)+GARCH(1,1)					
Distribution	Box Test (Res)	Box Test (Res^2)	AIC	KS Test	AD test
Norm	0.432	0.704	-5.400	0.36	0.15
STD	0.407	0.714	-5.425	0.52	0.26
SSTD	0.278	0.786	-5.454	0.002	0.0003
GED	0.427	0.687	-5.426	0.63	0.33
AR(8)+GARCH(2,2)					
Distribution	Box Test (Res)	Box Test (Res^2)	AIC	KS Test	AD test
Norm	0.151	0.030	-5.414	0.64	0.21
STD	0.200	0.032	-5.434	0.57	0.23
SSTD	0.0009	0.036	-5.470	0.001	0.00008
GED	0.216	0.018	-5.438	0.58	0.24
AR(1)+GARCH(2,2)					
Distribution	Box Test (Res)	Box Test (Res^2)	AIC	KS Test	AD test
Norm	0.392	0.695	-5.391	0.39	0.16
STD	0.348	0.710	-5.415	0.63	0.27
SSTD	0.214	0.775	-5.445	0.004	0.0004
GED	0.388	0.680	-5.415	0.78	0.34
AR(8)+GARCH(3,3)					
Distribution	Box Test (Res)	Box Test (Res^2)	AIC	KS Test	AD test
Norm	0.168	0.039	-5.408	0.64	0.17
STD	0.232	0.074	-5.422	0.44	0.21
SSTD	0.002	0.013	-5.458	0.001	0.00008
GED	0.238	0.048	-5.428	0.50	0.20
AR(1)+GARCH(3,3)					
Distribution	Box Test (Res)	Box Test (Res^2)	AIC	KS Test	AD test
Norm	0.437	0.838	-5.384	0.41	0.16
STD	0.377	0.847	-5.405	0.57	0.27
SSTD	0.233	0.836	-5.432	0.003	0.0004
GED	0.420	0.841	-5.405	0.63	0.32

Parameter Testing for FTSE100					
AR(0)+GARCH(1,1)					
Distribution	Box Test (Res)	Box Test (Res^2)	AIC	KS Test	AD test
Norm	0.420	0.873	-8.397	0.37	0.24
STD	0.417	0.889	-8.446	0.51	0.51
SSTD	0.402	0.877	-8.455	0.0002	0.002
GED	0.424	0.891	-8.428	0.35	0.36
AR(1)+GARCH(1,1)					
Distribution	Box Test (Res)	Box Test (Res^2)	AIC	KS Test	AD test
Norm	0.302	0.798	-8.400	0.22	0.20
STD	0.226	0.821	-8.454	0.27	0.36
SSTD	0.144	0.812	-8.470	0.0002	0.0005
GED	0.224	0.830	-8.438	0.41	0.20

AR(0)+GARCH(2,2)					
Distribution	Box Test (Res)	Box Test (Res^2)	AIC	KS Test	AD test
Norm	0.435	0.875	-8.393	0.40	0.28
STD	0.379	0.898	-8.435	0.54	0.54
SSTD	0.382	0.891	-8.443	0.0002	0.003
GED	0.443	0.911	-8.418	0.41	0.37
AR(1)+GARCH(2,2)					
Distribution	Box Test (Res)	Box Test (Res^2)	AIC	KS Test	AD test
Norm	0.302	0.806	-8.396	0.22	0.21
STD	0.155	0.843	-8.446	0.22	0.39
SSTD	0.096	0.848	-8.463	0.0002	0.0005
GED	0.178	0.842	-8.427	0.38	0.23
AR(0)+GARCH(3,3)					
Distribution	Box Test (Res)	Box Test (Res^2)	AIC	KS Test	AD test
Norm	0.268	0.920	-8.384	0.37	0.33
STD	0.283	0.952	-8.428	0.57	0.57
SSTD	0.273	0.929	-8.437	0.0002	0.004
GED	0.291	0.950	-8.410	0.55	0.45
AR(1)+GARCH(3,3)					
Distribution	Box Test (Res)	Box Test (Res^2)	AIC	KS Test	AD test
Norm	0.169	0.839	-8.387	0.12	0.23
STD	0.123	0.894	-8.438	0.23	0.42
SSTD	0.074	0.829	-8.455	0.0002	0.0007
GED	0.129	0.898	-8.420	0.43	0.26

We now use our uniform marginals for copula modelling. The BiCopSelect tests the best fit under the AIC framework for 10 different copulas plus various rotations.

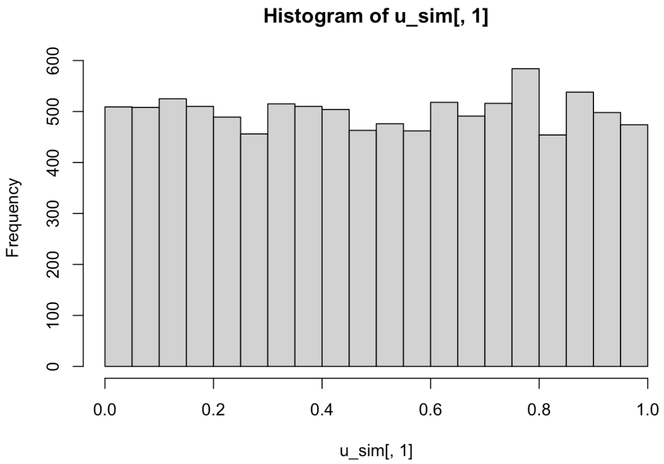
```
model=BiCopSelect(u1, u2, familyset=NA, selectioncrit="AIC", indeptest=TRUE, level=0.05,se = TRUE)
model
```

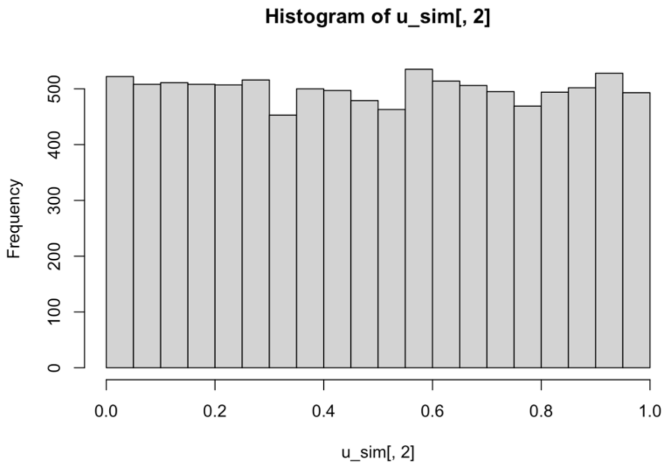
```
## Bivariate copula: Gaussian (par = 0.84, tau = 0.63)
```

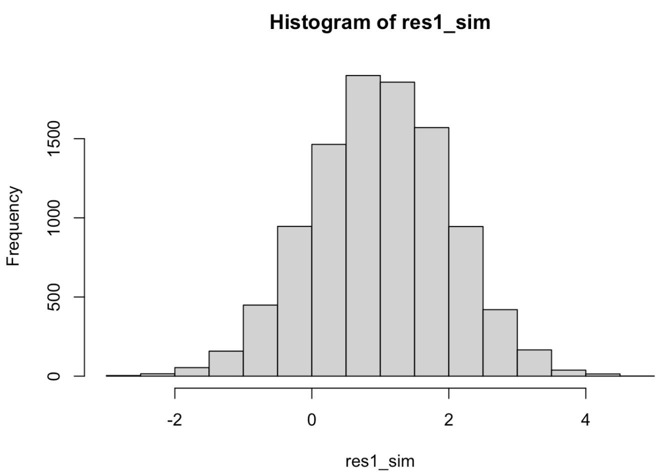
In this case, the best fit under the AIC framework is the Gaussian copula with parameters (par = 0.84, tau = 0.63).

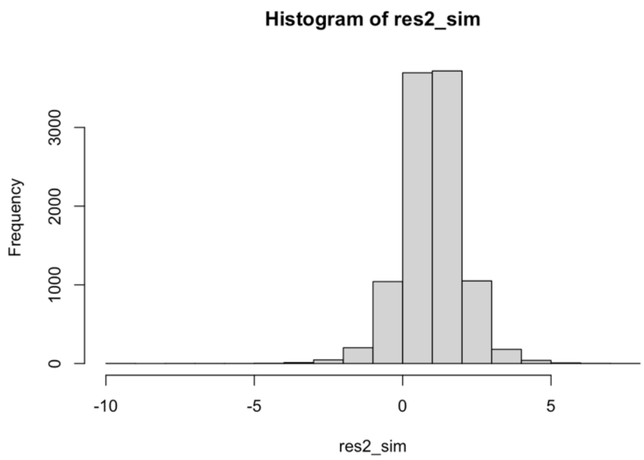
We now can make use of this copula in order to estimate the portfolio Value-at-Risk uisng Monte Carlo simulation.

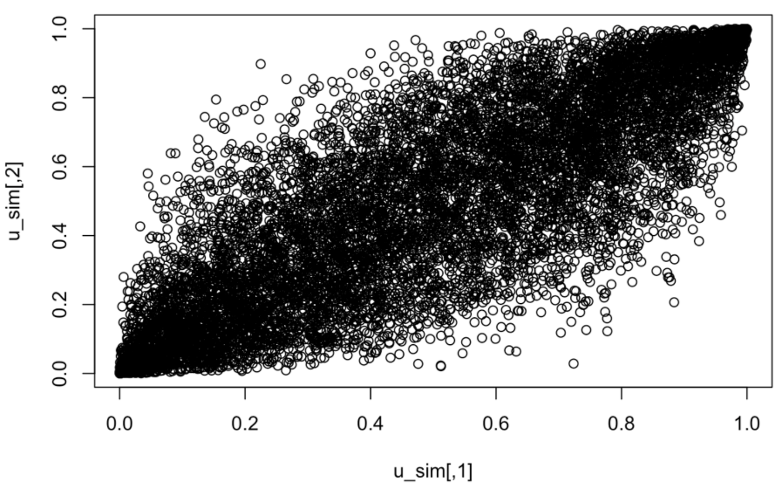
```
N=10000
u_sim=BiCopSim(N, family=model$family, model$par,  model$par2)
res1_sim=qged(u_sim[,1], mean = 1, sd = 1)
res2_sim=qstd(u_sim[,2], mean = 1, sd = 1)
uu1 = u_sim[,1]
uu2 = u_sim[,2]
```









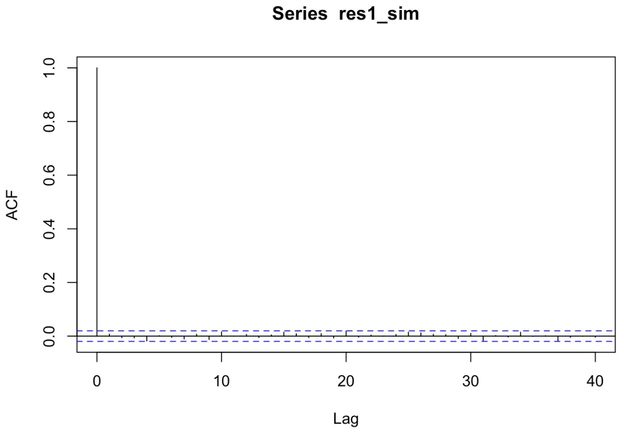


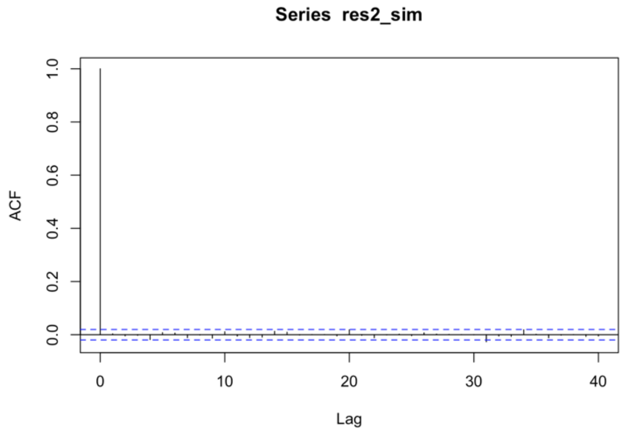
This plot is interesting, since it clearly

shows the heavy-tailed dependence of the Gaussian copula.

However, “res1_sim” and “res2_sim” are i.i.d. This can be seen in their ACF plots:

```
acf(res1_sim) #We see no significant lags because this is a simulated i.i.d process
acf(res2_sim) #We see no significant lags because this is a simulated i.i.d process
```





As such, we are required to give a treatment of re-introduction of autocorrelation and GARCH effects observed in the historical data.

First for asset 1 (S&P500):

```
OMEGA_1 = coef(model1)[3]
MU_1 = coef(model1)[1]
AR_1 = coef(model1)[2]
ALPH_1 = coef(model1)[4]
BETA_1 = coef(model1)[5]

Meansim1 <- matrix(0, nrow = N, ncol = 1)
Meansim1 [1] = MU_1 + AR_1*ret1[331]
for(i in seq(2, N, 1)) {
  Meansim1[i] = MU_1 + AR_1*Meansim1[i-1]
}

Variancesim1 <- matrix(0, nrow = N, ncol = 1)
Variancesim1 [1] = OMEGA_1 + ALPH_1*(model1@h.t[331])*(ret1[331])^2 + BETA_1*model1@h.t[331]
for(i in seq(2, N, 1)) {
  Variancesim1[i] = OMEGA_1 + ALPH_1*(uu1[i-1])^2 + BETA_1*Variancesim1[i-1]
}

Returnsim1 <- matrix(0, nrow = N, ncol = 1)
for(i in seq(1, N, 1)) {
  Returnsim1[i] = Meansim1[i]+Variancesim1[i]*res1_sim[i]
}
```

Then for asset 2 (FTSE100):

```
OMEGA_2 = coef(model2)[3]
MU_2 = coef(model2)[1]
AR_2 = coef(model2)[2]
ALPH_2 = coef(model2)[4]
BETA_2 = coef(model2)[5]

Meansim2 <- matrix(0, nrow = N, ncol = 1)
Meansim2 [1] = MU_2 + AR_2*ret2[331]
for(i in seq(2, N, 1)) {
  Meansim2[i] = MU_2 + AR_2*Meansim2[i-1]
}

Variancesim2 <- matrix(0, nrow = N, ncol = 1)
Variancesim2 [1] = OMEGA_2 + ALPH_2*(model2@h.t[331])*(ret2[331])^2 + BETA_2*model2@h.t[331]
for(i in seq(2, N, 1)) {
  Variancesim2[i] = OMEGA_2 + ALPH_2*(uu2[i-1])^2 + BETA_2*Variancesim2[i-1]
}

Returnsim2 <- matrix(0, nrow = N, ncol = 1)
for(i in seq(1, N, 1)) {
  Returnsim2[i] = Meansim2[i]+Variancesim2[i]*res2_sim[i]
}
```

We can then make use of the simulated returns and our portfolio weights to calculate the estimated VaR for different alpha levels. After carefully adjusting the portfolio distribution of returns to ensure an equal weighting, we can take the quantile function for the required alpha levels to arrive at the required VaR values.

```
portsim <- matrix(0, nrow = N, ncol = 1)
varsim <- matrix(0, nrow = 1, ncol = 2)
portsim=log(1+(exp(Returnsim1)-1)*(portfolio_weights[1])+(exp(Returnsim2)-1)*(portfolio_weights[2]))
varsim= abs(quantile(portsim,c(0.01,0.05)))
varsim
```

```
##        1%        5%
## 0.2594879 0.1066821
```