class3

Mohamed Khalifa

January 29, 2017

```r
#' # Imagine this ...
#'
#' # Stylized facts
#' ...of the market
#'
#' ## Learned the hard Way: not independent, volatile volatility, extreme
#' - Financial stock, bond, commodity...you name it...have highly interdependent relationships.
#' - Volatility is rarely constant and often has a structure (mean reversion) and is dependent on the p
#' - Past shocks persist and may or may not dampen (rock in a pool).
#' - Extreme events are likely to happen with other extreme events.
#' - Negative returns are more likely than positive returns (left skew).
#'
#' ***
#' Examples from the 70's, 80's, and 90's have lots of global events going on. Load up some computation
#'
#'
library(fBasics)
```

```
## Loading required package: timeDate
## Loading required package: timeSeries
##
## Rmetrics Package fBasics
## Analysing Markets and calculating Basic Statistics
## Copyright (C) 2005-2014 Rmetrics Association Zurich
## Educational Software for Financial Engineering and Computational Science
## Rmetrics is free software and comes with ABSOLUTELY NO WARRANTY.
## https://www.rmetrics.org --- Mail to: info@rmetrics.org
```

```r
library(evir)
library(qrmdata)
library(zoo)
```

```
##
## Attaching package: 'zoo'
## The following object is masked from 'package:timeSeries':
##
##     time<-
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
data(OIL_Brent)
str(OIL_Brent)
```

```
## An 'xts' object on 1987-05-20/2015-12-28 containing:
##   Data: num [1:7258, 1] 18.6 18.4 18.6 18.6 18.6 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : NULL
##   ..$ : chr "OIL_Brent"
##   Indexed by objects of class: [Date] TZ: UTC
##   xts Attributes:
##  NULL
```

```r
#'
#'
#' ***
#'
Brent.price <- as.zoo(OIL_Brent)
Brent.return <- diff(log(Brent.price))[-1] * 100
colnames(Brent.return) <- "Brent.return"
head(Brent.return, n = 5)
```

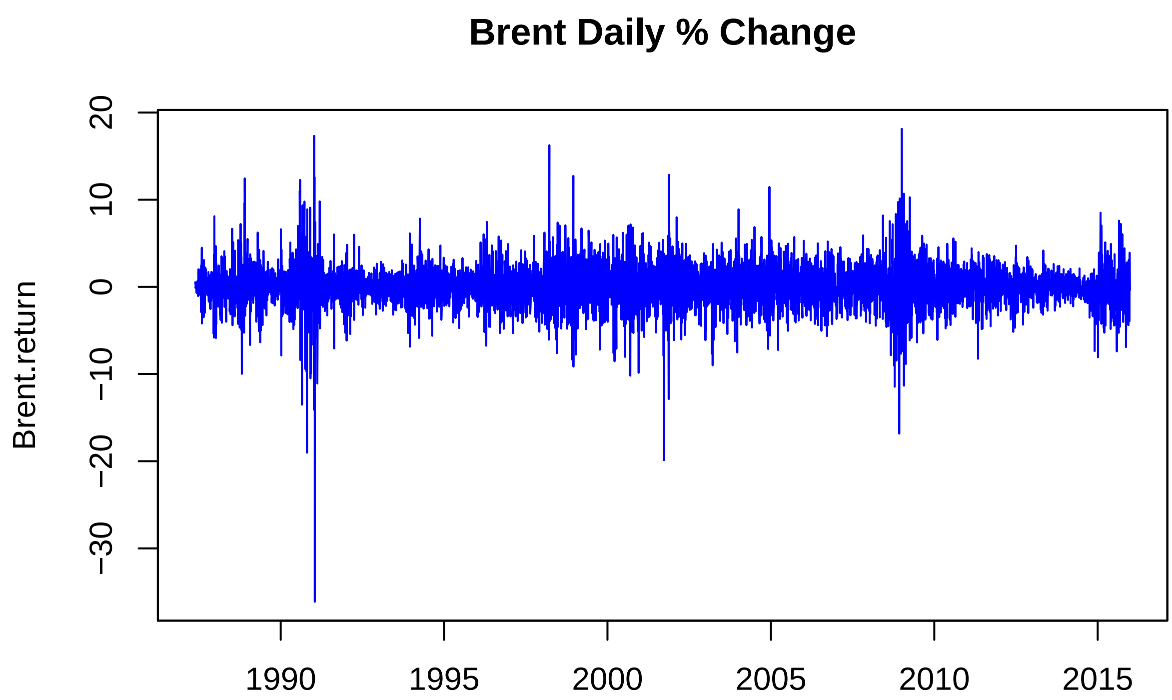
```
##            Brent.return
## 1987-05-22    0.5405419
## 1987-05-25    0.2691792
## 1987-05-26    0.1611604
## 1987-05-27   -0.1611604
## 1987-05-28    0.0000000
```

```r
tail(Brent.return, n = 5)
```

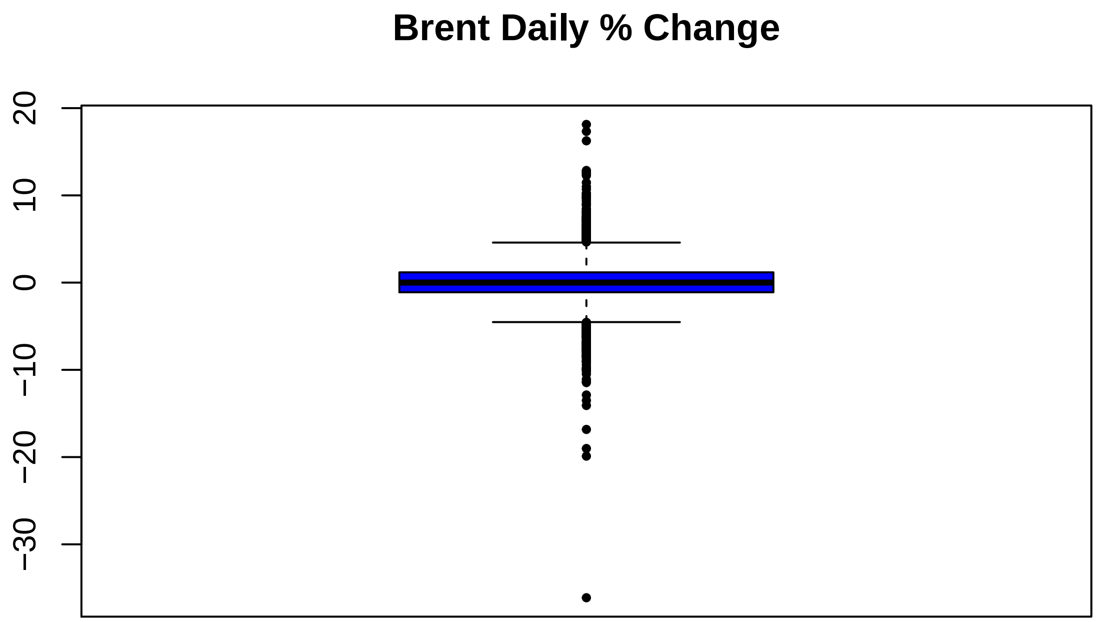
```
##            Brent.return
## 2015-12-21   -3.9394831
## 2015-12-22   -0.2266290
## 2015-12-23    1.4919348
## 2015-12-24    3.9177726
## 2015-12-28   -0.3768511
```

```r
#'
#'
#' # Try this...
#'
#' Let's look at this data with box plots and autocorrelation functions. Box plots will show minimum to
#'
#'
plot(Brent.return, title = FALSE, xlab = "", main = "Brent Daily % Change", col = "blue")
```

```
## Warning in plot.window(...): "title" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "title" is not a graphical parameter

## Warning in axis(side, at = z, labels = labels, ...): "title" is not a
## graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "title" is not
## a graphical parameter

## Warning in box(...): "title" is not a graphical parameter

## Warning in title(...): "title" is not a graphical parameter

## Warning in plot.xy(xy.coords(x, y), type = type, ...): "title" is not a
## graphical parameter
```




```
#'
#'
#' 1. Run the plot and comment.
#'
#' ***
#' Now run this:
#'
#'
boxplot(as.vector(Brent.return), title = FALSE, main = "Brent Daily % Change", col = "blue", cex = 0.5,
```



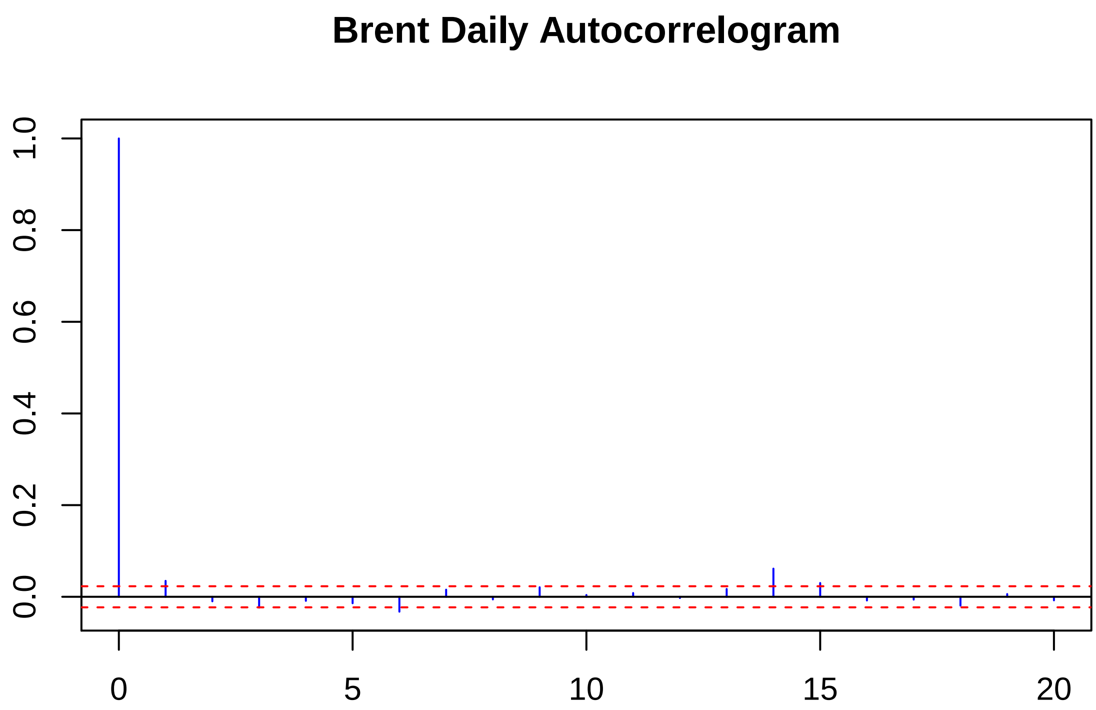

```
skewness(Brent.return)
```

```
## [1] -0.6210447
## attr(,"method")
## [1] "moment"
```

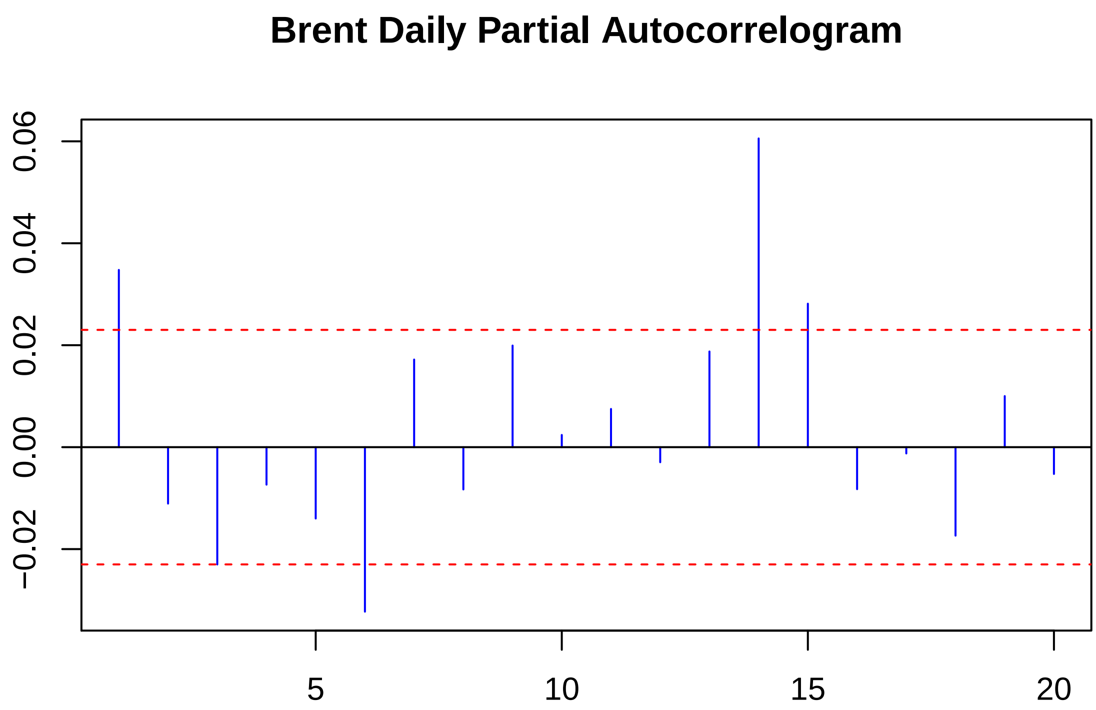
```
kurtosis(Brent.return)
```

```
## [1] 14.62226
## attr(,"method")
## [1] "excess"
```

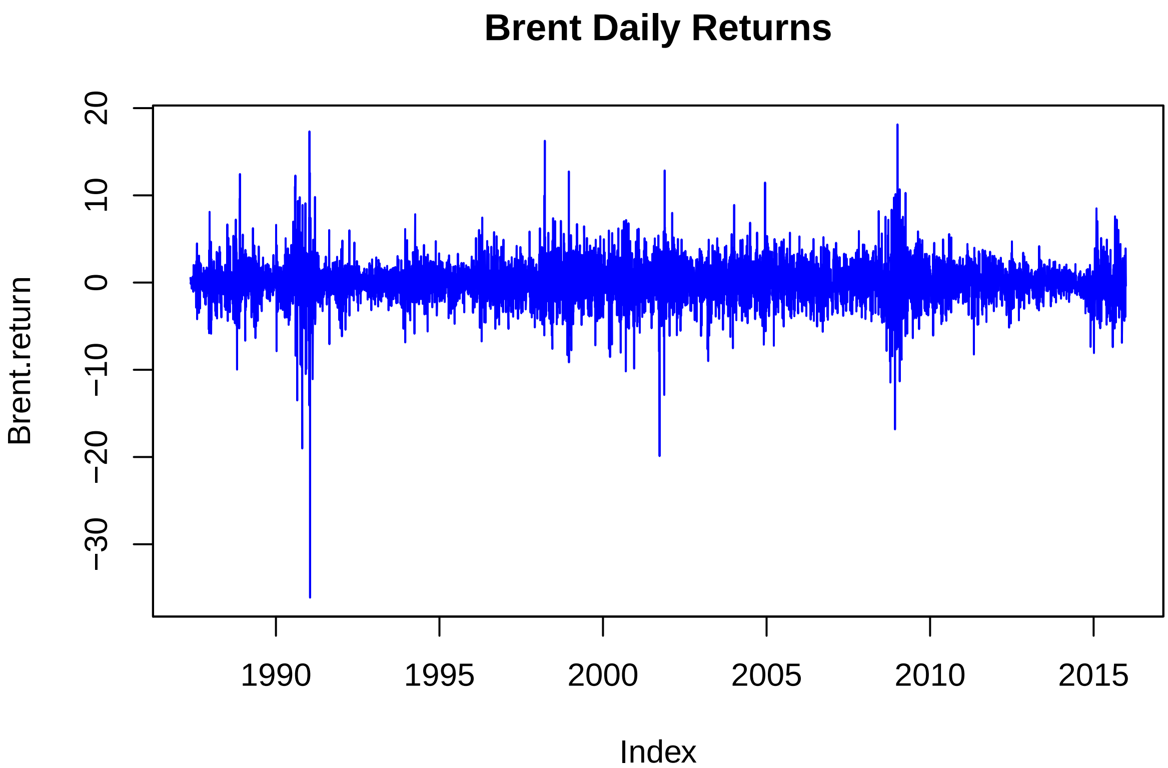
```
#'
#'
#' 2. Comment on the likelihood of positive versus negative returns. You might want to look up skewness
#'
#' ***
#' Now to look at persistence:
#'
#'
acf(coredata(Brent.return), main = "Brent Daily Autocorrelogram", lag.max = 20, ylab = "", xlab = "", c
```



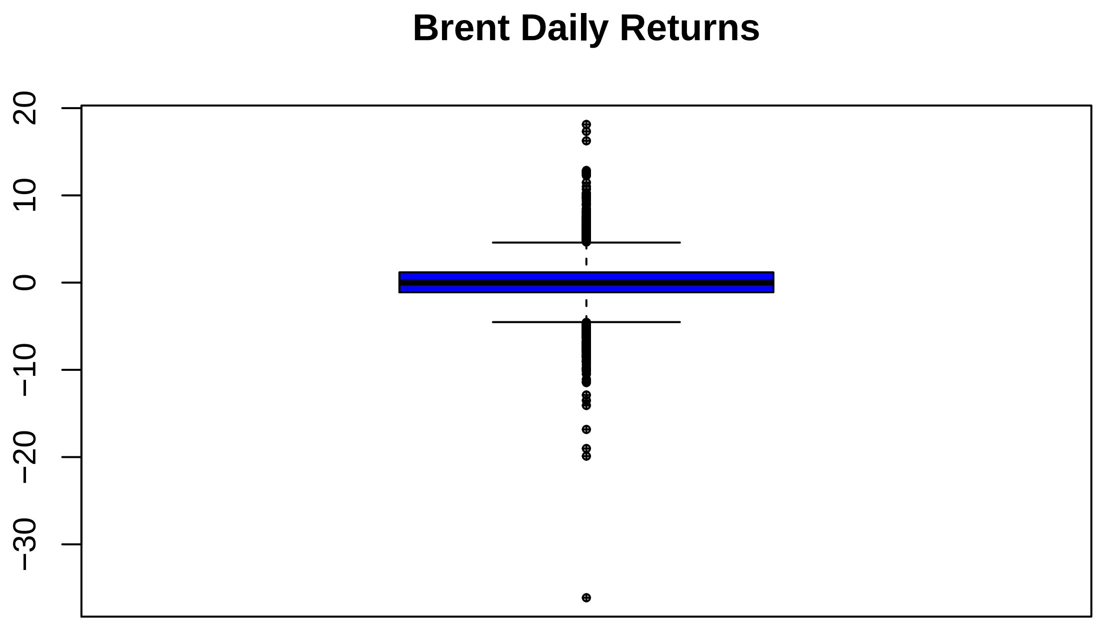

```
pacf(coredata(Brent.return), main = "Brent Daily Partial Autocorrelogram", lag.max = 20, ylab = "", xlal
```




```r
#'
#'
#' Confidence intervals are the red dashed lines. ACF at lag 6 means the correlation of current Brent r
#'
#' 3. How many trading days in a typical week or in a month? Comment on the spikes (blue lines that gro
#'
#' ***
#' Thinking...
#'
#' # We have results
#' 1. The plot
#'
#'
plot(Brent.return, main = "Brent Daily Returns", col = "blue")
#'
#'
#' ***
#'
plot(Brent.return, main = "Brent Daily Returns", col = "blue")
```




```
#'
#'
#' ***
#' This time series plot shows lots of return clustering and spikes, especially negative ones.
#'
#' ## Performing some "eyeball econometrics" these clusters seem to occur around
#' - The oil embargo of the '70s
#' - The height of the new interest rate regime of Paul Volcker at the Fed
#' - "Black Monday" stock market crash in 1987
#' - Gulf I
#' - Barings and other derivatives business collapses in the '90s
#'
#' ***
#' 2. How thick is that tail?
#'
#' Here is a first look:
#'
#'
boxplot(as.vector(Brent.return), title = FALSE, main = "Brent Daily Returns", col = "blue", cex = 0.5, p
#'
#'
#' ***
#'
boxplot(as.vector(Brent.return), title = FALSE, main = "Brent Daily Returns", col = "blue", cex = 0.5, p
```




```r
#'
#'
#' ***
#' ... with some basic stats to back up the eyeball econometrics in the box plot:
#'
#'
skewness(Brent.return)
```

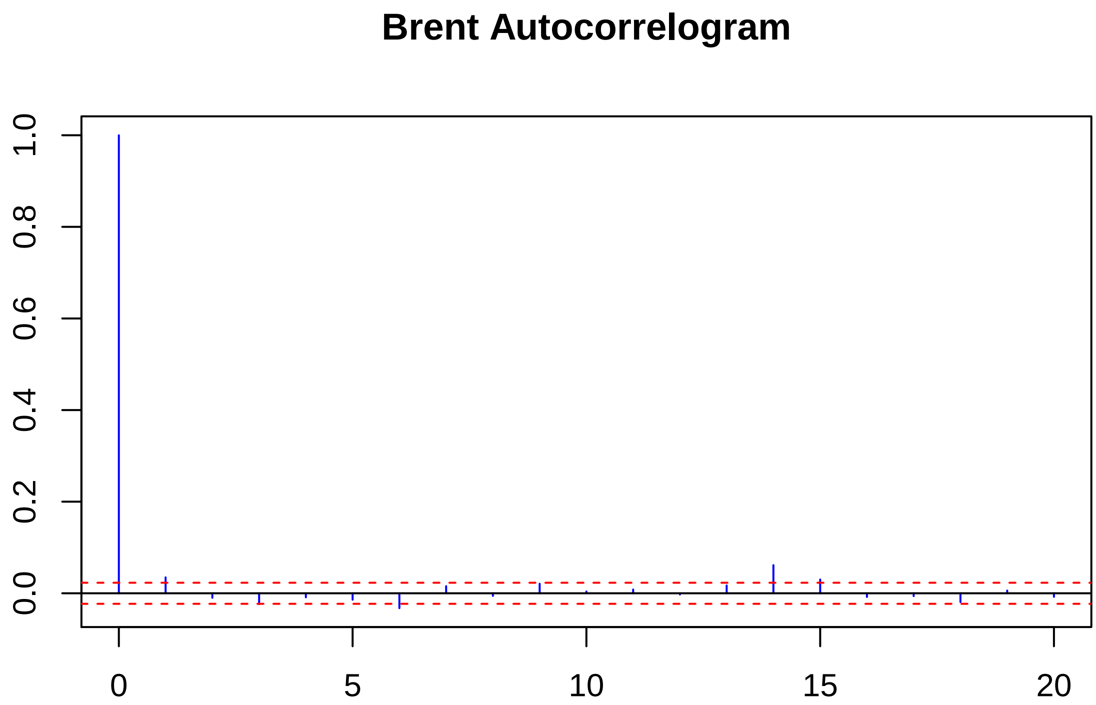
```
## [1] -0.6210447
## attr(,"method")
## [1] "moment"
```

```r
kurtosis(Brent.return)
```

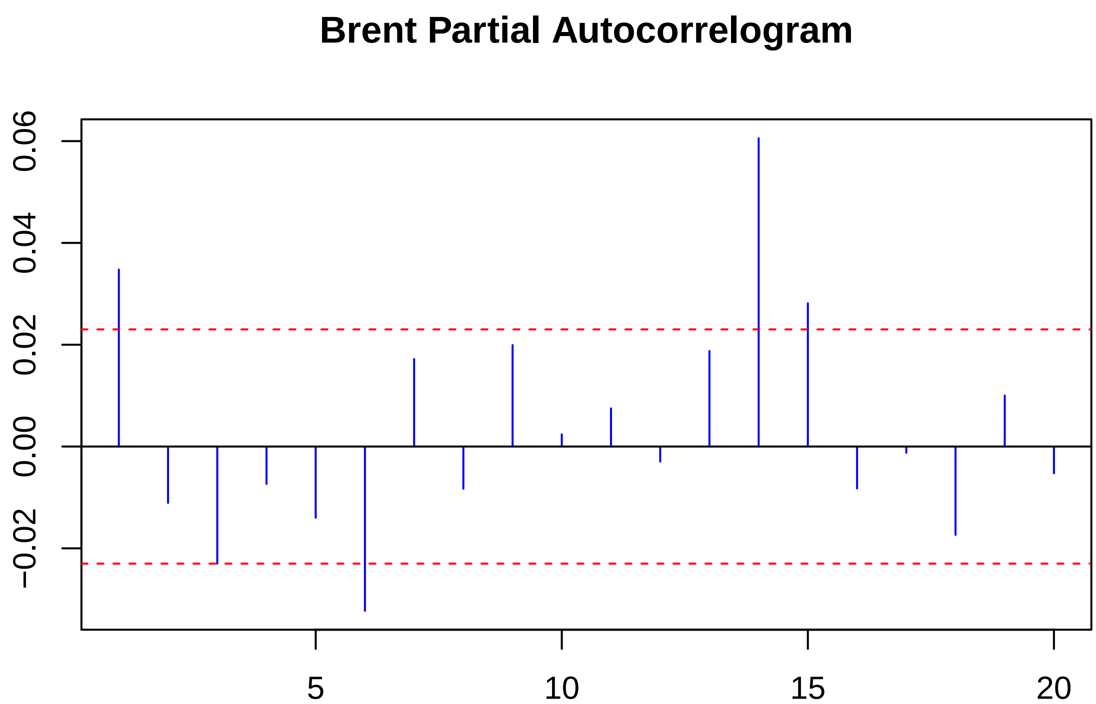
```
## [1] 14.62226
## attr(,"method")
## [1] "excess"
```

```r
#'
#'
#' - A negative skew means there are more observations less than the median than greater.
#' - This high a kurtosis means a pretty heavy tail, especially in negative returns. That means they ha
#' - A preponderance of negative returns frequently happening spells trouble for the holding of these a
#'
#' ***
#' ## Implications
#' - Budget for the body of the distribution from the mean and out to positive levels.
#' - Build a comprehensive playbook for the strong possibility that bad tail events frequently happen a
```

```r
#'
#' ***
#' 3. Now for something really interesting
#'
#'
acf(coredata(Brent.return), main = "Brent Autocorrelogram", lag.max = 20, ylab = "", xlab = "", col = "
```

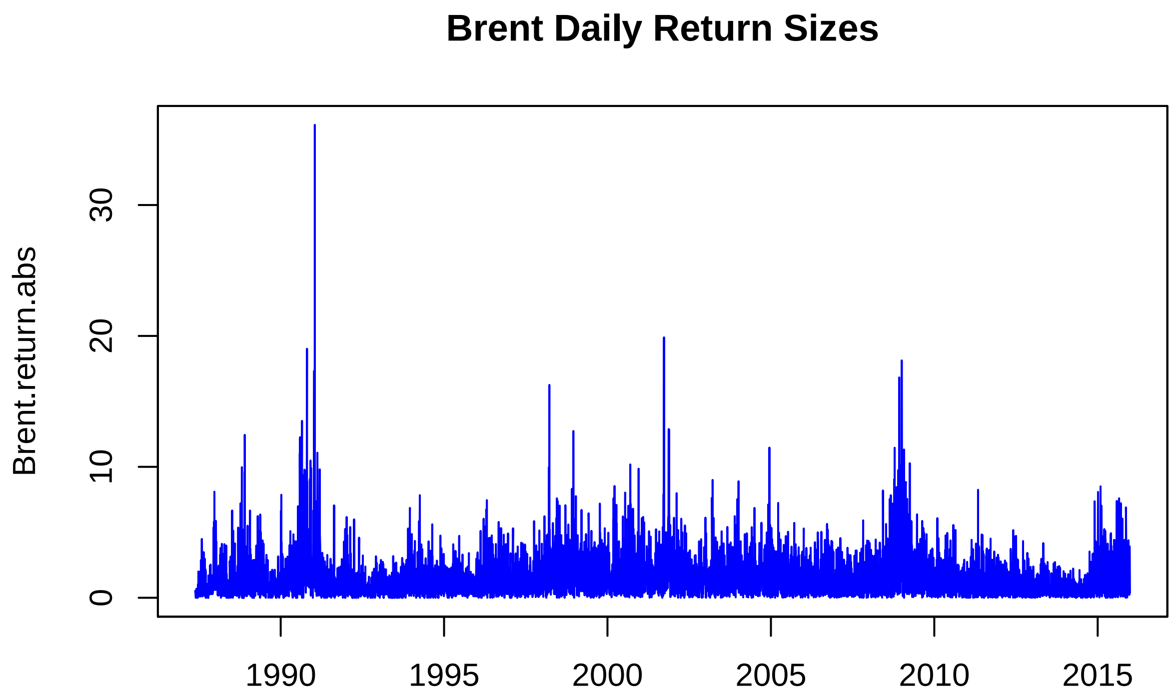



```r
#'
#'
#' ***
#'
pacf(coredata(Brent.return), main = "Brent Partial Autocorrelogram", lag.max = 20, ylab = "", xlab = ""
```



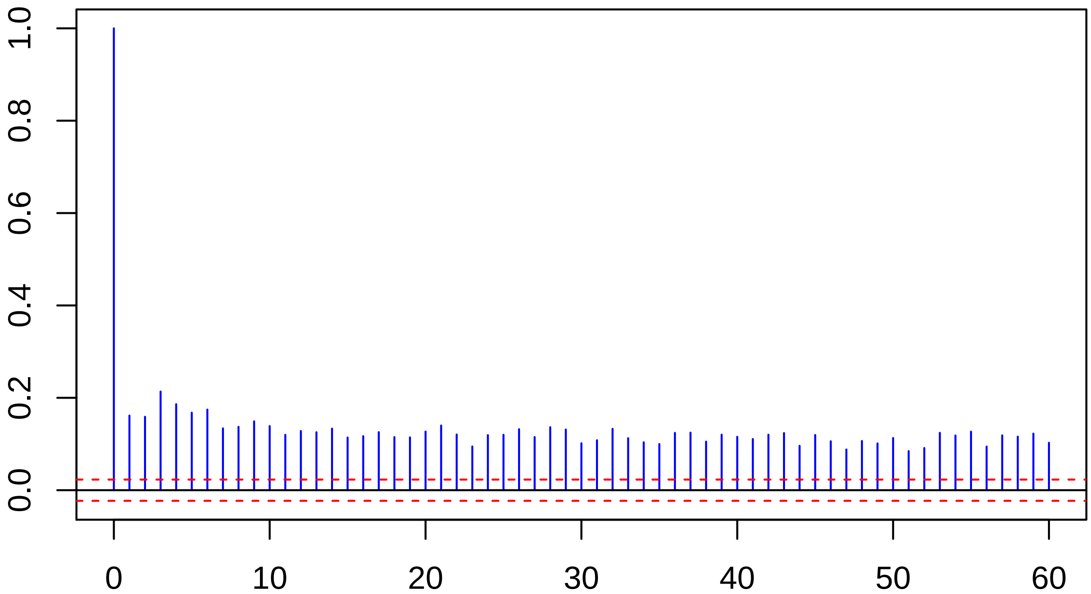

```
#'
#'
#' ***
#' On average there are 5 days in the trading week and 20 in the trading month.
#'
#' ##Some thoughts:
#' - There seems to be positive weekly and negative monthly cycles.
#' - On a weekly basis negative rates (5 trading days ago) are followed by negative rates (today) and v
#' - On a monthly basis negative rates (20 days ago) are followed by positive rates (today).
#' - There is memory in the markets: positive correlation at least weekly up to a month ago reinforces
#' - Run the PACF for 60 days to see a 40-day negative correlation as well.
#'
#' ***
#' # Now for somthing really interesting...again
#' Let's look just at the size of the Brent returns. The absolute value of the returns (think of oil an
#'
#'
Brent.return.abs <-  abs(Brent.return)
# Trading position size matters
Brent.return.tail <- tail(Brent.return.abs[order(Brent.return.abs)], 100)[1]
# Take just the first of the 100 observations and pick the first
index <- which(Brent.return.abs > Brent.return.tail, arr.ind = TRUE)
# Build an index of those sizes that exceed the heavy tail threshold
Brent.return.abs.tail <- timeSeries(rep(0, length(Brent.return)), charvec = time(Brent.return))
# just a lot of zeros we will fill up next
Brent.return.abs.tail[index, 1] <- Brent.return.abs[index]
```

```r
# A Phew! is in order
#'
#'
#' ***
#' What did we do? Let's run some plots next.
#'
#' ***
#'
plot(Brent.return.abs, xlab = "", main = "Brent Daily Return Sizes", col = "blue")
```

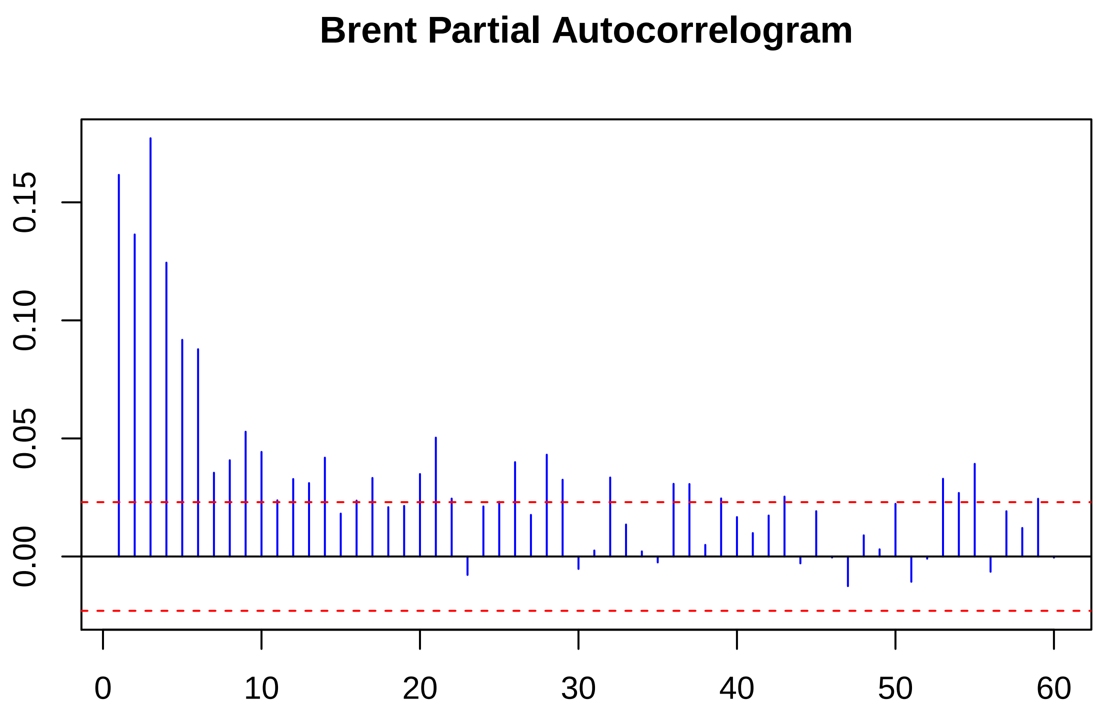



```r
#'
#'
#' ***
#' ## Lots of return volatility -- just in pure size
#' - Same event
#' - Correlated with financial innovations from the '80s and '90s
#' - Gulf 1, Gulf 2, Great Recession, and its 9/11 antecedents
#'
#' ***
#'
acf(coredata(Brent.return.abs), main = "Brent Autocorrelogram", lag.max = 60, ylab = "", xlab = "", col
```

Brent Autocorrelogram




```
#'
#'
#'
#' ***
#'
pacf(coredata(Brent.return.abs), main = "Brent Partial Autocorrelogram", lag.max = 60, ylab = "", xlab =
```

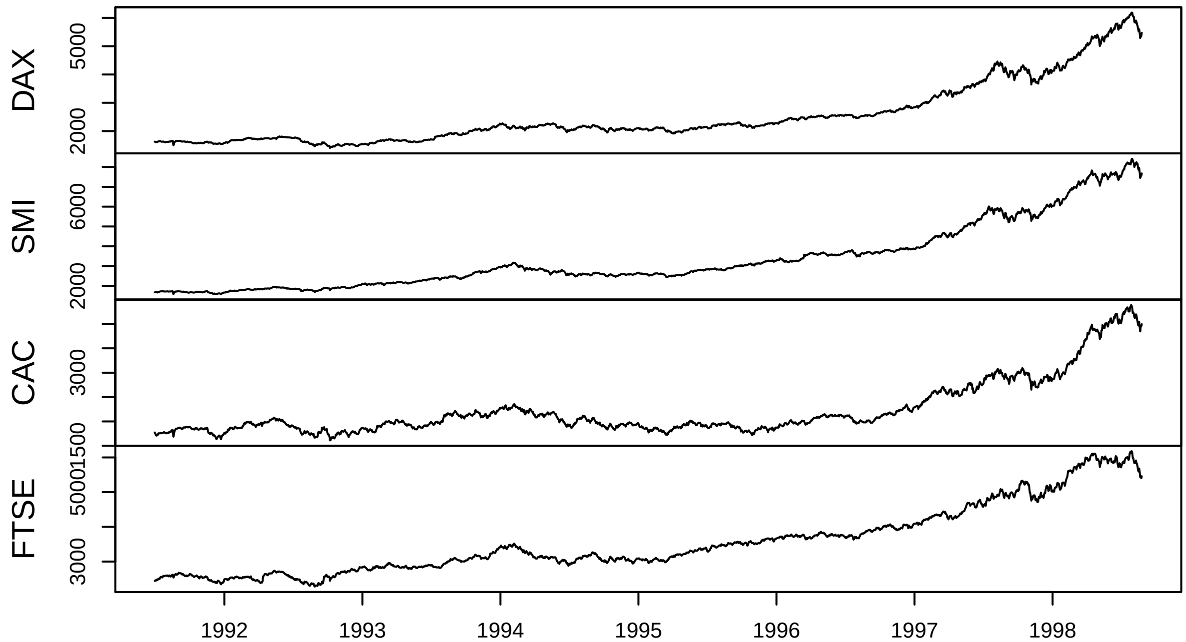



```
#'
#'
#' ***
#' ## *Volatility Clustering* galore
#' - Getting strong persistent lags of absolute movements in returns
#' - Dampening with after shocks past trading 10 days 10 ago: monthly volatility affects today's perfor
#'
#'
#'
#' Next: What are the relationships among financial variables?
#'
#' ***
#'
#'
#' # Getting caught in the cross-current
#'
#' ## Now our job is to ask the really important questions:
#' Suppose I am banking my investment in certain sectors of an economy, with its GDP, financial capabil
#'
#' ## then ...
#' - How will I decide to contract for goods and services, segment vendors, segment customers, based on
#' - How do I construct my portfolio of business opportunities?
#' - How do I identify insurgent and relational risks and build a playbook to manage these?
#' - How will changes in one sector's factors (say, finance, political will) affect factors in another?
#'
#' ***
```

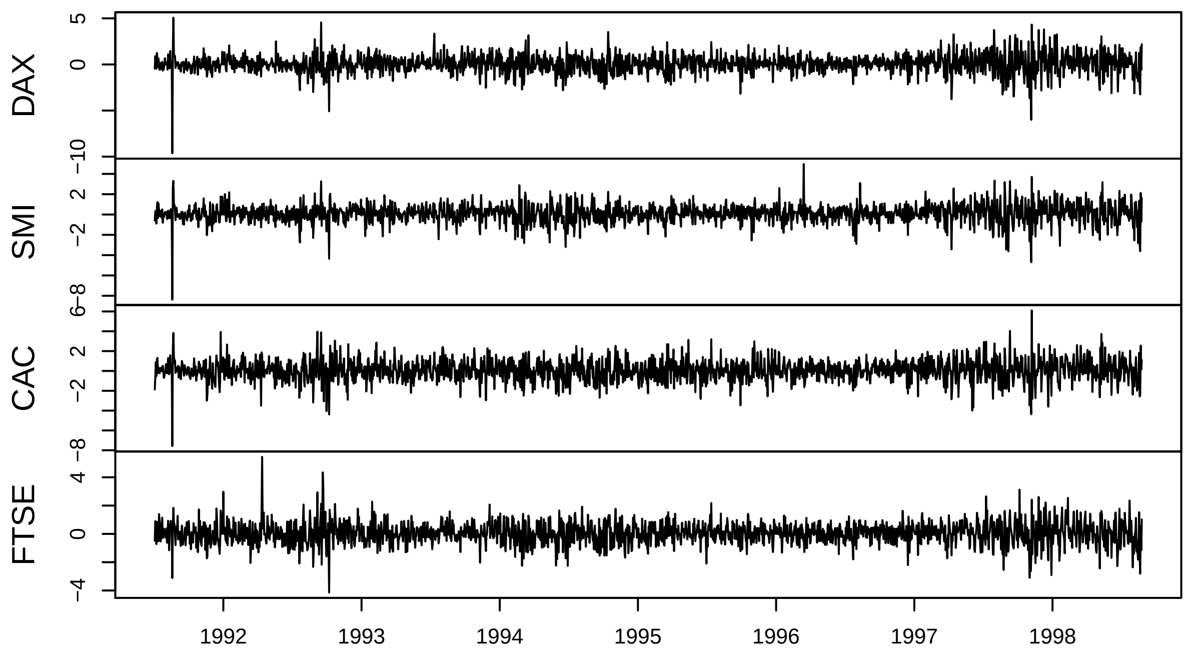
```
#' - We will now stretch out a bit and look at **cross-correlations** to help us get the ground truth a
#' - ...and _begin_ to answer some of these business questions in a more specific context.
#'
#' ***
#' Let's load the `zoo` and `qrmdata` libraries first and look at the `EuroStoxx50` data set. Here we c
#'
#' ## Our customers might be the companies based in these countries as our target market.
#' - The data: 4 stock exchange indices across Europe (and the United Kingdom)
#' - This will allow us to profile the forward capabilities of these companies across their economies.
#'
#' ***
#'
require(zoo)
require(qrmdata)
require(xts)
```

```
## Loading required package: xts
```

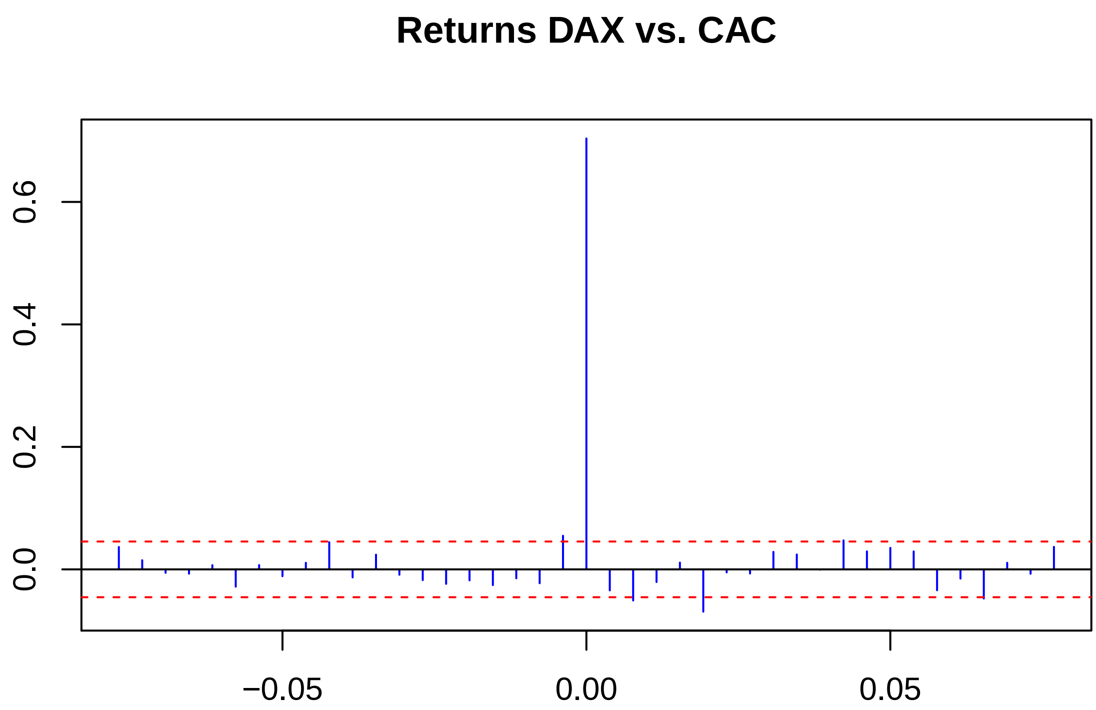
```
data("EuStockMarkets")
EuStockMarkets.price <- as.zoo(EuStockMarkets)
EuStockMarkets.return <- diff(log(EuStockMarkets.price))[-1] * 100
#'
#'
#' ***
#' Plot the levels and returns.
#'
#'
plot(EuStockMarkets.price, xlab = " ", main = " ")
#'
#'
#' ***
#'
plot(EuStockMarkets.price, xlab = " ", main = " ")
```



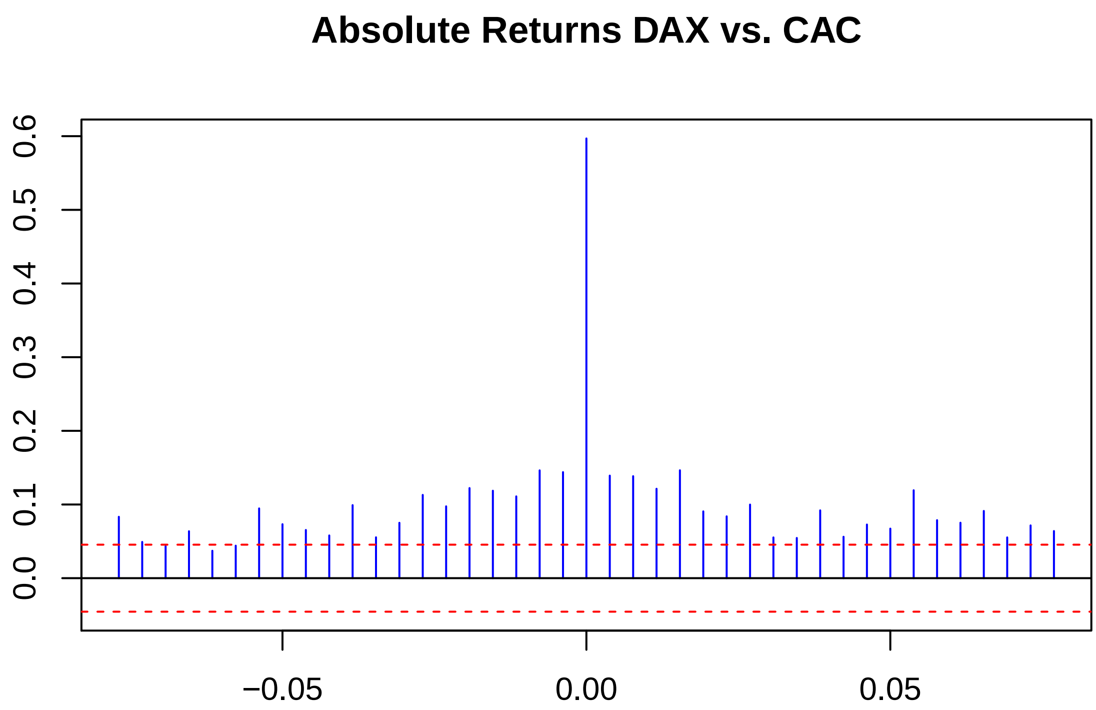

```
#'
#'
#' ***
#'
plot(EuStockMarkets.return, xlab = " ", main = " ")
```




```r
#'
#'
#' ***
#'
plot(EuStockMarkets.return, xlab = " ", main = " ")
#'
#'
#' ***
#' We see much the same thing as Brent oil with volatility clustering and heavily weighted tails.
#'
#' ***
#' Let's look at cross-correlations among one pair of these indices to see how they are related across
#'
#'
ccf(EuStockMarkets.return[, 1], EuStockMarkets.return[, 2], main = "Returns DAX vs. CAC", lag.max = 20,
#'
#'
#' ***
#'
ccf(EuStockMarkets.return[, 1], EuStockMarkets.return[, 2], main = "Returns DAX vs. CAC", lag.max = 20,
```

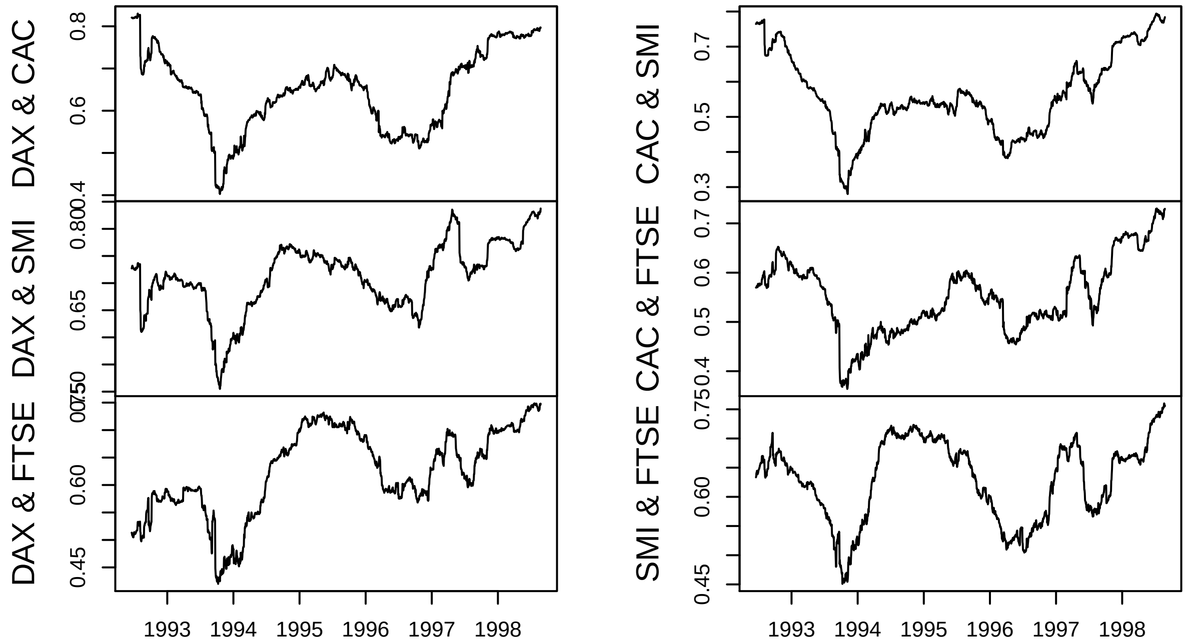



```
#'
#'
#' ***
#'
ccf(abs(EuStockMarkets.return[, 1]), abs(EuStockMarkets.return[, 2]), main = "Absolute Returns DAX vs. C
#'
#'
#' ***
#'
ccf(abs(EuStockMarkets.return[, 1]), abs(EuStockMarkets.return[, 2]), main = "Absolute Returns DAX vs. C
```




```
#'
#'
#'
#' ***
#' We see some small raw correlations across time with raw returns. More revealing, we see volatility o
#'
#'
corr.rolling <- function(x) {
  dim <- ncol(x)
  corr.r <- cor(x)[lower.tri(diag(dim), diag = FALSE)]
  return(corr.r)
}
#'
#'
#' ***
#' Embed our rolling correlation function, `corr.rolling`, into the function `rollapply` (look this one
#'
#' ***
#'
corr.returns <- rollapply(EuStockMarkets.return, width = 250, corr.rolling, align = "right", by.column =
colnames(corr.returns) <- c("DAX & CAC", "DAX & SMI", "DAX & FTSE", "CAC & SMI", "CAC & FTSE", "SMI & F
plot(corr.returns, xlab = "", main = "")
#'
#'
#' ***
#'
```

```r
corr.returns <- rollapply(EuStockMarkets.return, width = 250, corr.rolling, align = "right", by.column =
colnames(corr.returns) <- c("DAX & CAC", "DAX & SMI", "DAX & FTSE", "CAC & SMI", "CAC & FTSE", "SMI & F
plot(corr.returns, xlab = "", main = "")
```




```r
#'
#'
#' ***
#' Again look at the volatility clustering the absolute sizes of returns. Economic performance is certa
#'
#' # Try this one now ...
#'
#' Let's redo some of the work we just did using another set of techniques. This time we are using the
#'
#'
fisher <- function(r)
{0.5 * log((1 + r)/(1 - r))}
#'
#'
#' 1. What is the stated purpose of the Fisher transformation. How can it possibly help us answer our b
#'
#' ***
#'
#' 2. For three Spanish companies, Iberdrola, Endesa, and Repsol, replicate the Brent and EU stock mark
#'
#' ***
#' Thinking...
```

```r
#'
#' # Results
#' ## 1. Fisher transformations
#' - Stabilizes the variance of a variate
#' - Pulls some of the shockiness (i.e., outliers and aberrant noise) out
#' - Helps us see the forest for the trees
#'
#' ***
#' ## 2. Replicating the Brent and EU stock market experiments.
#'
#' Load some packages and get some data using `quantmod`'s `getSymbols` off the Madrid stock exchange.
#'
#'
require(xts)
require(qrmdata)
require(quantreg)
```

```
## Loading required package: quantreg

## Loading required package: SparseM

##
## Attaching package: 'SparseM'

## The following object is masked from 'package:base':
##
##     backsolve
```

```r
require(quantmod)
```

```
## Loading required package: quantmod

## Loading required package: TTR

##
## Attaching package: 'TTR'

## The following object is masked from 'package:fBasics':
##
##     volatility

## Version 0.4-0 included new data defaults. See ?getSymbols.
```

```r
require(matrixStats)
```

```
## Loading required package: matrixStats

## matrixStats v0.51.0 (2016-10-08) successfully loaded. See ?matrixStats for help.

##
## Attaching package: 'matrixStats'

## The following objects are masked from 'package:fBasics':
##
##     rowMaxs, rowMins, rowProds, rowQuantiles, rowSds, rowVars

## The following objects are masked from 'package:timeSeries':
##
##     colCummaxs, colCummins, colCumprods, colCumsums, colMaxs,
##     colMins, colProds, colQuantiles, colSds, colVars, rowCumsums
```

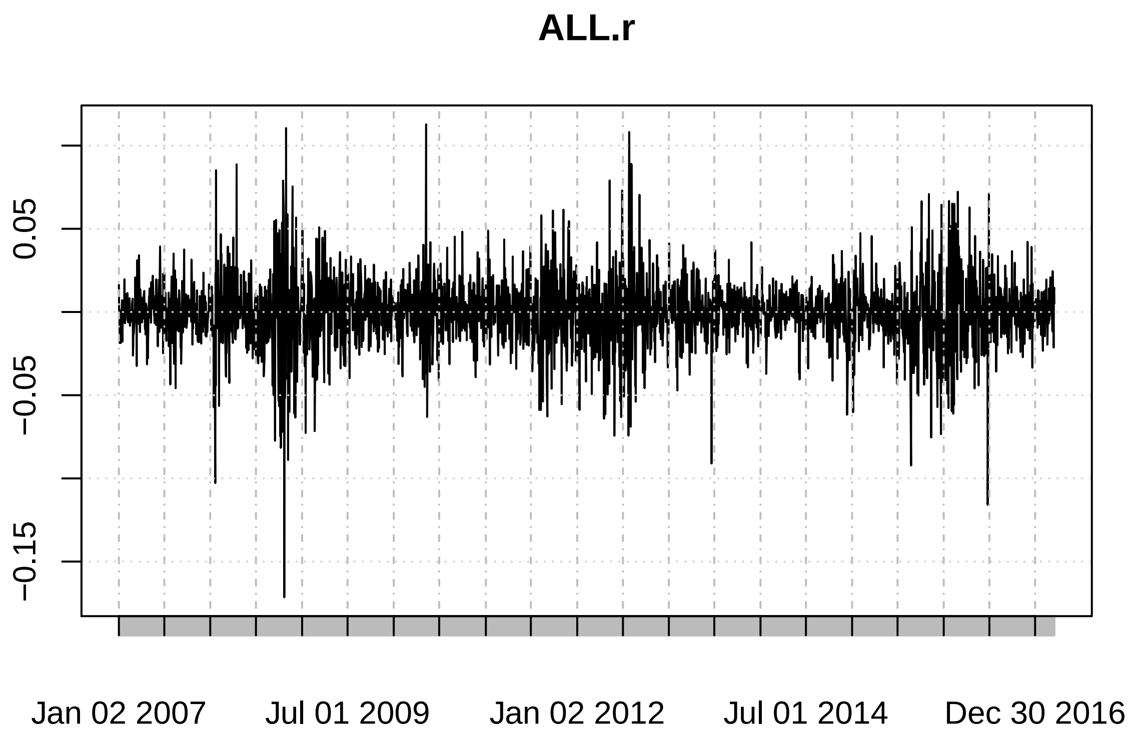
```r
tickers <- c("ELE.MC", "IBE.MC", "REP.MC")
getSymbols(tickers)
```

```
##     As of 0.4-0, 'getSymbols' uses env=parent.frame() and
##  auto.assign=TRUE by default.
##
##  This  behavior  will be  phased out in 0.5-0  when the call  will
##  default to use auto.assign=FALSE. getOption("getSymbols.env") and
##  getOptions("getSymbols.auto.assign") are now checked for alternate defaults
##
##  This message is shown once per session and may be disabled by setting
##  options("getSymbols.warning4.0"=FALSE). See ?getSymbols for more details.
```

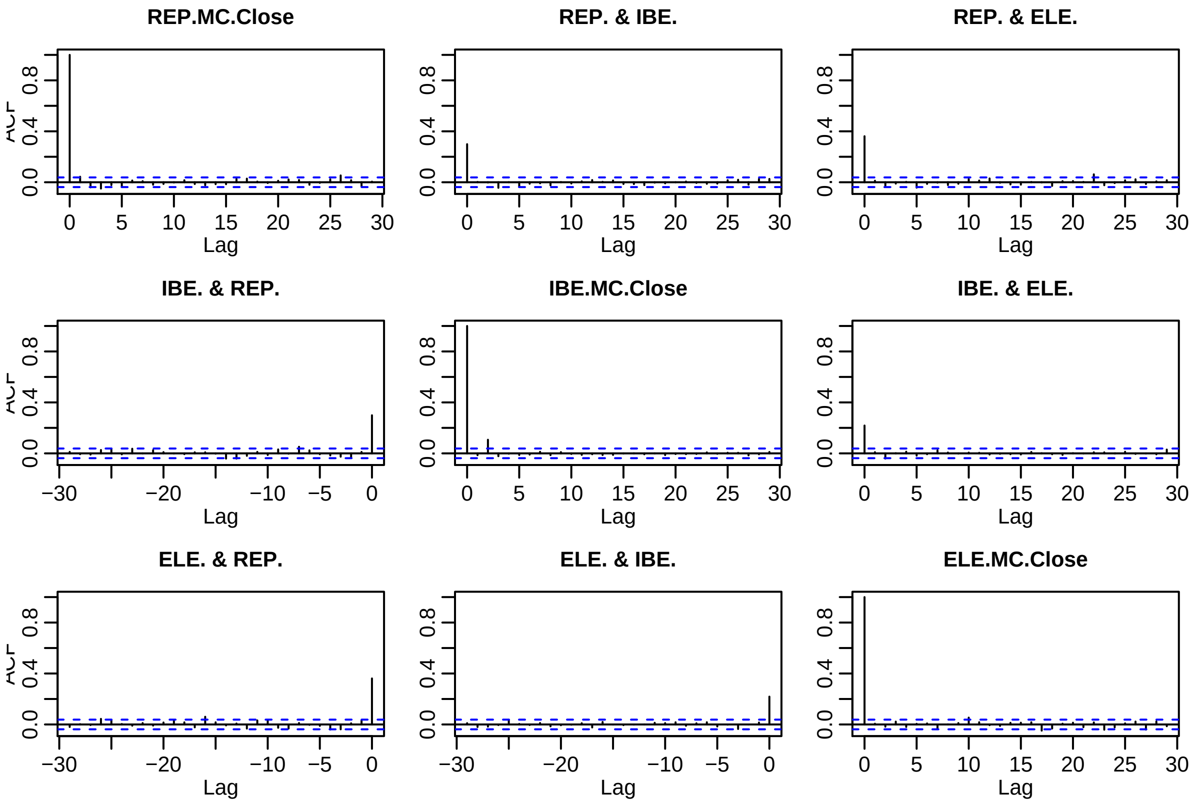
```
## [1] "ELE.MC" "IBE.MC" "REP.MC"
```

```r
REP.r <- diff(log(REP.MC[, 4]))[-1]
IBE.r <- diff(log(IBE.MC[, 4]))[-1]
ELE.r <- diff(log(ELE.MC[, 4]))[-1]

ALL.r <- merge(REP = REP.r, IBE = IBE.r, ELE = ELE.r, all = FALSE)
#'
#'
#' ***
#' Next plot the returns and their absolute values, acf and pacf, all like we did in Brent.
#'
#' ## Notice
#' 1. The persistence of returns
#' 2. The importance of return size
#' 3. Clustering of volatility
#'
#' ***
#'
plot(ALL.r)
```

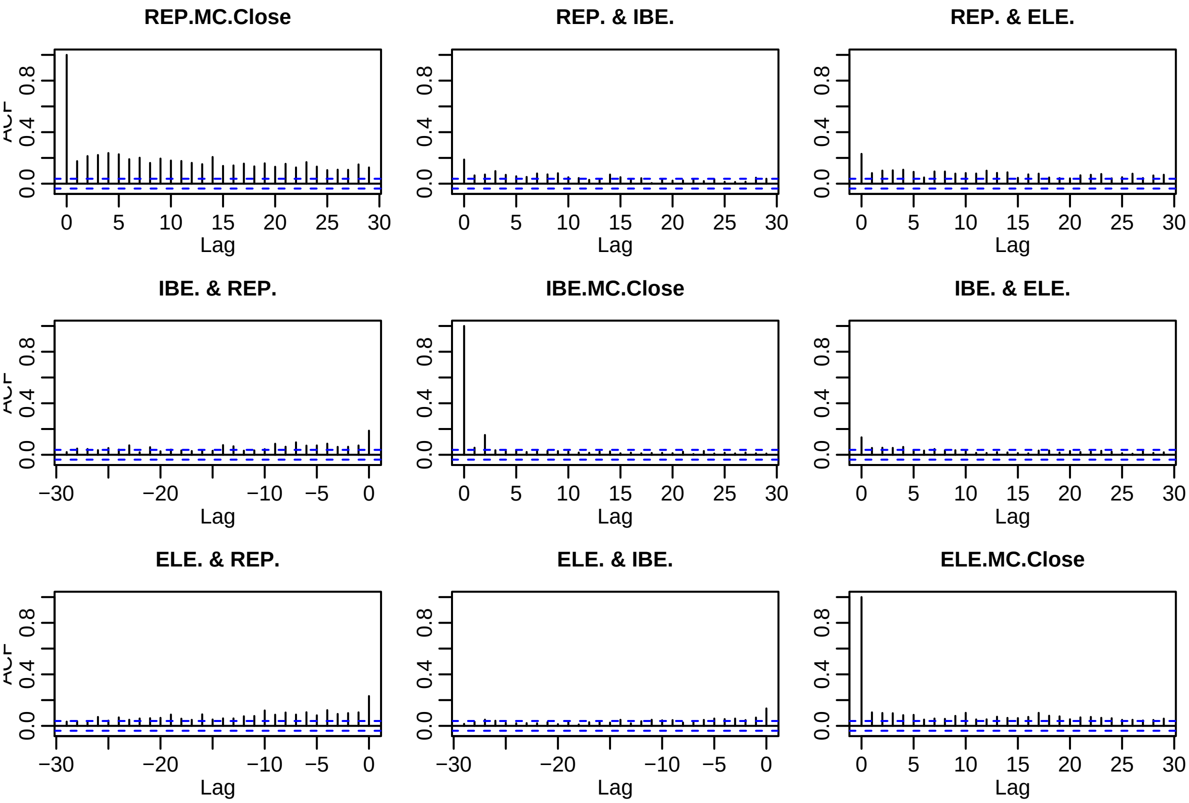
```
## Warning in plot.xts(ALL.r): only the univariate series will be plotted
```



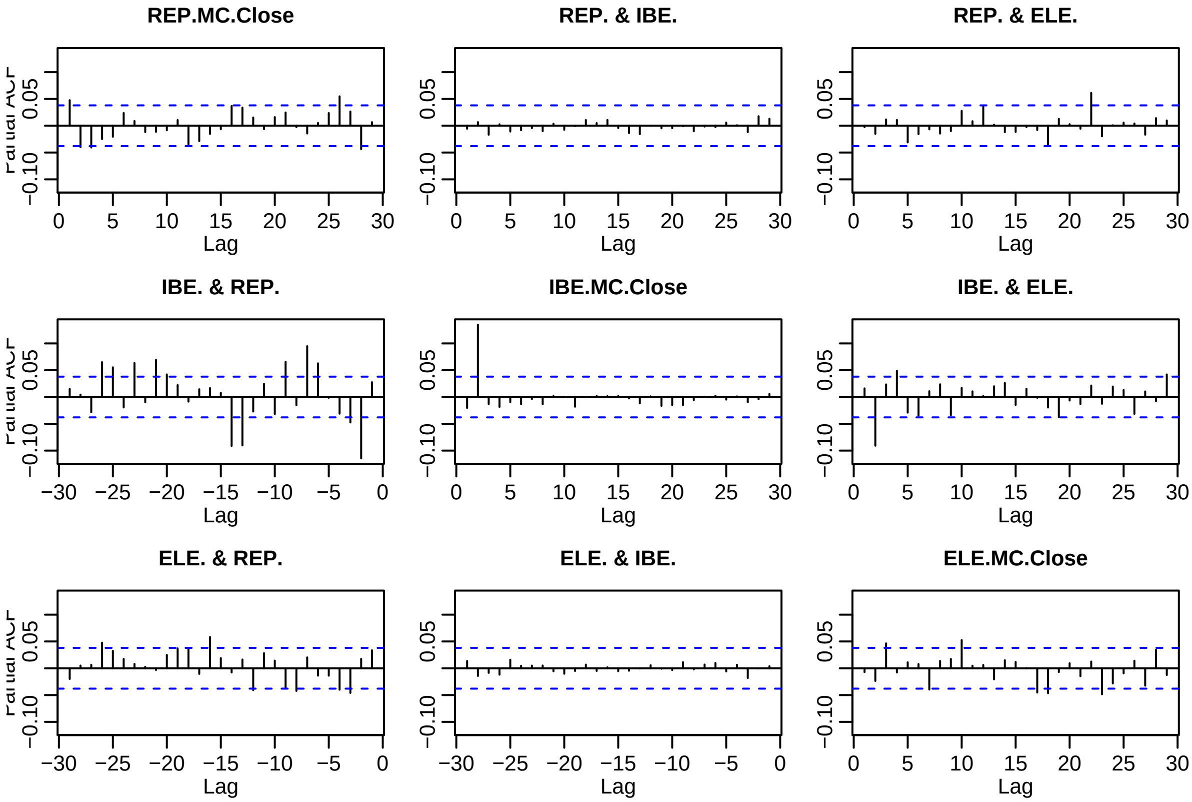

```r
#'
#'
#' ***
#'
par(mfrow = c(2,1))
acf(ALL.r)
```



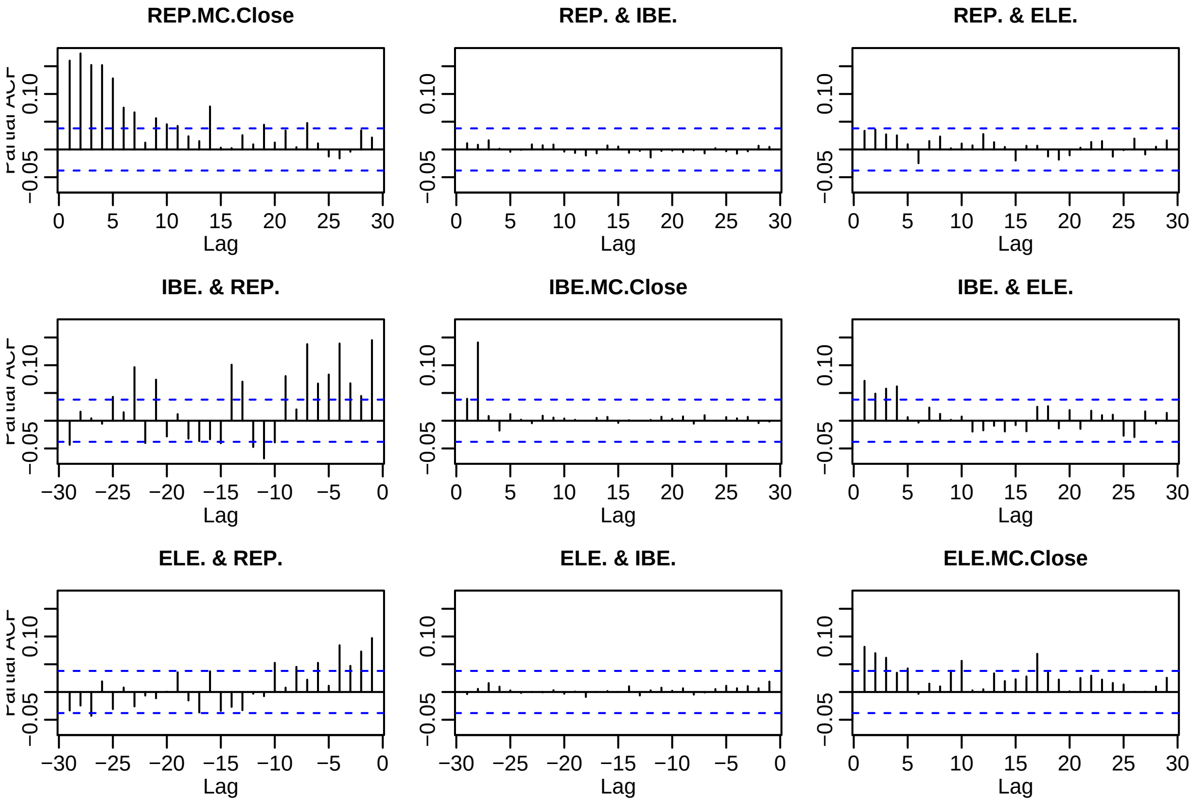

```
#'
#'
#' ***
#'
par(mfrow = c(2,1))
acf(abs(ALL.r))
```




```r
#'
#'
#'
#' ***
#'
par(mfrow = c(2,1))
pacf(ALL.r)
```




```r
#'
#'
#' ***
#'
par(mfrow = c(2,1))
pacf(abs(ALL.r))
```




```r
#'
#'
#'
#' ***
#' Now to examine the correlation structure of markets.
#'
#' ## Notice
#' 1. The relationship between correlation and volatility
#' 2. How quantile regression gets us to an understanding of high stress (high and low quantile) episod
#'
#' ***
#'
R.corr <- apply.monthly(ALL.r, FUN = cor)
R.vols <- apply.monthly(ALL.r, FUN = colSds) # from MatrixStats
head(R.corr, 3)
```

```
##             [,1]      [,2]        [,3]      [,4] [,5]       [,6]
## 2007-01-31    1 0.3613898 -0.27543509 0.3613898    1 0.10413703
## 2007-02-28    1 0.5662019 -0.09854683 0.5662019    1 0.10760374
## 2007-03-30    1 0.4500958 -0.08873714 0.4500958    1 0.08537969
##                   [,7]       [,8] [,9]
## 2007-01-31 -0.27543509 0.10413703    1
## 2007-02-28 -0.09854683 0.10760374    1
## 2007-03-30 -0.08873714 0.08537969    1
```

```r
head(R.vols, 3)
```

```
##            REP.MC.Close IBE.MC.Close ELE.MC.Close
## 2007-01-31  0.009787612  0.007892759  0.009777421
## 2007-02-28  0.009181144  0.014571945  0.007674825
## 2007-03-30  0.015317317  0.012719792  0.010919166
```

```r
#'
#'
#' ***
#'
R.corr.1 <- matrix(R.corr[1,], nrow = 3, ncol = 3, byrow = FALSE)
rownames(R.corr.1) <- tickers
colnames(R.corr.1) <- tickers
head(R.corr.1)
```

```
##            ELE.MC    IBE.MC     REP.MC
## ELE.MC  1.0000000 0.3613898 -0.2754351
## IBE.MC  0.3613898 1.0000000  0.1041370
## REP.MC -0.2754351 0.1041370  1.0000000
```

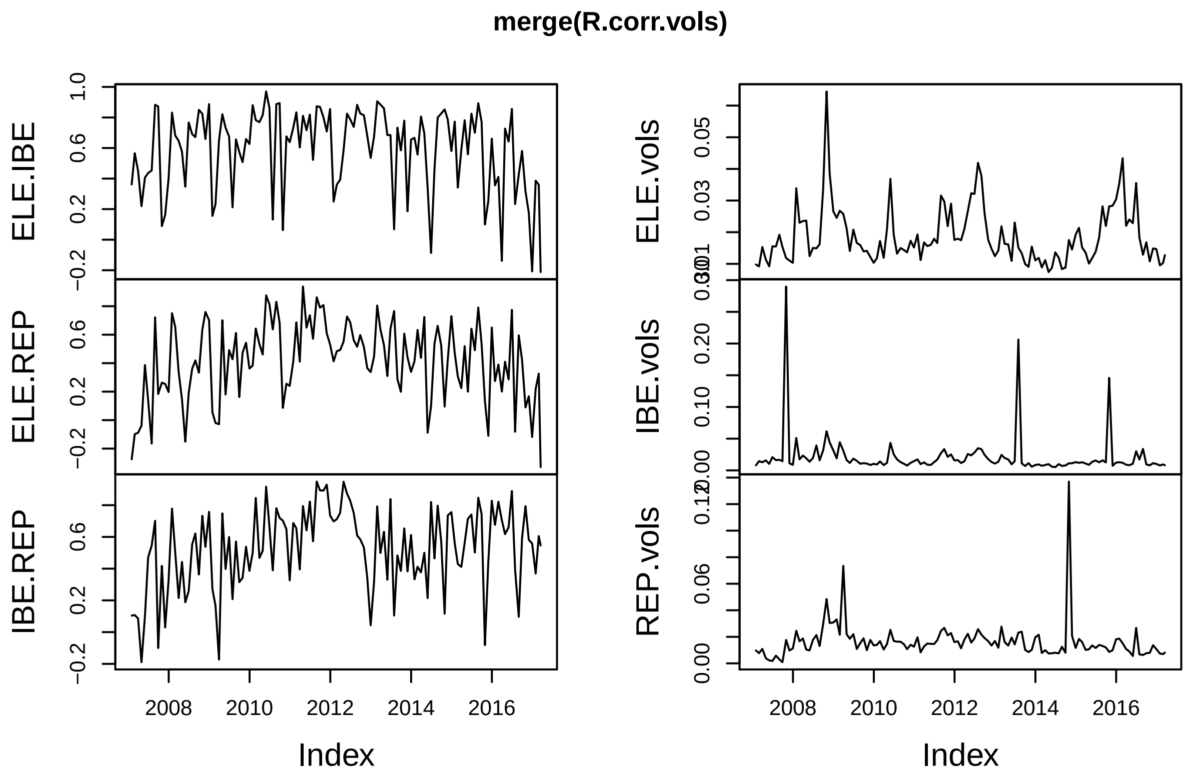
```r
#'
#'
#' ***
#'
R.corr <- R.corr[, c(2, 3, 6)]
colnames(R.corr) <- c("ELE.IBE", "ELE.REP", "IBE.REP")
colnames(R.vols) <- c("ELE.vols", "IBE.vols", "REP.vols")
head(R.corr, 3)
```

```
##              ELE.IBE     ELE.REP    IBE.REP
## 2007-01-31 0.3613898 -0.27543509 0.10413703
## 2007-02-28 0.5662019 -0.09854683 0.10760374
## 2007-03-30 0.4500958 -0.08873714 0.08537969
```

```r
head(R.vols, 3)
```

```
##               ELE.vols    IBE.vols    REP.vols
## 2007-01-31 0.009787612 0.007892759 0.009777421
## 2007-02-28 0.009181144 0.014571945 0.007674825
## 2007-03-30 0.015317317 0.012719792 0.010919166
```

```r
R.corr.vols <- merge(R.corr, R.vols)
#'
#'
#' ***
#'
plot.zoo(merge(R.corr.vols))
```

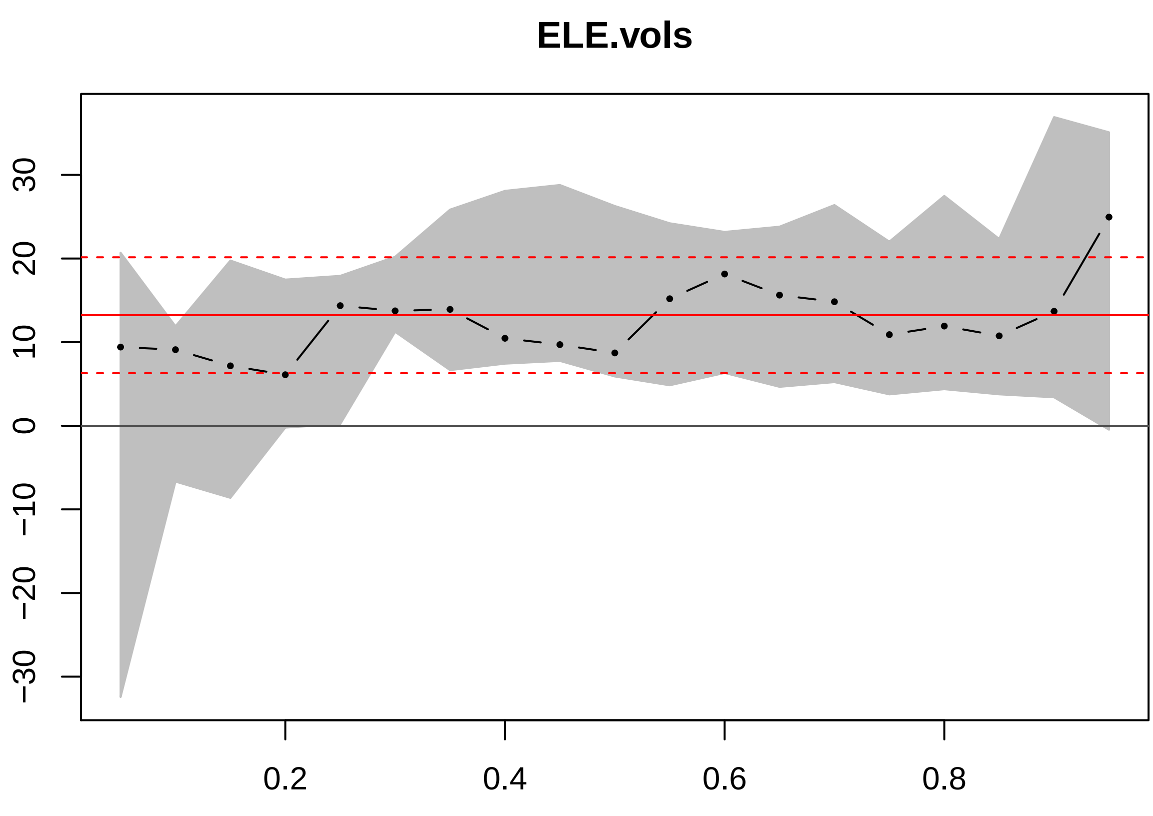



```
#'
#'
#' ***
#'
ELE.vols <- as.numeric(R.corr.vols[,"ELE.vols"])
IBE.vols <- as.numeric(R.vols[,"IBE.vols"])
REP.vols <- as.numeric(R.vols[,"REP.vols"])
length(ELE.vols)
```

```
## [1] 123
```

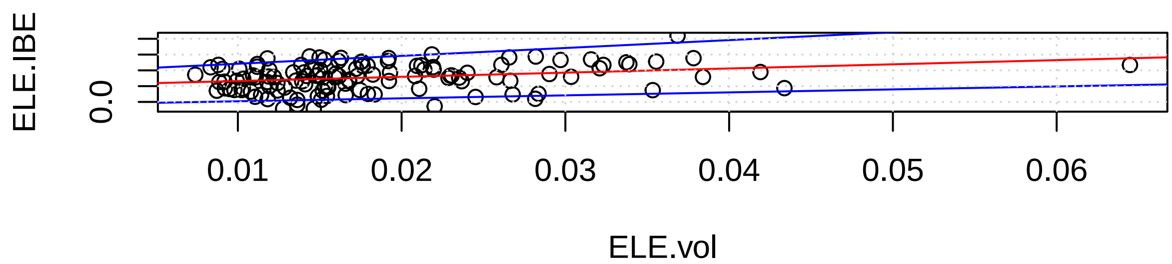
```
#'
#'
#' ***
#'
fisher <- function(r)
{0.5 * log((1 + r)/(1 - r))}
rho.fisher <- matrix(fisher(as.numeric(R.corr.vols[,1:3])), nrow = length(ELE.vols), ncol = 3, byrow= F
#'
#'
#' ***
#' Here is the quantile regression part of the package.
#'
#' ## Notice
#' 1. We set `taus` as the quantiles of interest.
#' 2. We run the quantile regression using the `quantreg` package and a call to the `rq` function.
#' 3. We can overlay the quantile regression results onto the standard linear model regression.
```

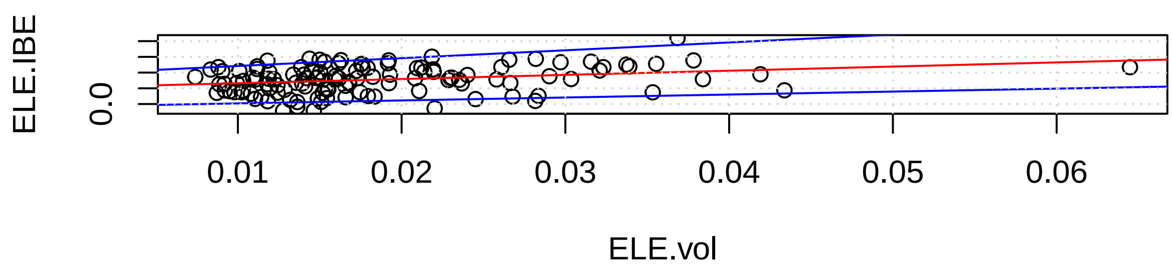
```r
#' 4. We can sensitize our analysis with the range of upper and lower bounds on the parameter estimates
#'
#'
taus <- seq(.05,.95,.05)
fit.rq.ELE.IBE <- rq(rho.fisher[,1] ~ ELE.vols, tau = taus)
fit.lm.ELE.IBE <- lm(rho.fisher[,1] ~ ELE.vols)
#'
#'
#' ***
#'
plot(summary(fit.rq.ELE.IBE), parm = "ELE.vols")
```




```r
#'
#'
#' ***
#' Here we build the estimations and plot the upper and lower bounds.
#'
#'
taus1 <- c(.05, .95) # fit the confidence interval (CI)
plot(ELE.vols,rho.fisher[, 1], xlab="ELE.vol", ylab="ELE.IBE")
abline(fit.lm.ELE.IBE, col = "red")
for (i in 1:length(taus1)){ # these lines will be the CI
    abline(rq(rho.fisher[,1] ~ ELE.vols, tau = taus1[i]), col = "blue")
}
grid()
#'
```

```r
#'
#' ***
#'
taus1 <- c(.05, .95) # fit the confidence interval (CI)
plot(ELE.vols,rho.fisher[, 1], xlab="ELE.vol", ylab="ELE.IBE")
abline(fit.lm.ELE.IBE, col = "red")
for (i in 1:length(taus1)){ # these lines will be the CI
    abline(rq(rho.fisher[,1] ~ ELE.vols, tau = taus1[i]), col = "blue")
}
grid()
```



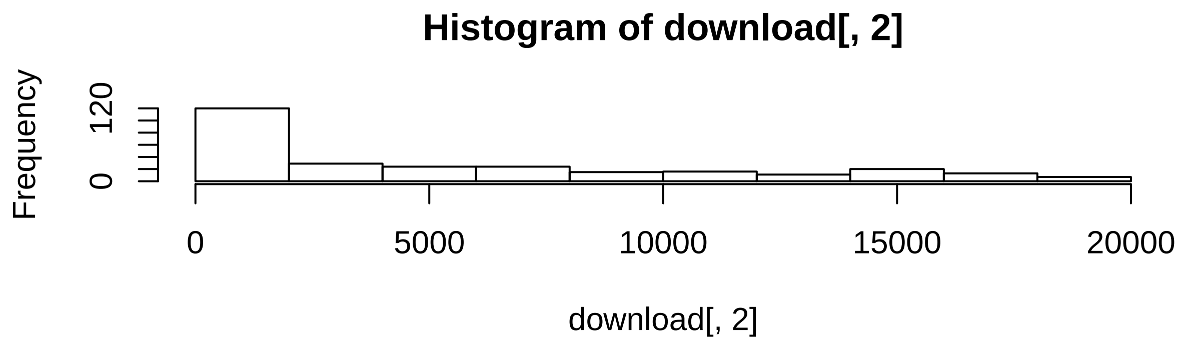


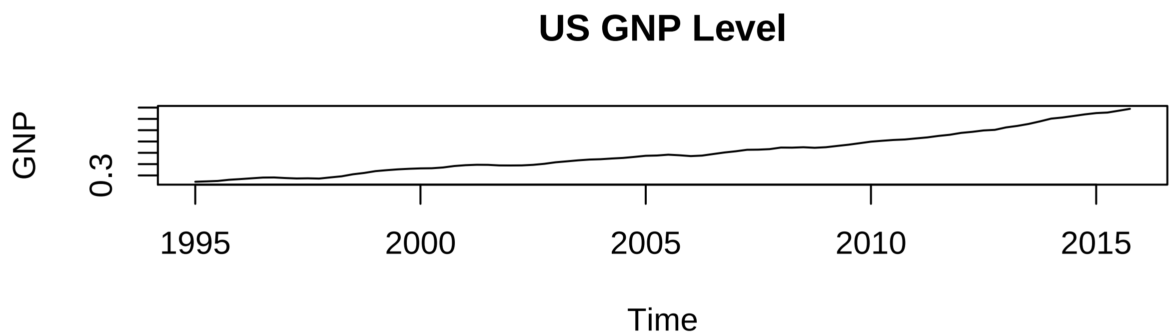

```r
#'
#'
#' ***
#' # Bounding our enthusiasm
#' 1. Quantile regression helps us to see the upper and lower bounds.
#' 2. Relationships between high-stress periods and correlation are abundant.
#' 3. These markets simply reflect normal buying behaviors across many types of exchanges: buying food
#'
#' ***
#'
#'
#' # Time is on our side...
#'
#' Let's start with some US Gross National Product (GNP) data from the St. Louis Fed's open data websit
#'
```

```r
#'
name <-  "GNP"
URL <-  paste("http://research.stlouisfed.org/fred2/series/", name,
"/", "downloaddata/", name, ".csv", sep = "")
download <-  read.csv(URL)
#'
#' ***
#' Look at the data:
#'
#'
hist(download[,2])
#'
#'
#' ***
#'
#'
summary(download[, 2])
```

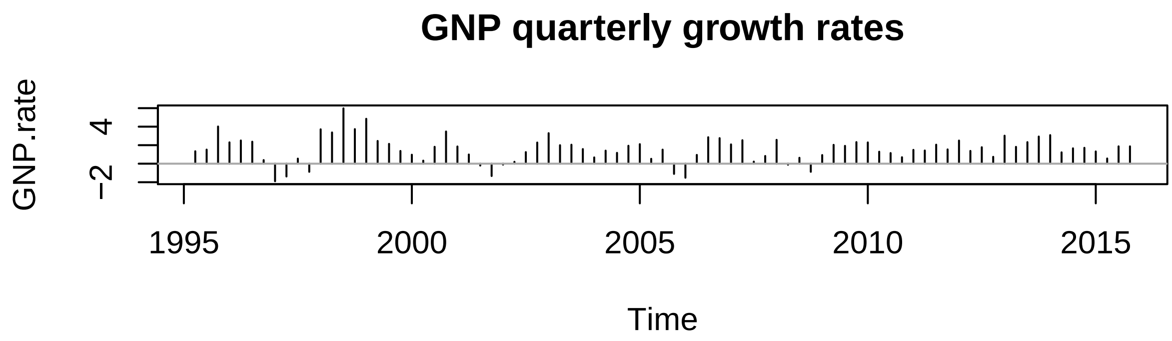
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   244.1   691.8  3310.0  5583.0  9527.0 18880.0
```

```r
#'
#'
#' ***
#' Create a raw time series object (rownames are dates...), select some data, and calculate growth rate
#'
#'
GNP <- ts(download[1:84, 2]/1000, start = c(1995, 1), freq = 4)
GNP.rate = 100 * diff(log(GNP))
#'
#'
#' # Try this ...
#' 1. Plot the GNP level and rate.
#' 2. Comment on the patterns.
#'
#' ***
#' Thinking...
#'
#' # Results
#'
plot(GNP, type = "l", main = "US GNP Level")
```

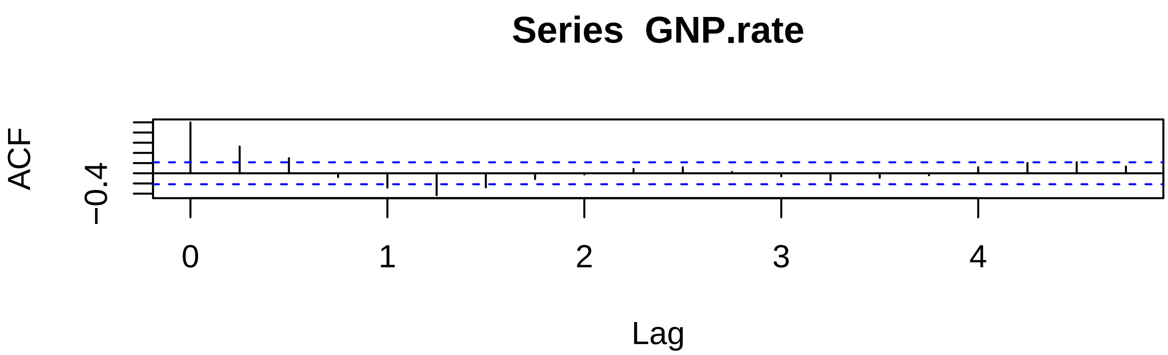


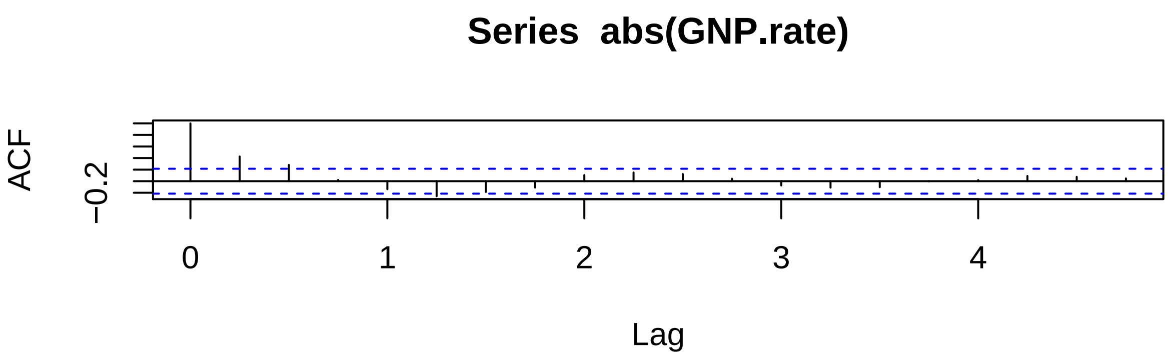



```
#'
#'
#' ***
#'
plot(GNP.rate, type = "h", main = "GNP quarterly growth rates")
abline(h = 0, col = "darkgray")
#'
#'
#' ***
#' ## What we call "nonstationary"
#' 1. The probability distribution (think `hist()`) would seem to change over time.
#' 2. This means that the standard deviation and mean changes as well.
#' 3. Lots of trend in the level and simply dampened sinusoidal in the rate.
#'
#' ## Can we forecast GNP?
#'
#' # Forecasting GNP
#'
#' As always let's look at ACF and PACF:
#'
#'
par(mfrow = c(2,1)) #stacked up and down
```

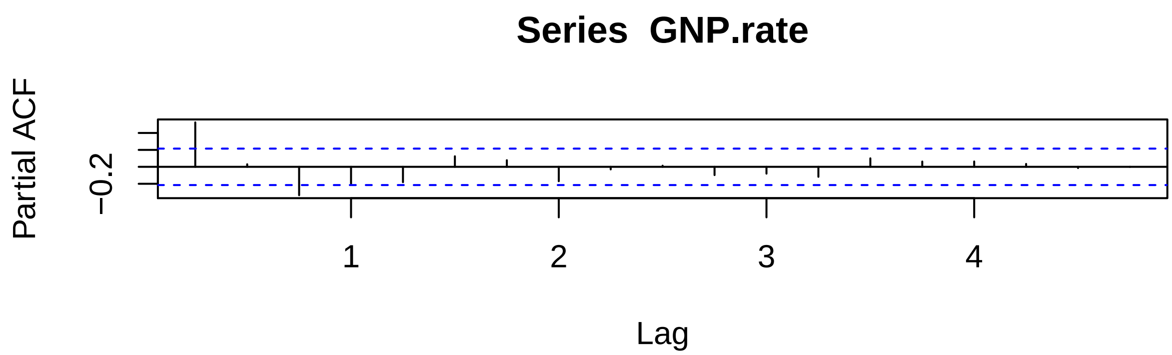


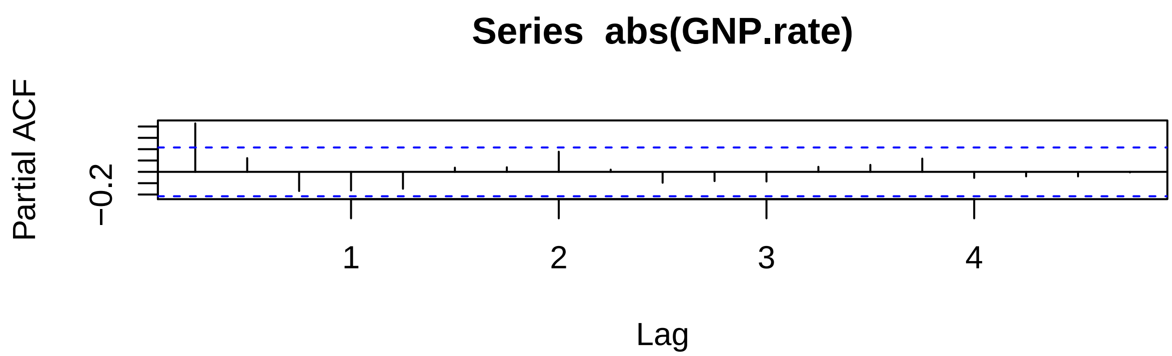

```
acf(GNP.rate)
acf(abs(GNP.rate))
#'
#'
#' ***
#'
par(mfrow = c(2,1)) #stacked up and down
acf(GNP.rate)
acf(abs(GNP.rate))
```






```
#'
#'
#'
#' # Try this...
#'
par(mfrow = c(2,1))
pacf(GNP.rate)
pacf(abs(GNP.rate))
par(mfrow = c(1,1)) #default setting
#'
#' What do you think is going on?
#'
#' ***
#' Thinking...
#'
#' # Result
#'
par(mfrow = c(2,1))
pacf(GNP.rate)
pacf(abs(GNP.rate))
```






```
par(mfrow = c(1,1)) #default setting
#'
#'
#' ***
#' ## What do you think?
#' - There are several significant autocorrelations within the last 4 quarters.
#' - Partial autocorrelation also indicates some possible relationship 8 quarters back.
#'
#' # Yet another regression (YAR)...
#' Let's use `R`'s time series estimation tool `arima`. We think there is a regression that looks like
#'
#' \[
#' x_t = a_0 + a_1 x_{t-1} ... a_p x_{t-p} + b_1 \epsilon_{t-1} + ... + b_q \epsilon_{t-q}
#' \]
#'
#' where $x_t$ is a first, $d = 1$, differenced level of a variable, here GNP. There are $p$ lags of th
#'
#' ***
#' Estimation is quick and easy.
#'
#'
fit.rate <- arima(GNP.rate, order = c(2, 0, 1))
#'
#'
#' The order is 2 lags of rates, 0 further differencing (already differenced once), and 1 lag of residu
#'
```

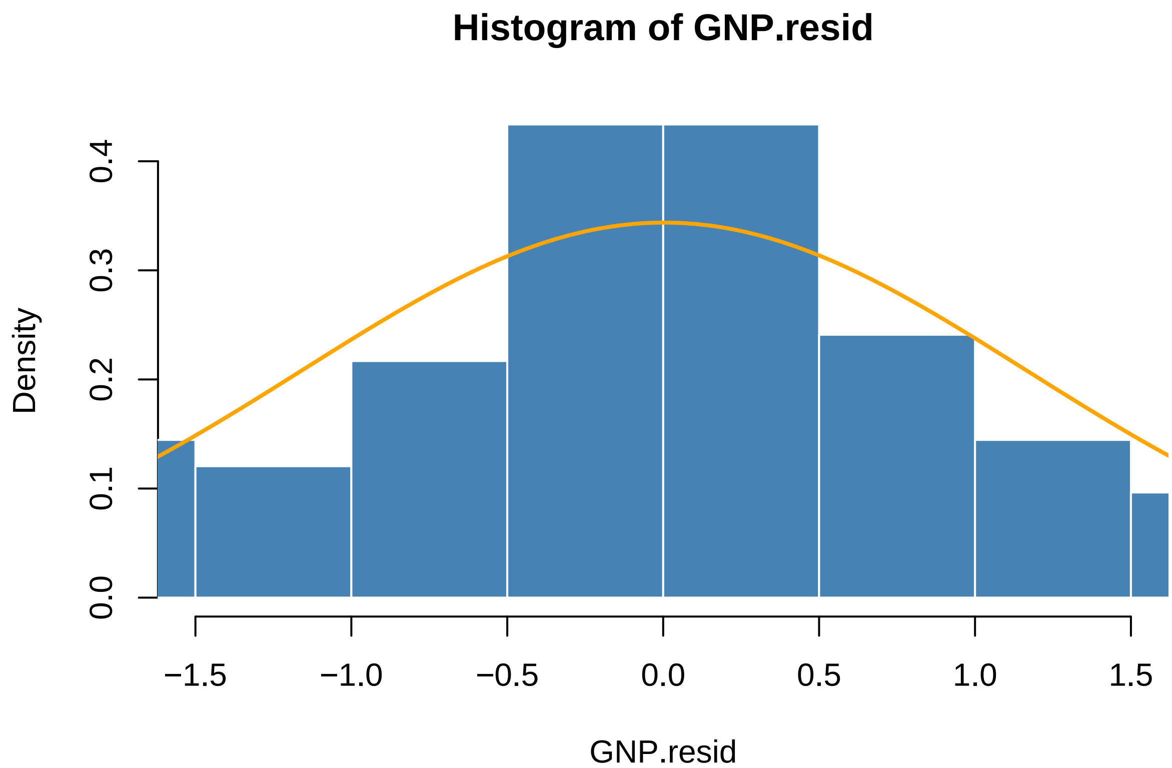
```
#' ***
#' What are the results?
#'
#'
fit.rate
```

```
##
## Call:
## arima(x = GNP.rate, order = c(2, 0, 1))
##
## Coefficients:
##           ar1     ar2     ma1  intercept
##       -0.2425  0.4844  0.7201     1.5582
## s.e.   0.2584  0.1310  0.2744     0.2826
##
## sigma^2 estimated as 1.33:  log likelihood = -129.82,  aic = 269.65
```

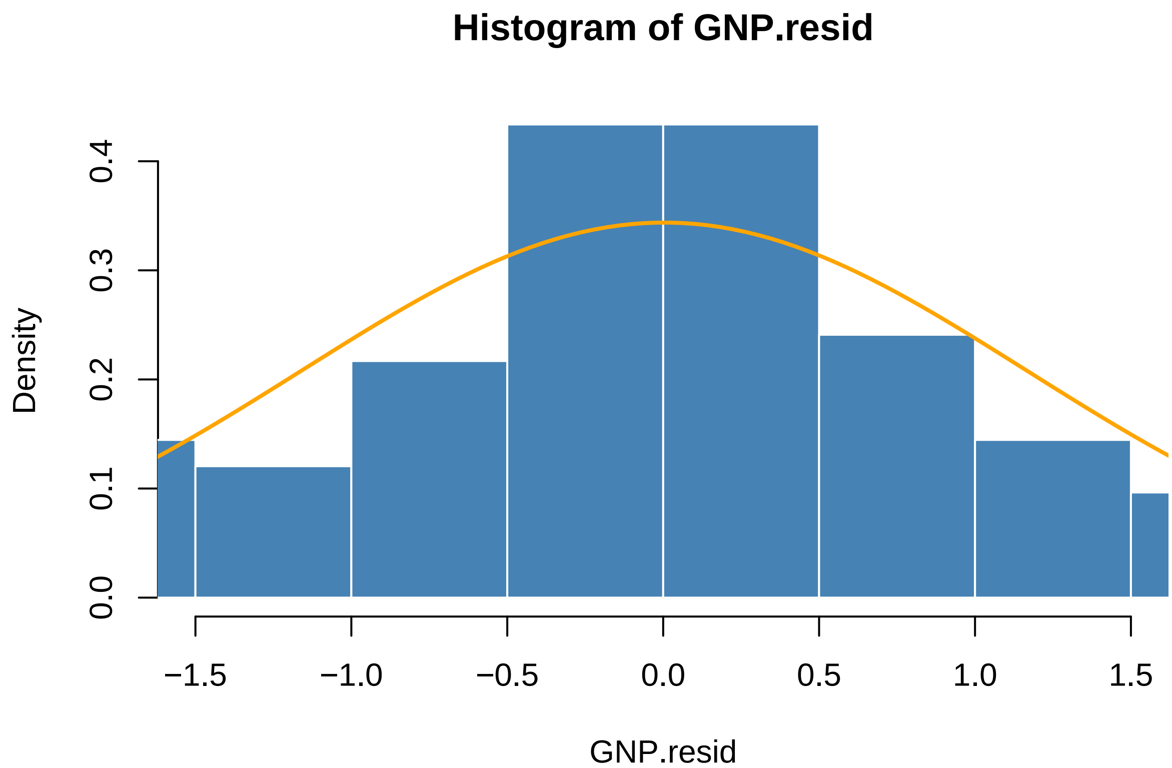
```
#'
#'
#' ***
#' Take out the moving average term and compare:
#'
#'
fit.rate.2 <- arima(GNP.rate, order = c(2,0,0))
fit.rate.2
```

```
##
## Call:
## arima(x = GNP.rate, order = c(2, 0, 0))
##
## Coefficients:
##          ar1     ar2  intercept
##       0.5036  0.0300     1.5586
## s.e.  0.1088  0.1085     0.2717
##
## sigma^2 estimated as 1.372:  log likelihood = -131.05,  aic = 270.09
```

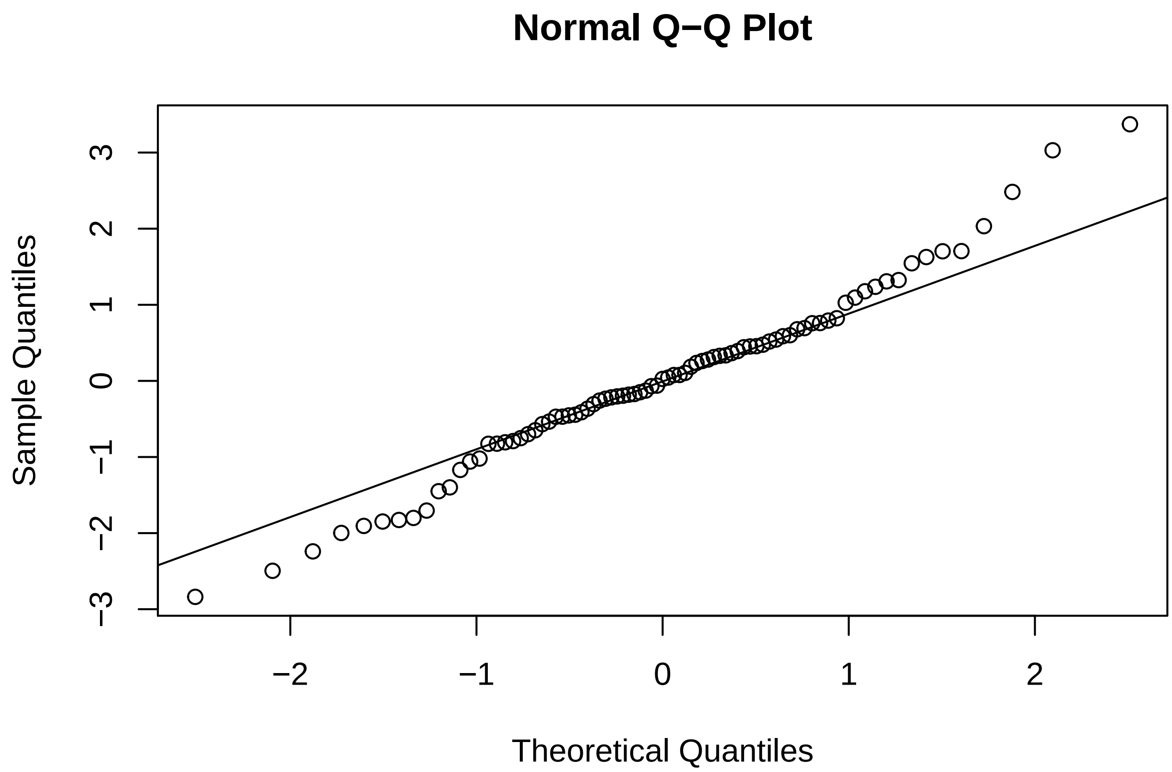
```
#'
#'
#' ***
#'
GNP.resid <- resid(fit.rate)
hist(GNP.resid, probability = TRUE, breaks = "FD", xlim = c(-1.5, 1.5), col = "steelblue", border = "whi
x = seq(-2, 2, length = 100)
lines(x, dnorm(x, mean = mean(GNP.resid), sd = sd(GNP.resid)), col = "orange", lwd = 2)
```



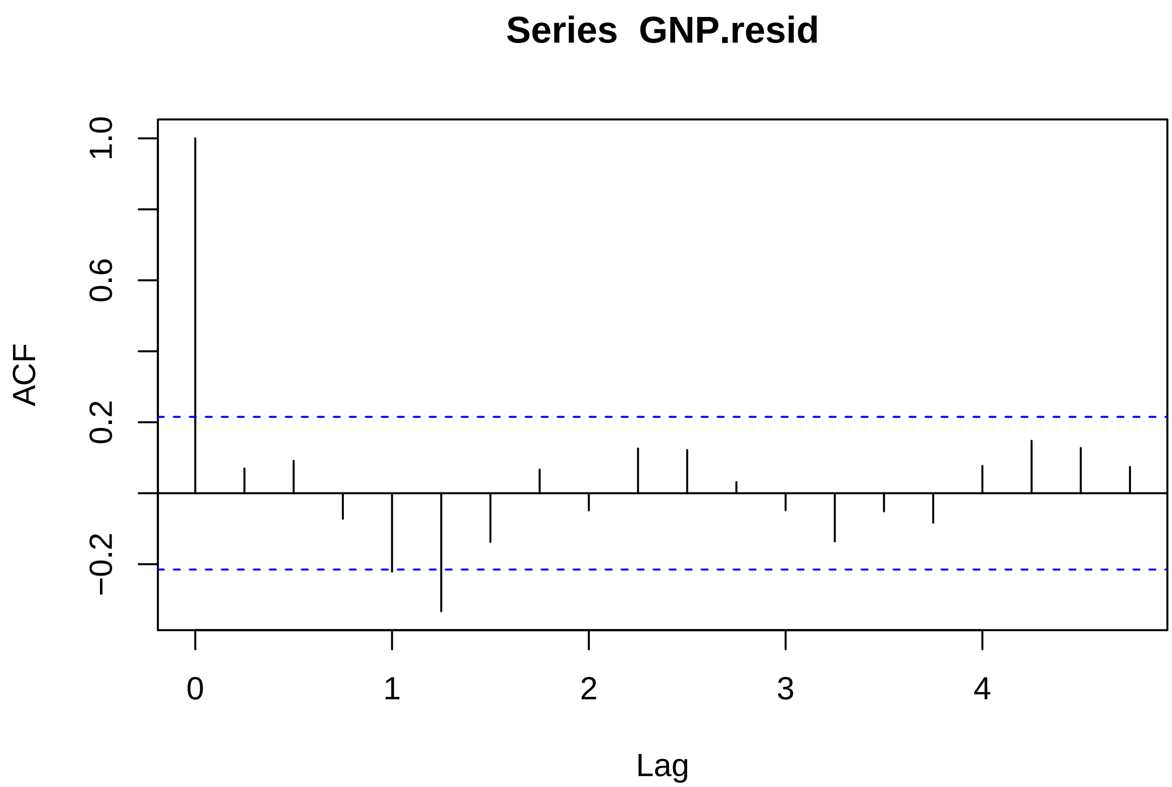

```
#'
#'
#' ***
#'
GNP.resid <- resid(fit.rate)
hist(GNP.resid, probability = TRUE, breaks = "FD", xlim = c(-1.5, 1.5), col = "steelblue", border = "whi
x = seq(-2, 2, length = 100)
lines(x, dnorm(x, mean = mean(GNP.resid), sd = sd(GNP.resid)), col = "orange", lwd = 2)
```



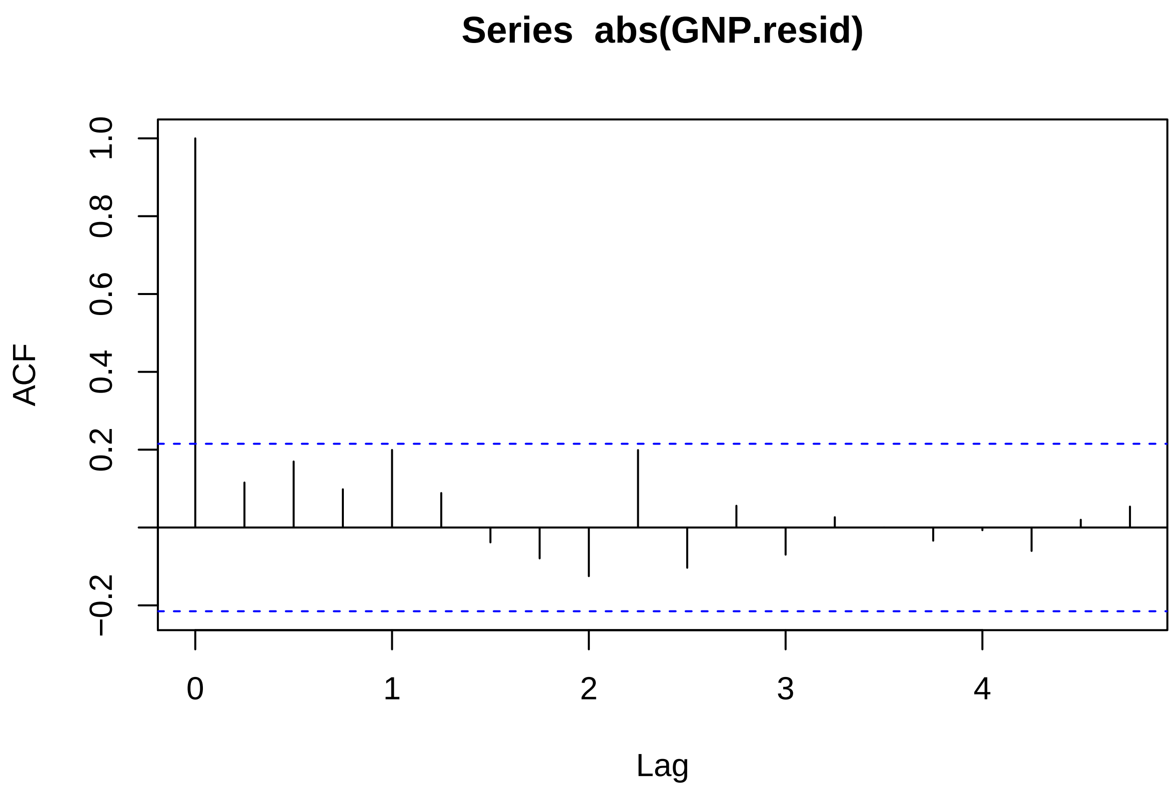

```
#'
#'
#'
#'
#' ***
#'
qqnorm(GNP.resid); qqline(GNP.resid)
```




```
#'
#'
#' ***
#' ##One way to read the qq-chart is
#' 1. The diagonal line is the normal distribution quantile line.
#' 2. Deviations of actual quantiles from the normal quantile line mean nonnormal.
#' 3. Especially deviations at either (or both) end of the line spell thick tails and lots more "shape"
#'
#' # Try this out
#'
#' Diagnose the GNP residuals using ACF and the `moments` package to calculate `skewness` and `kurtosis
#'
#' ***
#' Thinking...
#'
#' # Results
#'
#' Very thick tailed and serially correlated as evidenced by the usual statistical suspects. But no vol
#'
#'
acf(GNP.resid)
```




```r
#'
#'
#' ***
#' Nice absolute values (i.e., GNP growth sizes):
#'
#'
acf(abs(GNP.resid))
```




```r
#'
#'
#' ***
#'
require(moments)
```

```
## Loading required package: moments
##
## Attaching package: 'moments'
## The following objects are masked from 'package:timeDate':
##
##     kurtosis, skewness
```

```r
skewness(GNP.resid)
```

```
## [1] 0.1539986
```

```r
kurtosis(GNP.resid)
```

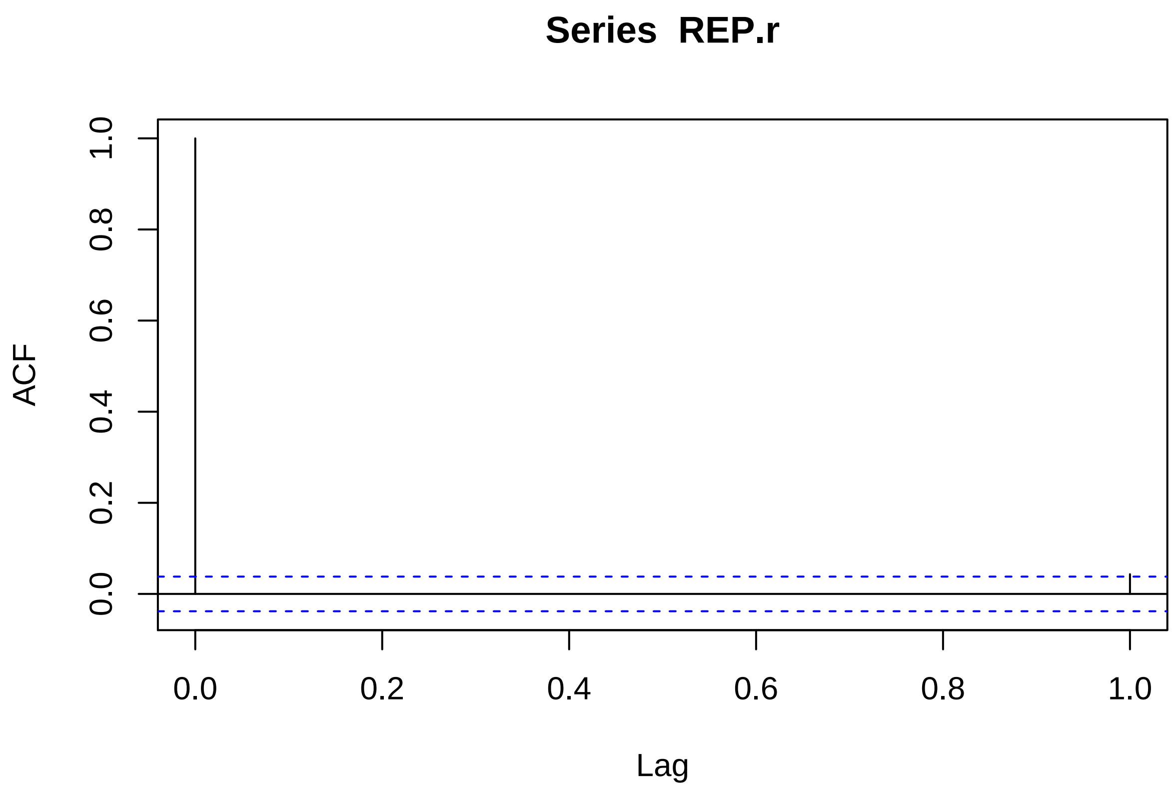
```
## [1] 3.596847
```

```r
#'
#' Positively skewed and thick tailed.
#'
#' ***
#' By the by: Where's the forecast?
#'
#'
```

```r
(GNP.pred <- predict(fit.rate, n.ahead = 8))
```

```
## $pred
##          Qtr1     Qtr2     Qtr3     Qtr4
## 2016 1.871913 1.635028 1.691508 1.563074
## 2017 1.621571 1.545178 1.592034 1.543671
##
## $se
##          Qtr1     Qtr2     Qtr3     Qtr4
## 2016 1.153446 1.278273 1.347110 1.357031
## 2017 1.367171 1.367728 1.369572 1.369573
```

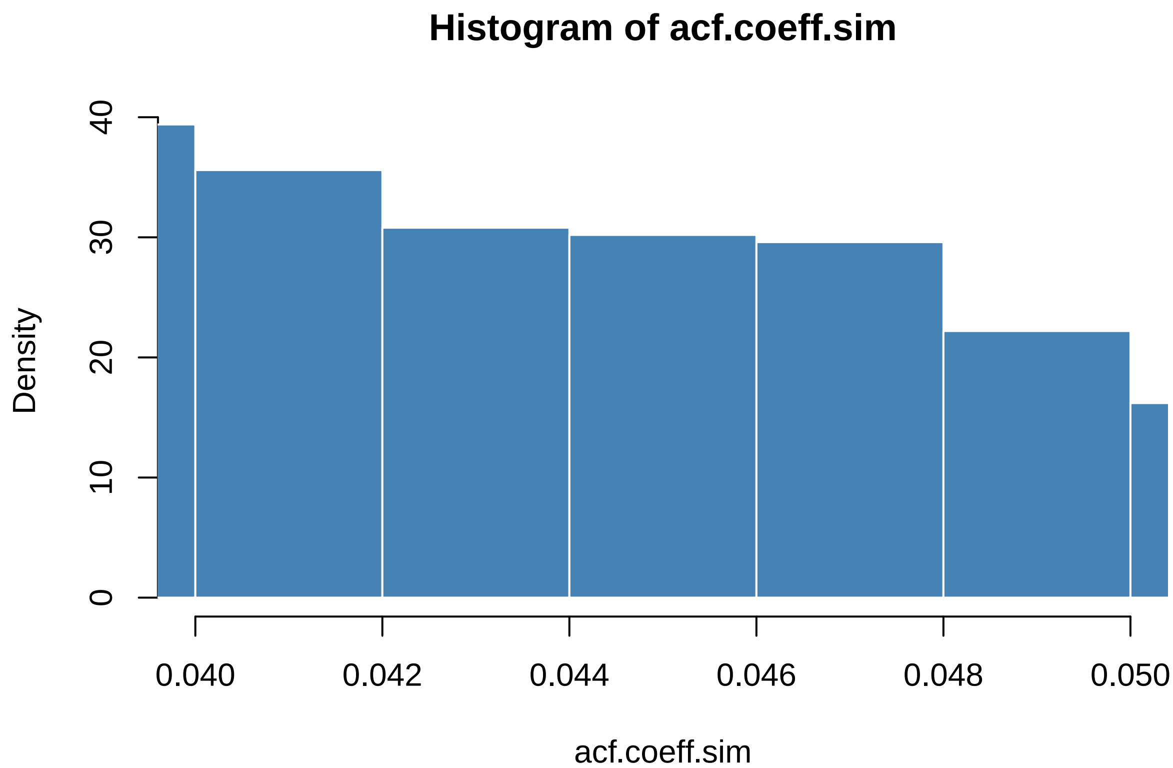
```r
#'
#'
#' ***
#'
#'
#' # Give it the boot
#'
#' ## Goal: An example of simulation-based inference.
#' - The context is just how dependent is today's stock return on yesterday's?
#' - We want to use the distribution of real-world returns data, without
#' needing assumptions about normality.
#' - The null hypothesis is lack of dependence (i.e., an efficient market).
#' - So repeatedly, the data is changed using the `replicate` function, and the sample ACF is computed.
#' - This gives us the distribution of the ACF under the null hypotheses, H0: independence while using
#'
#' ***
#' Let's use the Repsol returns. Pull the 1st autocorrelation from the sample:
#'
acf(REP.r, 1)
```




```r
#'
#'
#' ***
#' Not much to see -- barely a blip -- but over the 95% line. Let's further test this idea.
#'
#' ***
#' - Obtain 2500 draws from the distribution of the first autocorrelation using the `replicate` functio
#' - We operate under the null hypothesis of independence, assuming rational markets (i.e, rational mar
#'
#'
set.seed(1016)
acf.coeff.sim <- replicate(2500, acf(sample(REP.r, size = 2500, replace = FALSE), lag = 2,plot=FALSE)$ac
summary(acf.coeff.sim)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.002306  0.031130  0.038500  0.038370  0.045940  0.077340
```

```r
#'
#'
#' ***
#'
hist(acf.coeff.sim, probability = TRUE, breaks = "FD", xlim = c(.04, .05), col = "steelblue", border =
```




```
#'
#'
#' # Try this out
#' Investigate tolerances of $5\%$ and $1\%$ from both ends of the distribution of the 1-lag acf coeffi
#'
#' ``` {r mysize=TRUE, size='\\footnotesize'}
#' # At 95% tolerance level
#' quantile(acf.coeff.sim, probs=c(.025,.975))
#' # At 99% tolerance level
#' quantile(acf.coeff.sim, probs=c(.005,.995))
#' # And the
#' (t.sim <- mean(acf.coeff.sim)/sd(acf.coeff.sim))
#' (1-pt(t.sim, df = 2))
#' ```
#'
#' ***
#' Thinking...
#'
#' # Results
#' ## Some (highly preliminary and provisional answers)
#' 1. Quantile values are very narrow...
#' 2. How narrow (feeling like rejecting the null hypothesis)?
#' 3. t-stat is huge, but...
#' 4. ...no buts!, the probability that we would be wrong to reject the null hypothesis is very small.
#'
#' ***
```

```
#' Plot the simulated density and lower and upper quantiles, along with the estimate of the lag-1 coeff
#'
#' ***
#' ``` {r mysize=TRUE, size='\\footnotesize', eval = FALSE}
#' plot(density(acf.coeff.sim), col="blue")
#' abline(v=0)
#' abline(v=quantile(acf.coeff.sim, probs=c(.025,.975)), lwd=2, col="red")
#' abline(v=acf(REP.r, 1, plot=FALSE)$acf[2], lty=2, lwd=4, col="orange")
#' ```
#'
#' ***
#' ``` {r mysize=TRUE, size='\\footnotesize', echo = FALSE}
#' plot(density(acf.coeff.sim), col="blue")
#' abline(v=0)
#' abline(v=quantile(acf.coeff.sim, probs=c(.025,.975)), lwd=2, col="red")
#' abline(v=acf(REP.r, 1, plot=FALSE)$acf[2], lty=2, lwd=4, col="orange")
#' ```
#'
#' ***
#' Can we reject the null hypothesis that the coefficient = 0? Is the market "efficient"?
#'
#' ***
#' 1. Reject the null hypothesis since there is a less than 0.02% chance that the coefficient is zero.
#' 2. Read [Fama(2013, p. 365-367)]<https://www.nobelprize.org/nobel_prizes/economic-sciences/laureates/
#' 3. If the model is correct (ACF lag-1) then the previous day's return can predict today's return acc
#' 4. This means we might be able to create a profitable trading strategy that makes use of the little
#'
#' ***
#'
#'
#' # The wrap
#'
#' - Lots more `R` practice
#' - ACF and PACF to do EDA on time series
#' - Stylized facts of financial returns
#' - Simulated coefficient inference to check efficient markets hypothesis
#' - Probability distributions
#' - Risk tolerance from an inference point of view
#' - Yahoo finance data graps
#' - Average regression and quantile regression
#'
#' # To prepare for the live session:
#'
#' ## List these:
#' 1. What are the top 3 key learnings for you from this segment?
#' 2. What pieces of this segment are still a mystery?
#' 3. What parts would you like more practice on?
#' 4. Review the assignment. What questions do you have about the assignment for the live session?
#'
#' ## Thanks! Till next week...
#'
#' ***
#'
```

```
#'
```

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

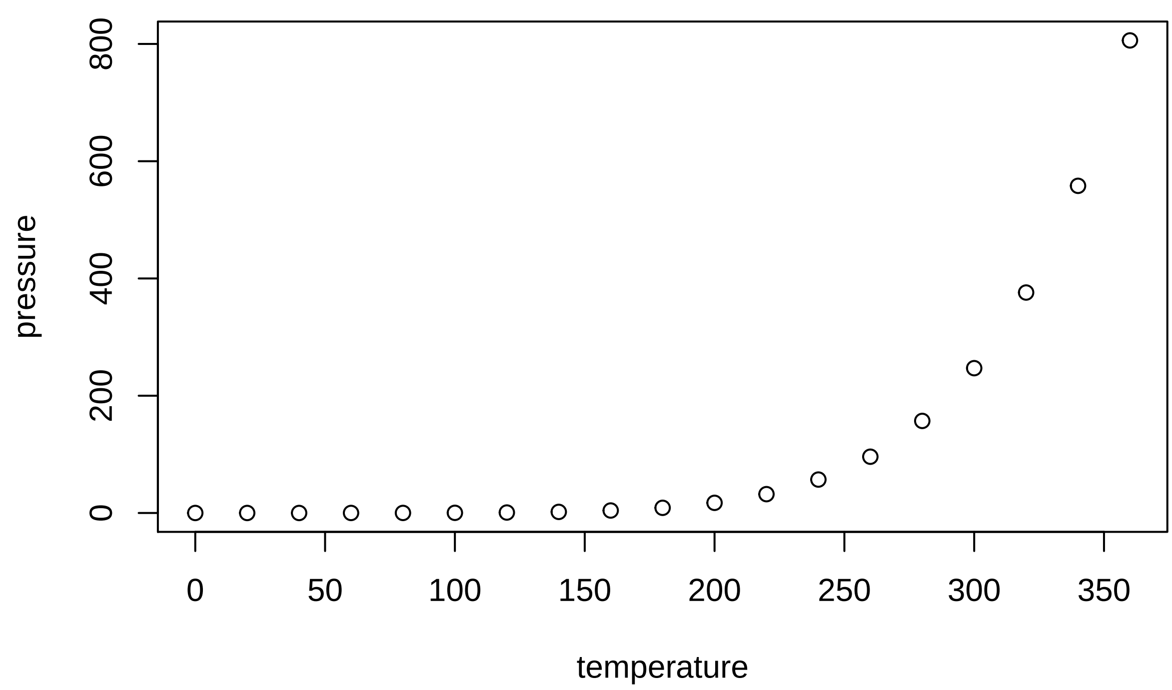
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed           dist       
##  Min.   : 4.0   Min.   :  2.00  
##  1st Qu.:12.0   1st Qu.: 26.00  
##  Median :15.0   Median : 36.00  
##  Mean   :15.4   Mean   : 42.98  
##  3rd Qu.:19.0   3rd Qu.: 56.00  
##  Max.   :25.0   Max.   :120.00
```

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.