Trading ETF's - The sequel, Daniel Emil Wiinberg S133232

```r
D <- read.table("finans2_data.csv", header = TRUE, sep = ";")
```

Introduction to the data

```r
head(D); tail(D)
```

```
##   ETF    Geo.mean Volatility maxTuW
## 1 SPY  0.10490378   2.478601    309
## 2 MDY  0.13954700   2.925897    327
## 3 EWJ -0.02620057   2.699671    327
## 4 EWH  0.11267242   3.373114    302
## 5 EWG  0.05038104   3.800206    177
## 6 EWW  0.08650049   4.073397    327
```

```
##    ETF    Geo.mean Volatility maxTuW
## 90 FXI  0.14410014  4.6620500    324
## 91 IAU  0.12507213  2.8521063    333
## 92 SLV  0.02909721  4.8307408    333
## 93 USO -0.27565080  4.8030638    319
## 94 SHY  0.01321481  0.1965787    375
## 95 TLT  0.08273749  2.0187989    182
```

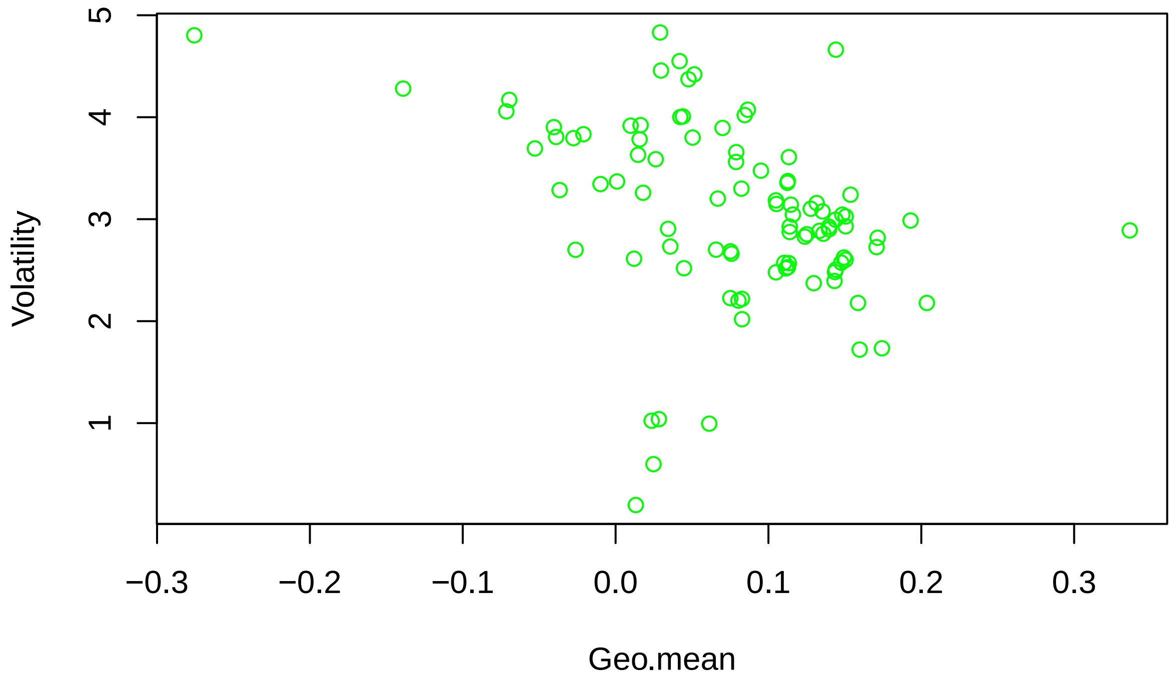
The data contain 95 rows of data (excluding header), one for each ETF. Each row contain 3 columns for the 3 variables: Geo.mean, Volatility and maxTuw. These variables are calculated from the data used in the first project, representing the 454 weeks returns of each ETF.

- **Geo.mean** contains observations of the geometric average rate of return for the ETFs in the dataset.
- **Volatility** represents the weekly volatility which is the standard deviation of the ratio between the price of an ETF at the beginning of a week and the end.
- **maxTuW** is Maximum Time under Water, and indicates the maximum number of weeks between two peak prices.

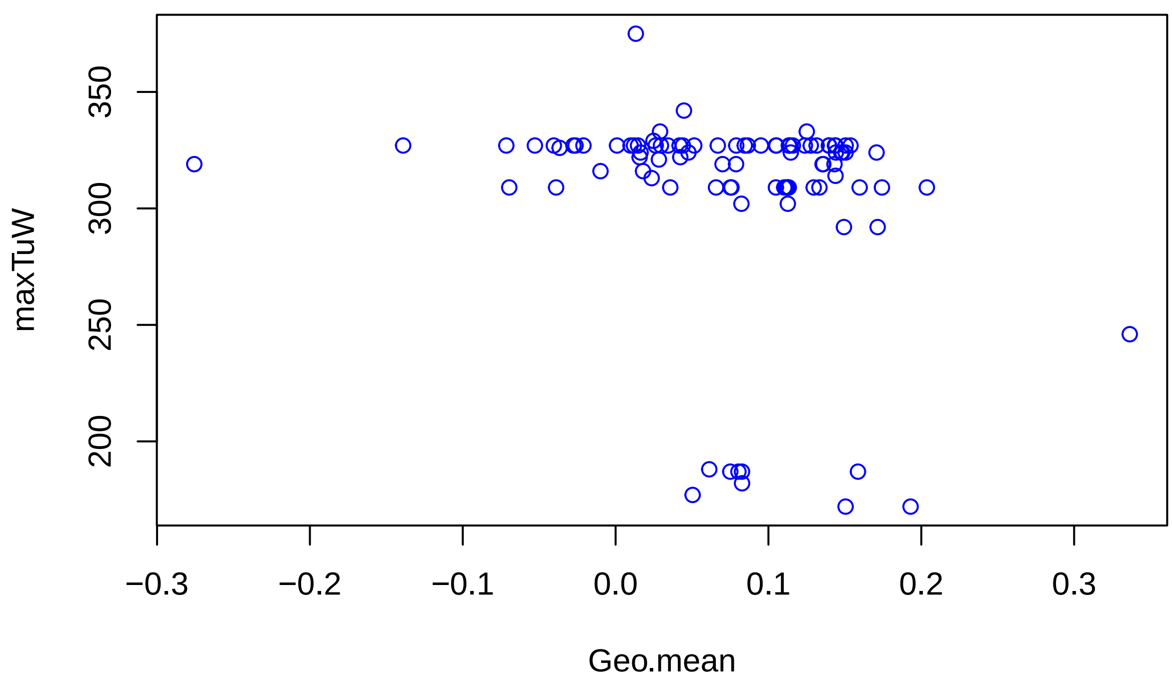
Scatterplots

First we plot the volatility and maxTuW variables against the geometric mean in a scatterplot.

```r
# Scatterplot
plot(c(D$Geo.mean), c(D$Volatility), type="p", xlab="Geo.mean",
     ylab="Volatility", col=c("green"), cex=1)
```



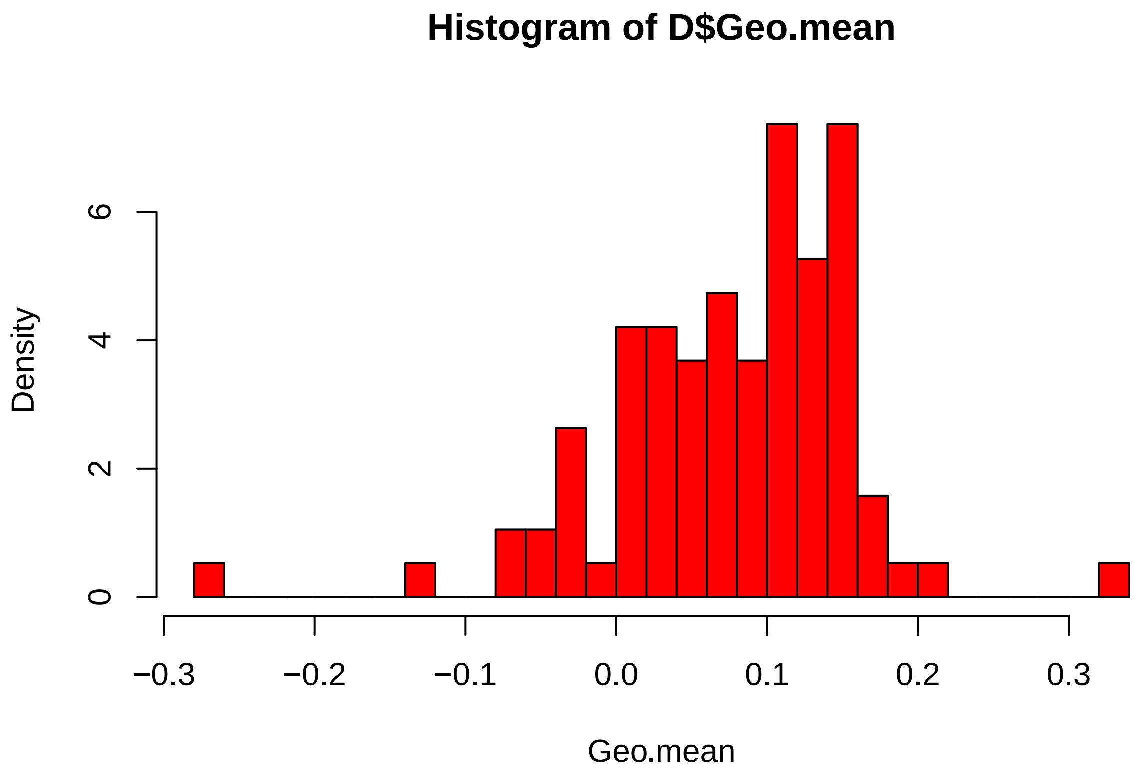

```
plot(c(D$Geo.mean), c(D$maxTuW), type="p", xlab="Geo.mean",
     ylab="maxTuW", col=c("blue"), cex=1)
```



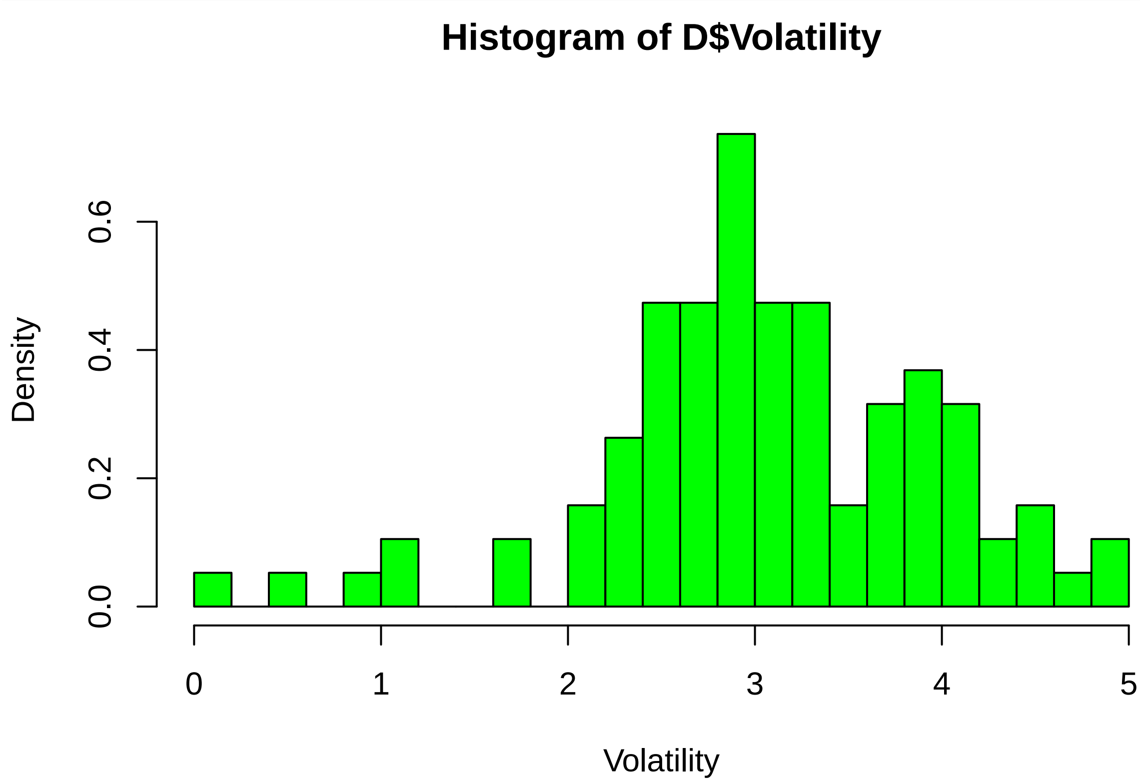
Histograms

Next we plot the three variables in a histogram plot to easily the distribution of values.

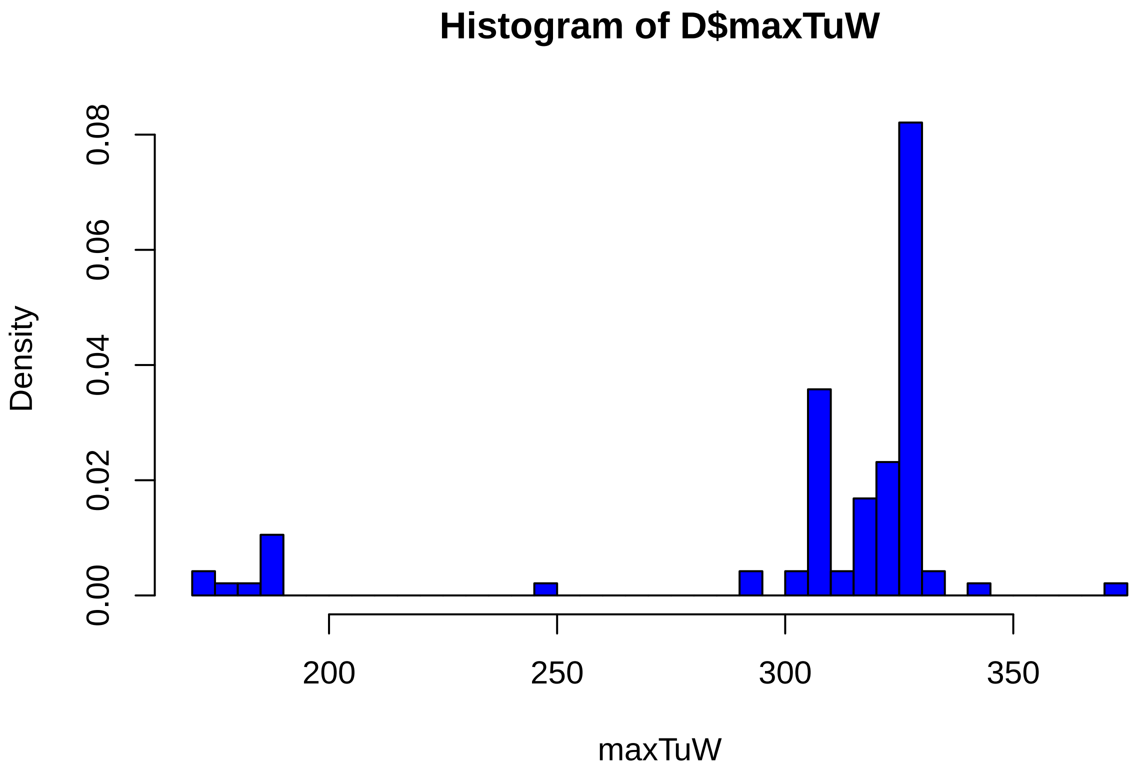
```r
# Histograms
hist(D$Geo.mean, xlab="Geo.mean", prob=TRUE, breaks = 30, col="red")
```




```r
hist(D$Volatility, xlab="Volatility", prob=TRUE, breaks = 30, col="green")
```



```
hist(D$maxTuW, xlab="maxTuW", prob=TRUE, breaks = 30, col="blue")
```



Summary table

We extract some general information about the variables using the summary R function and display it in a table as shown below.

```
# summary(D)
# sd(D$Geo.mean)
# sd(D$Volatility
# sd(D$maxTuW)
```

	n	mean	sd	median	q1	q3
Geo.mean	95	0.07690	0.08087	0.08274	0.02871	0.1344
Volatility	95	3.060	0.8790	3.026	2.588	3.675
maxTuW	95	307.3	42.77	324.0	309.0	327.0

Multiple linear regression

Our linear regression model with for the geometric mean with our two variables for volatility and maxTuW is:

$$Y_i = \beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + \epsilon_i$$

Where Y_i is the geometric mean, x_1 is Volatility and x_2 is maxTuW.

Next we divide out data set into two pools, where we will use the bigger pool to create a prediction model, and then use the smaller test set to see how well our prediction model behaves.

```
# Subset containing only AGG, VAW, IWN and SPY (for validation)
D_test = subset(D, ETF %in% c("AGG","VAW","IWN","SPY"))
# Subset containing only the 91 remaining ETFs (for model estimation)
D_model = subset(D, !(ETF %in% c("AGG","VAW","IWN","SPY")))
```

Here we create our linear regression for the geometric mean with our two variables.

```
# Estimate multiple linear regression model
fit = lm(Geo.mean ~ Volatility + maxTuW, data = D_model)

# Show parameter estimates etc.
summary(fit)
```

```
##
## Call:
## lm(formula = Geo.mean ~ Volatility + maxTuW, data = D_model)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.28946 -0.05088  0.02092  0.04937  0.23927
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.2528394  0.0584827   4.323 4.04e-05 ***
## Volatility  -0.0351310  0.0097229  -3.613 0.000503 ***
## maxTuW      -0.0002203  0.0001910  -1.154 0.251795
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.07582 on 88 degrees of freedom
## Multiple R-squared:  0.1704, Adjusted R-squared:  0.1516
## F-statistic: 9.039 on 2 and 88 DF,  p-value: 0.0002689
```

From the summary is can be seen that:

Name	Value
β_0	0.2528394
β_1	-0.0351310
β_2	-0.0002203
$\hat{\sigma}^2$	0.07582
Degree of freedom	88
R^2	0.1516

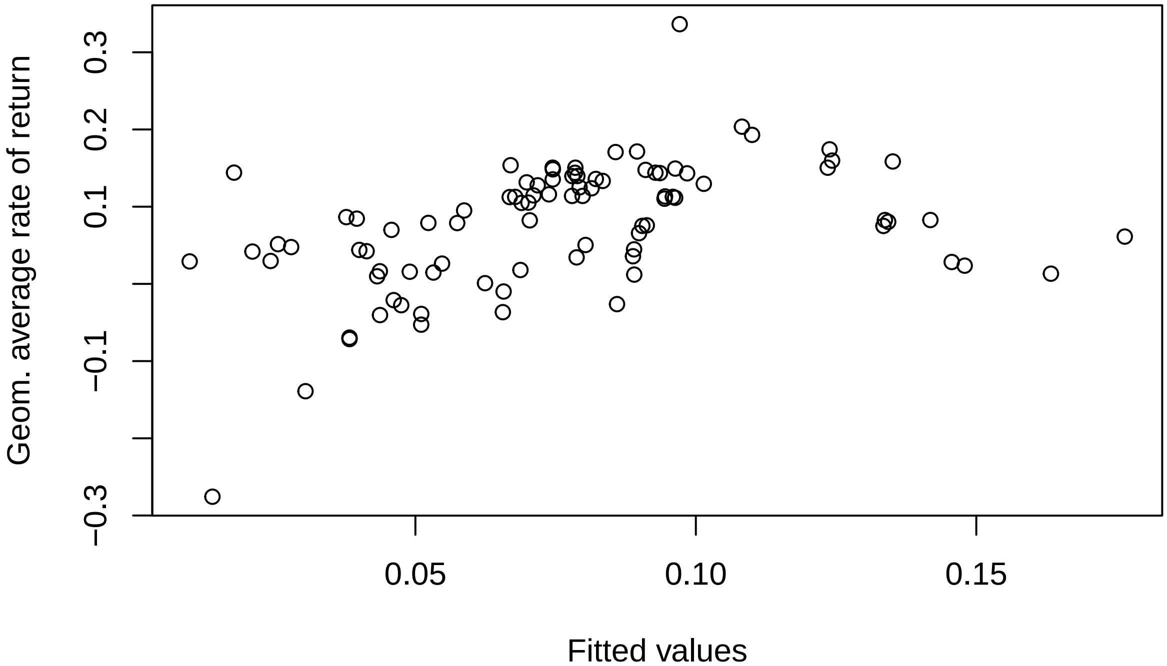
β_0 is the interception with $x = 0$ and β_1 and β_2 are the slopes of the regression line. Meaning the effect that β_i has on x_i.

Since β_1 and β_2 are both slightly negative it means that volatility and Maximum Time under Water has a negative influence on the average return.

Model validation

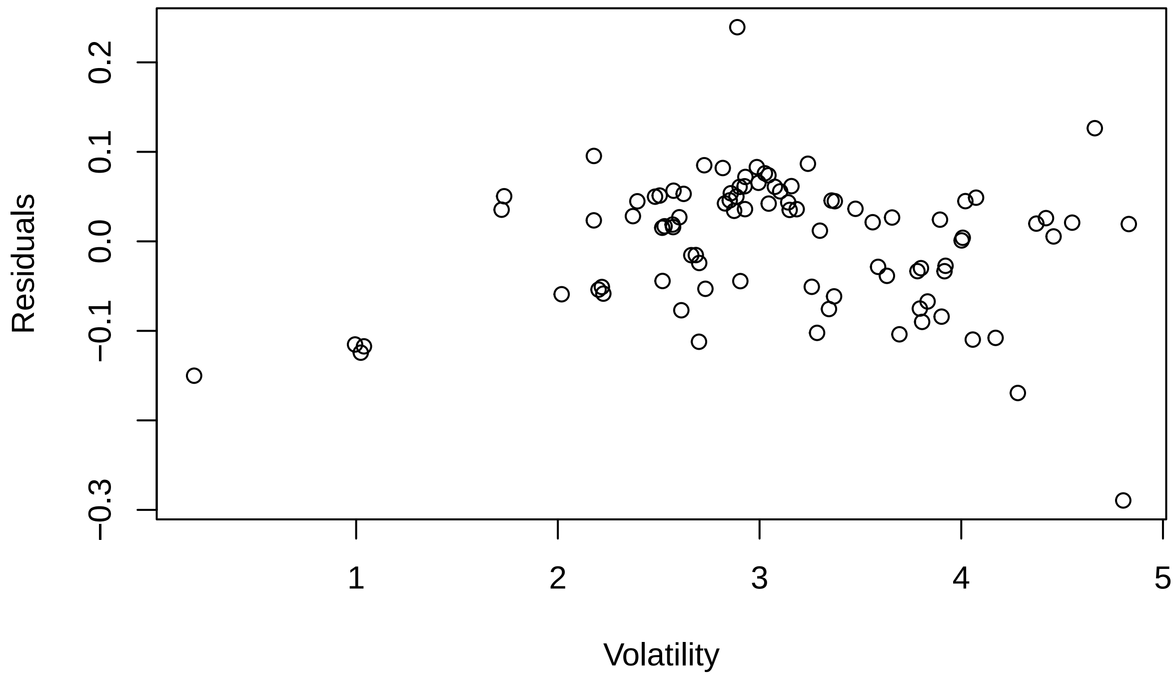
We start by plotting the observed geometric mean values against our fitted values.

```
# Plots for model validation
# Observations against fitted values
plot(fit$fitted.values, D_model$Geo.mean, xlab = "Fitted values",
       ylab = "Geom. average rate of return")
```

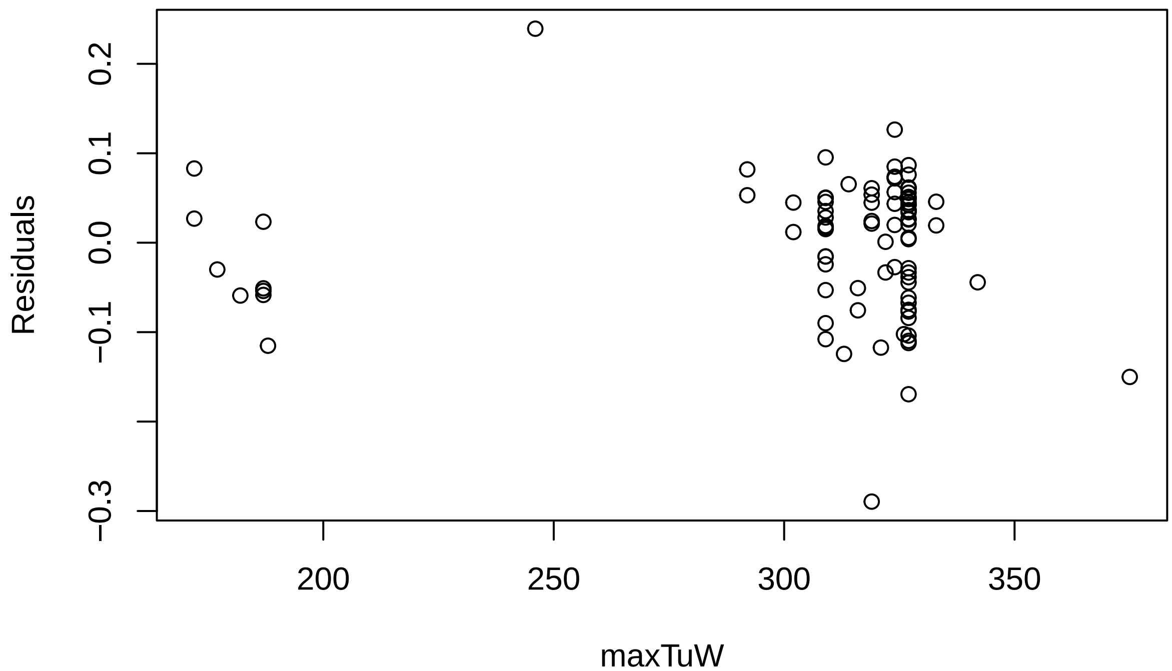


Next we plot the fitted values and each of our explanatory variables against the residuals. We are looking systematic dependences between the values and the residuals.

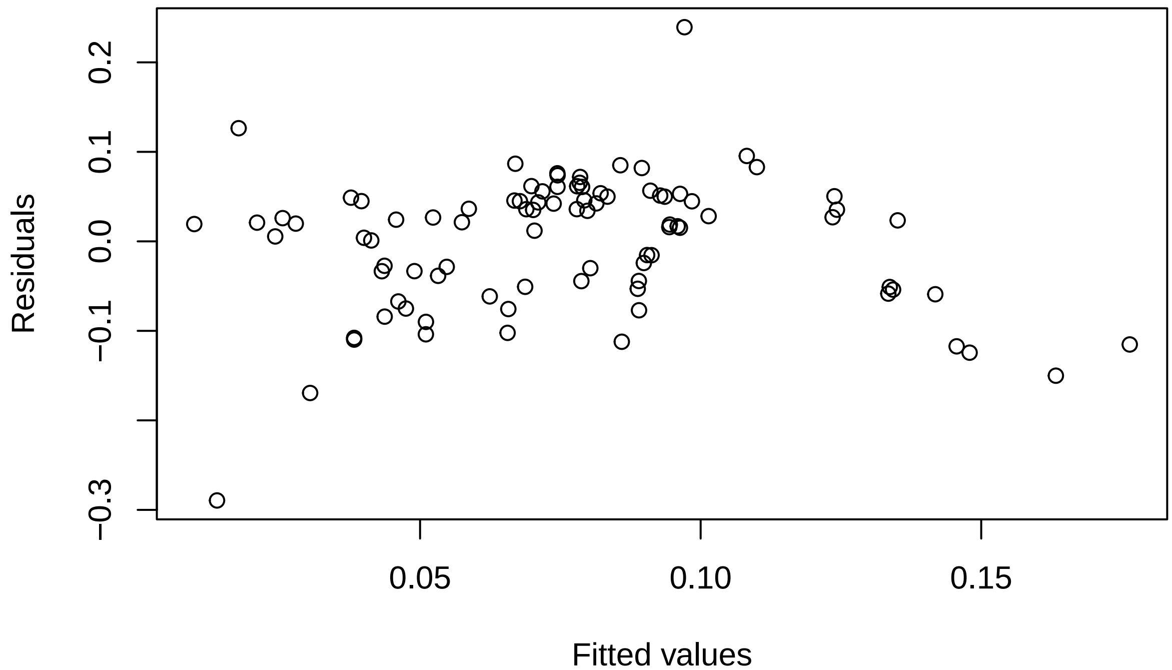
```
# Residuals against each of the explanatory variables
plot(D_model$Volatility, fit$residuals,
     xlab = "Volatility", ylab = "Residuals")
```




```r
plot(D_model$maxTuW, fit$residuals,
     xlab = "maxTuW", ylab = "Residuals")
```



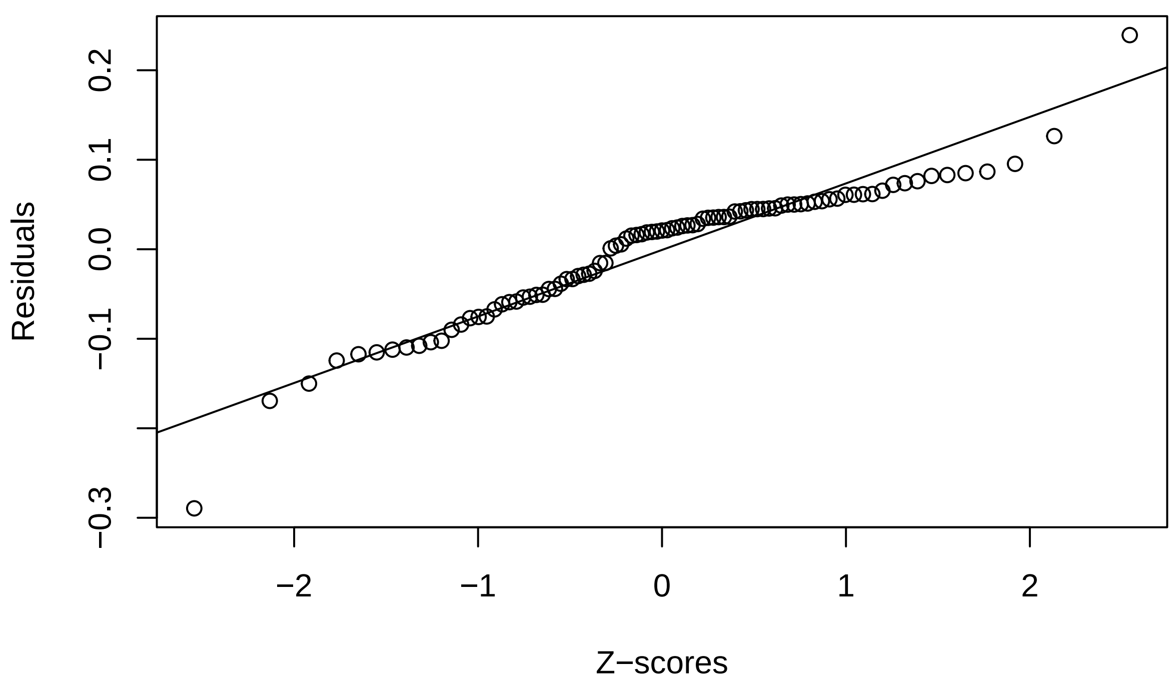

```r
# Residuals against fitted values
plot(fit$fitted.values, fit$residuals, xlab = "Fitted values",
     ylab = "Residuals")
```



From the plots of the residuals against the explanatory variables we do not see a systematic dependence, which means that our model is sound.

Next we check if the residuals are normal distributed by doing a qq-plot.

```
# Normal QQ-plot of the residuals
qqnorm(fit$residuals, ylab = "Residuals", xlab = "Z-scores",
       main = "")
qqline(fit$residuals)
```



From the qq-plot and qq-line we can see that the residuals are normal distributed, which means that the normality assumption holds.

Confidence interval for β_1

The confidence interval for parameter β_1 is given by:

$$\hat{\beta}_1 \pm t_{1-\alpha/2}\hat{\sigma}_{\beta_1}$$

```
#Confidence interval for beta_1
-0.0351310 + c(1,-1)*qt(0.975, df=88)*0.0097229
```

```
## [1] -0.01580878 -0.05445322
```

```
# Confidence intervals for the model coefficients
confint(fit, level = 0.95)
```

```
##                     2.5 %        97.5 %
## (Intercept)  0.1366172663  0.3690616050
## Volatility  -0.0544532736 -0.0158088161
## maxTuW      -0.0005999244  0.0001592431
```

We can see that the values calculated for β_1 using the formula are the same as when using the R-function. The confidence intervals for β_0 and β_2 are also given from the R-function above.

Hypothesis testing

We are testing whether β_1 might be -0.06, meaning we are testing:

$$H_{0,1} : \beta_1 = \beta_{0,1} \Rightarrow \beta_1 = -0.06$$

Against the other hypothesis:

$$H_{1,1} : \beta_1 \neq \beta_{0,1} \Rightarrow \beta_1 \neq -0.06$$

We are testing with a significance level of $\alpha = 0.05$.

```
tobs = (-0.0351310 - (-0.06)) / 0.0097229
tobs
```

```
## [1] 2.557776
```

```
# The p value becomes
2*(1-pt(tobs,df=88))
```

```
## [1] 0.01224591
```

We can see that with a $p - value = 0.0122$ we have some evidence against the hypothesis. This could also be seen from the confidence interval for β_1 being $\{-0.0545; -0.0158\}$, since -0.06 is outside the interval.

Backward selection

If we go back to inspect our model, we can see that that maxTuW is not a significant parameter in our model, since its p-value is 0.252. Using backwards selection we remove this parameter from our model, and fit the data again.

```r
# Estimate multiple linear regression model
final_model = lm(Geo.mean ~ Volatility, data = D_model)

# Show parameter estimates etc.
summary(final_model)
```

```
##
## Call:
## lm(formula = Geo.mean ~ Volatility, data = D_model)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.28686 -0.03830  0.02009  0.04689  0.25202
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.19486    0.02996   6.504 4.44e-09 ***
## Volatility  -0.03824    0.00936  -4.085 9.62e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.07596 on 89 degrees of freedom
## Multiple R-squared:  0.1579, Adjusted R-squared:  0.1484
## F-statistic: 16.69 on 1 and 89 DF,  p-value: 9.624e-05
```

We can now see that the slope is highly significant with a p-value of $9.62 * 10^{-5}$, meaning we cannot reduce our model further.

Therefore our final model with estimations of our parameters is:

$$Y_i = \beta_0 + \beta_1 x_{1,i} = 0.19486 - 0.03824 x_1$$

Comparing D_{model} and D_{test}

First we predict the data for the geometric mean using our final.

```
# Predictions and 95% prediction intervals
pred = predict(final_model, newdata = D_test,
               interval = "prediction", level = 0.95)
```

Next we use the cbind function to easily compare the actual geometric mean from our 4 test models to the values that we just got from our prediction.

```
# Observed values and predictions
compare = cbind(id = D_test$ETF, Geo.mean = D_test$Geo.mean, pred)
compare
```

```
##    id    Geo.mean      fit                  lwr
## 1  "SPY" "0.10490378"  "0.100090940900893"  "-0.0520845445000064"
## 40 "IWN" "0.066849084" "0.0724485877394012" "-0.0793227571820516"
## 69 "AGG" "0.024793652" "0.172012988491345"  "0.0133582944803555"
## 80 "VAW" "0.113346442" "0.05689659980963"   "-0.0951704082967189"
##    upr
## 1  "0.252266426301793"
## 40 "0.224219932660854"
## 69 "0.330667682502335"
## 80 "0.208963607915979"
```

We can see that the predictions are all fairly close, even though some of the predictions are better than others. However they are all within the 95% confidence interval.