In [ ]:
```
install.packages('lmtest')
install.packages('ggplot2')
install.packages("TTR")
install.packages("quadprog", repos="http://cran.rstudio.com")
install.packages("IntroCompFinR", repos="http://R-Forge.R-project.org")
install.packages('psych')
```

In [ ]:
```
library('ggplot2')
library("TTR")
source('normality.r')
library('lmtest')
library('IntroCompFinR')
library('psych')
```

In [317]:
```
d = read.csv('ALLDATA.csv',head = TRUE)
```

In [318]:
```
d$Date_formated = as.Date(d$Dates,tryFormats = "%d-%m-%Y")
```

In [319]:
```
A_1 = ts(d$ASSET_1); A_2 = ts(d$ASSET_2);
A_3 = ts(d$ASSET_3); A_4 = ts(d$ASSET_4);
A_5 = ts(d$ASSET_5); A_6 = ts(d$ASSET_6);
A_7 = ts(d$ASSET_7); A_8 = ts(d$ASSET_8);
A_9 = ts(d$ASSET_9); A_10 = ts(d$ASSET_10);A_11 = ts(d$ASSET_11);
```

In [579]:
```
# Plots
jpeg("Question1_plots/Timeseries_plots_first6_Assets.jpg")
  par(mfrow=c(2,3))
  ts.plot(A_1);ts.plot(A_2)
  ts.plot(A_3);ts.plot(A_4)
  ts.plot(A_5);ts.plot(A_6)
dev.off()
```

png: 2

In [580]:
```
jpeg("Question1_plots/Timeseries_plots_next5_Assets.jpg")
  par(mfrow=c(2,3))
  ts.plot(A_7);ts.plot(A_8)
  ts.plot(A_9);ts.plot(A_10)
  ts.plot(A_11);
dev.off()
```

png: 2

In [322]:
```
summary(d)
```

```
    Dates               ASSET_1          ASSET_2          ASSET_3
 Length:1147        Min.   : 4518   Min.   :15278   Min.   : 6322
 Class :character   1st Qu.: 8033   1st Qu.:20620   1st Qu.: 9077
 Mode  :character   Median : 9023   Median :24817   Median :10531
                    Mean   : 9138   Mean   :24316   Mean   :10475
                    3rd Qu.:10750   3rd Qu.:27331   3rd Qu.:11636
                    Max.   :12010   Max.   :32444   Max.   :14698
    ASSET_4          ASSET_5          ASSET_6           ASSET_7
 Min.   :19136   Min.   : 9435   Min.   : 987.2   Min.   :1496
 1st Qu.:23969   1st Qu.:10810   1st Qu.:1881.5   1st Qu.:2431
 Median :28342   Median :13505   Median :2554.2   Median :2886
 Mean   :26968   Mean   :13434   Mean   :2475.6   Mean   :2871
 3rd Qu.:30002   3rd Qu.:15532   3rd Qu.:3031.4   3rd Qu.:3339
 Max.   :32912   Max.   :22493   Max.   :3642.7   Max.   :4196
    ASSET_8          ASSET_9        ASSET_10         ASSET_11
 Min.   : 6432   Min.   :1087   Min.   : 8581   Min.   :146.6
 1st Qu.: 8709   1st Qu.:2459   1st Qu.:11418   1st Qu.:205.7
 Median : 9389   Median :2941   Median :13873   Median :251.8
 Mean   : 9574   Mean   :2776   Mean   :13576   Mean   :245.5
 3rd Qu.:10504   3rd Qu.:3233   3rd Qu.:15416   3rd Qu.:280.6
 Max.   :12321   Max.   :4069   Max.   :17914   Max.   :372.2
 Date_formated
 Min.   :2016-03-14
 1st Qu.:2017-05-13
 Median :2018-07-05
 Mean   :2018-07-09
 3rd Qu.:2019-09-05
 Max.   :2020-10-30
```

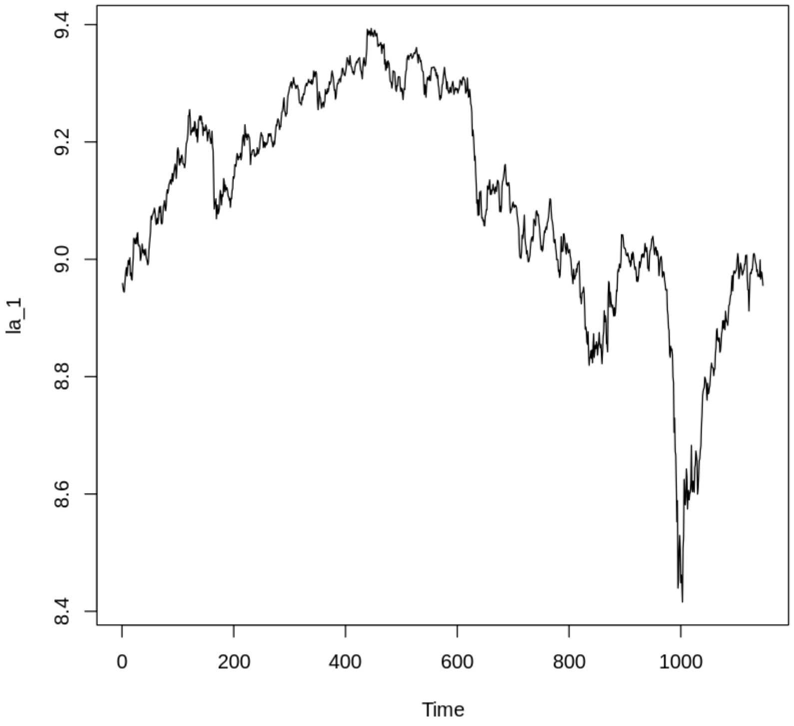
Q1. How can you identify Trends in the given Time series data? Please write appropriate code to calculate and visualize the Trends for each of the 11 Assets. (The plots can be simple and need not be fancy)

Method 1 Naive approach: Linear regression with Quadratic trend after smoothening

'It can be observed from the below plot that there appears to be no seasonal component. However presence of irregular components or randomness is evident. To estimate the trend component of a non-seasonal time series that can be described using an additive model, it is common to use a smoothing method, such as calculating the simple moving average of the time series.'

In [324]:

```
la_1 = log(A_1)
ts.plot(la_1)
```

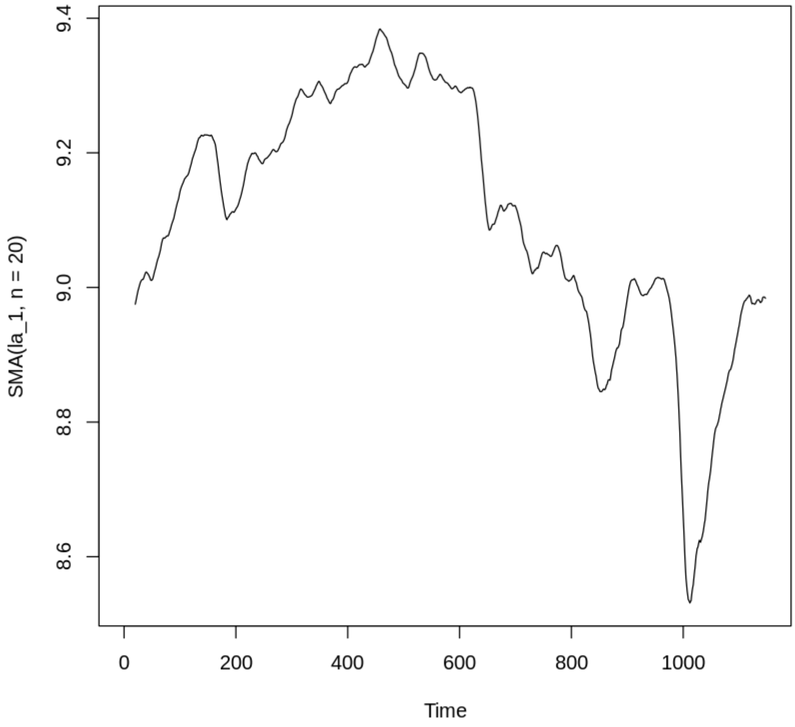


In [323]:

```
t = 1:dim(d)[1]
tsqr = t^2
```

Applying log transformation

In [325]:
```
s_ta1 = SMA(la_1,n=20)      #increased n from n=1 till satisfaction
ts.plot(SMA(la_1,n=20))
```



In [326]:
```
model = lm(s_ta1~t+tsqr) #Modeling the prices with a linear and a quadratic term y =
 B0 + B1T + B2Tsqr +Error; trend = B0+B1T+B2Tsqr
```

In [327]:
```
summary(model)
```

```
Call:
lm(formula = s_ta1 ~ t + tsqr)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.33902 -0.07479  0.01072  0.07614  0.30278 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  9.068e+00  1.030e-02  880.27   <2e-16 ***
t            8.674e-04  4.067e-05   21.33   <2e-16 ***
tsqr        -1.050e-06  3.381e-08  -31.07   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1077 on 1125 degrees of freedom
  (19 observations deleted due to missingness)
Multiple R-squared:  0.6706,	Adjusted R-squared:  0.6701 
F-statistic:  1145 on 2 and 1125 DF,  p-value: < 2.2e-16
```

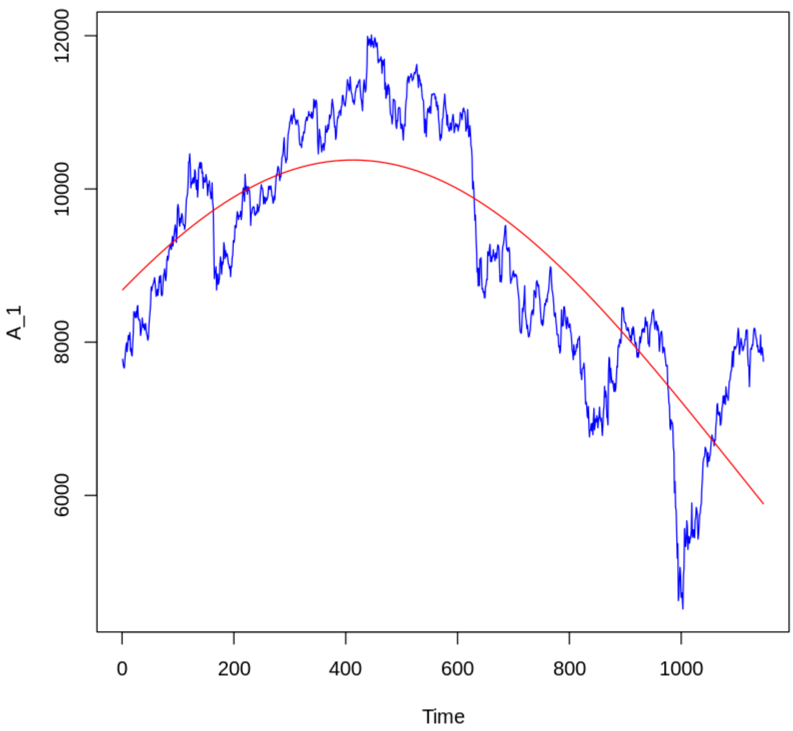
In [328]:
```
anova(model)
```

A anova: 3 × 5

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
	<int>	<dbl>	<dbl>	<dbl>	<dbl>
t	1	15.37214	15.37213950	1325.4659	2.139445e-192
tsqr	1	11.19562	11.19562153	965.3448	1.564222e-153
Residuals	1125	13.04723	0.01159754	NA	NA

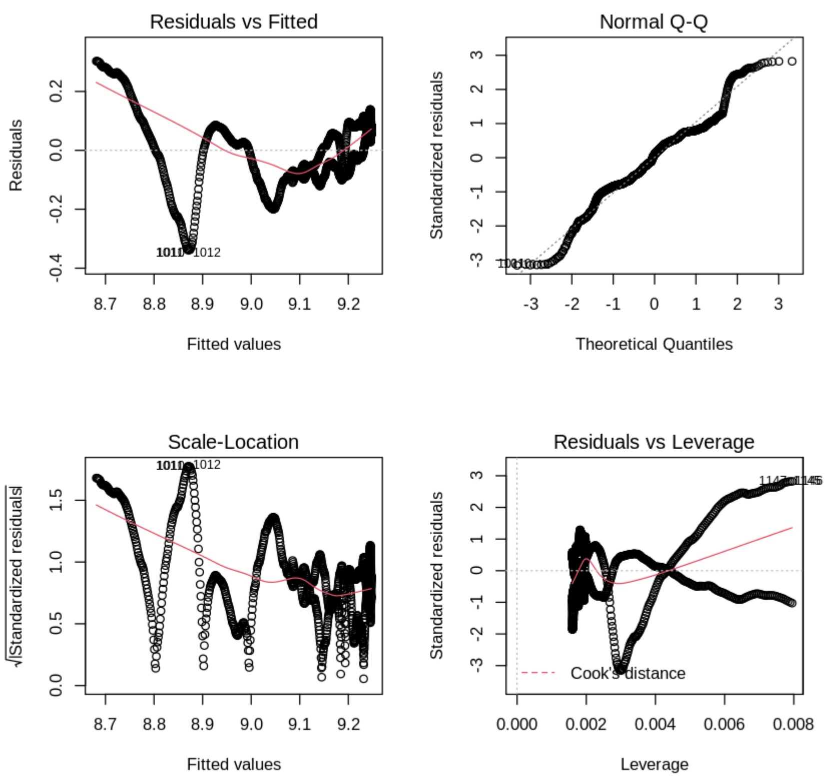
'The model explains a significant amount of variability that can be seen from the Fstatistic and p-value arrived by looking at the proportion of variance explained by the regression model compared to SYY'

In [330]:
```
pred <- exp(predict.lm(model,s_ta1)) # applying exp as the ouput is the log of the re
sult.
plot(A_1,col="blue")
lines(pred,col="red")
```



However The residuals exhibits patterns indicating that the model has not captured. Without even any tests one can spot that the residuals are not homoskedastic and Normal from the qq plot and scale-location plots

In [331]:
```
par(mfrow=c(2,2))
plot(model)
```



In [332]:
```
'Test for normality of the residuals failed. Here the Null Hypothesis is the residual
s are normal
with this p value < 0.05 it can be confirmed the residuals are not normal'
normtest(residuals(model))
```

'Test for normality of the residuals failed. Here the Null Hypothesis is the residuals are normal\nwith this p value < 0.05 it can be confirmed the residuals are not normal'

```
Warning message in cvm.test(x):
“p-value is smaller than 7.37e-10, cannot be computed more accurately”
```

A data.frame: 5 × 2

Method	P.value
<chr>	<dbl>
Shapiro-Wilk normality test	4.052384e-14
Anderson-Darling normality test	4.505686e-23
Cramer-von Mises normality test	7.370000e-10
Lilliefors (Kolmogorov-Smirnov) normality test	7.715916e-16
Shapiro-Francia normality test	9.105221e-13

In [333]:
```
'Test for homoskedasticity fails as well'
bptest(model)
```

'Test for homoskedasticity fails as well'

```
	studentized Breusch-Pagan test

data:  model
BP = 409.34, df = 2, p-value < 2.2e-16
```

In [334]:
```
objects =list(A_1,A_2,A_3,A_4,A_5,A_6,A_7,A_8,A_9,A_10,A_11)
```

In [335]:
```
#Repeating same for all 11 Assets
compute_trend <- function(objects)
{

res = c()
j = 0;
for (i in objects)
{
  j=j+1
  la = log(i) # Apply log transformation
  s_la = SMA(la,n=25) #Smoothening using Simple Moving Average
  if(j!= 5)
  {
  model = lm(s_la~t+tsqr) # Buiding an Additive model with quadratic term
  }
  else
  {
    model = lm(s_la~t) # For asset number 5 Tsqr term is insignificant
  }
  rSquared <- summary(model)$r.squared
  pVal <- anova(model)$'Pr(>F)'[1] #Significane of the model
  cat("Asset: ",j,"Has an Rsquare of :",rSquared," and Pvalue :",pVal,"\n")
  if(pVal >0.05)
  {
    cat("The pVal in accordance with Fstatistic for asset :",j," is not significant")
  }
  for(x in summary(model)$coefficients[,4]) #Check Pvalues of coeffiecients
      {
        if(x> 0.05) #Pvalue cutoff
        {
          cat("Not all coefficients are Significant for asset :",j,'\n')
        }
      }
  title <- paste("Question1_plots/Trendplot_asset",j,".jpg")
  jpeg(title) # All trend plots are saved as jpg can be available in Question1_plots
 folder
    pred <- exp(predict.lm(model,s_la)) # applying exp as the ouput is the log of the
result.
    plot(i,col="blue")
    lines(pred,col="red")
  dev.off()
}

}
```

In [336]:
```
compute_trend(objects) #Trend plots for each asset is present in Question1_plots fold
er
```

```
Asset:  1 Has an Rsquare of : 0.6815857  and Pvalue : 4.884668e-199
Asset:  2 Has an Rsquare of : 0.7588357  and Pvalue : 1.039911e-209
Asset:  3 Has an Rsquare of : 0.83344  and Pvalue : 0
Asset:  4 Has an Rsquare of : 0.9020171  and Pvalue : 0
Asset:  5 Has an Rsquare of : 0.7653623  and Pvalue : 0
Asset:  6 Has an Rsquare of : 0.8860482  and Pvalue : 0
Asset:  7 Has an Rsquare of : 0.7549589  and Pvalue : 1.237821e-65
Asset:  8 Has an Rsquare of : 0.5757708  and Pvalue : 7.024079e-126
Asset:  9 Has an Rsquare of : 0.8476692  and Pvalue : 0
Asset:  10 Has an Rsquare of : 0.7497403  and Pvalue : 2.826192e-186
Asset:  11 Has an Rsquare of : 0.5166371  and Pvalue : 3.768046e-25
```

Q2)In any approach you've chosen to take in the previous question (there are many), what do you think are the shortcomings of the approach?

```
Ans)
In Method1 The shortcomings are:
1) Setting n for Simple moving average based smoothening is a trial and error process ther
efore
was not certain on how further to increase.
2) Assumption of Independancy within Xs for the Linear regression model is compromised.
3) The residuals still have patterns left indicating the residuals left are not pure nois
e.
4) Residuals fail to satisfy Normality and homoskedasticity
```

Efficient Frontier

In [375]:
```
library(tidyr)
library(dplyr)
```

In [376]:
```
df <- d %>%
  select(Date_formated, ASSET_1, ASSET_2, ASSET_3, ASSET_4, ASSET_5, ASSET_6, ASSET_7
, ASSET_8, ASSET_9, ASSET_10, ASSET_11) %>%
  gather(key = "variable", value = "value", -Date_formated)
```

In [377]:
```
ggplot(df, aes(x = Date_formated, y = value)) +
  geom_line(aes(color = variable), size = 0.7) +
  theme_minimal()
```

In [378]:
```
summary(d)
```

```
    Dates               ASSET_1         ASSET_2         ASSET_3     
 Length:1147        Min.   : 4518   Min.   :15278   Min.   : 6322  
 Class :character   1st Qu.: 8033   1st Qu.:20620   1st Qu.: 9077  
 Mode  :character   Median : 9023   Median :24817   Median :10531  
                    Mean   : 9138   Mean   :24316   Mean   :10475  
                    3rd Qu.:10750   3rd Qu.:27331   3rd Qu.:11636  
                    Max.   :12010   Max.   :32444   Max.   :14698  
    ASSET_4         ASSET_5         ASSET_6          ASSET_7    
 Min.   :19136   Min.   : 9435   Min.   : 987.2   Min.   :1496  
 1st Qu.:23969   1st Qu.:10810   1st Qu.:1881.5   1st Qu.:2431  
 Median :28342   Median :13505   Median :2554.2   Median :2886  
 Mean   :26968   Mean   :13434   Mean   :2475.6   Mean   :2871  
 3rd Qu.:30002   3rd Qu.:15532   3rd Qu.:3031.4   3rd Qu.:3339  
 Max.   :32912   Max.   :22493   Max.   :3642.7   Max.   :4196  
    ASSET_8         ASSET_9        ASSET_10        ASSET_11    
 Min.   : 6432   Min.   :1087   Min.   : 8581   Min.   :146.6  
 1st Qu.: 8709   1st Qu.:2459   1st Qu.:11418   1st Qu.:205.7  
 Median : 9389   Median :2941   Median :13873   Median :251.8  
 Mean   : 9574   Mean   :2776   Mean   :13576   Mean   :245.5  
 3rd Qu.:10504   3rd Qu.:3233   3rd Qu.:15416   3rd Qu.:280.6  
 Max.   :12321   Max.   :4069   Max.   :17914   Max.   :372.2  
 Date_formated       
 Min.   :2016-03-14  
 1st Qu.:2017-05-13  
 Median :2018-07-05  
 Mean   :2018-07-09  
 3rd Qu.:2019-09-05  
 Max.   :2020-10-30  
```

3. Given the set of 11 assets, plot the Efficient Frontier, assuming a Risk Free Rate of 0%.

```
First step is to compute the expected returns and sd(returns):
Returns could be Simple returns or
Log returns or continously compounded returns (Assuming any gains made is reinvested)

I am going to use simple returns
Given as
Rt = (R(t)-R(t-1))/R(t-1) or R(t)/R(t-1) - 1

The time period to compute returns is taken as Quaterly returns
ie)
1 week = 5 working days
1 Month = 20 Working days
1 Quarter = 3 Months or 60 working days

(Note: Holidays are ignored. Returns will be calculated with 60 working days intervals)
```

In [403]:
```
idx <- seq(1,1147,59) # Generating indexes from 1 separated by 60 days
prices<-d[idx,c(13,2:12)] # Just reordered columns along with using the indexes
prices
```

A data.frame: 20 × 12

	Date_formated	ASSET_1	ASSET_2	ASSET_3	ASSET_4	ASSET_5	ASSET_6	ASSET_7	ASSET_8
	<date>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	2016-03-14	7777.15	15277.80	6374.15	19535.15	10866.85	2343.00	1843.85	11719.20
60	2016-06-10	8688.30	17828.60	7236.10	20704.50	11229.05	2590.85	2113.75	10657.60
119	2016-09-06	10348.55	20426.20	8385.85	22972.80	10545.30	2961.40	2607.05	11612.70
178	2016-12-02	8888.70	18247.65	7433.35	20179.85	9922.75	2554.20	2717.60	11038.05
237	2017-02-27	9666.85	20613.05	8302.75	22419.60	10706.70	2969.20	3110.90	10571.70
296	2017-05-25	10588.10	23190.80	9348.75	25056.65	10738.35	3034.95	2899.80	9199.15
355	2017-08-18	10676.35	24074.45	9942.35	25718.70	10570.00	2948.45	3384.65	8639.40
414	2017-11-14	11156.70	25284.60	10292.45	25579.50	11168.75	3159.45	3772.75	9208.45
473	2018-02-07	11214.75	25670.00	10539.95	26733.55	12450.05	3344.55	3879.75	8935.80
532	2018-05-08	11437.20	26090.50	10864.55	28555.15	13379.65	3355.45	3689.10	8753.25
591	2018-07-30	10925.05	27842.60	11738.85	30836.60	14420.55	2839.95	3320.30	9115.55
650	2018-10-29	8723.90	24959.70	10472.45	28257.50	14132.65	2424.20	3369.50	9590.40
709	2019-01-23	8612.25	27250.75	11556.60	29991.40	14888.75	2373.75	2896.90	8946.20
768	2019-04-22	8813.90	29687.95	12347.05	30418.10	16150.60	2358.60	3054.10	9285.45
827	2019-07-17	7648.15	30735.50	13485.30	29614.25	15675.95	2067.45	2848.80	8330.75
886	2019-10-16	7673.80	28538.80	12978.40	30899.00	15344.90	1752.25	2388.30	7475.35
945	2020-01-10	8225.50	32097.40	14609.65	30370.45	15959.90	1770.65	2840.75	8058.50
1004	2020-04-07	4951.55	19062.50	9278.30	28745.30	12588.95	1045.50	1637.00	8124.85
1063	2020-07-03	6971.05	21852.40	10870.90	30529.10	15285.90	1370.45	2008.50	9963.40
1122	2020-09-24	7418.85	20456.85	10134.80	29044.95	18973.00	1435.45	2111.85	11334.60

In [426]:
```
returns<- data.frame(indx = 1:19) # Initializing a dataframe to hold the returns of a
ssets data
```

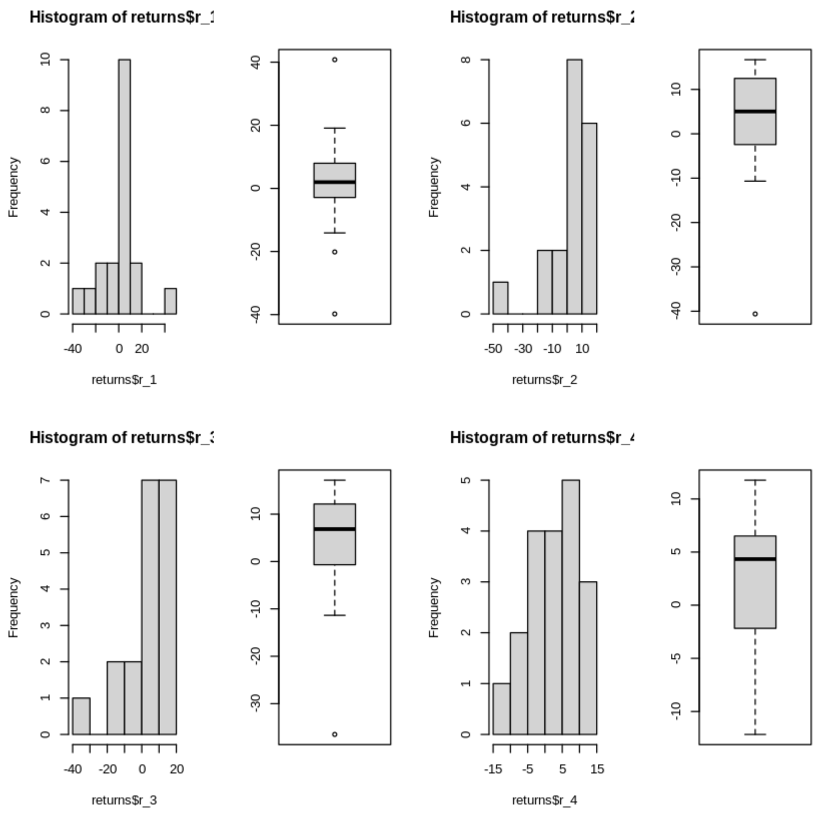
In [475]:
```
returns$r_1 <- (prices$ASSET_1[-1] / prices$ASSET_1[-length(prices$ASSET_1)] - 1 )
returns$r_2 <- (prices$ASSET_2[-1] / prices$ASSET_2[-length(prices$ASSET_2)] - 1 )
returns$r_3 <- (prices$ASSET_3[-1] / prices$ASSET_3[-length(prices$ASSET_3)] - 1 )
returns$r_4 <- (prices$ASSET_4[-1] / prices$ASSET_4[-length(prices$ASSET_4)] - 1 )
returns$r_5 <- (prices$ASSET_5[-1] / prices$ASSET_5[-length(prices$ASSET_5)] - 1 )
returns$r_6 <- (prices$ASSET_6[-1] / prices$ASSET_6[-length(prices$ASSET_6)] - 1 )
returns$r_7 <- (prices$ASSET_7[-1] / prices$ASSET_7[-length(prices$ASSET_7)] - 1 )
returns$r_8 <- (prices$ASSET_8[-1] / prices$ASSET_8[-length(prices$ASSET_8)] - 1 )
returns$r_9 <- (prices$ASSET_9[-1] / prices$ASSET_9[-length(prices$ASSET_9)] - 1 )
returns$r_10 <- (prices$ASSET_10[-1] / prices$ASSET_10[-length(prices$ASSET_10)] - 1
)
returns$r_11 <- (prices$ASSET_11[-1] / prices$ASSET_11[-length(prices$ASSET_11)] - 1
)
```

In [476]:
```
returns
```

A data.frame: 19 × 12

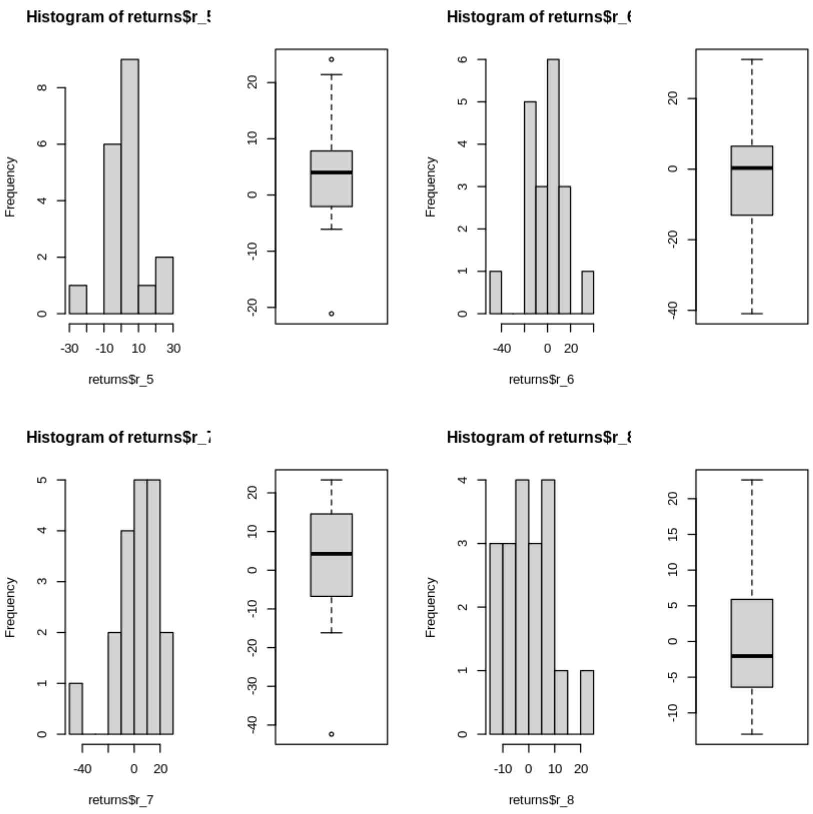
indx	r_1	r_2	r_3	r_4	r_5	r_6	r_7	
<int>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	
1	0.117157313	0.16696121	0.13522587	0.059858767	0.033330726	0.105783184	0.14637850	-0.0
2	0.191090317	0.14569848	0.15889084	0.109555894	-0.060891171	0.143022560	0.23337670	0.0
3	-0.141068072	-0.10665469	-0.11358419	-0.121576386	-0.059035779	-0.137502533	0.04240425	-0.0
4	0.087543735	0.12962765	0.11695938	0.110989428	0.079005316	0.162477488	0.14472329	-0.0
5	0.095299917	0.12505427	0.12598236	0.117622527	0.002956093	0.022144012	-0.06785818	-0.1
6	0.008334829	0.03810347	0.06349512	0.026422127	-0.015677455	-0.028501293	0.16720119	-0.0
7	0.044991968	0.05026698	0.03521300	-0.005412404	0.056646168	0.071563025	0.11466474	0.0
8	0.005203151	0.01524248	0.02404675	0.045116206	0.114721880	0.058586146	0.02836127	-0.0
9	0.019835485	0.01638099	0.03079711	0.068139099	0.074666367	0.003259033	-0.04913976	-0.0
10	-0.044779317	0.06715471	0.08047273	0.079896271	0.077797252	-0.153630661	-0.09997018	0.0
11	-0.201477339	-0.10354277	-0.10788110	-0.083637625	-0.019964564	-0.146393422	0.01481794	0.0
12	-0.012798175	0.09178997	0.10352401	0.061360701	0.053500228	-0.020810989	-0.14025820	-0.0
13	0.023414323	0.08943607	0.06839814	0.014227412	0.084751910	-0.006382306	0.05426490	0.0
14	-0.132262676	0.03528536	0.09218801	-0.026426700	-0.029389001	-0.123441872	-0.06722111	-0.1
15	0.003353752	-0.07147110	-0.03758908	0.043382831	-0.021118337	-0.152458342	-0.16164701	-0.1
16	0.071893977	0.12469340	0.12568961	-0.017105732	0.040078463	0.010500785	0.18944437	0.0
17	-0.398024436	-0.40610454	-0.36491976	-0.053510896	-0.211213729	-0.409538870	-0.42374373	0.0
18	0.407852087	0.14635541	0.17164782	0.062055362	0.214231528	0.310808226	0.22693952	0.2
19	0.064237095	-0.06386255	-0.06771288	-0.048614273	0.241209219	0.047429676	0.05145631	0.1

In [442]:
```
par(mfrow=c(2,4))
hist(returns$r_1)
boxplot(returns$r_1)
hist(returns$r_2)
boxplot(returns$r_2)
hist(returns$r_3)
boxplot(returns$r_3)
hist(returns$r_4)
boxplot(returns$r_4)
```



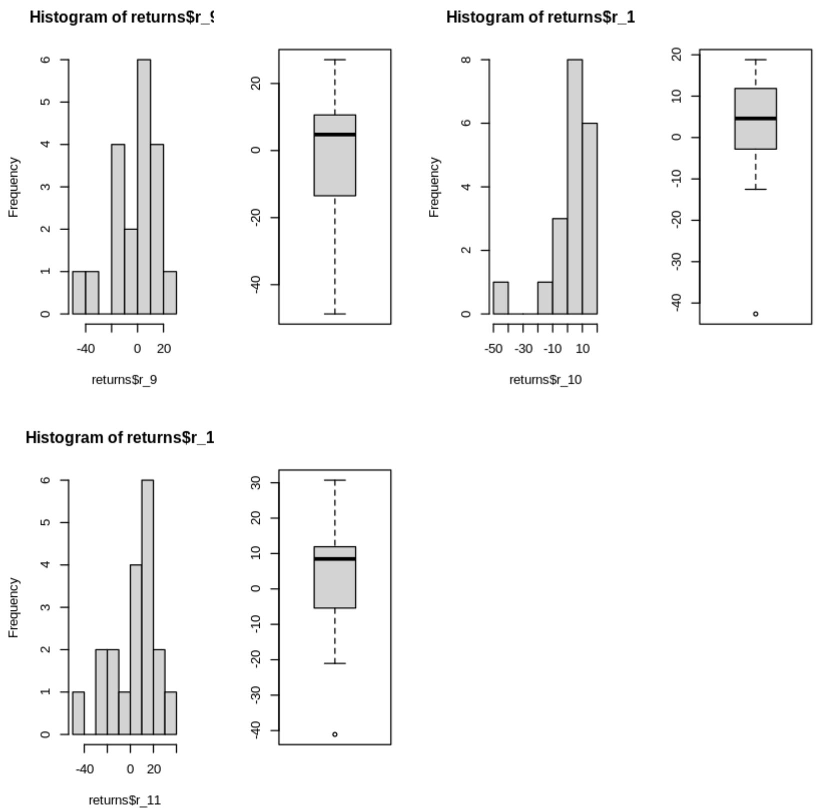
In [443]:

```
par(mfrow=c(2,4))
hist(returns$r_5)
boxplot(returns$r_5)
hist(returns$r_6)
boxplot(returns$r_6)
hist(returns$r_7)
boxplot(returns$r_7)
hist(returns$r_8)
boxplot(returns$r_8)
```



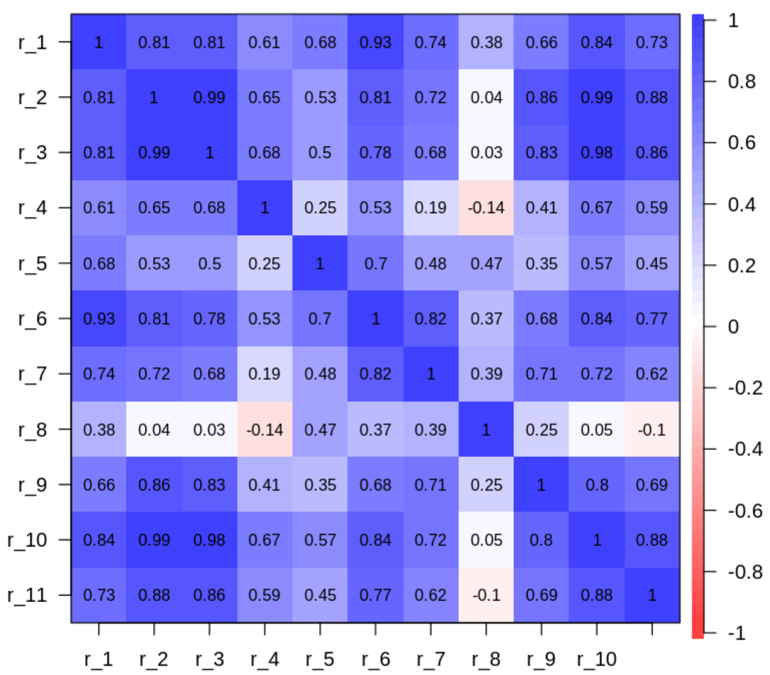
In [444]:

```
par(mfrow=c(2,4))
hist(returns$r_9)
boxplot(returns$r_9)
hist(returns$r_10)
boxplot(returns$r_10)
hist(returns$r_11)
boxplot(returns$r_11)
```

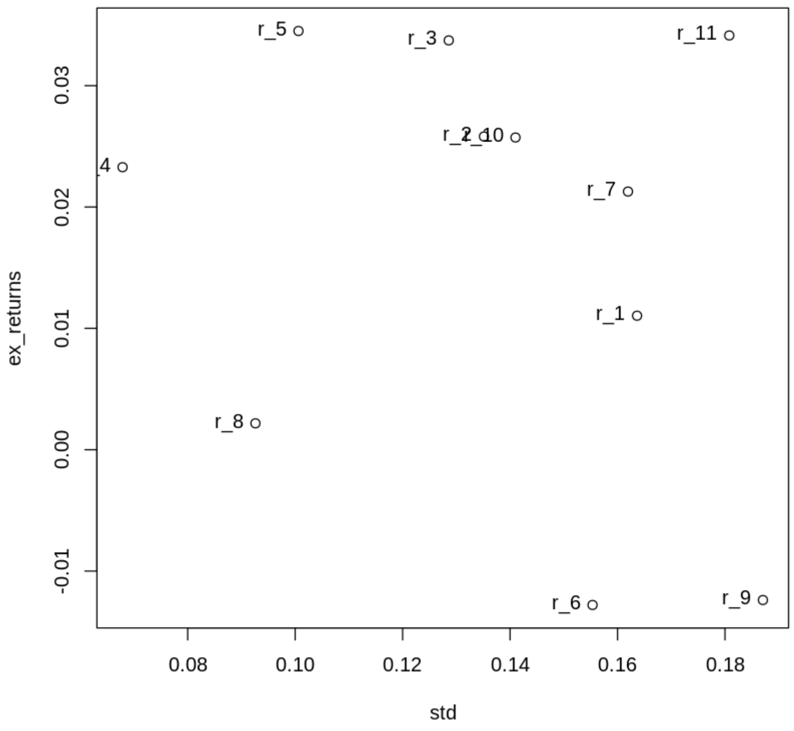


In [496]:

```
cor.plot(cor(returns[,2:12]),numbers=T) #Correlation plot to check how the assets mov
e
```



In [508]:
```
ex_returns = colMeans(returns[,2:12])
std = apply(returns[2:12],2,sd)
plot(std,ex_returns)
text(std, ex_returns, labels=names(ex_returns), pos=2)
```



Removing asset 9 and asset 6 as its expected returns are negative

In [594]:
```
r<-returns[2:12][-c(6,9)]
```

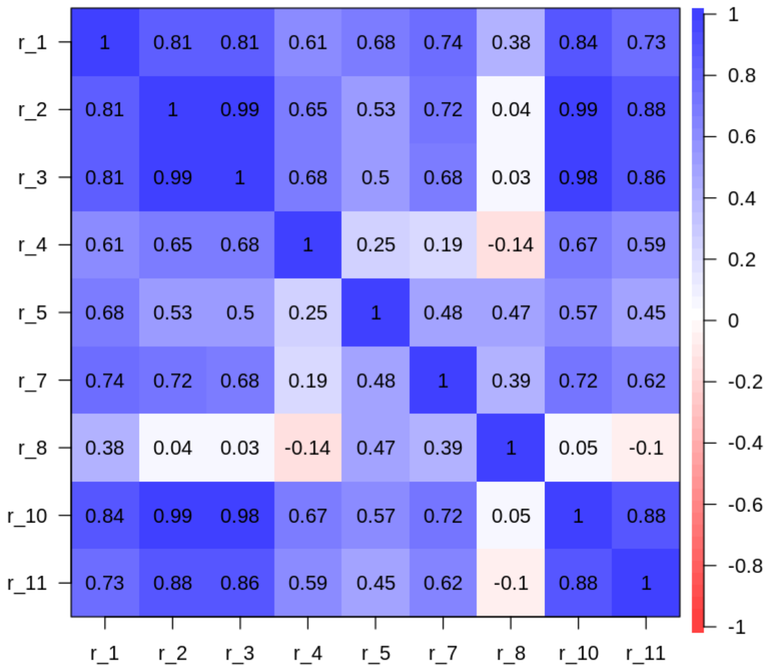
In [595]:

```
r
```

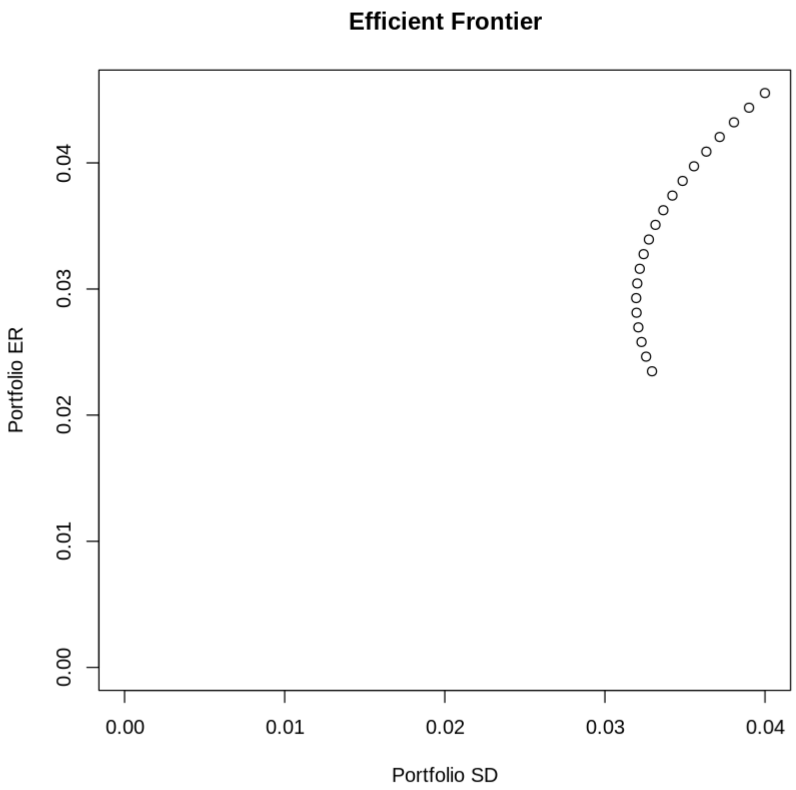
A data.frame: 19 × 9

r_1	r_2	r_3	r_4	r_5	r_7	r_8	r_
<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<d
0.117157313	0.16696121	0.13522587	0.059858767	0.033330726	0.14637850	-0.090586388	0.18813363
0.191090317	0.14569848	0.15889084	0.109555894	-0.060891171	0.23337670	0.089616799	0.12802757
-0.141068072	-0.10665469	-0.11358419	-0.121576386	-0.059035779	0.04240425	-0.049484616	-0.12515650
0.087543735	0.12962765	0.11695938	0.110989428	0.079005316	0.14472329	-0.042249310	0.13631823
0.095299917	0.12505427	0.12598236	0.117622527	0.002956093	-0.06785818	-0.129832477	0.12435550
0.008334829	0.03810347	0.06349512	0.026422127	-0.015677455	0.16720119	-0.060848013	0.04774351
0.044991968	0.05026698	0.03521300	-0.005412404	0.056646168	0.11466474	0.065866843	0.03124976
0.005203151	0.01524248	0.02404675	0.045116206	0.114721880	0.02836127	-0.029608675	0.03454693
0.019835485	0.01638099	0.03079711	0.068139099	0.074666367	-0.04913976	-0.020429061	0.04220701
-0.044779317	0.06715471	0.08047273	0.079896271	0.077797252	-0.09997018	0.041390341	0.04609328
-0.201477339	-0.10354277	-0.10788110	-0.083637625	-0.019964564	0.01481794	0.052092304	-0.09848753
-0.012798175	0.09178997	0.10352401	0.061360701	0.053500228	-0.14025820	-0.067171338	0.09460473
0.023414323	0.08943607	0.06839814	0.014227412	0.084751910	0.05426490	0.037921129	0.09168383
-0.132262676	0.03528536	0.09218801	-0.026426700	-0.029389001	-0.06722111	-0.102816772	0.00082656
0.003353752	-0.07147110	-0.03758908	0.043382831	-0.021118337	-0.16164701	-0.102679831	-0.05676395
0.071893977	0.12469340	0.12568961	-0.017105732	0.040078463	0.18944437	0.078009725	0.11279358
-0.398024436	-0.40610454	-0.36491976	-0.053510896	-0.211213729	-0.42374373	0.008233542	-0.42640551
0.407852087	0.14635541	0.17164782	0.062055362	0.214231528	0.22693952	0.226287255	0.17879619
0.064237095	-0.06386255	-0.06771288	-0.048614273	0.241209219	0.05145631	0.137623703	-0.06175939

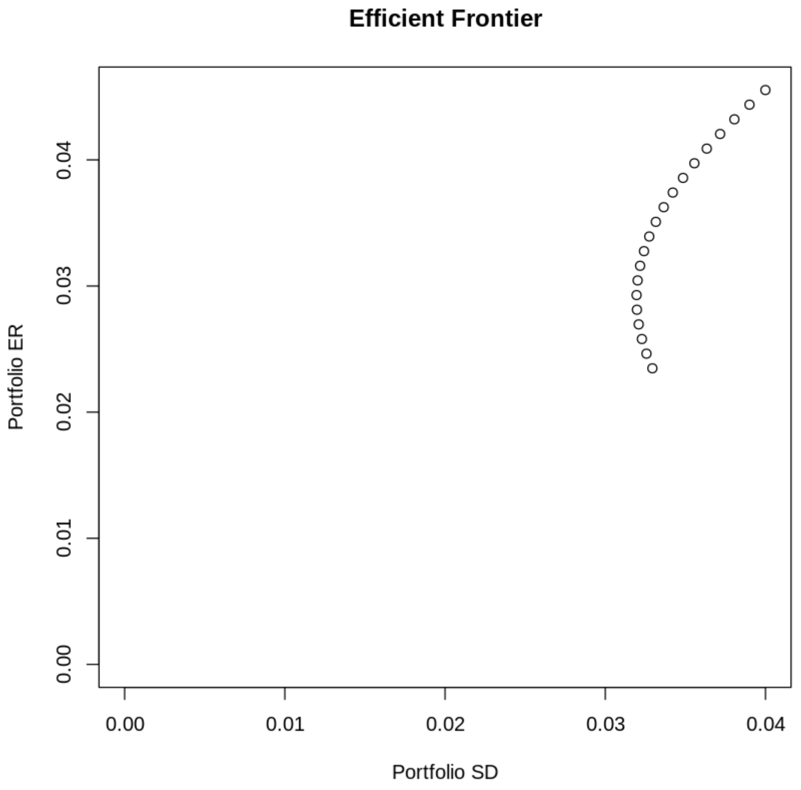
In [596]:
```
cor.plot(cor(r),numbers=T) #Correlation plot to check how the assets move
```



In [597]:
```
er = colMeans(r)
covmat = cov(r)
r.free = 0.00
# compute portfolio frontier
ef_woshort <- efficient.frontier(er, covmat,  nport=20,shorts= FALSE) #Without shorti
ng
plot(ef)
```



In [598]:
```
er = colMeans(r)
covmat = cov(r)
r.free = 0.00
# compute portfolio frontier
ef_wshort <- efficient.frontier(er, covmat, alpha.min=-2,
                        alpha.max=2,  nport=20,shorts= TRUE) #With shorting
plot(ef)
```



4. For the same 11 assets, identify the following portfolios:

a. Minimum Variance Portfolio

b. Tangency Portfolio

In [599]:
```
# tangency portfolio without shorting
tan1.port <- tangency.portfolio(er, covmat, r.free,shorts= FALSE)
# compute global minimum variance portfolio without shorting
gmin1.port = globalMin.portfolio(er, covmat,shorts = FALSE)
```

In [600]:
```
print(tan1.port$weights)
```

```
     r_1      r_2      r_3      r_4      r_5      r_7      r_8     r_10
0.000000 0.000000 0.000000 0.597232 0.402768 0.000000 0.000000 0.000000
    r_11
0.000000
```

In [601]:
```
print(gmin1.port$weights)
```

```
     r_1      r_2      r_3      r_4      r_5      r_7      r_8     r_10
0.000000 0.000000 0.000000 0.632648 0.000000 0.000000 0.367352 0.000000
    r_11
0.000000
```

In [602]:
```
# tangency portfolio with shorting
tan2.port <- tangency.portfolio(er, covmat, r.free,shorts= TRUE)
# compute global minimum variance portfolio with shorting
gmin2.port = globalMin.portfolio(er, covmat,shorts = TRUE)
```

In [603]:
```
print(tan2.port$weights)
```

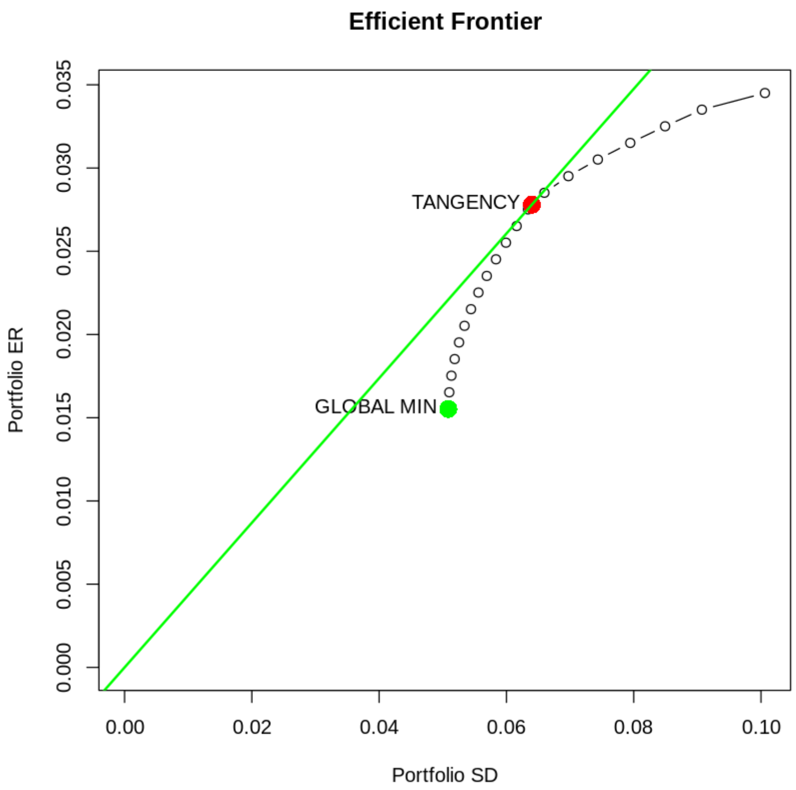
```
         r_1          r_2          r_3          r_4          r_5          r_7 
-0.506325822 -1.197308503  1.367656745  0.794454345  0.489295574  0.381848567 
         r_8         r_10         r_11 
-0.004135741 -0.431583657  0.106098491 
```

In [604]:
```
print(gmin2.port$weights)
```

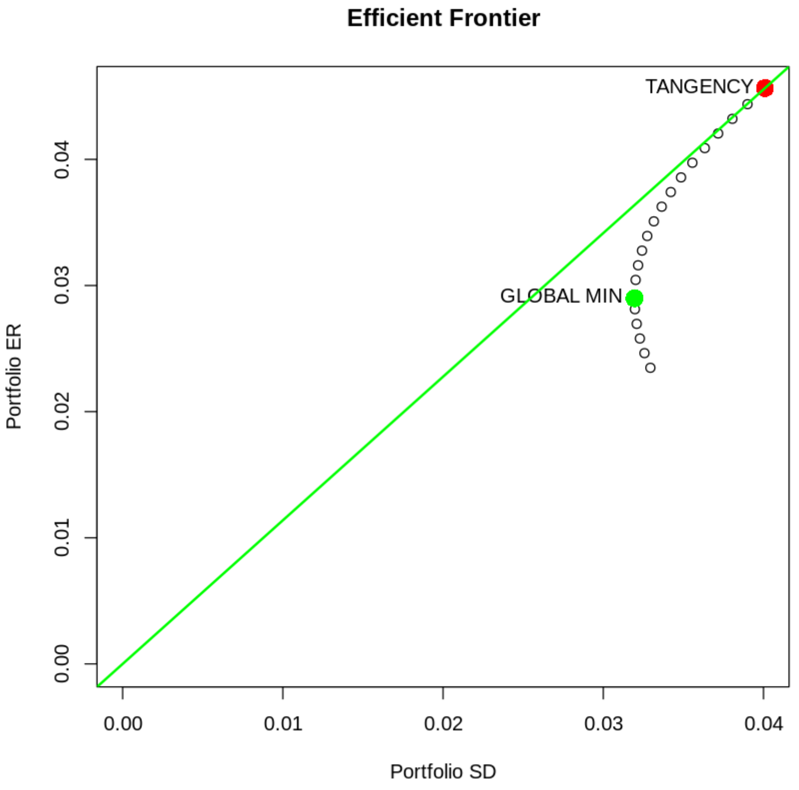
```
        r_1         r_2         r_3         r_4         r_5         r_7 
-0.39657060 -0.19957670  0.35530815  0.79330847  0.24676136  0.23333098 
        r_8        r_10        r_11 
 0.22363242 -0.33865065  0.08245657 
```

5. Show the above 2 portfolios on the Efficient Frontier plot.

In [605]:
```
#Effcient Frontier without shorting
plot(ef_woshort)
points(gmin1.port$sd, gmin1.port$er, col="green", pch=16, cex=2) #marking the global
 minimum portfolio green
points(tan1.port$sd, tan1.port$er, col="red", pch=16, cex=2) #marking the tangency po
rtfolio red
text(gmin1.port$sd, gmin1.port$er, labels="GLOBAL MIN", pos=2) #Marking labels
text(tan1.port$sd, tan1.port$er, labels="TANGENCY", pos=2)
sr.tan1 = (tan1.port$er - r.free)/tan1.port$sd # calculating sharpe's ratio
abline(a=r.free, b=sr.tan1, col="green", lwd=2)
```



In [606]:
```
#Effcient Frontier with shorting
plot(ef_wshort)
points(gmin2.port$sd, gmin2.port$er, col="green", pch=16, cex=2) #marking the global
 minimum portfolio green
points(tan2.port$sd, tan2.port$er, col="red", pch=16, cex=2) #marking the tangency po
rtfolio red
text(gmin2.port$sd, gmin2.port$er, labels="GLOBAL MIN", pos=2) #Marking labels
text(tan2.port$sd, tan2.port$er, labels="TANGENCY", pos=2)
sr.tan2 = (tan2.port$er - r.free)/tan2.port$sd # calculating sharpe's ratio
abline(a=r.free, b=sr.tan2, col="green", lwd=2)
```



6. Write a short note on what aspect of this assignment did you find challenging (if any).

1. For the first part where we were asked to estimate the trend and plot it. I was stuck with looking at many methods. There were non-Parameteric methods like kalman's filtering, cubicspline. Since i haven't used them and couldnt understand the math underneath given the time constraint didnt use them. Turned back to parameteric method or regression analysis.
2. Had to think a lot on how to compute the returns for each of the asset (in terms of what duration to use like yearly,monthly,quaterly)
3. It didnt seem good to add a asset with negative return to the portfolio which could have been beneficial with shorting. Would love to hear one of your views on what to do in such occurances.