Data Exploration on the Returns of 27 Stocks

2023-10-29

Data Preprocessing

Load the required packages

```
library(quantmod)
library(tidyverse)
library(Hmisc)
library(moments)
library(reshape2)
library(fGarch)
```

Define the stock symbol and date range

```
tickers <- c("AAPL", "AMGN", "AXP", "BA", "CAT", "CRM", "CSCO", "CVX", "DIS", "DOW",
             "GS", "HD", "HON", "IBM", "INTC", "JNJ", "JPM", "KO", "MCD", "MMM",
             "MRK", "MSFT", "NKE", "PG", "TRV", "UNH", "V", "VZ", "WBA", "WMT")
start_date <- as.Date("2000-01-01")
end_date <- as.Date("2020-12-31")
```

Fetch the stock prices

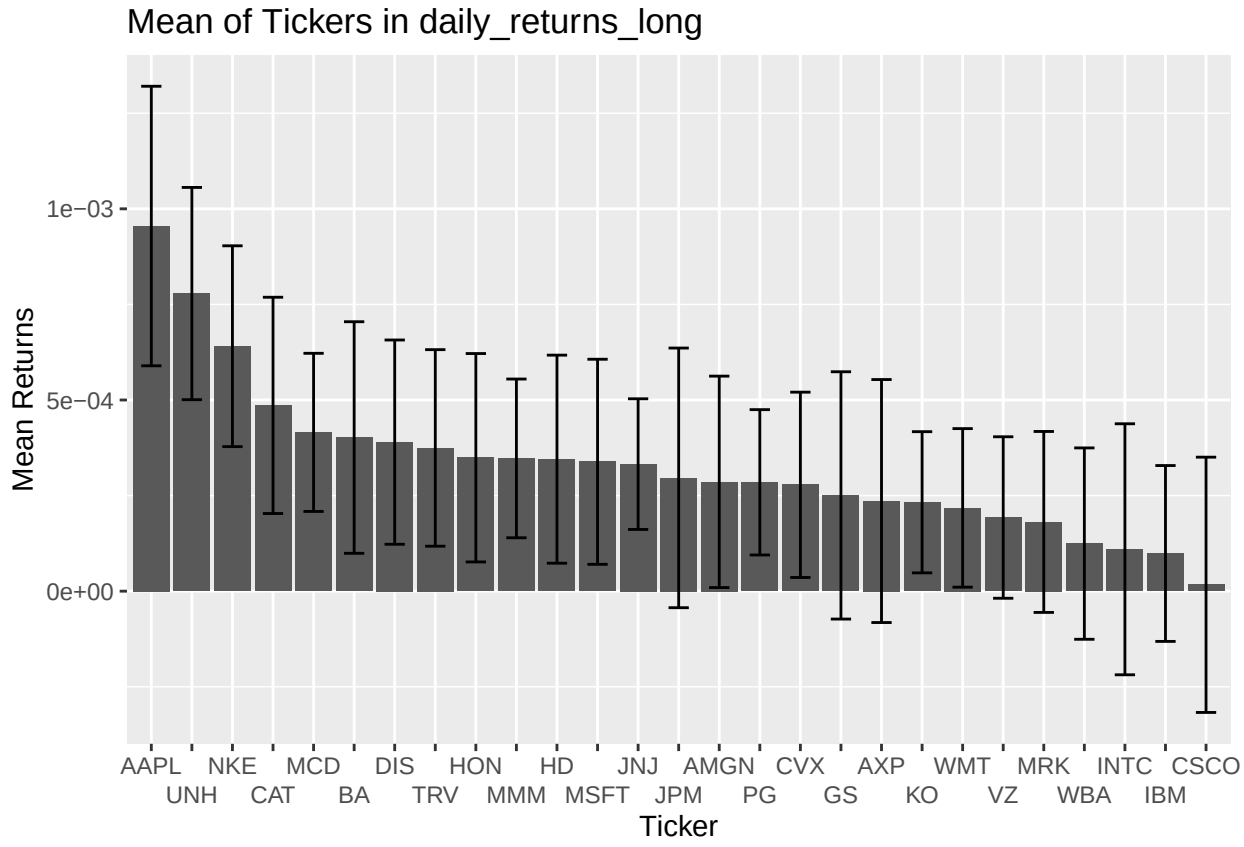
```
closing_prices <- lapply(tickers, function(ticker) {
  getSymbols(ticker, src = 'yahoo', from = start_date, to = end_date, auto.assign = FALSE)[,6]
})

closing_prices <- as_tibble(do.call(cbind, closing_prices))

date <- index(getSymbols("AAPL", src = 'yahoo', from = start_date, to = end_date, auto.assign = FALSE))
closing_prices <- cbind(date, closing_prices)

# Remove columns with NA & clean column names
closing_prices <- closing_prices[ , colSums(is.na(closing_prices))==0]

names(closing_prices)[-1] <- substr(names(closing_prices)[-1], 1, nchar(names(closing_prices)[-1]) - 9)

# Calculate daily returns
daily_returns <- closing_prices %>%
  mutate_at(vars(-1), ~log(.) - log(lag(.))) %>%
```

```
  na.omit()

days <- nrow(daily_returns)

daily_returns_long <- pivot_longer(daily_returns, cols = -1, names_to = "ticker", values_to = "returns")
```

Exploratory Data Analysis

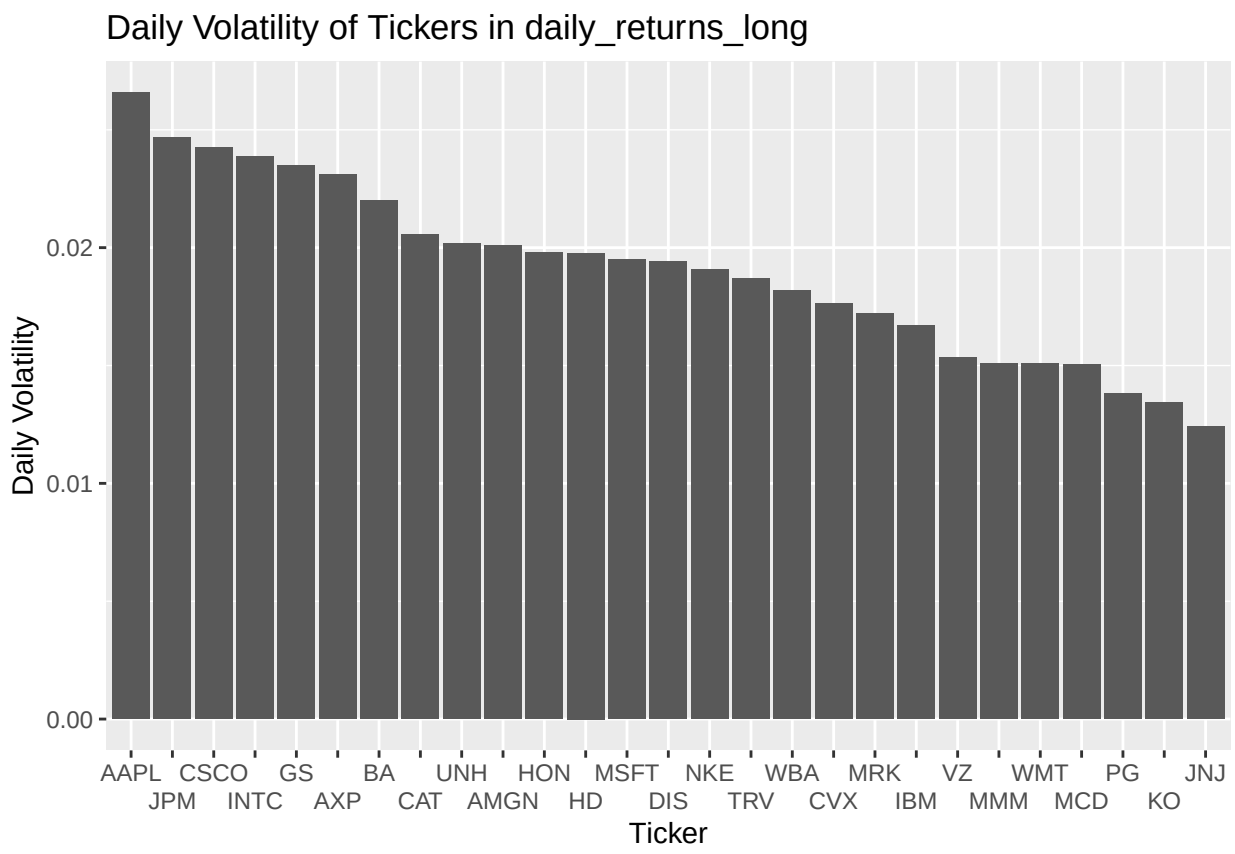
First Four Moments

```
ggplot(daily_returns_long, aes(x = reorder(ticker, -returns), y = returns)) +
  geom_bar(stat = "summary", fun = "mean") +
  labs(title = "Mean of Tickers in daily_returns_long", x = "Ticker", y = "Mean Returns") +
  stat_summary(fun.data = mean_sdl, geom = "errorbar", width = .5, fun.args = list(mult = 1/sqrt(days))) +
  scale_x_discrete(guide = guide_axis(n.dodge=2))
```

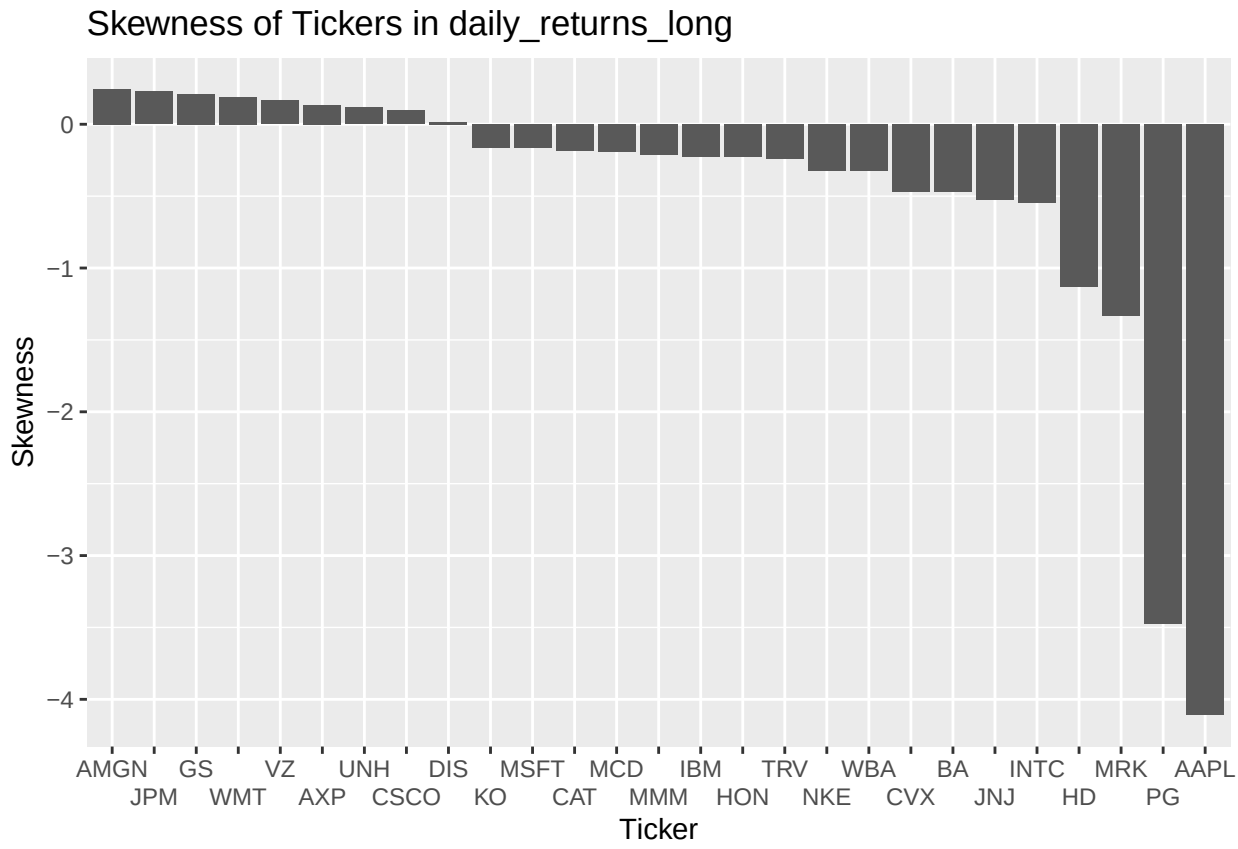



```
returns_moments <- daily_returns_long %>%
  group_by(ticker) %>%
  summarise(mean = mean(returns),
            sd = sd(returns),
            skewness = skewness(returns),
            kurtosis = kurtosis(returns))
```

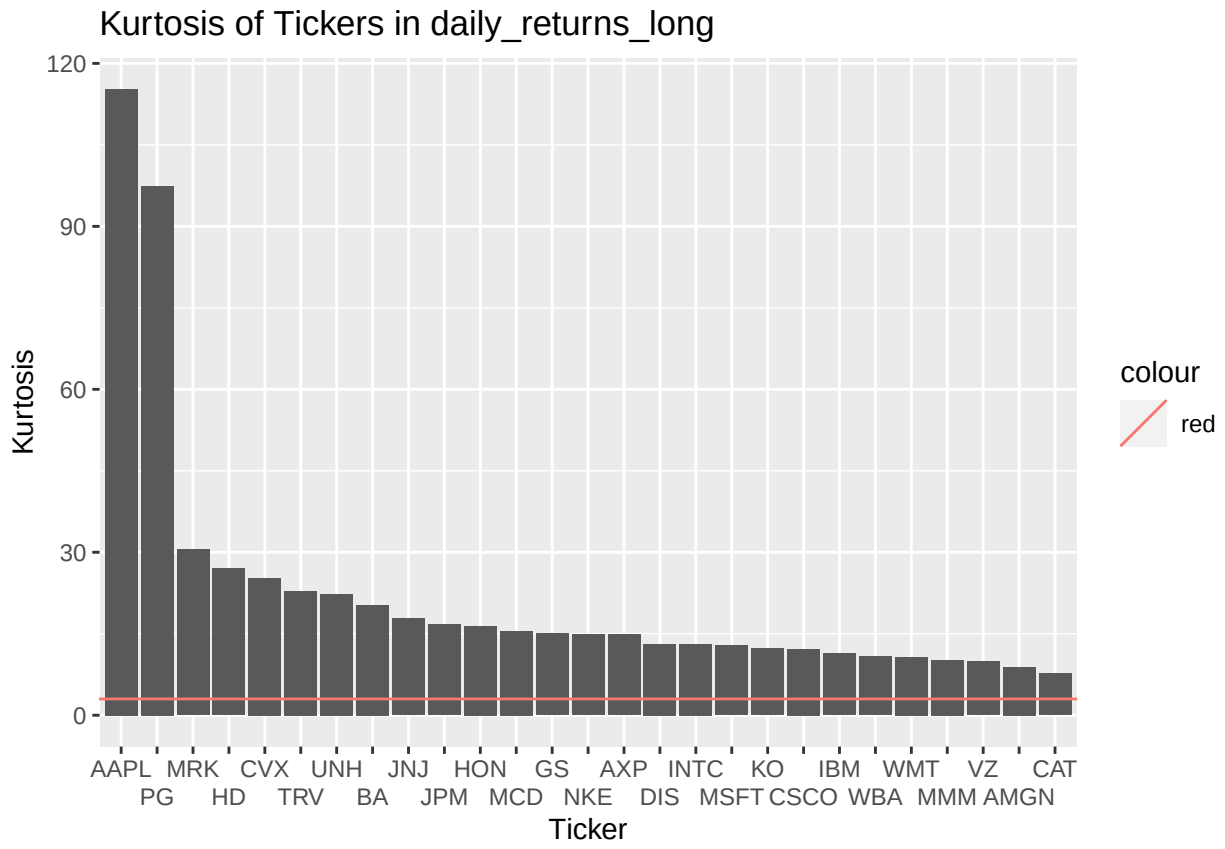
```
ggplot(returns_moments, aes(x = reorder(ticker, -sd), y = sd)) +
  geom_col() +
  labs(title = "Daily Volatility of Tickers in daily_returns_long", x = "Ticker", y = "Daily Volatility
  scale_x_discrete(guide = guide_axis(n.dodge=2))
```




```
ggplot(returns_moments, aes(x = reorder(ticker, -skewness), y = skewness)) +
  geom_col() +
  labs(title = "Skewness of Tickers in daily_returns_long", x = "Ticker", y = "Skewness") +
  scale_x_discrete(guide = guide_axis(n.dodge=2))
```

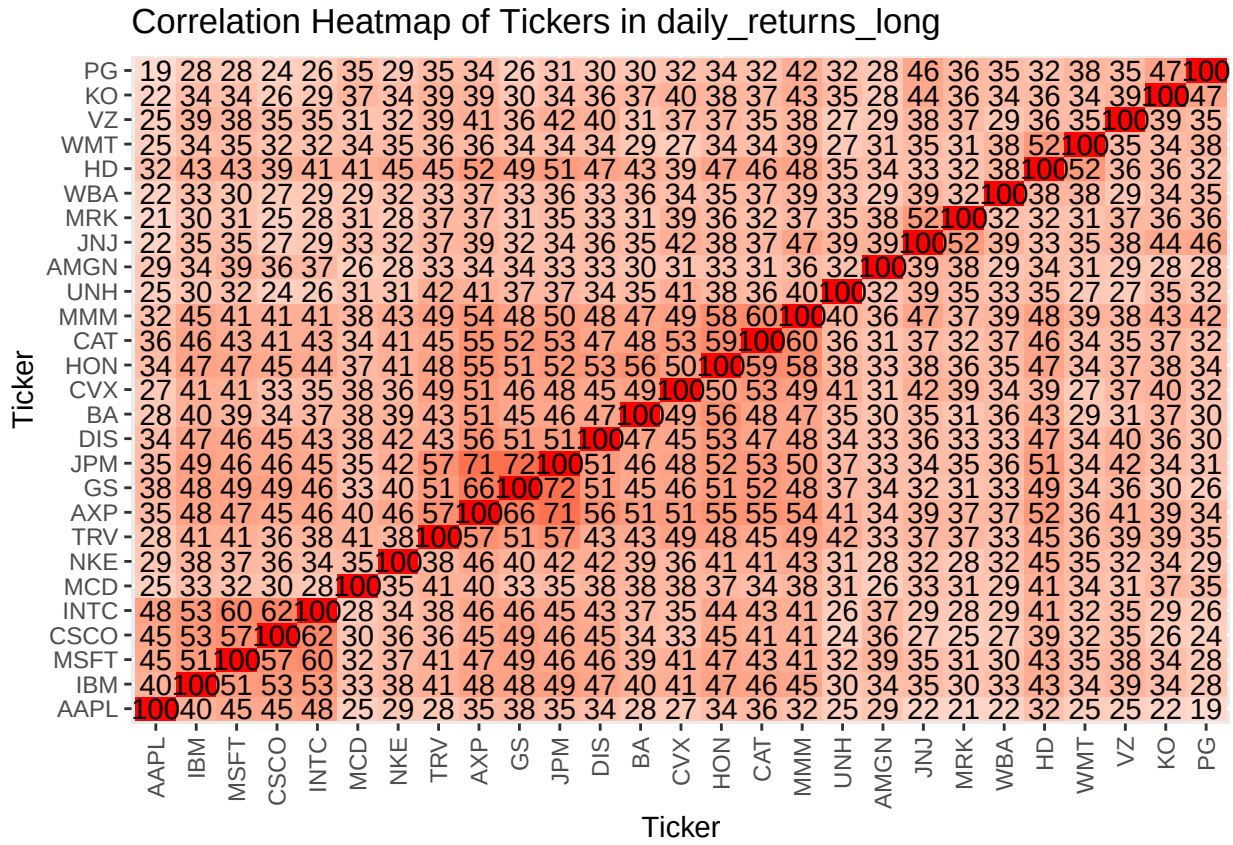



```
ggplot(returns_moments, aes(x = reorder(ticker, -kurtosis), y = kurtosis)) +
  geom_col() +
  labs(title = "Kurtosis of Tickers in daily_returns_long", x = "Ticker", y = "Kurtosis") +
  geom_abline(aes(slope = 0, intercept = 3, color = "red")) +
  scale_x_discrete(guide = guide_axis(n.dodge=2))
```



Correlation between returns

```r
# Calculate correlation matrix
cor_matrix <- cor(daily_returns[, -1])

# Reorder rows and columns by hierarchical clustering
hc_rows <- hclust(as.dist(1 - cor_matrix))
hc_cols <- hclust(as.dist(1 - t(cor_matrix)))
cor_matrix_reordered <- cor_matrix[hc_rows$order, hc_cols$order]

# Create heatmap
ggplot(melt(cor_matrix_reordered), aes(x = Var1, y = Var2, fill = value)) +
  geom_tile() +
  scale_fill_gradient2(low = "blue", mid = "white", high = "red", midpoint = 0) +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  labs(title = "Correlation Heatmap of Tickers in daily_returns_long", x = "Ticker", y = "Ticker") +
  geom_text(aes(label=round(value*100))) +
  guides(fill = "none")
```



Helper functions

```
# function to fetch the correlation matrix given start and end dates


# function to get prediction of standard deviations given start and end dates
```

Fit Individual $Garch(1,1)$ Models

```
fit <- garchFit(data = daily_returns$AAPL)
```

```
##
## Series Initialization:
##  ARMA Model:                 arma
##  Formula Mean:               ~ arma(0, 0)
##  GARCH Model:                garch
##  Formula Variance:           ~ garch(1, 1)
##  ARMA Order:                 0 0
##  Max ARMA Order:             0
##  GARCH Order:                1 1
```

```
##  Max GARCH Order:           1
##  Maximum Order:             1
##  Conditional Dist:          norm
##  h.start:                   2
##  llh.start:                 1
##  Length of Series:          5282
##  Recursion Init:            mci
##  Series Scale:              0.02656627
##
## Parameter Initialization:
##  Initial Parameters:          $params
##  Limits of Transformations:   $U, $V
##  Which Parameters are Fixed?  $includes
##  Parameter Matrix:
##                      U         V   params includes
##     mu     -0.35946004   0.35946 0.035946     TRUE
##     omega   0.00000100 100.00000 0.100000     TRUE
##     alpha1  0.00000001   1.00000 0.100000     TRUE
##     gamma1 -0.99999999   1.00000 0.100000    FALSE
##     beta1   0.00000001   1.00000 0.800000     TRUE
##     delta   0.00000000   2.00000 2.000000    FALSE
##     skew    0.10000000  10.00000 1.000000    FALSE
##     shape   1.00000000  10.00000 4.000000    FALSE
##  Index List of Parameters to be Optimized:
##     mu  omega alpha1  beta1
##      1      2      3      5
##  Persistence:                 0.9
##
##
## --- START OF TRACE ---
## Selected Algorithm: nlminb
##
## R coded nlminb Solver:
##
##   0:     6713.5472: 0.0359460 0.100000 0.100000 0.800000
##   1:     6632.1827: 0.0359479 0.0732812 0.103908 0.788065
##   2:     6573.7512: 0.0359519 0.0641143 0.129782 0.798933
##   3:     6572.1707: 0.0359564 0.0355571 0.136201 0.795078
##   4:     6529.5713: 0.0359578 0.0457899 0.143315 0.802989
##   5:     6512.7043: 0.0359631 0.0354149 0.151679 0.809336
##   6:     6507.9232: 0.0359665 0.0359301 0.156529 0.815457
##   7:     6503.9257: 0.0359720 0.0283751 0.155942 0.817410
##   8:     6499.0186: 0.0359821 0.0291555 0.155925 0.825193
##   9:     6495.7413: 0.0359943 0.0238199 0.153741 0.830476
##  10:     6493.0356: 0.0360110 0.0236222 0.152025 0.838096
##  11:     6490.7998: 0.0360359 0.0196201 0.148437 0.843744
##  12:     6489.1965: 0.0360838 0.0189570 0.145261 0.850741
##  13:     6488.1321: 0.0361962 0.0162154 0.141168 0.855959
##  14:     6487.3455: 0.0364337 0.0165534 0.138712 0.859343
##  15:     6483.3211: 0.0407996 0.0126066 0.112087 0.881903
##  16:     6483.0930: 0.0453396 0.0163171 0.114242 0.878100
##  17:     6480.6455: 0.0476100 0.0140274 0.115355 0.878768
##  18:     6479.8278: 0.0522621 0.0126185 0.118024 0.880869
##  19:     6479.0158: 0.0569142 0.0123441 0.119153 0.877357
```

```
##  20:     6477.7124: 0.0615665 0.0139841 0.121500 0.875352
##  21:     6477.1521: 0.0662192 0.0139128 0.123315 0.872861
##  22:     6476.8118: 0.0708732 0.0144253 0.123657 0.872667
##  23:     6476.5027: 0.0728760 0.0133730 0.116367 0.879315
##  24:     6476.4571: 0.0749097 0.0134789 0.117166 0.878245
##  25:     6476.4561: 0.0752965 0.0135064 0.117233 0.878115
##  26:     6476.4561: 0.0752979 0.0135032 0.117237 0.878117
##  27:     6476.4561: 0.0752961 0.0135039 0.117236 0.878117
##
## Final Estimate of the Negative LLH:
##  LLH:  -12687.24    norm LLH:  -2.401976
##           mu        omega       alpha1        beta1
## 2.000337e-03 9.530590e-06 1.172358e-01 8.781167e-01
##
## R-optimhess Difference Approximated Hessian Matrix:
##                   mu         omega        alpha1         beta1
## mu      -15555356.17 -2.901144e+08     -13027.47     -78771.79
## omega  -290114397.57 -1.998820e+12 -342863998.23 -554950356.95
## alpha1     -13027.47 -3.428640e+08    -123066.53    -156724.72
## beta1      -78771.79 -5.549504e+08    -156724.72    -223025.63
## attr(,"time")
## Time difference of 0.08769608 secs
##
## --- END OF TRACE ---
##
##
## Time to Estimate Parameters:
##  Time difference of 0.345458 secs
```

```
pred <- predict(fit, n.ahead = 10)
```