Project 1

2025-03-09

Project 1 - Team 12

Needed packages

Exercise 4: Market Portfolio Returns

Download monthly Fama-French data

```r
start_date <- "1927-01-01"
end_date <- "2024-12-31"

#monthly
factors_ff_monthly_raw <- download_french_data("Fama/French 3 Factors")
```

```
## New names:
## New names:
## * `` -> `...1`
```

```r
# first data set is the monthly data
mkt_rf_ff_monthly <- factors_ff_monthly_raw$subsets$data[[1]] %>%
  mutate(
    # transform the data to be first day of the month and of the form "yyyy-mm-dd"
    date = floor_date(ymd(paste0(date, "01")), "month"),
    # data in percentage format, therefore convert them to decimal form
    across(c(RF, `Mkt-RF`, SMB, HML), ~as.numeric(.) / 100),
    # drop all other columns
    .keep = "none"
  ) %>%
  # rename all columns for better readability
  rename_with(str_to_lower) %>%
  rename(mkt_excess = `mkt-rf`) %>%
  # filter for the given time frame
  filter(date >= start_date & date <= end_date) %>%
  # Rearranging columns
  select(date, mkt_excess, rf)

head(mkt_rf_ff_monthly)
```

```
## # A tibble: 6 x 3
##   date       mkt_excess     rf
##   <date>          <dbl>  <dbl>
## 1 1927-01-01    -0.0006 0.0025
## 2 1927-02-01     0.0418 0.0026
## 3 1927-03-01     0.0013 0.003
## 4 1927-04-01     0.0046 0.0025
## 5 1927-05-01     0.0544 0.003
```

```
## 6 1927-06-01    -0.0234 0.0026
```

Calculate Market Portfolio Returns

The mkt-excess data set in the Fama-French data is the market excess return. Therefore, we need to add the riskfree rate "rf" to receive the market portfolio returns.

```
mkt_ret_monthly <- mkt_rf_ff_monthly %>%
  mutate(
    mkt_ret = mkt_excess + rf
    ) %>%
  select(-mkt_excess,-rf)

# Function to calculate rolling cumulative returns
# width = the number of data we want to cumulate over --> roll monthly
# Fun = calculates the cumulative return
# align = "right" --> assure we take last month of the window
rolling_cum_return <- function(return_series, window) {
  rollapply(return_series, width = window, FUN = function(x) prod(1 + x) - 1, align = "right", fill = NA
}

# Compute rolling returns
mkt_ret_rolling <- mkt_ret_monthly %>%
  arrange(date) %>%
  mutate(
    ret_1yr  = rolling_cum_return(mkt_ret, 12), # 1 year
    ret_10yr = rolling_cum_return(mkt_ret, 120), # 10 years
    ret_20yr = rolling_cum_return(mkt_ret, 240) # 20 years
  ) %>%
  select(-mkt_ret)

# 1. Average if rolling every month

# Compute average return for each investment duration
avg_ret_1yr_m  <- round(mean(mkt_ret_rolling$ret_1yr, na.rm = TRUE),4) *100
avg_ret_10yr_m <- round(mean(mkt_ret_rolling$ret_10yr, na.rm = TRUE),4) *100
avg_ret_20yr_m <- round(mean(mkt_ret_rolling$ret_20yr, na.rm = TRUE),4) *100

# 2. Average if rolling every year

mkt_ret_rolling_y <- mkt_ret_rolling %>%
  filter(month(date) == 1)

avg_ret_1yr_y  <- round(mean(mkt_ret_rolling_y$ret_1yr, na.rm = TRUE),4) *100
avg_ret_10yr_y <- round(mean(mkt_ret_rolling_y$ret_10yr, na.rm = TRUE),4) *100
avg_ret_20yr_y <- round(mean(mkt_ret_rolling_y$ret_20yr, na.rm = TRUE),4) *100


results <- data.frame(
  Rolling_Type = c("Monthly", "Yearly"),
  Average_1_year_Return = c(avg_ret_1yr_m, avg_ret_1yr_y),
  Average_10_year_Return = c(avg_ret_10yr_m, avg_ret_10yr_y),
  Average_20_year_Return = c(avg_ret_20yr_m, avg_ret_20yr_y)
)
```

Average Returns of the Market Portfolio [1927-2024] (in %)

Rolling Type	Avg. 1 Year Ret.	Avg. 10 Year Ret.	Avg. 20 Year Ret.
Monthly	12.14	198.14	797.02
Yearly	11.97	197.51	793.07

```r
colnames(results) <- c("Rolling Type","Avg. 1 Year Ret.", "Avg. 10 Year Ret.", "Avg. 20 Year Ret.")

results %>%
  gt() %>%
  tab_style(
    style = cell_borders(sides = "right",
                         color = "grey80",
                         weight = px(2)),
    locations = cells_body(columns = everything())
  ) %>%
  tab_style(
    style = cell_borders(sides = "right",
                         color = "grey80",
                         weight= px(2)),
    locations = cells_column_labels(columns = everything())  # Apply to column headers
  ) %>%
  tab_header(title = "Average Returns of the Market Portfolio [1927-2024] (in %)")
```

Annualize Market Portfolio Returns

```r
#function for annualizing returns with given input of the months = T
annualize_returns <- function(return,T){
  (1+return)^(12/T) - 1
}

# Annualize returns in the df mkt_ret_rolling using the yearly rolling timeframe

mkt_ret_rolling_y_ann <- mkt_ret_rolling_y %>%
  mutate(
    ret_10yr_ann = annualize_returns(ret_10yr,120),
    ret_20yr_ann = annualize_returns(ret_10yr,240)
  ) %>%
  select(-ret_10yr,-ret_20yr)

avg_ret_10yr_y_ann <- round(mean(mkt_ret_rolling_y_ann$ret_10yr_ann, na.rm = TRUE),4) *100
avg_ret_20yr_y_ann <- round(mean(mkt_ret_rolling_y_ann$ret_20yr_ann, na.rm = TRUE),4) *100

results <- data.frame(
  Rolling_Type = c("Yearly"),
  Average_1_year_Return = c(avg_ret_1yr_y),
  Average_10_year_Return = c(avg_ret_10yr_y_ann),
  Average_20_year_Return = c(avg_ret_20yr_y_ann)
)

colnames(results) <- c("Rolling Type","Avg. 1 Year Ret.", "Avg. 10 Year Ret.", "Avg. 20 Year Ret.")
```

Average Annualized Returns of the Market Portfolio [1927-2024] (in %)

Rolling Type	Avg. 1 Year Ret.	Avg. 10 Year Ret.	Avg. 20 Year Ret.
Yearly	11.97	10.5	5.09

```r
results %>%
  gt() %>%
  tab_style(
    style = cell_borders(sides = "right",
                         color = "grey80",
                         weight = px(2)),
    locations = cells_body(columns = everything())
  ) %>%
  tab_style(
    style = cell_borders(sides = "right",
                         color = "grey80",
                         weight= px(2)),
    locations = cells_column_labels(columns = everything())  # Apply to column headers
  ) %>%
  tab_header(title = "Average Annualized Returns of the Market Portfolio [1927-2024] (in %)")
```

Resample from the actual data and calculate the bootstrap standard error

```r
# We want to sample from the actual return data series
# Repeat the sample 10,000 times and calculate the mean for every sample
# standard error is the sd of the sample means
# We use the annualized return series
bootstrap_se <- function(returns, num_samples = 10000) {
  boot_means <- replicate(num_samples, mean(sample(returns, size = length(returns), replace = TRUE)))
  return(sd(boot_means))
}

se_1yr <- bootstrap_se(na.omit(mkt_ret_rolling_y_ann$ret_1yr))
se_10yr <- bootstrap_se(na.omit(mkt_ret_rolling_y_ann$ret_10yr_ann))
se_20yr <- bootstrap_se(na.omit(mkt_ret_rolling_y_ann$ret_20yr_ann))

result_se <- data.frame(
  Metric = c("Bootstrap SE"),
  Average_1_year_Return = round(se_1yr,4) *100 ,
  Average_10_year_Return = round(se_10yr,4)*100,
  Average_20_year_Return = round(se_20yr,4)*100
)

colnames(result_se) <- c("Rolling Type","Avg. 1 Year Ret.", "Avg. 10 Year Ret.", "Avg. 20 Year Ret.")

result_se %>%
  gt() %>%
  tab_style(
    style = cell_borders(sides = "right",
                         color = "grey80",
                         weight = px(2)),
```

Bootstrap Standard Error of Average Annualized Returns (Ret. in %)

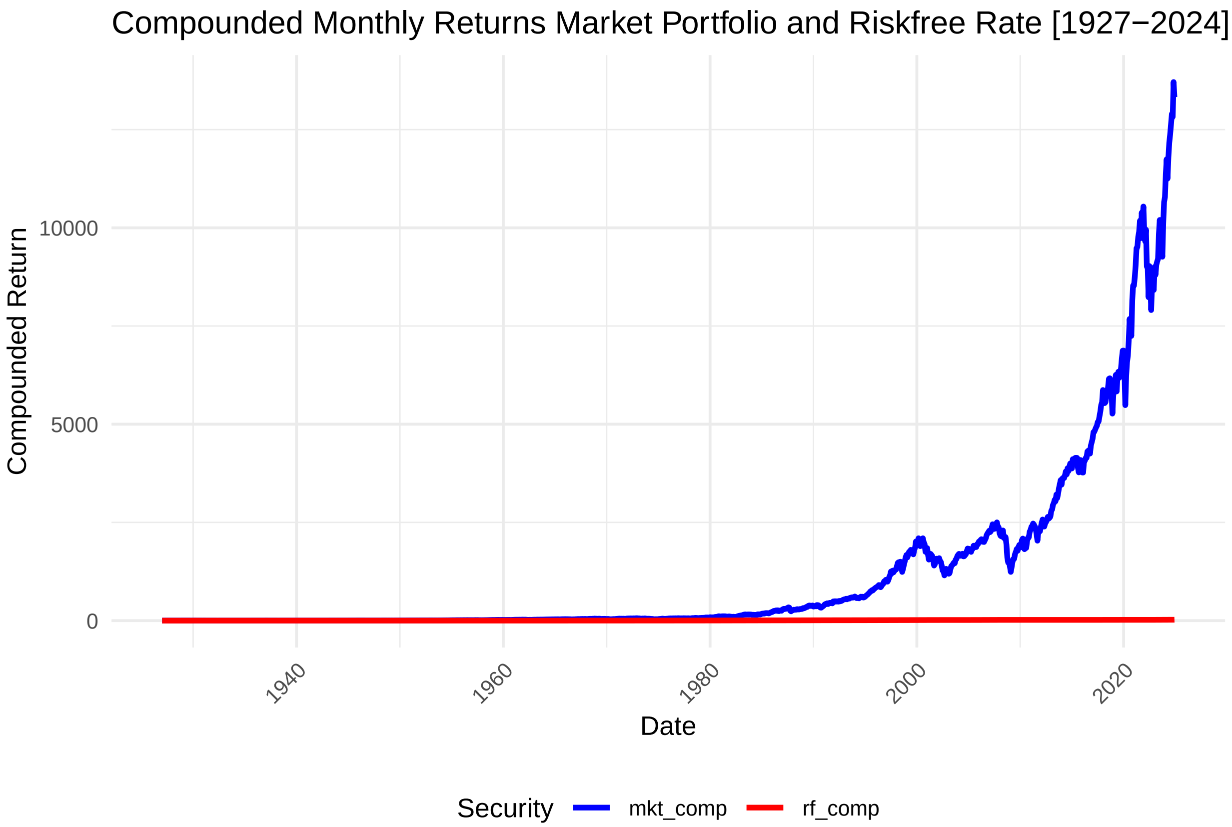
Rolling Type	Avg. 1 Year Ret.	Avg. 10 Year Ret.	Avg. 20 Year Ret.
Bootstrap SE	2.01	0.56	0.27

```
    locations = cells_body(columns = everything())
  ) %>%
  tab_style(
    style = cell_borders(sides = "right",
                         color = "grey80",
                         weight= px(2)),
    locations = cells_column_labels(columns = everything())  # Apply to column headers
  ) %>%
  tab_header(title = "Bootstrap Standard Error of Average Annualized Returns (Ret. in %)")
```

Comparison of the returns of investment of the MP with riskfree rate

```
mkt_ret_monthly <- mkt_rf_ff_monthly %>%
  mutate(
    mkt_ret = mkt_excess + rf
    ) %>%
  select(-mkt_excess)

mkt_ret_monthly <- mkt_ret_monthly %>%
  mutate(
    mkt_comp = cumprod(1 + mkt_ret) - 1,
    rf_comp = cumprod(1 + rf) - 1
  )

# Reshape data for plotting
mkt_ret_monthly_long <- na.omit(mkt_ret_monthly) %>%
  select(date, mkt_comp , rf_comp) %>%
  pivot_longer(cols = c(mkt_comp,rf_comp),
               names_to = "Security",
               values_to = "Compound_Return")


# Plot the compounded returns for Market Portfolio and riskfree rate
ggplot(mkt_ret_monthly_long, aes(x = date, y = Compound_Return, color = Security)) +
  geom_line(linewidth = 1) +  # Line plot for compounded returns
  labs(
    title = "Compounded Monthly Returns Market Portfolio and Riskfree Rate [1927-2024]",
    x = "Date",
    y = "Compounded Return",
    color = "Security"
  ) +
  theme_minimal() +  # Clean theme
  theme(
    legend.position = "bottom",
    text = element_text(size = 10),
    axis.text.x = element_text(angle = 45, hjust = 1)
```

```r
  ) +
  scale_color_manual(values = c("blue", "red"))
```



Housekeeping for Git

```r
save.image(file = "Pj1_Task4.RData")
```

Exercise 5: Find a proxy for the Market Portfolio

As Alnylam Pharmaceuticals, Inc. is a publicly traded large cap corporation in the US and it is part of the Russel 1000 and Russel 3000, we think that it is suitable to use the wider Russel 3000 index as the market portfolio.

First download the data from Excel

The price data for the Russel 3000 and for Alnylam Pharmaceuticals is in an Excel file, which we downloaded from Bloomberg.

```r
ALNY_PX <- read.xlsx("Daily_Data_Case_1.xlsx", sheet = 2)
ALNY_PX$Date <- as.Date(ALNY_PX$Date,origin = "1899-12-30")

RUS_3_PX <- read.xlsx("Daily_Data_Case_1.xlsx", sheet = 1)
RUS_3_PX$Date <- as.Date(RUS_3_PX$Date,origin = "1899-12-30")

# Observe NA values

ALNY_NA <- sum(is.na(ALNY_PX$Last.Price))
```

```
RUS_3_NA <- sum(is.na(RUS_3_PX$Last.Price))

# One NA in ALNY_PX --> Lin. Interpolate

na <- which(is.na(ALNY_PX$Last.Price))

ALNY_PX$Last.Price[na] <- (ALNY_PX$Last.Price[na-1] + ALNY_PX$Last.Price[na+1]) * 0.5
```

Calculate Arithmic Average Annual Return and Compound Annual Return

```
ALNY_PX_xts <- xts(ALNY_PX$Last.Price, order.by = as.Date(ALNY_PX$Date))
RUS_3_PX_xts <- xts(RUS_3_PX$Last.Price, order.by = as.Date(RUS_3_PX$Date))

PX_data <- list(ALNY_PX_xts,RUS_3_PX_xts)

time <- c("daily", "weekly", "monthly")
time_2 <- c(252, 52, 12)
type <- c("Alnylam", "Russel 3000 Index")

ret <- list()  # List to store returns
avg_an_ret <- vector("numeric", 3)
comp_an_ret <- vector("numeric", 3)
summary_ret <- list()

for (j in 1:2){
  for (i in 1:3) {
    # Calculate returns
    ret[[i]] <- coredata(periodReturn(PX_data[[j]]["/2019-01-01"], period = time[i], type = "arithmetic

    # Average Arithmetic Annual Return
    avg_an_ret[i] <- (1 + mean(ret[[i]])) ^ time_2[i] - 1

    # Compound Annual Return
    comp_returns <- cumprod(1 + ret[[i]])
    # Annualize the return by adjusting for the number of periods
    comp_an_ret[i] <- (comp_returns[length(comp_returns)]) ^ (time_2[i] / length(comp_returns)) - 1
  }

summary_ret[[j]] <- data.frame(
  "Period" = time,
  "Art_Avg_An_Ret (in %)" = round(avg_an_ret,4)*100,
  "CAGR (in %)" = round(comp_an_ret,4)*100
    )
colnames(summary_ret[[j]]) <- c("Period", "Arithmetic Average Annual Return (in %)", "Compound Annual Re
}


summary_ret[[1]] %>%
    gt() %>%
    tab_style(
      style = cell_borders(sides = "right",
                           color = "grey80",
                           weight = px(2)),
```

Arithmetic Average Annualized Returns vs Compound Annual Returns of Alnylam

Period	Arithmetic Average Annual Return (in %)	Compound Annual Return (in %)
daily	33.00	11.25
weekly	32.66	11.16
monthly	30.51	11.23

Arithmetic Average Annualized Returns vs Compound Annual Returns of Russel 3000 Index

Period	Arithmetic Average Annual Return (in %)	Compound Annual Return (in %)
daily	12.26	10.63
weekly	12.00	10.56
monthly	11.74	10.63

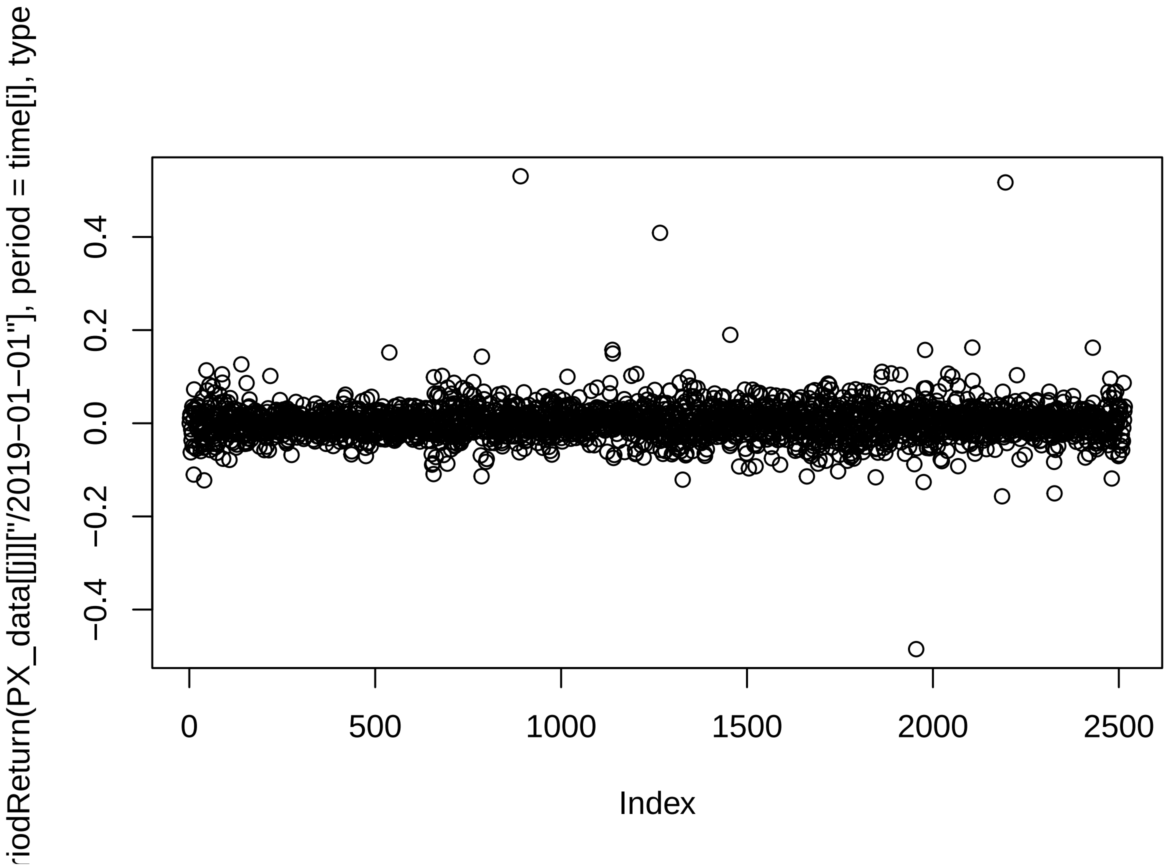
```
      locations = cells_body(columns = everything())
    ) %>%
    tab_style(
      style = cell_borders(sides = "right",
                           color = "grey80",
                           weight= px(2)),
      locations = cells_column_labels(columns = everything())
    ) %>%
    tab_header(title = paste0("Arithmetic Average Annualized Returns vs Compound Annual Returns of ", ty
```

```
summary_ret[[2]] %>%
    gt() %>%
    tab_style(
      style = cell_borders(sides = "right",
                           color = "grey80",
                           weight = px(2)),
      locations = cells_body(columns = everything())
    ) %>%
    tab_style(
      style = cell_borders(sides = "right",
                           color = "grey80",
                           weight= px(2)),
      locations = cells_column_labels(columns = everything())
    ) %>%
    tab_header(title = paste0("Arithmetic Average Annualized Returns vs Compound Annual Returns of ", ty
```

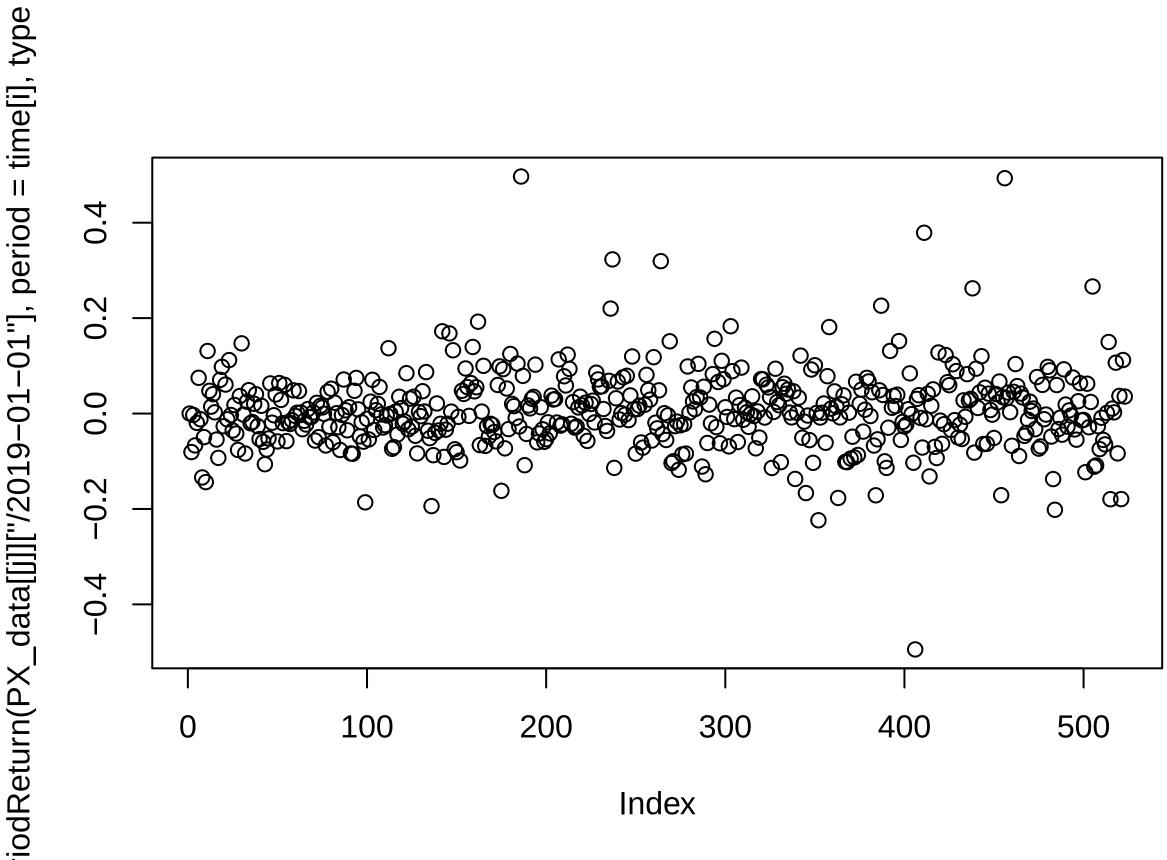
Estimate Return Volatility

```
st_dev <- vector("numeric",3)

summary_dev <- data.frame(
  "Period" = time
)

for(j in 1:2){
```

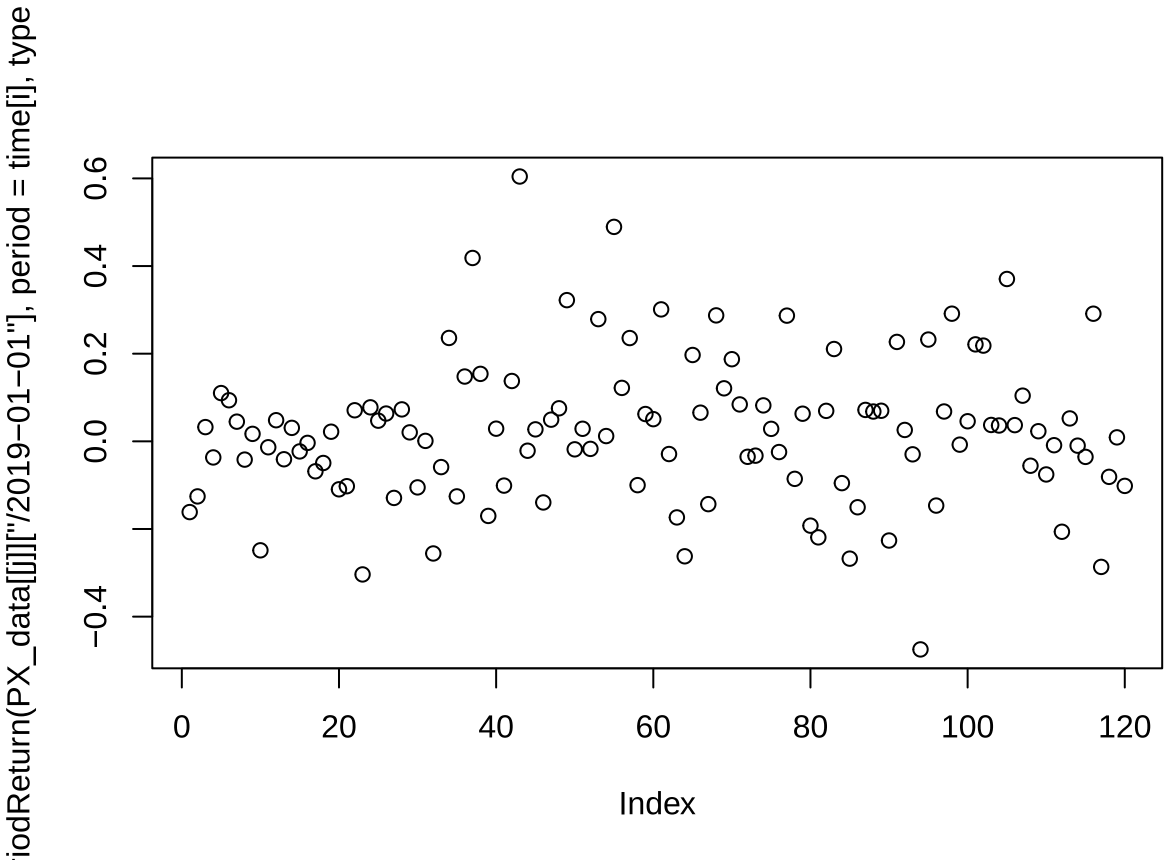
```
  for (i in 1:3){
    st_dev[i] <- round(sd(coredata(periodReturn(PX_data[[j]]["/2019-01-01"], period = time[i], type = "a
    print(plot(coredata(periodReturn(PX_data[[j]]["/2019-01-01"], period = time[i], type = "arithmetic")
  }
  summary_dev <- cbind(summary_dev, st_dev)
}
```




```
## NULL
```

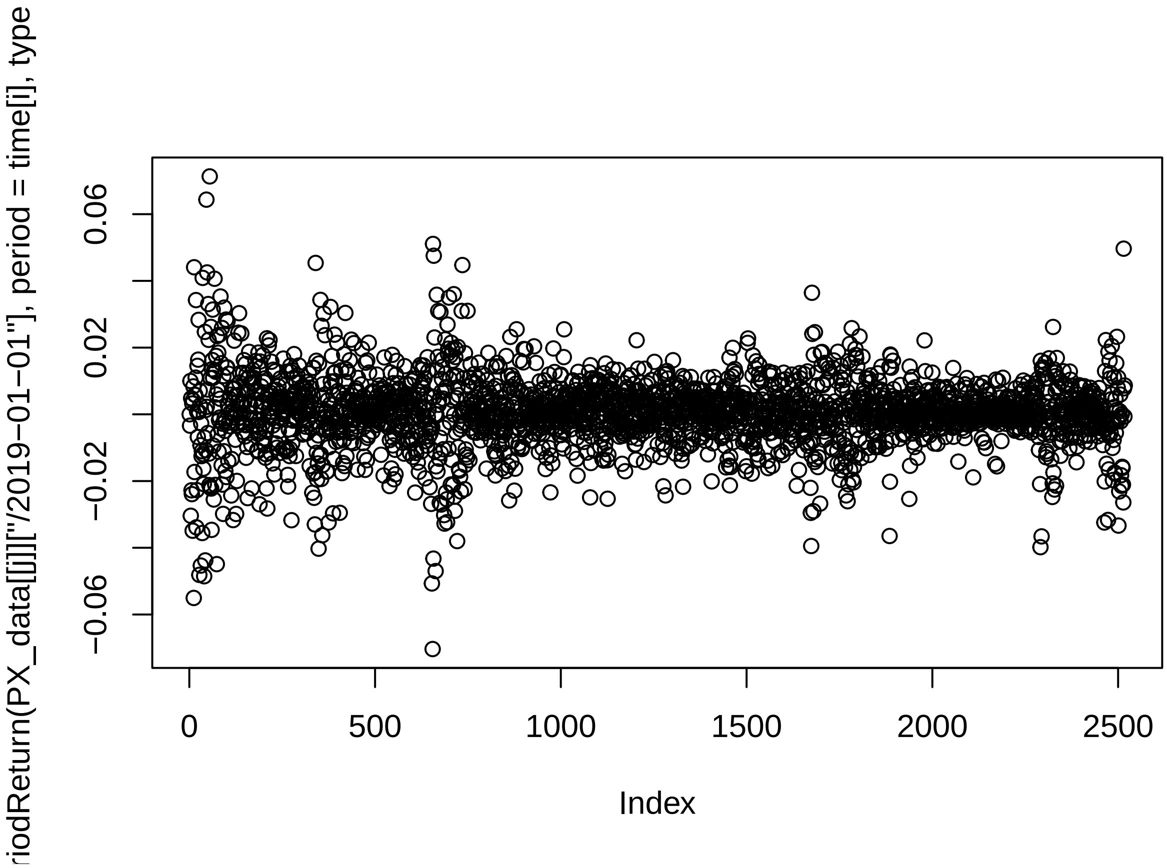

riodReturn(PX_data[[j]]["/2019−01−01"], period = time[i], type
−0.4
−0.2
0.0
0.2
0.4
0
100
200
300
400
500
Index

```
## NULL
```

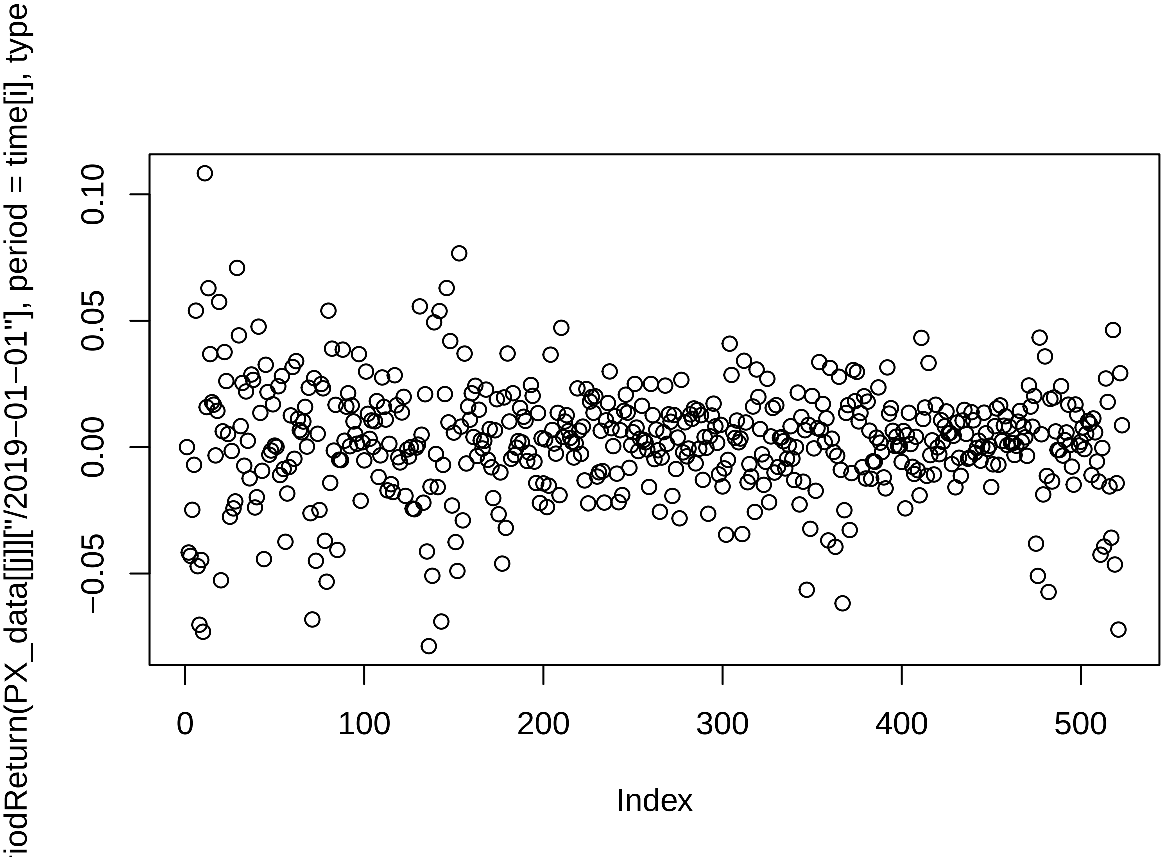


riodReturn(PX_data[[j]]["/2019−01−01"], period = time[i], type
−0.4
0.0
0.2
0.4
0.6
0
20
40
60
80
100
120
Index

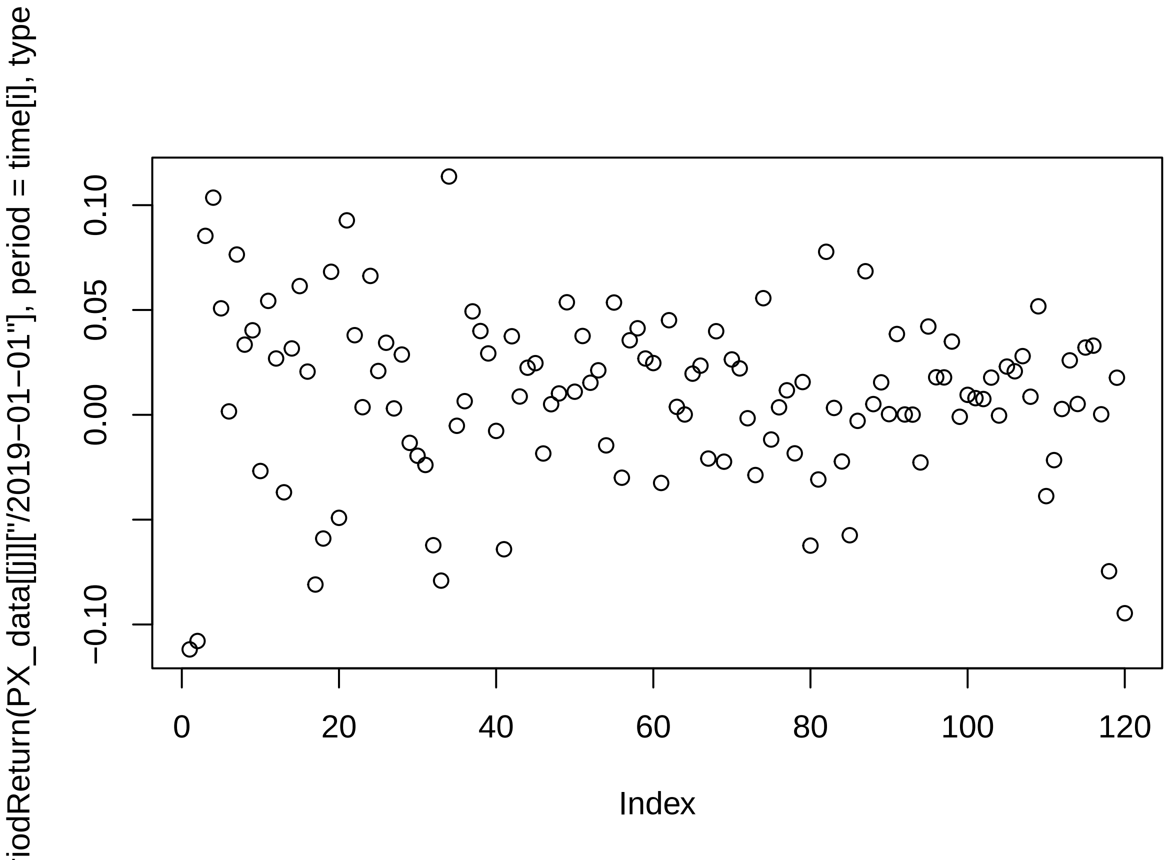

```
## NULL
```


riodReturn(PX_data[[j]][["/2019−01−01"], period = time[i], type
0.06
0.02
−0.02
−0.06
0
500
1000
1500
2000
2500
Index

```
## NULL
```


riodReturn(PX_data[[j]][["/2019−01−01"], period = time[i], type
−0.05
0.00
0.05
0.10
0
100
200
300
400
500
Index

```
## NULL
```




```
## NULL
```

```
colnames(summary_dev) <- c("Period Type", "Alnylam","Russel 3000 Index")

summary_dev %>%
gt() %>%
tab_style(
  style = cell_borders(sides = "right",
                       color = "grey80",
                       weight = px(2)),
  locations = cells_body(columns = everything())
) %>%
tab_style(
  style = cell_borders(sides = "right",
                       color = "grey80",
                       weight= px(2)),
  locations = cells_column_labels(columns = everything())  # Apply to column headers
) %>%
tab_header(title = "Estimated Return Volatility for Alnylam and Russel 3000 Index")
```

Plot the rolling 2 years (104 weeks) Return Volatility for your firm and your

Market Portfolio with weekly returns from 1/1/2009 to 12/31/2024

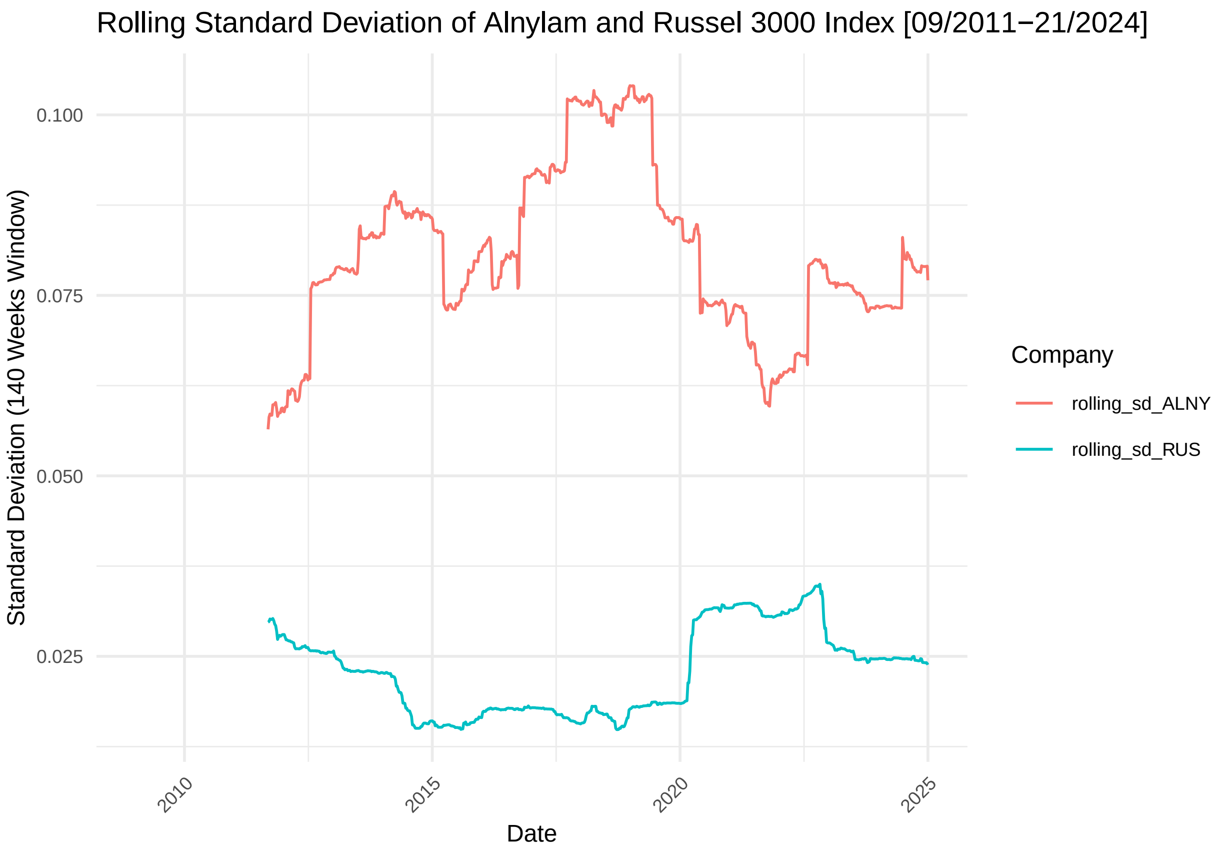
```
returns_weekly_ALNY <- periodReturn(PX_data[[1]], period = "weekly", type = "arithmetic")
returns_weekly_RUS <- periodReturn(PX_data[[2]], period = "weekly", type = "arithmetic")
```

Estimated Return Volatility for Alnylam and Russel 3000 Index

Period Type	Alnylam	Russel 3000 Index
daily	0.0380	0.0108
weekly	0.0832	0.0223
monthly	0.1667	0.0410

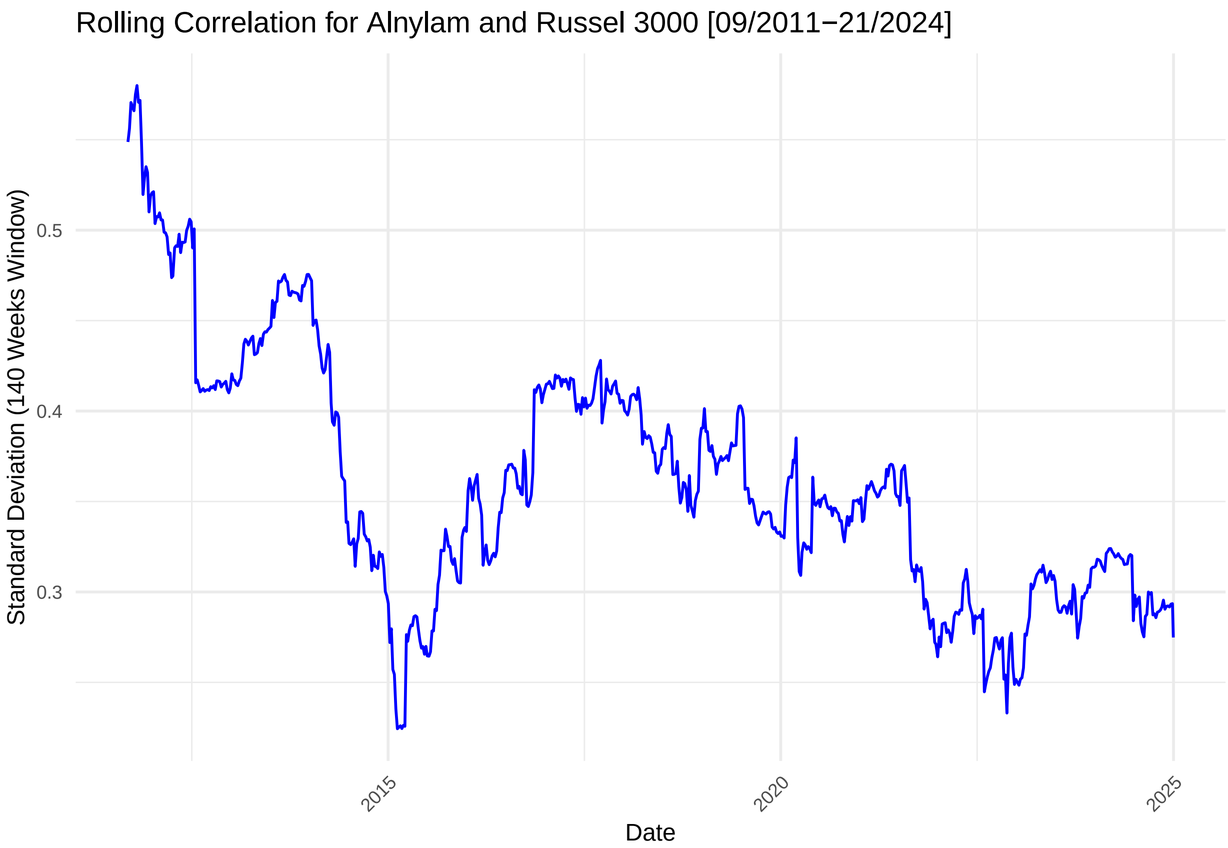
```
returns_weekly_ALNY_df <- data.frame(Date = index(returns_weekly_ALNY), coredata(returns_weekly_ALNY))
returns_weekly_RUS_df <- data.frame(Date = index(returns_weekly_RUS), coredata(returns_weekly_RUS))

returns_weekly <- inner_join(returns_weekly_ALNY_df, returns_weekly_RUS_df, by = "Date")
colnames(returns_weekly) <- c("Date","ALNY","Russel3000")
returns_weekly <- returns_weekly[-1,]


returns_weekly$rolling_sd_ALNY <- rollapply(returns_weekly$ALNY, width = 140, FUN = sd, align = "right"
returns_weekly$rolling_sd_RUS <- rollapply(returns_weekly$Russel3000, width = 140, FUN = sd, align = "r

returns_weekly$rolling_cor <- runCor(
  x = returns_weekly$ALNY,
  y = returns_weekly$Russel3000,
  n = 140
)

# For Plotting, put into long format
sd_long <- returns_weekly %>%
  pivot_longer(cols = c(rolling_sd_ALNY, rolling_sd_RUS),
               names_to = "Company",
               values_to = "Rolling_SD")

ggplot(sd_long, aes(x = Date, y = Rolling_SD, color = Company)) +
  geom_line() +
  labs(title = "Rolling Standard Deviation of Alnylam and Russel 3000 Index [09/2011-21/2024]",
       x = "Date",
       y = "Standard Deviation (140 Weeks Window)") +
  theme_minimal() +
  theme(
    axis.text.x = element_text(angle = 45, hjust = 1),
        text = element_text(size = 9))
```

```
## Warning: Removed 278 rows containing missing values or values outside the scale range
## (`geom_line()`).
```




```
ggplot(returns_weekly, aes(x = Date, y = rolling_cor)) +
  geom_line(color = "blue") +
  labs(title = "Rolling Correlation for Alnylam and Russel 3000 [09/2011-21/2024]",
       x = "Date",
       y = "Standard Deviation (140 Weeks Window)") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        text = element_text(size = 9))+
  xlim(as.Date("2011-09-09"), as.Date("2024-12-31"))
```

```
## Warning: Removed 139 rows containing missing values or values outside the scale range
## (`geom_line()`).
```



Rolling Correlation for Alnylam and Russel 3000 [09/2011−21/2024]
Standard Deviation (140 Weeks Window)
0.5
0.4
0.3
2015
2020
2025
Date