Solutions to Exercises

Portfolio Optimization: Theory and Application
Chapter 2 – Financial Data: Stylized Facts

Daniel P. Palomar (2025). *Portfolio Optimization: Theory and Application.*
Cambridge University Press.

portfoliooptimizationbook.com

Choose one or several assets (e.g., stocks or cryptocurrencies) for the following exercises.

Exercise 2.1: Price time series

Choose one asset and plot the price time series using both a linear and a logarithmic scale. Compare the plots and comment.

Solution

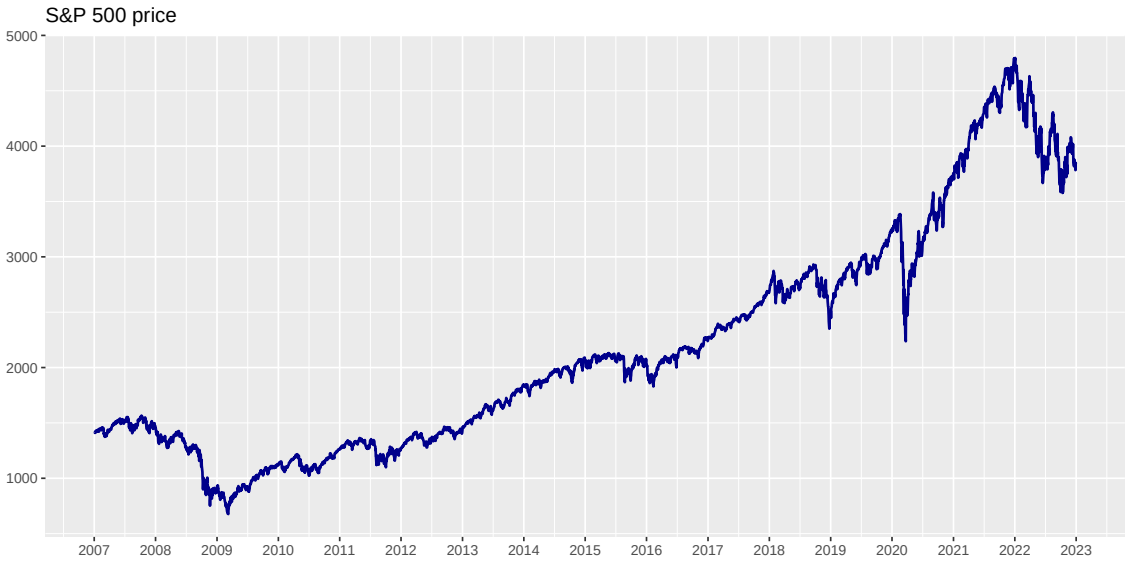
Load data for S&P 500:

```
library(quantmod)
library(ggplot2)

# Get data from Yahoo!Finance
sp500_prices  <- Ad(getSymbols("^GSPC",
                               from = "2007-01-01", to = "2022-12-31",
                               auto.assign = FALSE))
sp500_logreturns <- diff(log(sp500_prices))[-1]
sp500_linreturns <- exp(sp500_logreturns) - 1
```

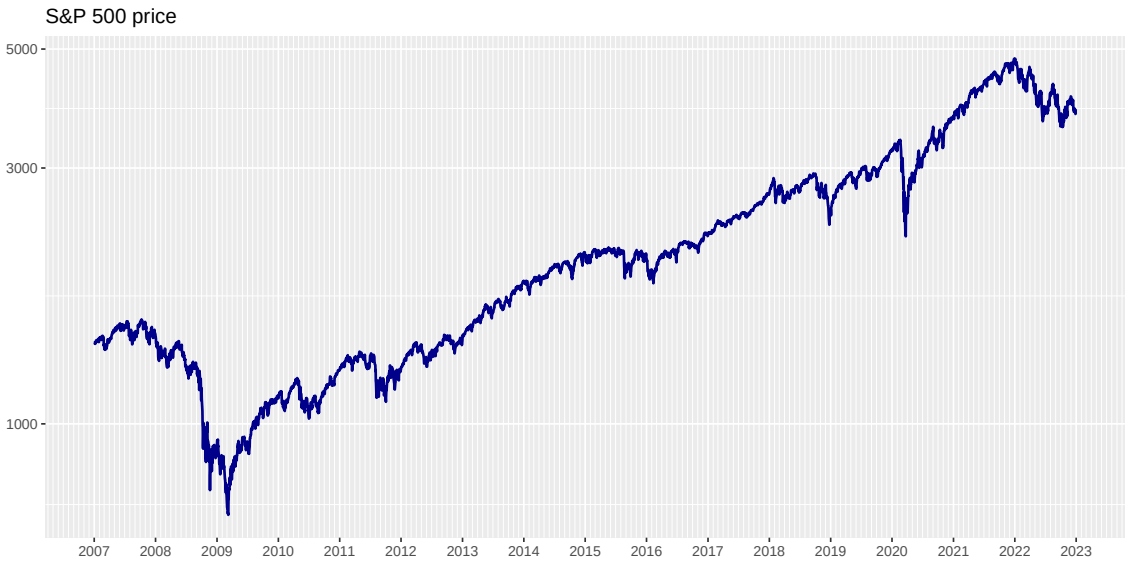
Plot in linear scale:

```
ggplot(fortify(sp500_prices, melt = TRUE), aes(x = Index, y = Value)) +
  geom_line(linewidth = 0.8, color = "darkblue") +
  scale_x_date(date_breaks = "1 year", date_labels = "%Y") +
  labs(title = "S&P 500 price", x = NULL, y = NULL)
```



Plot in logarithmic scale:

```
ggplot(fortify(sp500_prices, melt = TRUE), aes(x = Index, y = Value)) +
  geom_line(linewidth = 0.8, color = "darkblue") +
  scale_x_date(date_breaks = "1 year", date_labels = "%Y", date_minor_breaks = "1 month") +
  scale_y_log10() +
  labs(title = "S&P 500 price", x = NULL, y = NULL)
```



The logarithmic scale allows a better representation of the whole dynamic range.

Exercise 2.2: Return time series

Choose one asset and plot the linear returns and log-returns. Compare the plots and comment.

Solution

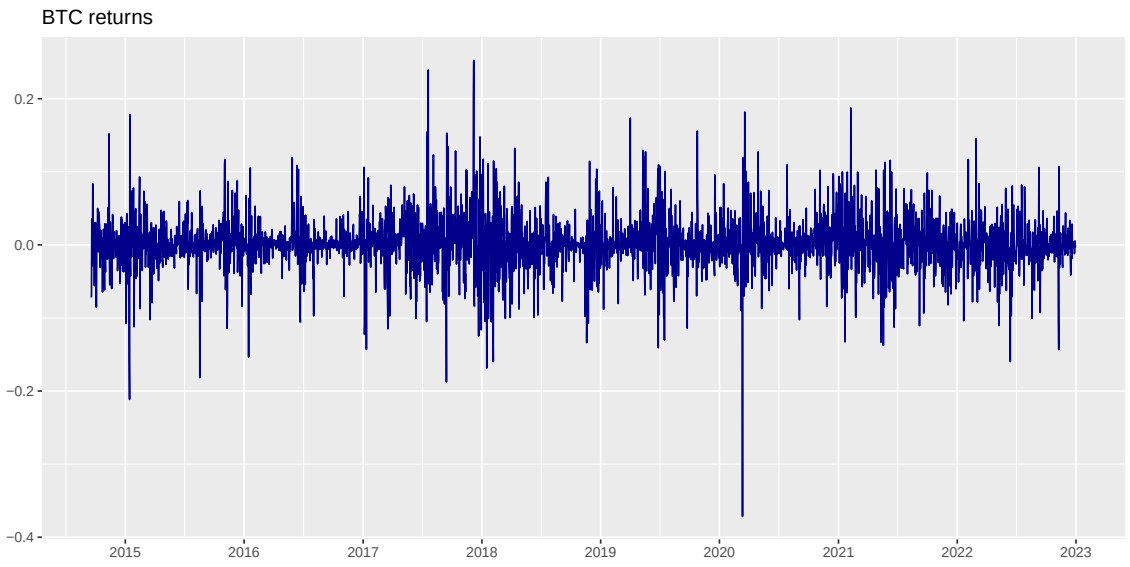
Load data for Bitcoin:

```
library(quantmod)
library(ggplot2)

# Get data from Yahoo!Finance
btc_prices  <- Ad(getSymbols("BTC-USD",
                             from = "2007-01-01", to = "2022-12-31",
                             auto.assign = FALSE))
btc_logreturns <- diff(log(btc_prices))[-1]
btc_linreturns <- exp(btc_logreturns) - 1
```

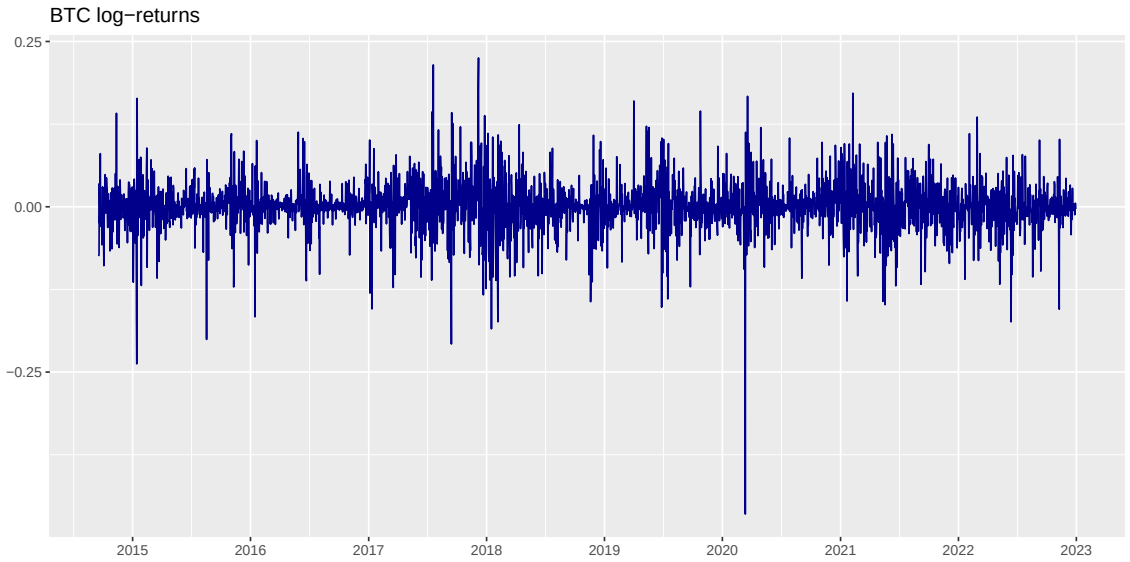
Plot linear returns:

```
ggplot(fortify(btc_linreturns, melt = TRUE), aes(x = Index, y = Value)) +
  geom_line(linewidth = 0.5, color = "darkblue") +
  scale_x_date(date_breaks = "1 year", date_labels = "%Y") +
  labs(title = "BTC returns", x = NULL, y = NULL)
```



Plot log-returns:

```
ggplot(fortify(btc_logreturns, melt = TRUE), aes(x = Index, y = Value)) +
  geom_line(linewidth = 0.5, color = "darkblue") +
  scale_x_date(date_breaks = "1 year", date_labels = "%Y") +
  labs(title = "BTC log-returns", x = NULL, y = NULL)
```



As expected, the linear returns and log-returns look virtually identical.

Exercise 2.3: Volatility envelope

Choose one asset and compute the volatility (square root of the average of the squared returns over k samples) on a rolling-window basis in two ways:

a. Left-aligned window: at each time t, use the samples $t-k+1, \ldots, t$. Try different values of k, observe the effect, and discuss.
b. Centered window: at each time t, use the samples $t - \lfloor k \rfloor/2, \ldots, t + \lceil k \rceil/2 - 1$. Try different values of k, observe the effect, and discuss.

Finally, compare the left-aligned and centered rolling-window approaches and discuss.

Solution

We will use a window of length $k = 20$ to compute the volatility envelope:

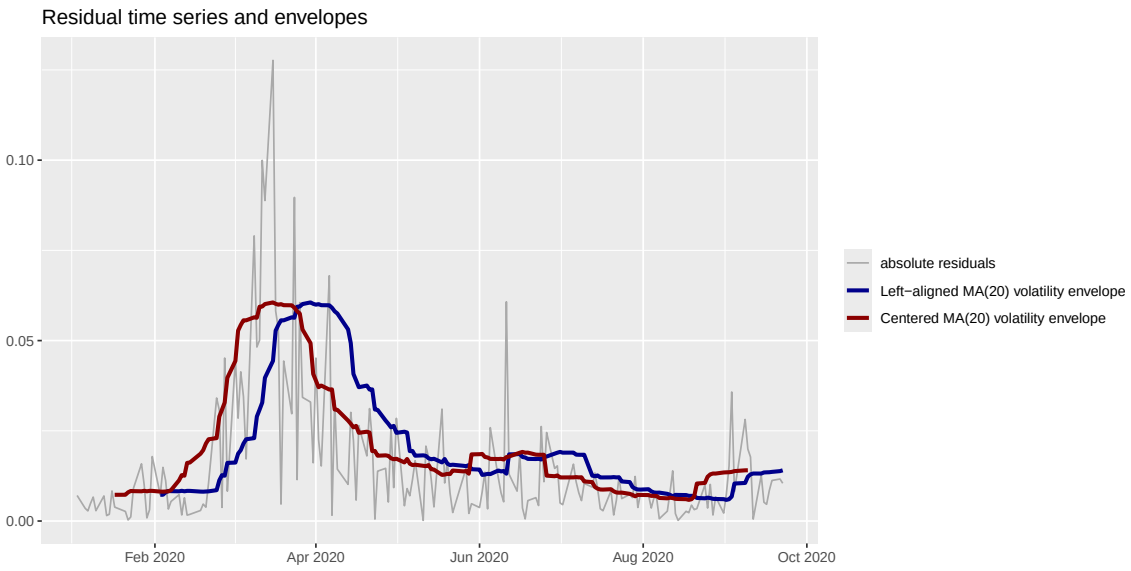
```r
library(pob)       # Market data used in the book
library(RcppRoll)  # Fast rolling means

# Load market data
y <- log(SP500_2015to2020$index["2020"])
x <- diff(y)[-1]
x_all <- abs(x)
colnames(x_all) <- "absolute residuals"

# Left-aligned MA(20) on squared returns x^2
vol_forecast <- xts(sqrt(roll_mean(x^2, n = 20, align = "right", fill = NA)),
                    index(x))
colnames(vol_forecast) <- "Left-aligned MA(20) volatility envelope"
x_all <- cbind(x_all, lag(vol_forecast)[-1], check.names = FALSE)

# Centered MA(20) on squared returns x^2
vol_forecast <- xts(sqrt(roll_mean(x^2, n = 20, align = "center", fill = NA)),
                    index(x))
colnames(vol_forecast) <- "Centered MA(20) volatility envelope"
x_all <- cbind(x_all, lag(vol_forecast)[-1], check.names = FALSE)
```

```r
# plot
ggplot(fortify(x_all, melt = TRUE),
       aes(x = Index, y = Value, color = Series, linewidth = Series)) +
  geom_line() +
  scale_color_manual(values = c("darkgray", "darkblue", "darkred")) +
  scale_linewidth_manual(values = c(0.5, 1.2, 1.2)) +
  scale_x_date(date_breaks = "2 months", date_labels = "%b %Y") +
  theme(legend.title = element_blank()) +
  labs(title = "Residual time series and envelopes", x = NULL, y = NULL)
```



The left-aligned version is causal and shows a delay, whereas the centered version requires future data but does not exhibit a delay.

Exercise 2.4: Return distribution

Choose one asset and perform the following tasks:

a. Plot histograms of the log-returns at different frequencies. Compare the plots and comment.
b. Draw Q–Q plots to focus on the tail distribution. Do the returns follow a Gaussian distribution?
c. Compute the skewness and kurtosis to see if they correspond to a Gaussian distribution.

Solution

Load data for S&P 500:

```
library(quantmod)

# Get data from Yahoo!Finance
sp500_prices  <- Ad(getSymbols("^GSPC",
                               from = "2007-01-01", to = "2022-12-31",
                               auto.assign = FALSE))
```

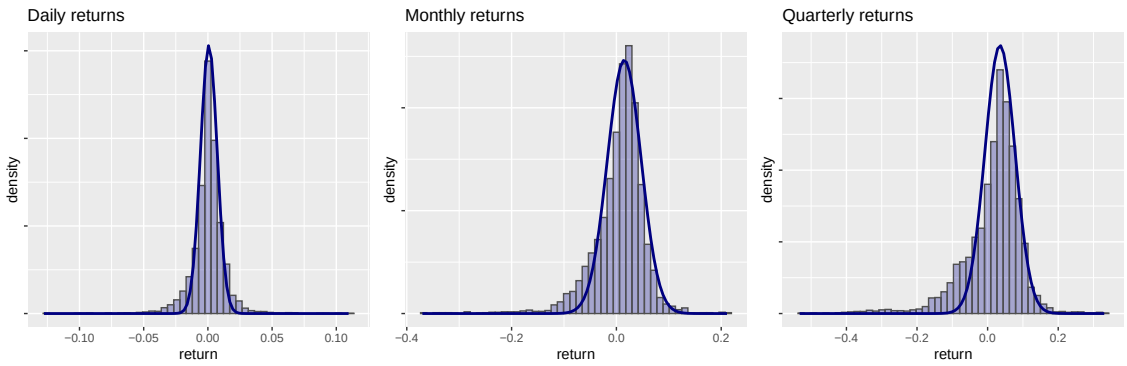
a. Histogram of S&P 500 log-returns at different frequencies (with Gaussian fit):

```
library(ggplot2)
library(patchwork)

plot_histogram <- function(returns,
                           title = "",
                           shift = 0.0, sd_factor = 1, nbins = 50) {
  ggplot(fortify(returns, melt = TRUE), aes(x = Value)) +
    geom_histogram(aes(y = after_stat(density)), bins = nbins, fill = "darkblue",
                   alpha = 0.3, col = "gray31") +
    stat_function(fun = dnorm,
                  args = list(mean = mean(returns, na.rm = TRUE) + shift,
                              sd = sd_factor*sd(returns, na.rm = TRUE)),
                  color = "navy", linewidth = 1) +
    theme(axis.text.y = element_blank()) +
    ggtitle(title) + xlab("return") + ylab("density")
}

sp500_dailyreturns     <- diff(log(sp500_prices))[-1]
sp500_monthlyreturns   <- diff(log(sp500_prices), 20)[-c(1:20)]
sp500_quarterlyreturns <- diff(log(sp500_prices), 60)[-c(1:60)]

p1 <- plot_histogram(sp500_dailyreturns, "Daily returns", shift = 5e-4, sd_factor = 0.50)
p2 <- plot_histogram(sp500_monthlyreturns, "Monthly returns", shift = 1e-2, sd_factor = 0.65)
p3 <- plot_histogram(sp500_quarterlyreturns, "Quarterly returns", shift = 2e-2, sd_factor = 0.52)
p1 | p2 | p3
```

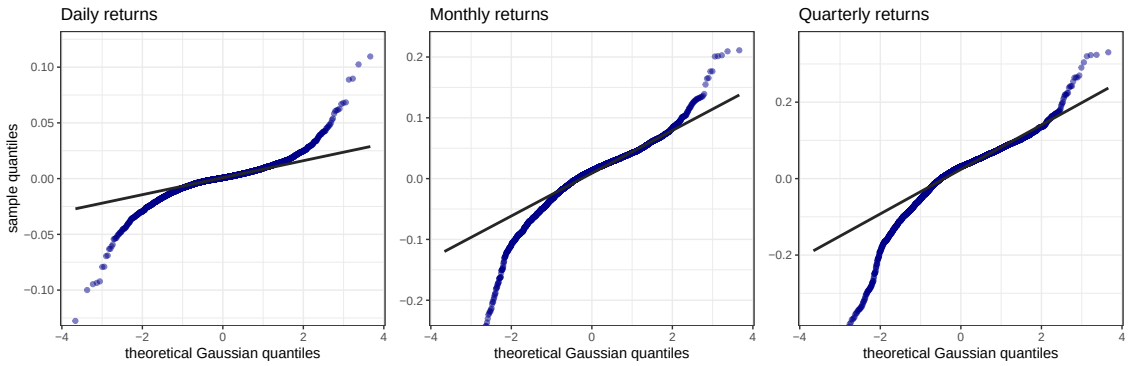


The distributions for different frequencies do not fit the Gaussian distribution due to heavy tails and asymmetry.

b. Q–Q plots of S&P 500 log-returns at different frequencies:

```
p1 <- ggplot(data.frame(y = as.vector(sp500_dailyreturns)), aes(sample = y)) +
  stat_qq(color = "darkblue", alpha = 0.5) +
  stat_qq_line(color = "gray15", linewidth = 1.0) +
  theme_bw() +  coord_cartesian(ylim = c(-0.12, 0.12)) +
  labs(title = "Daily returns", x = "theoretical Gaussian quantiles", y = "sample quantiles")
p2 <- ggplot(data.frame(y = as.vector(sp500_monthlyreturns)), aes(sample = y)) +
  stat_qq(color = "darkblue", alpha = 0.5) +
  stat_qq_line(color = "gray15", linewidth = 1.0) +
  theme_bw() + coord_cartesian(ylim = c(-0.22, 0.22)) +
  labs(title = "Monthly returns", x = "theoretical Gaussian quantiles", y = NULL)
p3 <- ggplot(data.frame(y = as.vector(sp500_quarterlyreturns)), aes(sample = y)) +
  stat_qq(color = "darkblue", alpha = 0.5) +
  stat_qq_line(color = "gray15", linewidth = 1.0) +
  theme_bw() +  coord_cartesian(ylim = c(-0.35, 0.35)) +
  labs(title = "Quarterly returns", x = "theoretical Gaussian quantiles", y = NULL)

p1 | p2 | p3
```



Real market data do not follow a Gaussian distribution as indicated by the deviation of the tails.

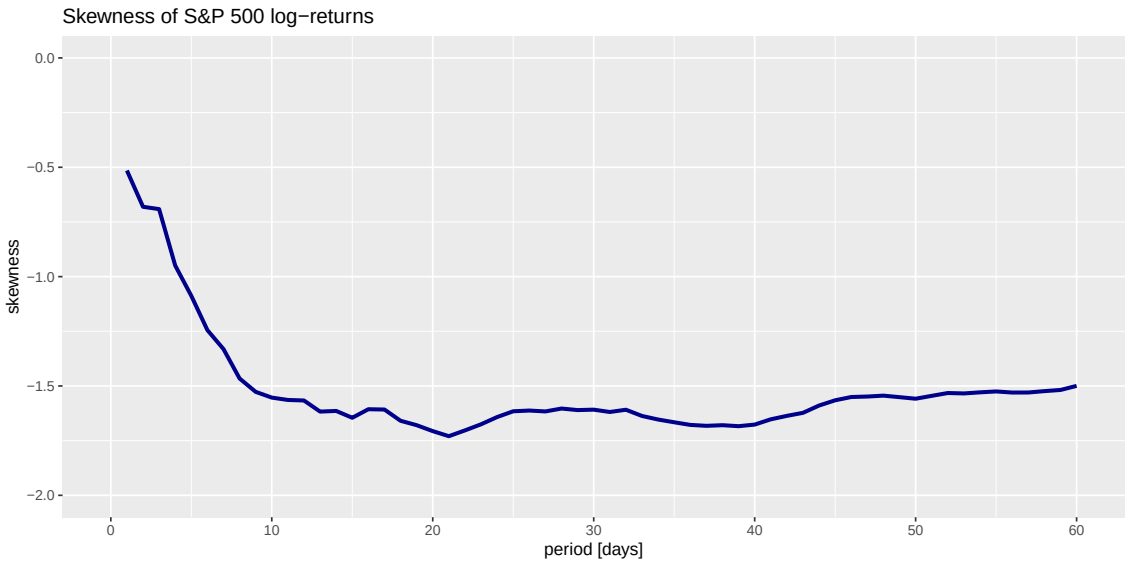
c. Skewness and kurtosis of S&P 500 and Bitcoin:

```
library(PerformanceAnalytics)

# compute kurtosis vs frequency (observe aggregational Gaussianity)
SP500_kurtosis <- NULL
SP500_skewness <- NULL
for(l in 1:60) {
  SP500_kurtosis[l] <- kurtosis(diff(log(sp500_prices), l), method="excess")
  SP500_skewness[l] <- skewness(diff(log(sp500_prices), l))
}
```

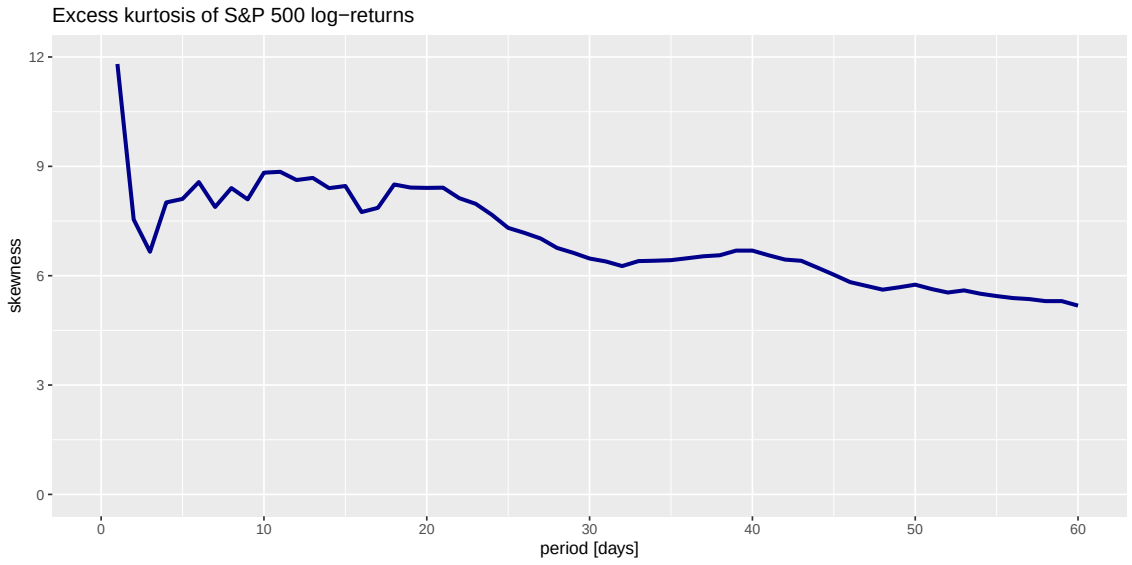
The skewness is clearly not zero but negative, indicating an asymmetry with heavier left tails:

```
ggplot(data.frame("x" = 1:60, "y" = SP500_skewness), aes(x, y)) +
  geom_line(color = "darkblue", linewidth = 1.2) +
  scale_x_continuous(limits = c(0, 60), breaks = seq(0, 60, by = 10)) +
  coord_cartesian(ylim = c(-2, 0)) +
  labs(title = "Skewness of S&P 500 log-returns",
       x = "period [days]", y = "skewness")
```



There is a clear excess kurtosis, indicating clear heavy tails:

```
ggplot(data.frame("x" = 1:60, "y" = SP500_kurtosis), aes(x, y)) +
  geom_line(color = "darkblue", linewidth = 1.2) +
  scale_x_continuous(limits = c(0, 60), breaks = seq(0, 60, by = 10)) +
  scale_y_continuous(limits = c(0, 12), breaks = seq(0, 12, by = 3)) +
  labs(title = "Excess kurtosis of S&P 500 log-returns",
       x = "period [days]", y = "skewness")
```



Exercise 2.5: Return autocorrelation

Choose one asset and perform the following tasks:

a. Plot the autocorrelation function of the log-returns at various frequencies. Compare the plots and comment.
b. Repeat the process using squared returns instead of log-returns. Compare these plots and comment.

Solution

Load data for S&P 500:

```
library(quantmod)

# Get data from Yahoo!Finance
sp500_prices  <- Ad(getSymbols("^GSPC",
                               from = "2007-01-01", to = "2022-12-31",
                               auto.assign = FALSE))
sp500_logreturns <- diff(log(sp500_prices))[-1]
```

a. Autocorrelation of S&P 500 daily log-returns (no linear structure can be appreciated):

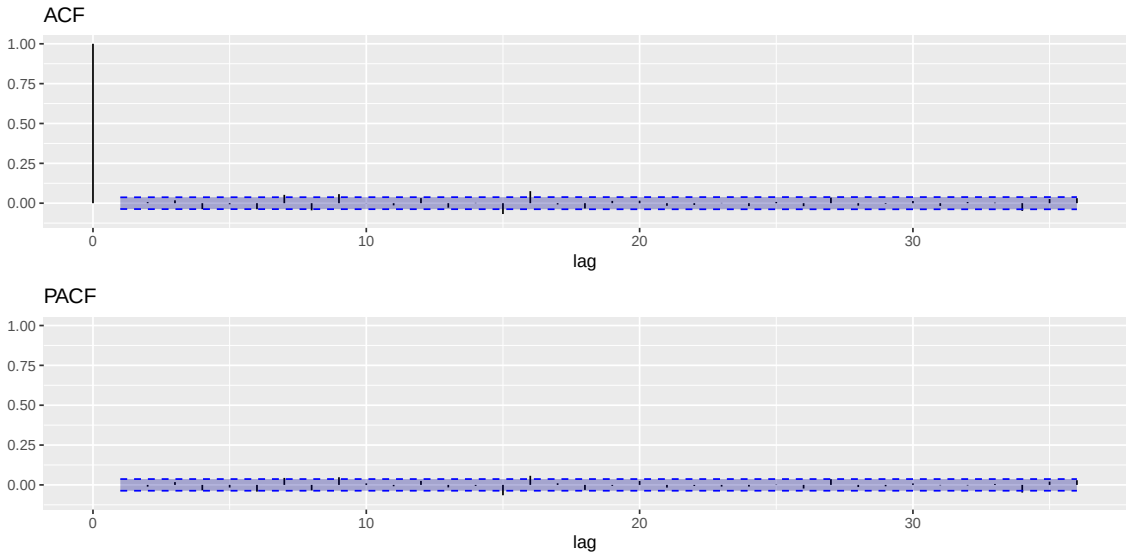
```r
library(ggfortify)

sp500_acf  <- acf(sp500_logreturns, plot = FALSE)
sp500_pacf <- pacf(sp500_logreturns, plot = FALSE)

p1 <- autoplot(sp500_acf,
               conf.int.fill = "darkblue", conf.int.value = 0.98, conf.int.type = "ma") +
  xlab("lag") + ylab(element_blank()) +
  scale_x_continuous(limits = c(0, NA)) + scale_y_continuous(limits = c(-0.10, 1)) +
  ggtitle("ACF")

p2 <- autoplot(sp500_pacf,
               conf.int.fill = "darkblue", conf.int.value = 0.98, conf.int.type = "ma") +
  xlab("lag") + ylab(element_blank()) +
  scale_x_continuous(limits = c(0, NA)) + scale_y_continuous(limits = c(-0.10, 1)) +
  ggtitle("PACF")

p1 / p2
```



b. Autocorrelation of absolute value of S&P 500 daily log-returns (nonlinear structure is obviously present):

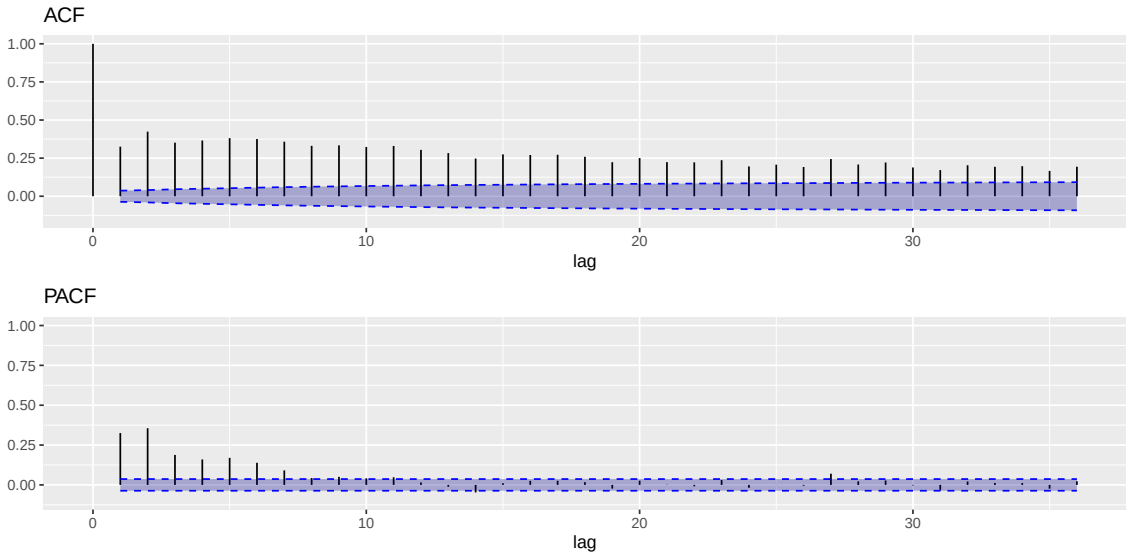
```r
library(ggfortify)

sp500_abs_acf  <- acf(abs(sp500_logreturns), plot = FALSE)
sp500_abs_pacf <- pacf(abs(sp500_logreturns), plot = FALSE)

p1 <- autoplot(sp500_abs_acf,
               conf.int.fill = "darkblue", conf.int.value = 0.98, conf.int.type = "ma") +
  xlab("lag") + ylab(element_blank()) +
  scale_x_continuous(limits = c(0, NA)) + scale_y_continuous(limits = c(-0.15, 1)) +
  ggtitle("ACF")

p2 <- autoplot(sp500_abs_pacf,
               conf.int.fill = "darkblue", conf.int.value = 0.98, conf.int.type = "ma") +
  xlab("lag") + ylab(element_blank()) +
  scale_x_continuous(limits = c(0, NA)) + scale_y_continuous(limits = c(-0.10, 1)) +
  ggtitle("PACF")

p1 / p2
```



Exercise 2.6: Asset correlation

Choose several stocks and perform the following tasks:

a. Compute the cross-correlations and plot a heatmap.
b. Compute the correlation between each of the stocks and the index. Discuss the results.
c. Compute the correlation between a stock and a cryptocurrency. Discuss the result and the implications.

Solution

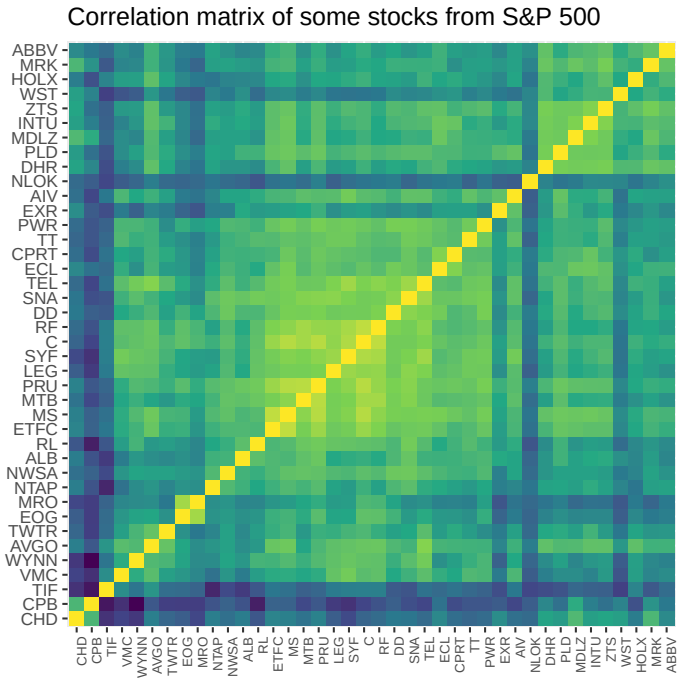
a. Heatmat of S&P 500 stocks:

```
library(pob)       # Market data used in the book
library(ggplot2)
library(reshape2)
library(viridisLite)

# use data from package pob
data(SP500_2015to2020)
set.seed(42)
prices <- tail(SP500_2015to2020$stocks[, sample(ncol(SP500_2015to2020$stocks), 40)], 200)
X <- diff(log(prices))[-1]
cormat_sp500 <- cor(X)

reorder_cormat <- function(cormat){
  dd <- as.dist((1 - cormat)/2)
  hc <- hclust(dd)
  cormat <-cormat[hc$order, hc$order]
}

cormat_sp500 |> reorder_cormat() |> melt() |>
  ggplot(aes(x=Var1, y=Var2, fill=value)) +
  geom_tile() +
  scale_fill_viridis_c(na.value = "white") +
  theme(axis.text.x = element_text(size = 7, angle = 90, hjust = 1, vjust = 1)) +
  theme(legend.position = "none") +
  labs(title = "Correlation matrix of some stocks from S&P 500", x = NULL, y= NULL)
```



b. Compute the correlation between each of the stocks and the index:

```
# Extract last 200 rows
prices_index <- tail(SP500_2015to2020$index, 200)
rets_index <- diff(log(prices_index))[-1]

# Compute correlation
cor(X, rets_index)
```

```
     SP500_INDEX
AVGO   0.8432922
NTAP   0.7099253
EOG    0.6283330
C      0.8334527
INTU   0.8816996
ECL    0.8130169
DD     0.7930437
ZTS    0.8462868
DHR    0.8338776
MTB    0.7692609
ALB    0.6961082
NWSA   0.7187634
PRU    0.8560165
CHD    0.5900068
```

```
EXR    0.6164350
CPRT   0.7900188
AIV    0.7169402
TIF    0.3677297
RL     0.6685407
RF     0.7761334
SNA    0.7936781
MRK    0.7726212
WST    0.6431932
MDLZ   0.8493114
TWTR   0.7245940
CPB    0.4113997
ABBV   0.7485481
HOLX   0.7157814
TT     0.7491973
PLD    0.8423284
PWR    0.7608415
LEG    0.7642701
NLOK   0.4814834
MRO    0.5369774
WYNN   0.6565635
ETFC   0.8504631
MS     0.8899857
SYF    0.7772663
TEL    0.8127803
VMC    0.6154780
```

As expected, most of the stocks are highly correlated with the market index.

c. Compute the correlation between a stock and a cryptocurrency:

```r
# We will use this data:
data(SP500_2015to2020)
data(cryptos_2017to2021)

set.seed(42)
stock_prices <- SP500_2015to2020$stocks[, sample(ncol(SP500_2015to2020$stocks), 40)]
btc_prices <- cryptos_2017to2021$daily[, "BTC"]


# Subset common dates
common_dates <- intersect(as.character(index(stock_prices)), as.character(index(btc_prices)))
btc_prices <- xts(coredata(btc_prices[common_dates]), as.Date(common_dates))
stock_prices <- xts(coredata(stock_prices[common_dates]), as.Date(common_dates))
btc_returns <- diff(log(btc_prices))[-1]
stock_returns <- diff(log(stock_prices))[-1]

# Compute correlation during period ("2017-11-01" to "2020-09-22")
first(index(stock_returns))
```

```
[1] "2017-11-01"
```

```r
last(index(stock_returns))
```

```
[1] "2020-09-22"
```

```r
cor(stock_returns, btc_returns)
```

```
            BTC
AVGO 0.16455707
NTAP 0.13637539
EOG  0.20622675
C    0.22324033
INTU 0.18934291
ECL  0.16604539
DD   0.14027893
ZTS  0.16684993
DHR  0.14875017
MTB  0.15225579
ALB  0.08165465
NWSA 0.15397732
PRU  0.23321770
CHD  0.04858052
EXR  0.09384341
CPRT 0.16190921
AIV  0.10486715
```

```
TIF  0.06064056
RL   0.14850637
RF   0.17686926
SNA  0.13471535
MRK  0.08853355
WST  0.09846893
MDLZ 0.13652339
TWTR 0.12030811
CPB  0.04593228
ABBV 0.10864012
HOLX 0.13863221
TT   0.14370147
PLD  0.12293025
PWR  0.13509171
LEG  0.16899363
NLOK 0.12529320
MRO  0.12302631
WYNN 0.11401163
ETFC 0.19270005
MS   0.22538054
SYF  0.21303821
TEL  0.17805503
VMC  0.15197974
```

Interestingly, the correlation between the stocks and Bitcoin does not seem to be large.