We just love to create portfolio optimization models based on Optimization theory and such models require a well-defined return scenario set which is nothing more than a matrix where we have a joint possible set of returns of all our assets under consideration. The easiest way is to use historical data for this purpose. While it is dangerous to use historical data in many price-based single asset strategies, it definitely makes sense for portfolio-based analysis because we capture the empirical dependency of the assets which works surprisingly well.

We focus on the S&P 100 here, but the code can be extended to any set of assets. Furthermore we download the data from Yahoo! Finance which optimally should also be replaced at least with e.g. [Alpha Vantage](https://www.alphavantage.co/).

We require two libraries to replicate all the code below, i.e.

library(quantmod)  
library(rvest)

First we need to get the list of all components of the S&P 100. Because we are lazy we use (and web-scrape) Wikipedia for this purpose. Please note that we hard-coded the fact that the list of assets and ticker symbols is in the third table of this Wikipedia page. If the structure of the page changes this number has to be adapted. Last check: August 5th, 2019.

table\_id <- 3

url <- "https://en.wikipedia.org/wiki/S%26P\_100"

sp100 <- url %>%

read\_html() %>%

html\_nodes("table") %>%

.[[table\_id]] %>%

html\_table()

Next we store the parsed info (i.e. ticker symbols and company names) into two vectors:

sp100\_ticker <- sp100$Symbol  
sp100\_company <- sp100$Name

Now we are ready to download all price data from Yahoo! Finance and store all separate data frames into one large list for easier post-processing:

quantmod::getSymbols(sp100\_ticker)

sp100\_data\_complete <- list()

for(current\_ticker in sp100\_ticker) {

sp100\_data\_complete[[length(sp100\_data\_complete) + 1]]

<- get(current\_ticker)

}

Next we check whether enough data for all stocks is available and if not we drop the stocks. In this case we require at least 1500 days of data availability.

rows <- sapply(sp100\_data\_complete, function(x) nrow(x))

nrow.cutoff <- 1500

pos <- which(nrows < nrow.cutoff)

sp100\_ticker[pos]

sp100\_data\_complete <- sp100\_data\_complete[-pos]  
sp100\_company <- sp100\_company[-pos]  
sp100\_ticker <- sp100\_ticker[-pos]

Now we are ready to compute returns. To do so we start by extracting the Adjusted Closing Prices only from the OHLC/VA data we have.

current\_pos <- 1

sp100\_adjusted <- Ad(sp100\_data\_complete[[current\_pos]])

for(current\_pos in 2:length(sp100\_ticker)) {

sp100\_adjusted <- merge(sp100\_adjusted,

Ad(sp100\_data\_complete[[current\_pos]]))

}

names(sp100\_adjusted) <- sp100\_ticker

Then we may select a certain time frame and compute daily returns from this set of Adjusted Closing Prices:

timeframe <- "2013/2019"

sp100\_returns <- dailyReturn(sp100\_adjusted[,1])[timeframe]

for(current\_pos in 2:length(sp100\_ticker)) {

sp100\_returns <- merge(sp100\_returns,

dailyReturn(sp100\_adjusted[,current\_pos])[timeframe])

}

names(sp100\_returns) <- sp100\_ticker

Now we are done! The full code contains one more (almost unnecessary) cleanup, but in any way we may now store our data for subsequent use.

scenario.set <- sp100\_returns

save(scenario.set, sp100\_ticker, sp100\_company,

file="oex-returns-1908.rda")

That’s it! Now you have a perfect scenario set to (back)test any portfolio optimization methods based on scenario input. Enjoy!