If you are familiar with the R Views posts, then you probably have a pretty good sense for the book’s style, prose, and code approach, but I’d like to add a few quick words of background.

The book’s practical motivations are: (1) to introduce R to finance professionals, or aspiring finance professionals, who wish to move beyond Excel for their quantitative work, and (2) to introduce various finance coding paradigms to R coders.

The softer motivation is to demonstrate and emphasize readable, reusable, and reproducible R code, data visualizations, and Shiny dashboards. It will be very helpful to have some background in the R programming language *or* in finance, but the most important thing is a desire to learn about the landscape of R code and finance packages.

An overarching goal of the book is to introduce the three major R paradigms for portfolio analysis: xts, the tidyverse, and tidyquant. As a result, we will frequently run the same analysis using three different code flows.

If that ‘three-universe’ structure seems a bit unclear, you’ll notice that we solve the same task and get the same result with different code paths.

For example, if we had portfolio returns saved in a tibble object called portfolio\_returns\_tq\_rebalanced\_monthly, and an xts object called portfolio\_returns\_xts\_rebalanced\_monthly, and our goal was to find the Sharpe Ratio of portfolio returns, we would start in the xts world with SharpeRatio() from the PerformanceAnalytics package.

# define risk free rate variable

rfr <- .0003

sharpe\_xts <-

SharpeRatio(portfolio\_returns\_xts\_rebalanced\_monthly,

Rf = rfr,

FUN = "StdDev") %>%

`colnames<-`("sharpe\_xts")

sharpe\_xts

## sharpe\_xts

## StdDev Sharpe (Rf=0%, p=95%): 0.2748752

We next would use the tidyverse and run our calculations in a piped flow:

sharpe\_tidyverse\_byhand <-

portfolio\_returns\_tq\_rebalanced\_monthly %>%

summarise(sharpe\_dplyr = mean(returns - rfr)/

sd(returns - rfr))

sharpe\_tidyverse\_byhand

## # A tibble: 1 x 1

## sharpe\_dplyr

##

## 1 0.275

And then head to the tidyquant paradigm where we can apply the SharpeRatio() function to a tidy tibble.

sharpe\_tq <-

portfolio\_returns\_tq\_rebalanced\_monthly %>%

tq\_performance(Ra = returns,

performance\_fun = SharpeRatio,

Rf = rfr,

FUN = "StdDev") %>%

`colnames<-`("sharpe\_tq")

We can compare our three Sharpe objects and confirm consistent results.

sharpe\_tq %>%

mutate(tidy\_sharpe = sharpe\_tidyverse\_byhand$sharpe\_dplyr,

xts\_sharpe = sharpe\_xts)

## # A tibble: 1 x 3

## sharpe\_tq tidy\_sharpe xts\_sharpe

##

## 1 0.275 0.275 0.275

We might be curious how the Sharpe-Ratio-to-standard-deviation ratio of our portfolio compares to those of the component ETFs and a ggplot scatter is a nice way to visualize that.

asset\_returns\_long %>%

na.omit() %>%

summarise(stand\_dev = sd(returns),

sharpe = mean(returns - rfr)/

sd(returns - rfr)) %>%

add\_row(asset = "Portfolio",

stand\_dev =

portfolio\_sd\_xts\_builtin[1],

sharpe =

sharpe\_tq$sharpe\_tq) %>%

ggplot(aes(x = stand\_dev,

y = sharpe,

color = asset)) +

geom\_point(size = 2) +

geom\_text(

aes(x =

sd(portfolio\_returns\_tq\_rebalanced\_monthly$returns),

y =

sharpe\_tq$sharpe\_tq + .02,

label = "Portfolio")) +

ylab("Sharpe Ratio") +

xlab("standard deviation") +

ggtitle("Sharpe Ratio versus Standard Deviation") +

# The next line centers the title

theme\_update(plot.title = element\_text(hjust = 0.5))

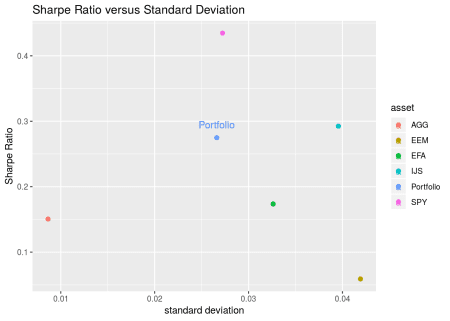


Figure 1: Sharpe versus Standard Deviation

Finally, we are ready to calculate and visualize the Sharpe Ratio of a custom portfolio with Shiny and a flexdashboard.

As in the above example, most tasks in the book end with data visualization and then Shiny (a few early readers have commented with happy surprise that all the charts and code are in full color in the book – thanks to CRC press for making that happen!). Data visualization and Shiny are heavily emphasized – more so than in other finance books – and that might seem unusual. After all, every day we hear about how the financial world is becoming more quantitatively driven as firms race towards faster, more powerful algorithms. Why emphasize good ol’ data visualization? I believe, and have observed first-hand, that the ability to communicate and tell the story of data in a compelling way is only going to become more crucial as the financial world becomes more complex. Investors, limited partners, portfolio managers, clients, risk managers – they might not want to read our code or see our data, but we still need to communicate to them the value of our work. Data visualization and Shiny dashboards are a great way to do that. By book’s end, a reader will have built a collection of live, functioning flexdashboards that can be the foundation for more complex apps in the future.