install.packages("DBI")

install.packages("RMySQL")

library(DBI)

library(RMySQL)

mydb<- dbConnect(MySQL(),user='ktruc002', password='35442fed', dbname='cn\_stock\_quote', host='172.19.3.250')

SQL\_statement<- "SELECT `trade\_date`, `index\_code`, `index\_name`, `last`, `input\_time`

FROM `cn\_stock\_index`.`daily\_quote`

WHERE index\_code IN('000001','801150','801260','000009','000005') AND trade\_date BETWEEN('2013-12-31 00:00:00') AND('2018-12-31 00:00:00')

ORDER BY `trade\_date` DESC, `index\_name` DESC " # 选择了上证指数、医药生物、申万消费、中国宝安、世纪星源2013年末至2018末的收盘价数据

dbGetQuery(mydb,SQL\_statement)

rowdata <- dbGetQuery(mydb,SQL\_statement) # 导入原始数据

install.packages("dplyr")

library(dplyr)

data1<-select(rowdata,trade\_date,index\_code,last) # 处理原始数据

write.csv(data1, file="D:/123/data1.csv") # 将处理后的数据导出到本地，以csv格式保存

#载入一些必要的包

install.packages("tidyverse")

install.packages("lubridate")

install.packages("readxl")

install.packages("highcharter")

install.packages("tidyquant")

install.packages("timetk")

install.packages("tibbletime")

install.packages("quantmod")

install.packages("PerformanceAnalytics")

install.packages("scales")

library(tidyverse)

library(lubridate)

library(readxl)

library(highcharter)

library(tidyquant)

library(timetk)

library(tibbletime)

library(quantmod)

library(PerformanceAnalytics)

library(scales)

# 第一部分：Get Returns

##Get Prices

symbols <- c("SZ","SJ", "BA", "YY","SW") # 为选取的股票设置名称作为标识。

#这里本应该选取股票代码，但是数字在选取过程中会出现错误。因此，选取首字母代替，“SZ即上证指数”、“SJ即世纪星源”、“BA即中国宝安”、“YY即医药生物”、“SW即申万消费”.

prices <-

read\_excel("F:/R.example/basic/data2.xlsx", # 将数据库中导出的csv格式数据表格进行初步处理及排序之后。保存为xlsx格式，并通过read\_excel()函数导入Rstudio中。

col\_types =

c("text", "numeric",

"numeric", "numeric",

"numeric", "numeric")) %>% # 对表格中数据进行规定格式，其中。将date数据规定为text格式，将所选取的五只股票和或股指的收盘价指定为numeric格式。

mutate(date = ymd(date)) %>%

tk\_xts(date\_var = date) # 将text格式的date数据调整为日期格式

#To Monthly Returns in the xts World

prices\_monthly <-

to.monthly(prices,

indexAt = "lastof",

OHLC = FALSE) # 将前面处理好的按日计算的prices调整为月度数据，并规定为月末计算

asset\_returns\_xts <-

na.omit(Return.calculate(prices\_monthly,

method = "log")) # 利用处理好的月度价格计算收益，并规定为log方式。

#To Monthly Returns in the tidyverse

asset\_returns\_dplyr\_byhand <-

prices %>%

to.monthly(indexAt = "lastof", OHLC = FALSE) %>%

data.frame(date = index(.)) %>% # 将标识为date的ndex调整为date列，即引入date列作为单独的一列.

remove\_rownames() %>% # 基于上述生成单独的date列，移除掉原有的index。

gather(asset, returns, -date) %>% # 将生成的date列、returns数据等汇入原有数据中,但并未进行实际计算，只是调整为整洁的格式

group\_by(asset) %>%

mutate(returns = (log(returns) - log(lag(returns)))) %>% # 计算用log表示的收益并加入到data frame中，而且，这其中，会对原始价格数据进行处理

spread(asset, returns) %>% # 将处理好的数据由长格式转回到宽格式，便于后续的数据比较、处理及阅读。

select(date, symbols) %>%

na.omit() #将调整好的数据进行重新排列，以恢复原有格式。

#To Monthly Returns in the tidyquant World 也就是在 tidyquant world中进行数据的调整，类似于上述在tidyverse中的调整，目的在于同上述数据调整进行比较

asset\_returns\_tq\_builtin <-

prices %>%

tk\_tbl(preserve\_index = TRUE,

rename\_index = "date") %>%

gather(asset, prices, -date) %>%

group\_by(asset) %>%

tq\_transmute(mutate\_fun = periodReturn,

period = "monthly",

type = "log") %>%

spread(asset, monthly.returns) %>%

select(date, symbols) %>%

na.omit()

#To Monthly Returns using tibbletime 也就是使用 tibbletime 进行数据的调整，类似于上述在tidyverse、tidyquant world中的调整，目的在于同上述数据调整进行比较

asset\_returns\_tbltime <-

prices %>%

to.monthly(indexAt = "lastof",

OHLC = FALSE) %>%

tk\_tbl(preserve\_index = TRUE,

rename\_index = "date") %>%

# this is the the tibbletime function

tbl\_time(index = "date") %>%

gather(asset, returns, -date) %>%

group\_by(asset) %>%

tq\_transmute(mutate\_fun = periodReturn,

type = "log") %>%

spread(asset, monthly.returns) %>%

select(date, symbols) %>%

na.omit()

#Tidy Asset Returns

asset\_returns\_long <-

asset\_returns\_dplyr\_byhand %>%

gather(asset, returns, -date)

#Portfolio Returns 这里主要是手动构建资产组合，为后续的分析建立基础。包括分配权重以及加权平均等。

w <- c(0.25,

0.25,

0.20,

0.20,

0.10)

#Portfolio Returns By-Hand

w\_1 <- w[1]

w\_2 <- w[2]

w\_3 <- w[3]

w\_4 <- w[4]

w\_5 <- w[5]

asset1 <- asset\_returns\_xts[,1]

asset2 <- asset\_returns\_xts[,2]

asset3 <- asset\_returns\_xts[,3]

asset4 <- asset\_returns\_xts[,4]

asset5 <- asset\_returns\_xts[,5]

portfolio\_returns\_byhand <-

(w\_1 \* asset1) +

(w\_2 \* asset2) +

(w\_3 \* asset3) +

(w\_4 \* asset4) +

(w\_5 \* asset5)

names(portfolio\_returns\_byhand) <- "returns"

#Portfolio Returns in the xts World 这里是利用xts World的方式进行资产组合的构建及回报的计算，目的在于同上述手动计算进行比较

portfolio\_returns\_xts\_rebalanced\_monthly <-

Return.portfolio(asset\_returns\_xts,

weights = w,

rebalance\_on = "months") %>%

`colnames<-`("returns")

#Portfolio Returns in the tidyverse 这里是利用tidyverse的方式进行资产组合的构建及回报的计算，目的在于同上述两种方式进行比较

portfolio\_returns\_dplyr\_byhand <-

asset\_returns\_long %>%

group\_by(asset) %>%

mutate(weights = case\_when(asset == symbols[1] ~ w[1],

asset == symbols[2] ~ w[2],

asset == symbols[3] ~ w[3],

asset == symbols[4] ~ w[4],

asset == symbols[5] ~ w[5]),

weighted\_returns = returns \* weights) %>%

group\_by(date) %>%

summarise(returns = sum(weighted\_returns))

#Portfolio Returns in the tidyquant World 这里是利用tidyquant World的方式进行资产组合的构建及回报的计算，目的在于同上述三种方式进行比较

portfolio\_returns\_tq\_rebalanced\_monthly <-

asset\_returns\_long %>%

tq\_portfolio(assets\_col = asset,

returns\_col = returns,

weights = w,

col\_rename = "returns",

rebalance\_on = "months")

#8.CAPM Beta

market\_returns\_xts <-

read\_excel("F:/R.example/basic/data-market.xlsx",

col\_types =

c("text", "numeric")) %>%

mutate(date = ymd(date)) %>%

tk\_xts(date\_var = date) %>%

to.monthly(indexAt = "lastof",

OHLC = FALSE) %>%

Return.calculate(.,

method = "log") %>%

na.omit() #这一部分主要是引入市场收益，所使用的方法和最开始引入构建投资组合所使用五只股票或股指方式相同。利用管道函数直接处理为月度收益。

market\_returns\_tidy <-

market\_returns\_xts %>%

tk\_tbl(preserve\_index = TRUE,

rename\_index = "date") %>%

na.omit() %>%

select(date, returns = SZ) # 对得到的数据进行处理，使用tidyverse的同时，加入一个tibble格式的数据。

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

mutate(market\_returns = market\_returns\_tidy$returns) %>%

head() # 为接下来投资组合的市场回报回归做准备，确保投资组合回报和市场回报的观察数目相同。

# 利用投资组合β = 投资组合收益和市场收益的协方差/市场收益的方差，来计算投资组合的β。

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

market\_returns\_tidy$returns)/

var(market\_returns\_tidy$returns)

# 这里得到的投资组合β接近1，即假设的市场组合β值，这很正常，毕竟我们选择的组合成员中，股指占据比例很大，尤其是上证指数“SZ”占据了25%。

# 利用投资组合的β = 投资组合中各组成资产的β和相应资产权重的加权。

beta\_assets <-

asset\_returns\_long %>%

nest(-asset)

beta\_assets # 对资产收益数据进行初步处理，将资产名称和资产收益并入同框并列示

beta\_assets <-

asset\_returns\_long %>%

nest(-asset) %>%

mutate(model =

map(data, ~

lm(returns ~ market\_returns\_tidy$returns,

data = .)))

beta\_assets # 利用map() + lm() 函数的组合，实现每个单独资产收益对市场收益的快速回归，并将所得结果保存为新列：model中。因此，现在列表中存在三组数据

library(broom)

beta\_assets <-

asset\_returns\_long %>%

nest(-asset) %>%

mutate(model =

map(data, ~

lm(returns ~ market\_returns\_tidy$returns,

data = .))) %>%

mutate(model = map(model, tidy))

beta\_assets

beta\_assets <-

asset\_returns\_long %>%

nest(-asset) %>% mutate(model =

map(data, ~ lm(returns ~ market\_returns\_tidy$returns,

data = .))) %>%

mutate(model = map(model, tidy)) %>%

unnest(model) %>%

filter(term != "(Intercept)") %>%

mutate\_if(is.numeric, funs(round(., 4))) %>%

select(-term)

beta\_assets # 主要是为了提高数据的可读性和整洁性

beta\_assets %>%

select(asset, estimate) %>%

filter(asset == "SZ") #利用市场组合的贝塔为1这一特点，对所构建的模型进行迅速检查。

# 接下来，利用这些资产β的甲醛组合来计算投资组合的β。

beta\_byhand <-

w[1] \* beta\_assets$estimate[1] +

w[2] \* beta\_assets$estimate[2] +

w[3] \* beta\_assets$estimate[3] +

w[4] \* beta\_assets$estimate[4] +

w[5] \* beta\_assets$estimate[5]

beta\_byhand

# 结果显示，同第一种方法计算出的投资组合β值相同。

# 在xts world中计算资产组合以及市场组合的β值。

beta\_builtin\_xts <-

CAPM.beta(portfolio\_returns\_xts\_rebalanced\_monthly,

market\_returns\_xts)

beta\_builtin\_xts

# 结果显示，同前面两种计算的结果相同。

# 利用tidyverse的方法进行计算

beta\_dplyr\_byhand <-

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

do(model =

lm(returns ~ market\_returns\_tidy$returns,

data = .)) %>%

tidy(model) %>%

mutate(term = c("alpha", "beta")) %>%

select(estimate)

beta\_dplyr\_byhand$estimate[2]

# 结果显示，同前面的结果相同。

# 最终，利用tq\_performance() 函数，将计算的结果应用到收益中。

beta\_builtin\_tq <-

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

mutate(market\_return =

market\_returns\_tidy$returns) %>%

na.omit() %>%

tq\_performance(Ra = returns,

Rb = market\_return,

performance\_fun = CAPM.beta) %>%

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

beta\_builtin\_tq %>%

mutate(dplyr\_beta = beta\_dplyr\_byhand$estimate[2],

byhand\_beta = beta\_byhand,

xts\_beta = coredata(beta\_builtin\_xts)) %>%

round(3)

# 结果列示，利用手动计算以及内嵌函数的方式计算出的资产组合β相同。

# 下面就是利用集中绘图方式，对CAPM的运用结果进行可视化列示。

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

mutate(market\_returns =

market\_returns\_tidy$returns) %>%

ggplot(aes(x = market\_returns,

y = returns)) +

geom\_point(color = "cornflowerblue") +

ylab("portfolio returns") +

xlab("market returns")

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

mutate(market\_returns =

market\_returns\_tidy$returns) %>%

ggplot(aes(x = market\_returns,

y = returns)) +

geom\_point(color = "cornflowerblue") +

geom\_smooth(method = "lm",

se = FALSE,

color = "green") +

ylab("portfolio returns") +

xlab("market returns")

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

mutate(market\_returns = market\_returns\_tidy$returns) %>%

ggplot(aes(x = market\_returns, y = returns)) +

geom\_point(color = "cornflowerblue") +

geom\_abline(aes(

intercept = beta\_dplyr\_byhand$estimate[1],

slope = beta\_dplyr\_byhand$estimate[2]),

color = "purple") +

ylab("portfolio returns") +

xlab("market returns")

# 从以上三幅图中可以较直白的看初，两者具有较强的线性，这也符合我们所计算出的结果。

# 从图像角度分析拟合的收益值同实际收益值之间的匹配程度。

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

mutate(market\_returns =

market\_returns\_tidy$returns) %>%

ggplot(aes(x = market\_returns,

y = returns)) +

geom\_point(color = "cornflowerblue") +

geom\_abline(

aes(intercept =

beta\_dplyr\_byhand$estimate[1],

slope = beta\_dplyr\_byhand$estimate[2]),

color = "purple") +

geom\_smooth(method = "lm",

se = FALSE,

color = "green") +

ylab("portfolio returns") +

xlab("market returns")

portfolio\_model\_augmented <-

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

do(model =

lm(returns ~

market\_returns\_tidy$returns, data = .)) %>%

augment(model) %>%

rename(mkt\_rtns = market\_returns\_tidy.returns) %>%

select(returns, mkt\_rtns, .fitted) %>%

mutate(date = portfolio\_returns\_tq\_rebalanced\_monthly$date)

head(portfolio\_model\_augmented, 3)

portfolio\_model\_augmented %>%

ggplot(aes(x = date)) +

geom\_line(aes(y = returns),

color = "cornflowerblue") +

geom\_line(aes(y = .fitted),

color = "green") +

xlab("date")

# 结果显示，拟合的收益值同实际收益值之间的匹配程度较高,符合要求。

# 利用highcharter重新绘制上述绘图，用以比较学习。

highchart() %>%

hc\_title(text = "Portfolio v. Market Returns Scatter") %>%

hc\_add\_series(portfolio\_model\_augmented,

type = "scatter",

color = "cornflowerblue",

hcaes(x = round(mkt\_rtns, 4),

y = round(returns, 4)),

name = "Returns") %>%

hc\_xAxis(title = list(text = "Market Returns")) %>%

hc\_yAxis(title = list(text = "Portfolio Returns")) %>%

hc\_add\_theme(hc\_theme\_flat()) %>%

hc\_exporting(enabled = TRUE)

highchart() %>%

hc\_title(text = "Portfolio v. Market Returns Scatter w/Date") %>%

hc\_add\_series(portfolio\_model\_augmented,

type = "scatter",

color = "cornflowerblue",

hcaes(x = round(mkt\_rtns, 4),

y = round(returns, 4),

date = date),

name = "Returns") %>%

hc\_xAxis(title = list(text = "Market Returns")) %>%

hc\_yAxis(title = list(text = "Portfolio Returns")) %>%

hc\_tooltip(formatter = JS("function(){

return ('port return: ' + this.y + ' <br> mkt return: ' + this.x +

' <br> date: ' + this.point.date)}")) %>%

hc\_add\_theme(hc\_theme\_flat()) %>%

hc\_exporting(enabled = TRUE)

highchart() %>%

hc\_title(text = "Scatter with Regression Line") %>%

hc\_add\_series(portfolio\_model\_augmented,

type = "scatter",

color = "cornflowerblue",

hcaes(x = round(mkt\_rtns, 4),

y = round(returns, 4),

date = date),

name = "Returns") %>%

hc\_add\_series(portfolio\_model\_augmented,

type = "line",

hcaes(x = mkt\_rtns, y = .fitted),

name = "CAPM Beta = Regression Slope") %>%

hc\_xAxis(title = list(text = "Market Returns")) %>%

hc\_yAxis(title = list(text = "Portfolio Returns")) %>%

hc\_tooltip(formatter = JS("function(){

return ('port return: ' + this.y + ' <br> mkt return: ' + this.x +

' <br> date: ' + this.point.date)}"))%>%

hc\_add\_theme(hc\_theme\_flat()) %>%

hc\_exporting(enabled = TRUE)