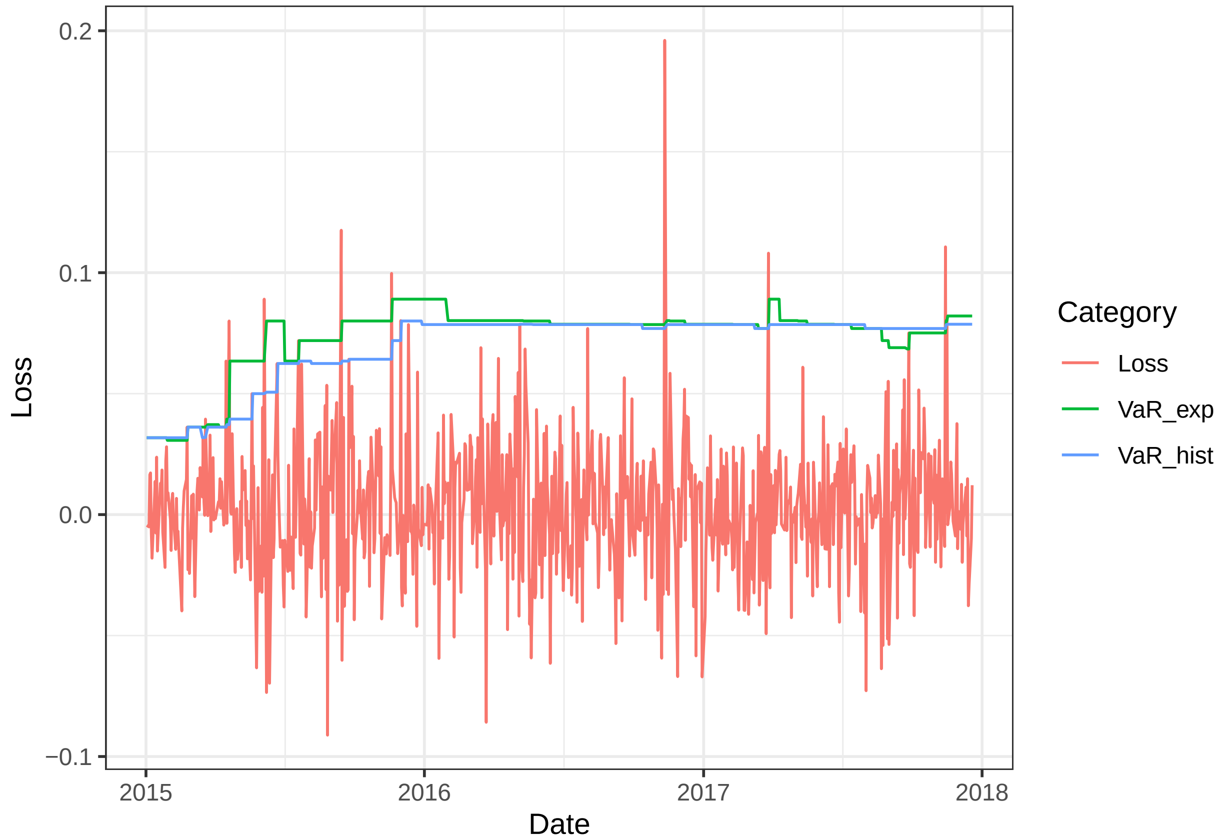
Homework 3: solution suggestions

April 24, 2020

Question 1

part 1

```
data <- read.csv('hw3_returns2.csv')
data$Date <- as.Date(data$Date, "%m/%d/%Y")

comp_dates <- data %>% filter(Date >= as.Date('2015-01-01')) %>% .$Date

exp_weight <- function(lambda,N){
  lambda^(N-1:N)*(1-lambda)/(1-lambda^N)
}


historical_VaR <- numeric(length(comp_dates))
exponential_VaR <- numeric(length(comp_dates))
for(t in 1:length(comp_dates)){
  cache <- data %>%
    filter(Date < comp_dates[t])
  historical_VaR[t] <-
    cache %>%
    arrange(Return) %>%
    .[ceiling(0.01*dim(cache)[1]),2]
  exponential_VaR[t] <-
    cache %>%
    mutate(weights = exp_weight(0.995,dim(cache)[1])) %>%
    arrange(Return) %>%
    mutate(cum_w = cumsum(weights)) %>%
    filter(cum_w >= 0.01) %>%
    .[1,2]
}

# let W_0 = 0
historical_VaR <- abs(pmin(historical_VaR,0))
exponential_VaR <- abs(pmin(exponential_VaR,0))

plotdata <-
  data.frame(Date = comp_dates,
             VaR_hist = historical_VaR,
             VaR_exp = exponential_VaR,
             Loss = -(data %>% filter(Date >= as.Date('2015-01-01')) %>% .$Return)) # loss not return


plotdata2 <- gather(plotdata, Category, Loss, VaR_hist:Loss)
ggplot(data = plotdata2,aes(x = Date, y = Loss, group = Category)) + geom_line(aes(color = Category)) +
```



2

```
# number exceptions historical approach
sum(plotdata$Loss > plotdata$VaR_hist)
```

```
## [1] 18
```

```
# number exceptions exponential approach
sum(plotdata$Loss > plotdata$VaR_exp)
```

```
## [1] 13
```

```
# probability of at least as many exception as with historical approach
1 - pbinom(sum(plotdata$Loss > plotdata$VaR_hist) - 1, dim(plotdata)[1], 0.01)
```

```
## [1] 0.0007138522
```

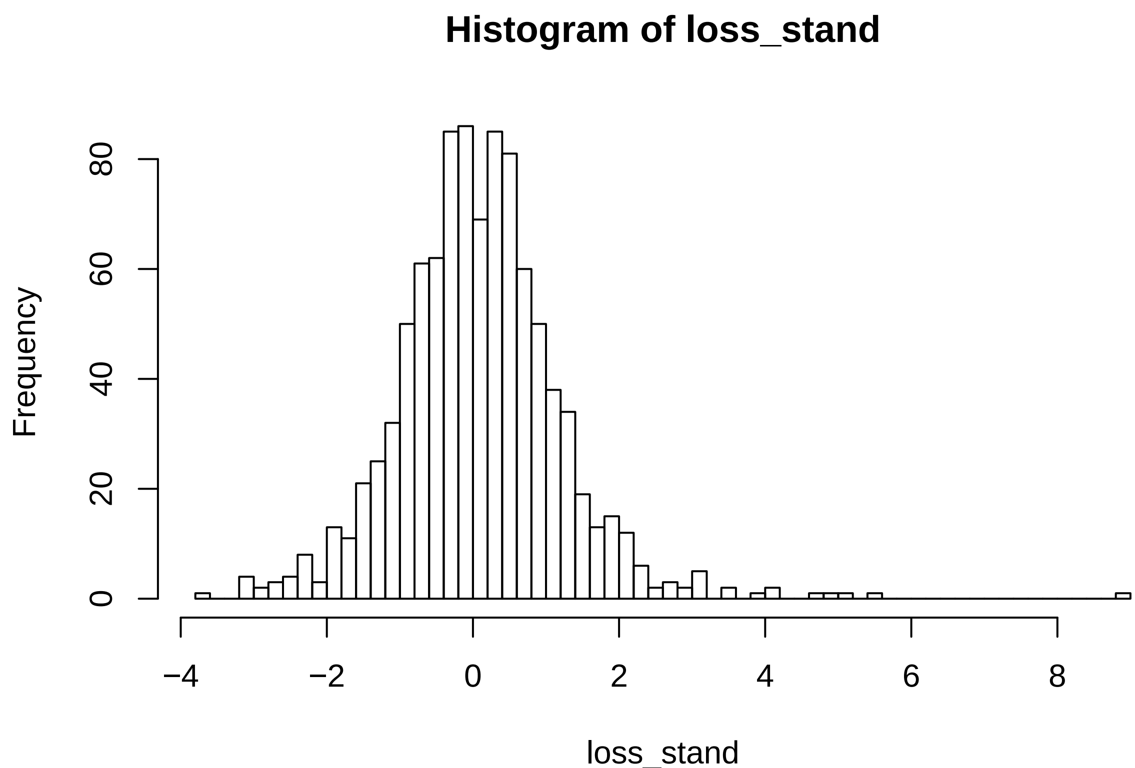
```
# probability of at least as many exception as with exponential approach
1 - pbinom(sum(plotdata$Loss > plotdata$VaR_exp) - 1, dim(plotdata)[1], 0.01)
```

```
## [1] 0.04112061
```

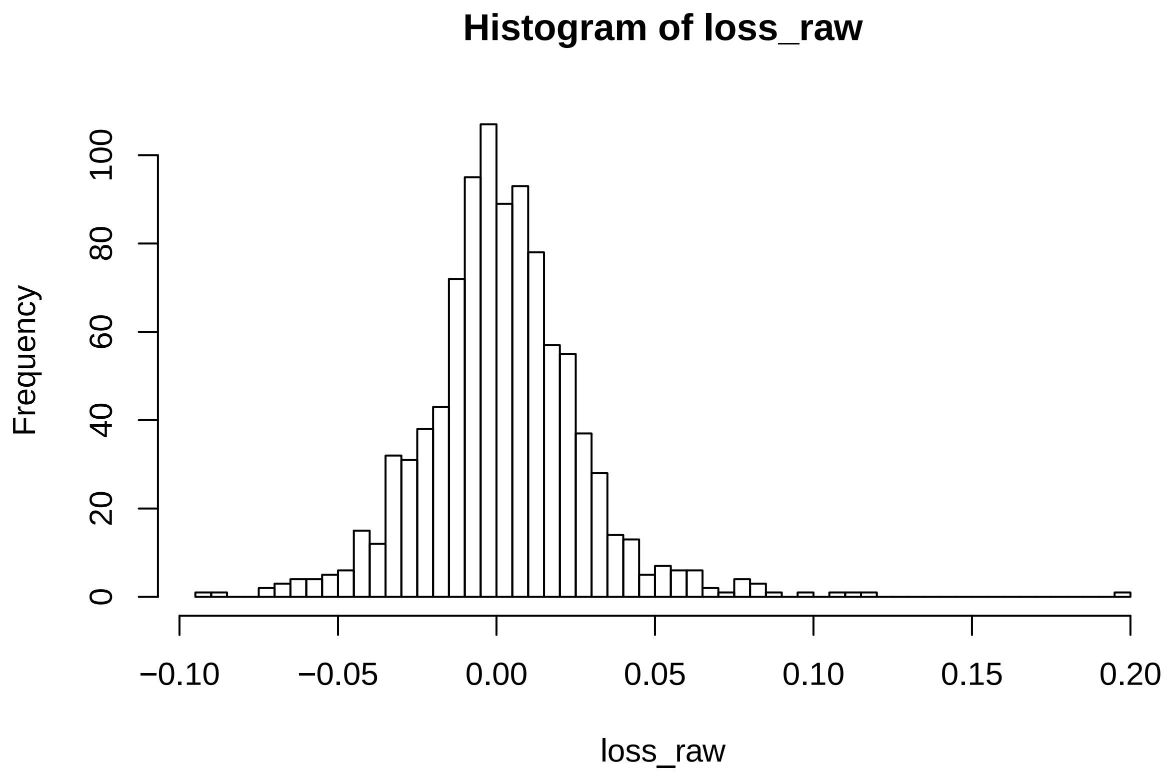
The number of exceptions is lower for the exponential approach, which puts more weight on recent observations. This already hints at the fact that there might be time-variation (e.g. of volatility) of returns, such that we can do better than just drawing with equal probability from past observations.

3

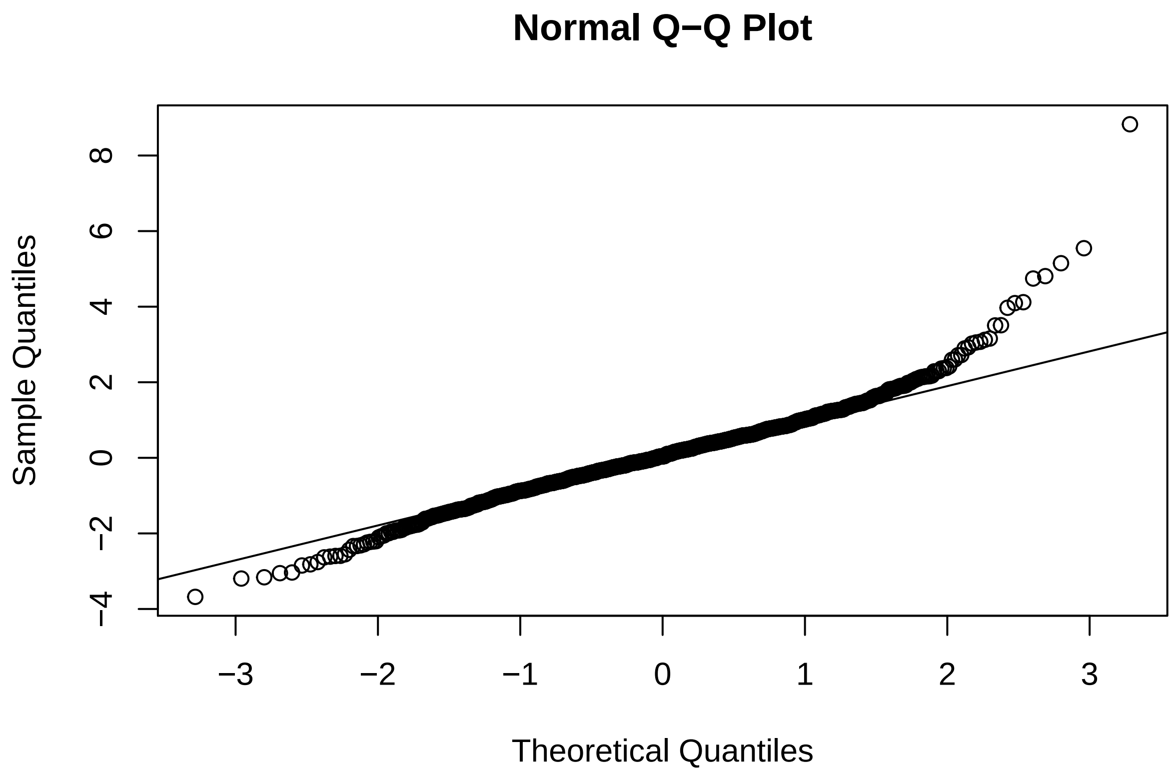
```r
loss_raw <- -data$Return

# use n previous trading days to estimate svol
N <- 1000
n <- 25
vol_est <- sapply((n+1):N,function(z)sd(loss_raw[(z-n):(z-1)]))

loss_raw <- loss_raw[-c(1:n)]
loss_stand <- loss_raw/vol_est
hist(loss_stand,breaks = 50)
```



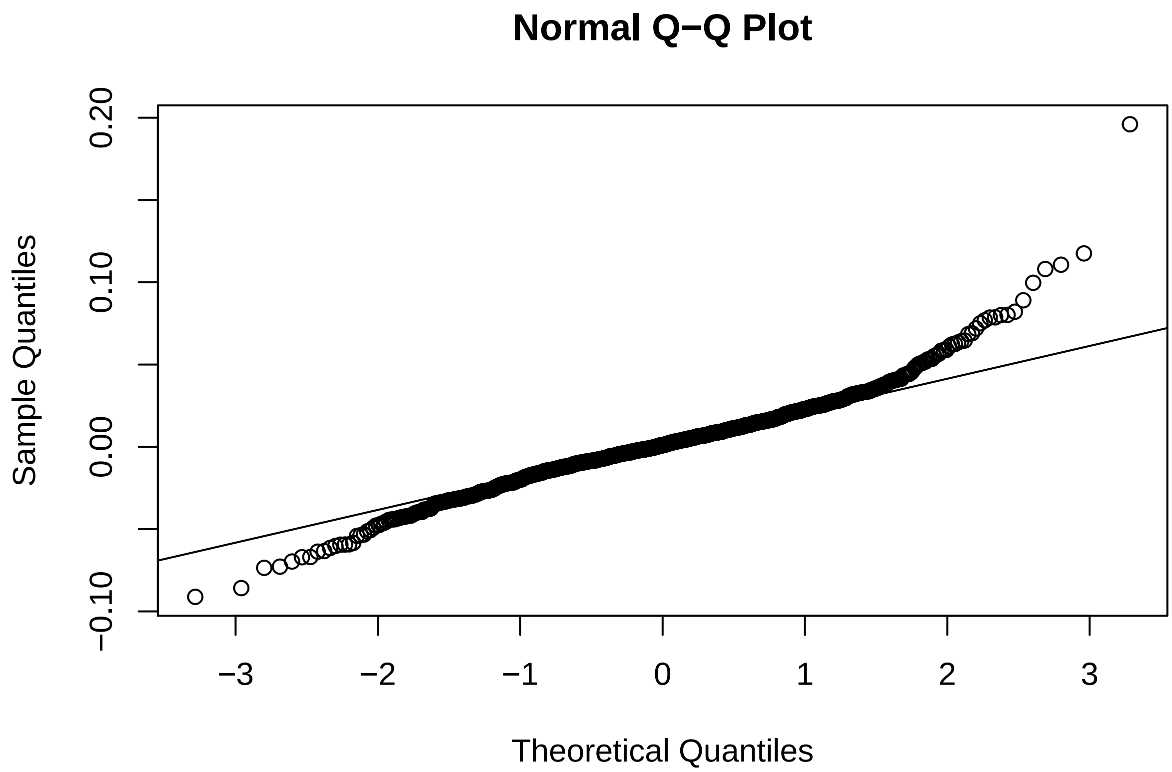

```r
hist(loss_raw,breaks = 50)
```




```
qqnorm(loss_stand)
qqline(loss_stand)
```




```
qqnorm(loss_raw)
qqline(loss_raw)
```



The loss idstribution now looks closer to normal, with the expection of the right tail of the loss distribution, which is exactly the days we are interested in for the VaR.

4

What you can do now is use the standardized losses to compute a VaR using the historical method (or using a standard normal, or using the exponential approach), and then multiply with the current estimate for volatility. The result should deliver fewer exceptions than before, but perhaps still too many, given that we still have a fat right tail of the loss distribution.

Question 2

1

We can proof this by contradiction. Assume to the contrary that 8 people are born in a three-year period, and there exists a case in which at most 2 of them are born within any one-year period. Then the maximum number of people who could have been born in the three years is $2 + 2 + 2 = 6 < 8$. The maximum over a set containing 8 being 6 is a contradiction.

2

You need to make an assumption here, e.g. normal distribution.

3

You can use the binomial test here.

4

There are many things you could do here. What they have in common is that ideally you will use data from previous scheduled FOMC meetings. For instance, you could assume that returns are normal, but where variance is higher on days of FOMC announcements. You could estimate this higher variance from previous announcements you have data about.

Setup Python

```r
#install.packages('reticulate')
library(reticulate)
```

We can now use `Python` code junks. The version used will be the one from your `PATH` directory:

```python
import sys
sys.version_info
```

```
## sys.version_info(major=3, minor=7, micro=3, releaselevel='final', serial=0)
```