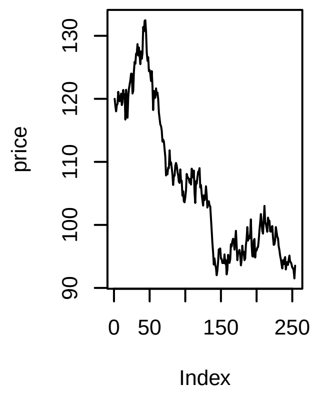
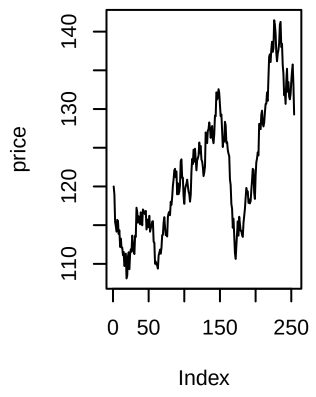
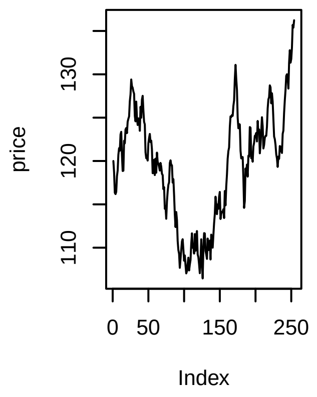
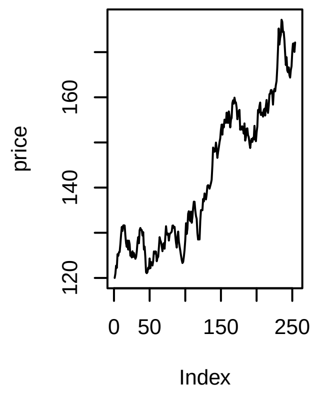
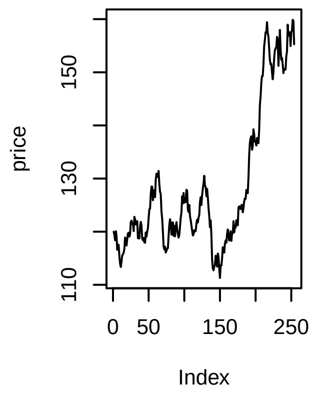
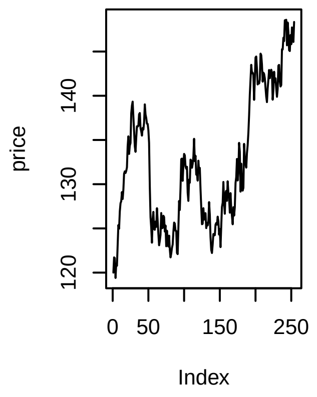
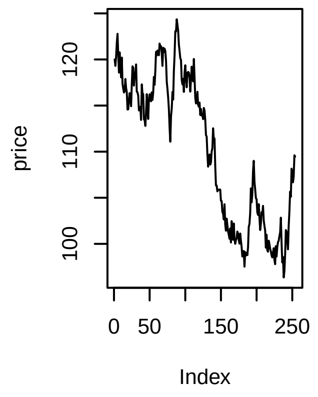
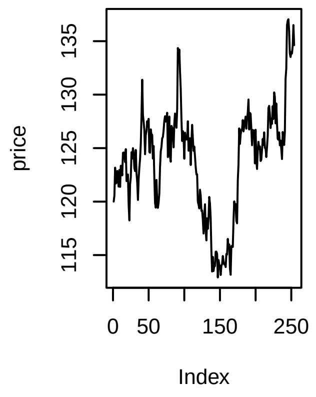
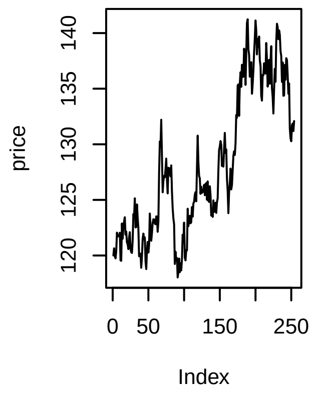
STAT 631 Homework 1

Jack Cunningham (jgavc@tamu.edu)

9/3/24

1)

```r
set.seed(2012)
n = 253
par(mfrow = c(3,3))
for (i in (1:9)){
  logr = rnorm(n, 0.05/253, 0.2/sqrt(253))
  price = c(120, 120 * exp(cumsum(logr)))
  plot(price, type = "l")
}
```


price
90
100
110
120
130
0
50
150
250
Index

price
110
120
130
140
0
50
150
250
Index

price
110
120
130
0
50
150
250
Index

price
120
140
160
0
50
150
250
Index

price
110
130
150
0
50
150
250
Index

price
120
130
140
0
50
150
250
Index

price
100
110
120
0
50
150
250
Index

price
115
120
125
130
135
0
50
150
250
Index

price
120
125
130
135
140
0
50
150
250
Index

Problem 9)

We are generating the daily log-return data by using a normal distribution with parameters $\mu = \frac{0.05}{253}$ and $\sigma = \frac{0.2}{\sqrt{253}}$. Thus for a one year period (with 253 trading days) the mean of the log-return is $\mu t = \frac{0.05}{253}(253) = 0.05$ and the standard deviation is $\sigma\sqrt{t} = \frac{0.2}{\sqrt{253}}(\sqrt{253}) = 0.2$.

Problem 10)

We can see many clumps of time where it appears positive (negative) returns are followed by positive (negative) returns. In this case we are sure that this appearance of momentum is an illusion since we generated this simulation with i.i.d log-returns. In other words each day's return was independent and was not impacted by the previous days.

Problem 11)

First we have the c() function which serves to concatenate vectors. Its first entry is the initial price of the asset, 120. Its second entry is a bit more involved. The variable logr has each days generated log return. The cumsum function creates a cumulative summation of the vector logr. This allows us to track the log return as it moves over time. We take constant e and raise it by the cumulative log return to bring us back to the cumulative return. We then multiply by 120 to get the prices for each day.

2.

Problem 17)

```r
data = read.csv('MCD_PriceDaily.csv')
logReturn = diff(log(data[,"Adj.Close"]))
head(logReturn)
```

```
[1] -0.0076229801 -0.0137181518  0.0073522812 -0.0009395848  0.0076786593
[6]  0.0053958639
```

```r
#Simulation of situation
niter = 10000
set.seed(2015)
outcome = rep(0, niter)

for(i in (1:niter)){
  logr = rnorm(20, mean = mean(logReturn), sd = sd(logReturn))
  price = 93.07*exp(cumsum(logr))

  #Boolean with check to see if price went under 84.5
  outcome[i] = ifelse(sum(price < 84.5) >= 1, 125, -1)
}
```

```
#Expected Value from Bet
ev <- mean(outcome)
s <- sd(outcome)
lower <- ev - qt(.975,df = niter - 1)*s/sqrt(niter)
upper <- ev + qt(.975,df = niter - 1)*s/sqrt(niter)
ev
```

```
[1] -0.0802
```

```
c(lower,upper)
```

```
[1] -0.2904632  0.1300632
```

The expected value from the bet is -0.0802 with a confidence interval of (-0.29, 0.13). With this information I would avoid taking this bet.

3)

a)

Log-returns are normally distributed with parameters $\mu = 0.001, \sigma = 0.015$.

We know:

$\log(P_1) = \log(P_0) + \log(1 + R_1(t))$.

In this case we are looking for the chance of initial price $P_0 = 1000$ dropping to less than $P_1 = 990$ in the next day. That is a return of $R_1 = \frac{990-1000}{1000} = -0.01$. So we need to find the chance of a $\log(1 - 0.01) = \log(.99)$ move in the log return or less. That is:

$P[r \leq \log(.99)] = P[r^* \leq \frac{\log(.99)-0.001}{0.015}]$

```
Z = (log(.99) - .001)/0.015
chance = pnorm(Z)
chance
```

```
[1] 0.2306557
```

There is a probability of 0.231 that tomorrow price for this investment will drop to below $990.

b)

Since $r_1, r_2, ...r_t$ are i.i.d with a normal distribution. When r_1, r_2, r_3, r_4, r_5 are added together their cumulative distribution is $N(5\mu, 5\sigma^2)$. We have return $R_1 = \frac{990-1000}{1000} = -0.01$. so we need to find the chance of a $\log(1-0.01) = \log(.99)$ move in the log return or less. That is:

$P[r_5 \leq log(.99) = P[r_5^* \leq \frac{log(.99)-0.001}{0.015\sqrt{5}}]$

```
Z = (log(.99) - 0.001)/(0.015*sqrt(5))
chance = pnorm(Z)
chance
```

```
[1] 0.370905
```

There is a probability of 0.371 that this investment will be worth less than $990 after 5 trading days.

4)

Given $P_1 = 95, P_2 = 103, P_3 = 98$. We need to find $r_3(2)$.

We know that $r_t(k) = r_t + r_{t-1} + \cdots + r_{t-k+1}$

And that $r_t = \log(\frac{P_t}{P_{t-1}}) = p_t - p_{t-1}$

In this case $t = 3$ and $k = 2$. So:

$r_3(2) = r_3 + r_2 = \log(\frac{98}{103}) + \log(\frac{103}{95}) = \log(98) - \log(103) + \log(103) - \log(95) = \log(98) - \log(95)$

```
r_3_2 <- log(98)-log(95)
r_3_2
```

```
[1] 0.03109059
```

$r_3(2)$=0.03109.

```
#Generating Symbols
library(quantmod)
getSymbols(Symbols = c("^GSPC","CVX","AMZN"),
           from = "2005-01-01", to = "2024-08-01")
```

```
[1] "GSPC" "CVX"  "AMZN"
```

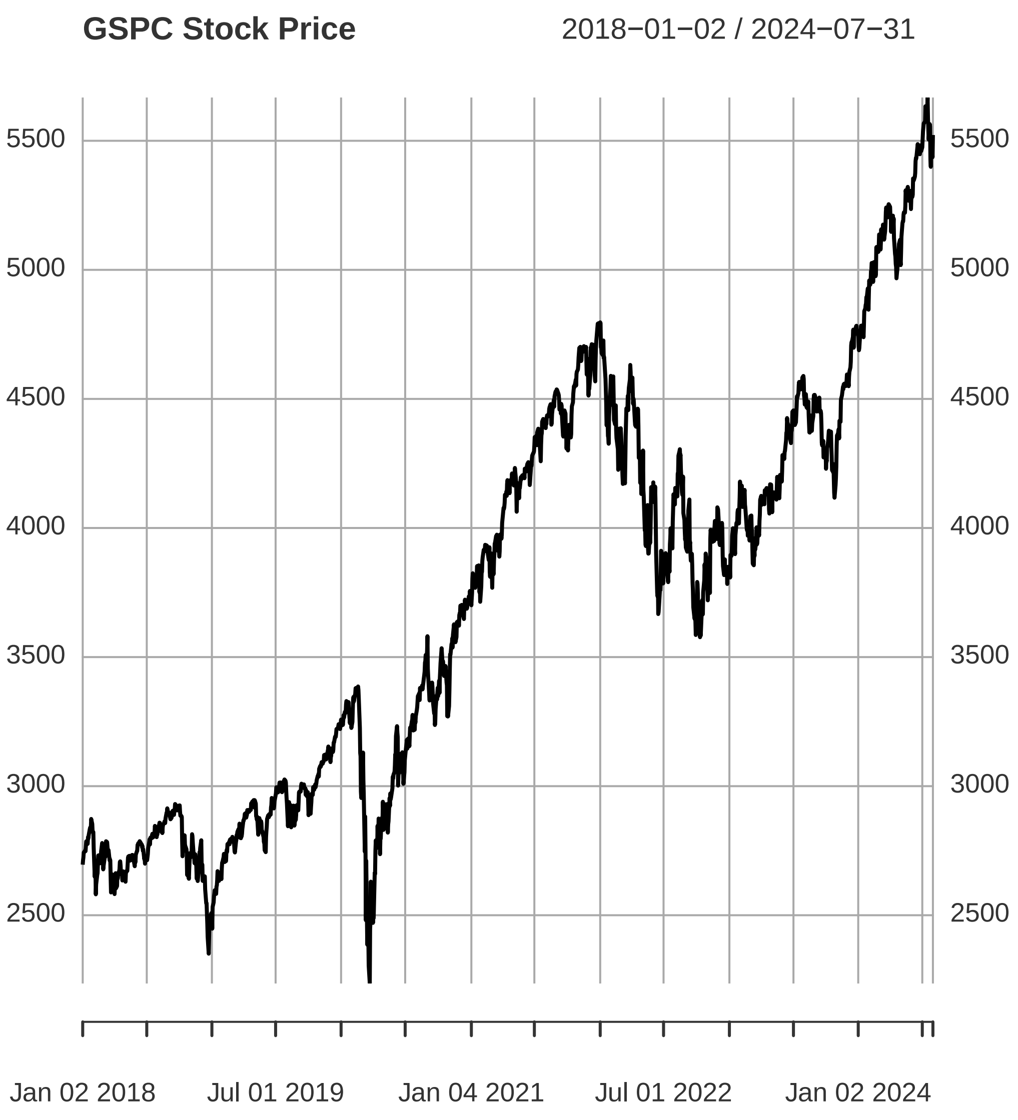
```r
#Getting Adjusted Close
SnP_500_AC <- GSPC[, "GSPC.Adjusted"]
Chevron_AC <- CVX[, "CVX.Adjusted"]
Amazon_AC <- AMZN[, "AMZN.Adjusted"]
```

```r
#Getting daily log return
SnP_500_dlr <- dailyReturn(SnP_500_AC, type = "log")
Chevron_dlr <- dailyReturn(Chevron_AC, type = "log")
Amazon_dlr <- dailyReturn(Amazon_AC, type = "log")
```

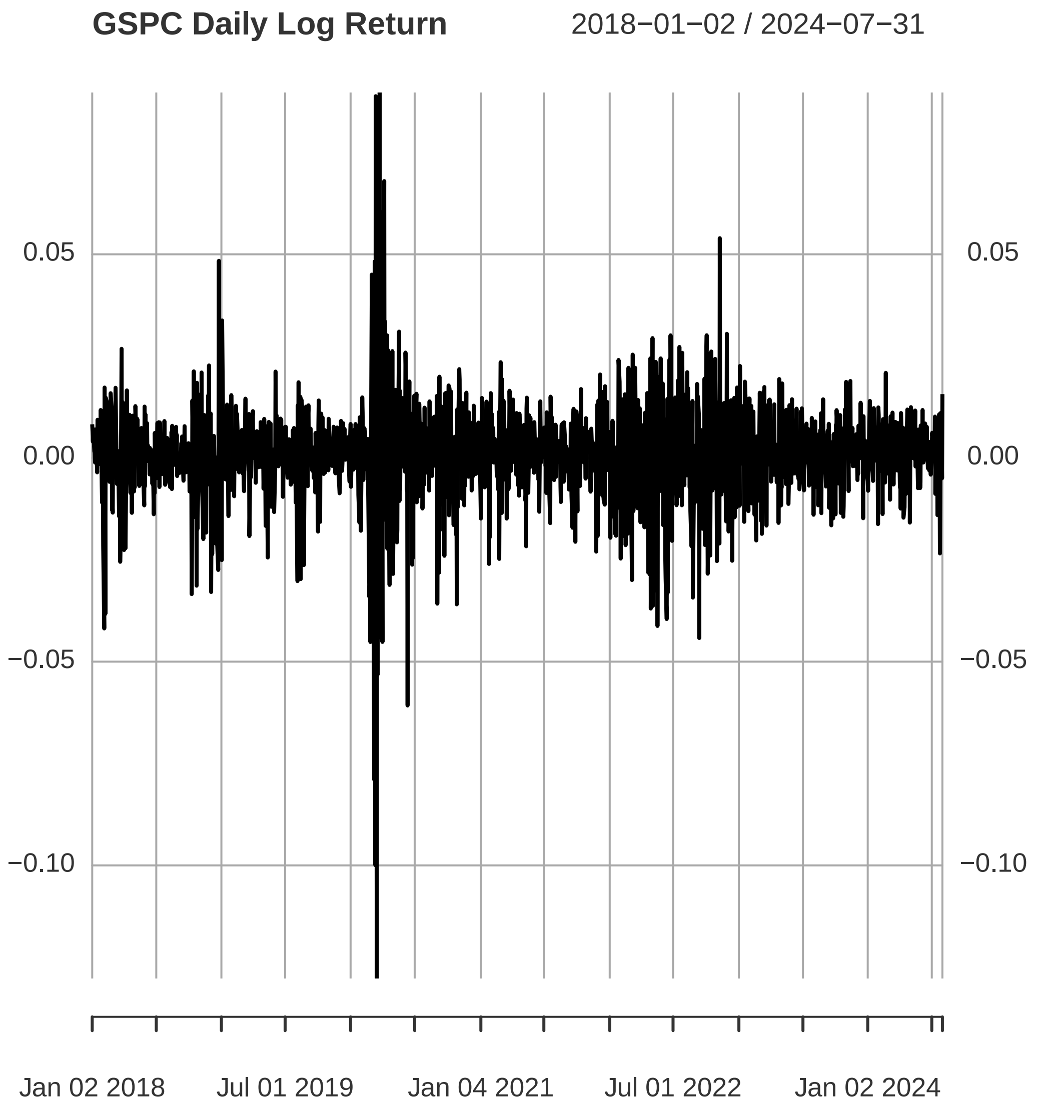
5.

a)

```r
plot(SnP_500_AC["2018-01-01::2024-8-01"], main = "GSPC Stock Price")
```



```r
plot(SnP_500_dlr["2018-01-01::2024-8-01"], main = "GSPC Daily Log Return")
```



One of the first major events in this time period was the sell off at the end of 2018. That December there were a few factors that drove volatility, most importantly in my opinion was a concern about the stability of the economy as the FED continued raising rates and reducing their balance sheet after a mediocre November 2018 jobs report. Another factor that drove a lot of volatility throughout 2018, and much of the Trump presidency, were the trade and tariff negotiations with China. There was a ton of action and counteraction between the US and China, tariffs and counter-tariffs would hit the news, or be tweeted directly from Trump himself, and cause a flurry of trading action. The reason I believe the overall economic condition concerns were the primary cause of the sell-off is because on December 1st 2018 the US and China agreed on a 90-day halt to new tariffs, seeming to indicate some positive traction on a trade deal. But it would remiss to not believe it was a combination of the two.

In August of 2019 there was a lot volatility despite a lack of direction on the month, this was mainly due to trade tensions between the US and China and a yield reversion in the bond market causing some recessionary fears. On August 5th Trump had accused China of currency manipulation after the yuan steeply fell in value, that day the S&P 500 dropped about 3% fearing a ignition in the trade war.

In Match of 2020 there was COVID-19. A historic amount of uncertainty was present in the market as investors debated how it would affect the global economy. Ultimately the recovery after the sell-off was rather steep.

Throughout 2022 and 2023 the market faced what appeared to be runaway inflation with CPI peaking at 9.1% and a new conflict as Russia invaded Ukraine. The ascent of inflation at this time forced the reluctant FED into action to begin raising rates. The FED had been consistent about the theory that inflation was transitory, a temporary blip in prices due to COVID supply chain disruption. However as 2022 progressed and the CPI kept hitting new highs the FED had to begin raising the Federal Funds rate, with 11 hikes between March of 2022 and July of 2023. The federal funds target went from 0-.25% to 5.25-5.50% in that time. At the same time the Russian invasion of Ukraine heightened the risk of a global conflict and lead to a realignment in the global economy as west powers began placing embargoes on Russia. The S&P would sell off and recover in 2023 and 2024.

Most recently there was a large amount of volatility at the start of August 2024. Part of the concern was a bad job report for July, raising concerns that a weakened job market would wreck the soft landing the FED has been striving for. Another aspect was a rate hike in Japan, the first since 2007, and an indication that there may be more hawkish actions to come. There was a large amount of institutions executing the “yen carry trade”. The trader would borrow yen for an extremely low interest rate and buy, for instance, a treasury bill in the US paying north of 5%. The trader faces foreign exchange risk in this transaction, if the value of the yen appreciates relative to the dollar the amount of yen they have to pay back becomes more expensive to buy. After the actions taken by the central back in Japan the unwinding of this transaction arguably caused a lot of volatility in these couple of days.

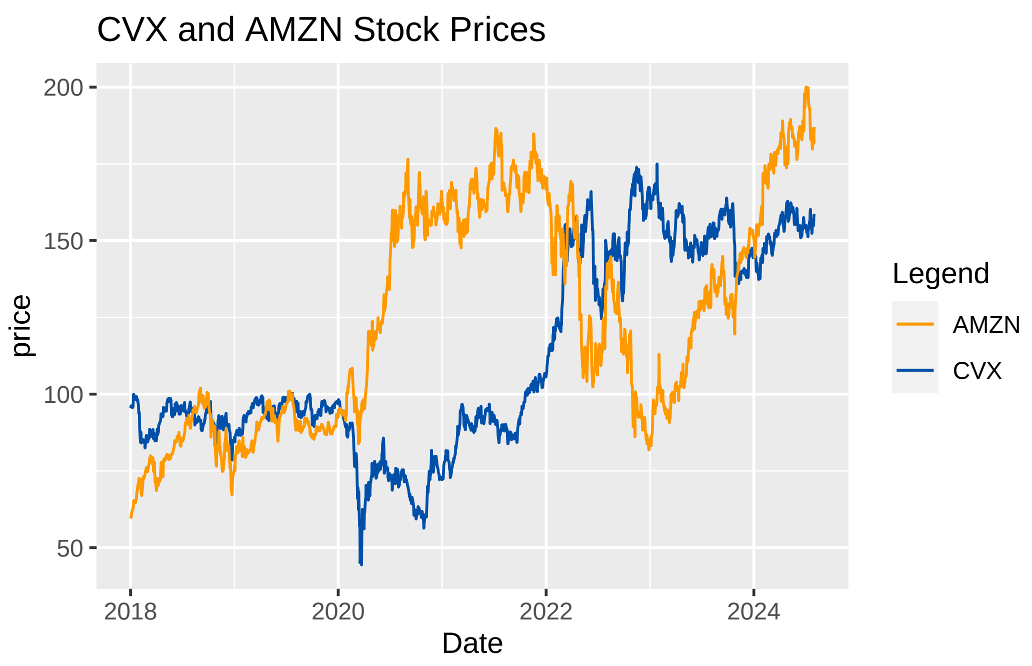
b)

```
library(tidyverse)
stock_data <- data.frame(
  date = index(GSPC),
  SnP_500_AD = as.numeric(GSPC[, "GSPC.Adjusted"]),
  Chevron_AD = as.numeric(CVX[, "CVX.Adjusted"]),
  Amazon_AD = as.numeric(AMZN[, "AMZN.Adjusted"]),
  SnP_500_dlr = as.numeric(dailyReturn(Ad(GSPC), type = "log")),
  Chevron_dlr = as.numeric(dailyReturn(Ad(CVX), type = "log")),
```

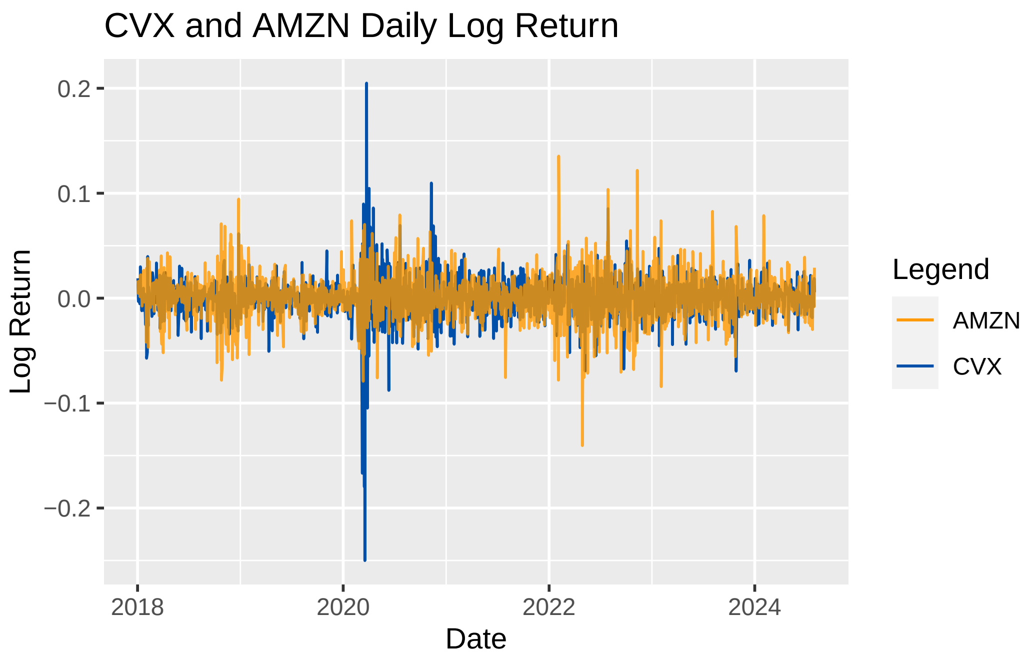
```r
  Amazon_dlr = as.numeric(dailyReturn(Ad(AMZN)), type = "log")
)
```

```r
library(tidyverse)
colors <- c("AMZN" = "#ff9900", "CVX" = "#0050AA")

stock_data |>
  filter(date > "2018-01-01") |>
  ggplot(aes(x = date)) +
  geom_line(aes(y = Chevron_AD, color = "CVX")) +
  geom_line(aes(y = Amazon_AD, color = "AMZN")) +
  labs(title = "CVX and AMZN Stock Prices", x = "Date", y = "price",
       color = "Legend") +
  scale_color_manual(values = colors)
```



```r
stock_data |>
  filter(date > "2018-01-01") |>
  ggplot(aes(x = date)) +
  geom_line(aes(y = Chevron_dlr, color = "CVX")) +
  geom_line(aes(y = Amazon_dlr, color = "AMZN"), alpha = 0.8) +
  labs(title = "CVX and AMZN Daily Log Return", x = "Date", y = "Log Return",
       color = "Legend") +
  scale_color_manual(values = colors)
```



The most striking impact to compare these two stocks is their performance in the COVID pandemic. Just by taking a look at the daily log return graph we can see the huge volatility at the start of the pandemic for Chevron while Amazon's was much more muted compared to Chevron and the overall market. Chevron is an energy corporation who fracks and produces oil and gas, and naturally is very exposed to the demand shrinking as it did under lock-downs in the early pandemic. Amazon on the other hand thrived in the lock-down era as many consumers preferred to order their essentials online instead of venturing into a store.

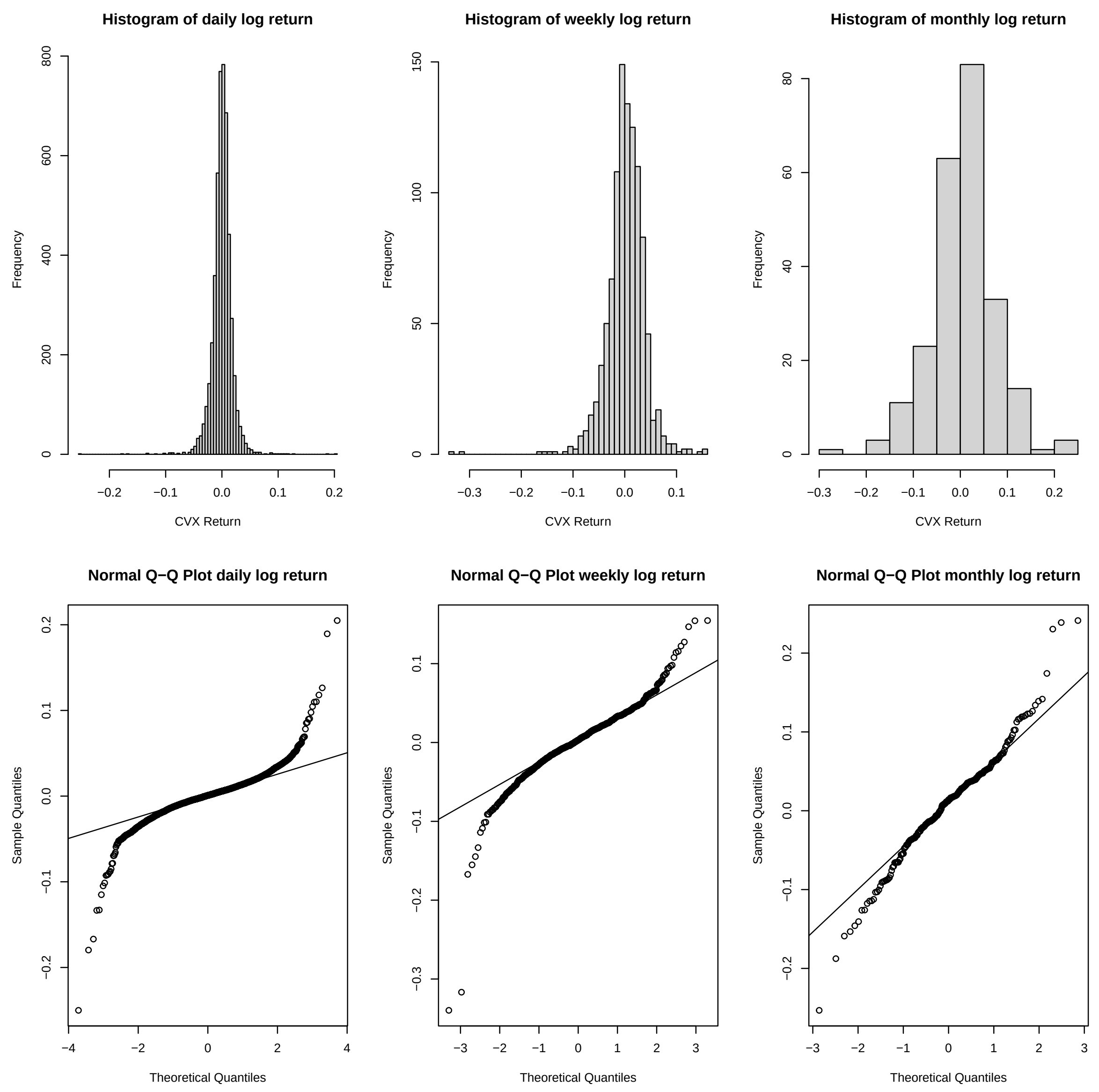
6)

a)

```
CVX_returns <- list(
  daily = dailyReturn(Ad(CVX), type = "log"),
  weekly = weeklyReturn(Ad(CVX), type = "log"),
  monthly = monthlyReturn(Ad(CVX), type = "log")
)

par(mfcol = c(2,3))
```

```r
for(i in  1:3 ){
  #Generating Histogram
  hist(CVX_returns[[i]], breaks = "scott",
       main = paste("Histogram of", names(CVX_returns)[i],"log return"),
       xlab = "CVX Return")
  #Generating qq plot
  qqnorm(CVX_returns[[i]],
         main = paste("Normal Q-Q Plot",names(CVX_returns)[i],"log return"))
  qqline(CVX_returns[[i]])
}
```



b)

In the daily log return quantile we can see the long and fat tails in the sample quantile when compared to the standard normal. As the time of return gets larger we the tails get less fat but still maintaining too many outliers to be well fit by a normal distribution.

We can also see that each distribution has a left skew although it becomes less severe in the monthly returns.

c)

```r
#Defining functions to compute skewness and kurtosis
Sk.fun <- function(x) { ## function to compute skewness
mean((x-mean(x))^3/sd(x)^3)
}
Kur.fun <- function(x){ ## function to compute kurtosis
mean((x-mean(x))^4/sd(x)^4)
}
```

```r
daily_skewness <- Sk.fun(CVX_returns[[1]])
daily_kurtosis <- Kur.fun(CVX_returns[[1]])
daily_shapiro_wilk <- shapiro.test((as.numeric(CVX_returns[[1]])))

weekly_skewness <- Sk.fun(as.numeric(CVX_returns[[2]]))
weekly_kurtosis <- Kur.fun(as.numeric(CVX_returns[[2]]))
weekly_shapiro_wilk <- shapiro.test(as.numeric(CVX_returns[[2]]))

monthly_skewness <- Sk.fun(as.numeric(CVX_returns[[3]]))
monthly_kurtosis <- Kur.fun(as.numeric(CVX_returns[[3]]))
monthly_shapiro_wilk <- shapiro.test(as.numeric(CVX_returns[[3]]))

paste("Daily statistics: Skewness:",daily_skewness,"Kurtosis:",daily_kurtosis)
```

```
[1] "Daily statistics: Skewness: -0.506890218875313 Kurtosis: 24.6939731638367"
```

```r
daily_shapiro_wilk
```

```

    Shapiro-Wilk normality test

data:  (as.numeric(CVX_returns[[1]]))
W = 0.87278, p-value < 2.2e-16
```

```r
paste("Weekly statistics: Skewness:",weekly_skewness,"Kurtosis:",weekly_kurtosis)
```

```
[1] "Weekly statistics: Skewness: -1.42259980992555 Kurtosis: 15.7761938857281"
```

```r
weekly_shapiro_wilk
```

```
    Shapiro-Wilk normality test

data:  as.numeric(CVX_returns[[2]])
W = 0.90397, p-value < 2.2e-16
```

```
paste("Monthly statistics: Skewness:",monthly_skewness,"Kurtosis:",
      monthly_kurtosis)
```

```
[1] "Monthly statistics: Skewness: -0.0355884507604814 Kurtosis: 4.69414276250485"
```

```
monthly_shapiro_wilk
```

```
    Shapiro-Wilk normality test

data:  as.numeric(CVX_returns[[3]])
W = 0.9743, p-value = 0.0002864
```

The Shapiro-Wilk normality tests the following hypothesis with $H_0 = Normaility$ and $H_1 = Non - Normality$ its test statistic is the correlation between the sample quantile and the theoretical normal quantile.

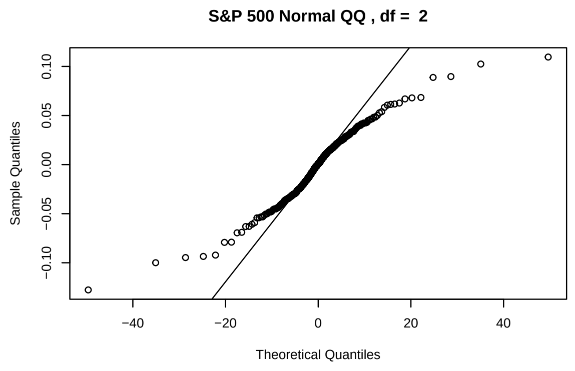
As the period of return gets larger we can see the test statistic for the Shapiro Wilk test get larger, going from 0.87 for daily returns to 0.97 for monthly returns, the fit of the normal distribution gets better as more return data is grouped. Despite, this the hypothesis of normality is rejected for each period, agreeing with our visual inspection of the normal qq-plots.

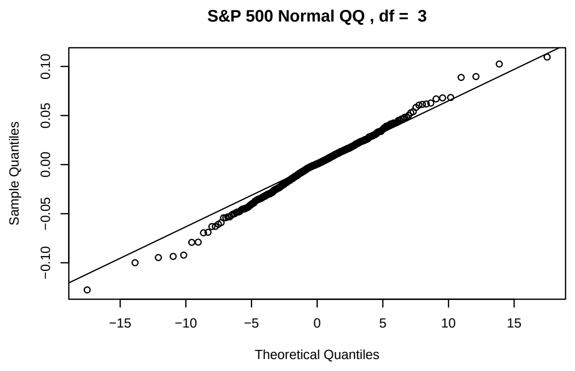
The skewness of the graph is moderately left skewed in the daily and weekly return and is close to unskewed in the monthly return reflecting some asymmetry in returns.

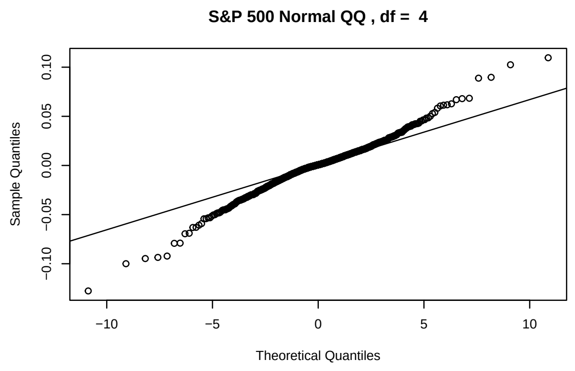
The kurtosis statistic gives us information that is harder to see in the histograms. We see extreme kurtosis in the daily return a measure of 24.69, far greater than the normal distributions kurtosis of 3. This indicates a lot of the weight in the data is in the center and tails. We can also see the kurtosis approach 3 as the period of return gets larger, with monthly return's kurtosis being 4.69. This reflects the principle of aggregational gaussianity in financial return data.

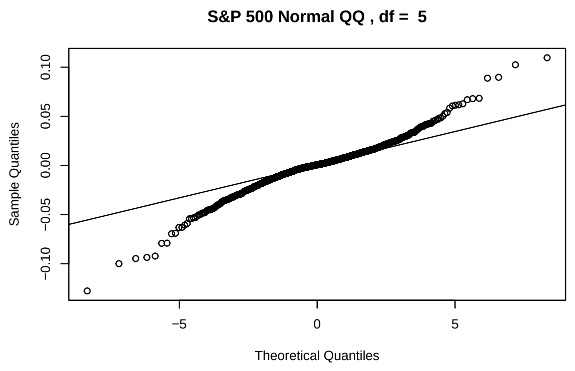
7)

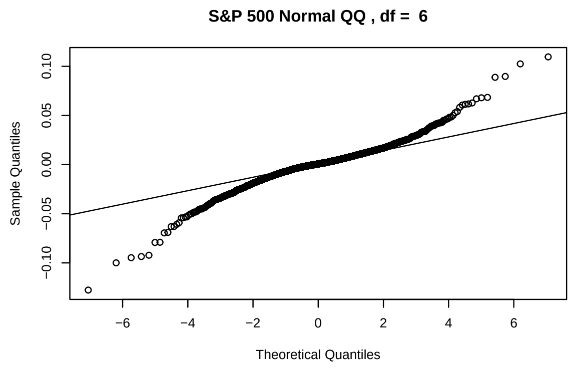
```r
n = length(SnP_500_dlr)
q_grid = (1:n) / (n + 1)
df_grid = 2:7
par(mfrow = c(3,2))
for(i in 1:6){
 qqplot(y = as.numeric(SnP_500_dlr), x = qt(q_grid,df_grid[i]),
 main = paste("S&P 500 Normal QQ", ", df = ", df_grid[i]),
 xlab = "Theoretical Quantiles", ylab = "Sample Quantiles")
  abline(lm(quantile(SnP_500_dlr, c(0.25, 0.75)) ~
             qt(c(0.25, 0.75), df = df_grid[i])))
}
```

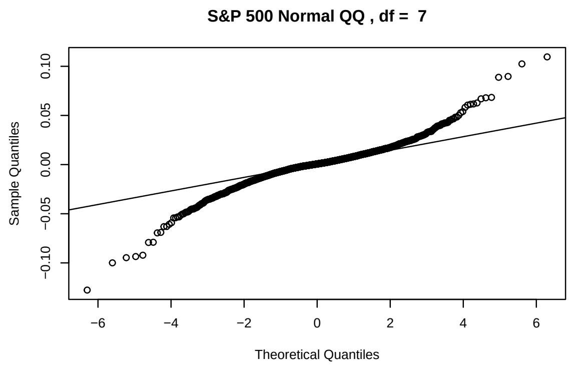












Problem 4)

The code q.grid = (1:n) / (n + 1) creates an evenly spaced out vector of length n that avoids inputs of probability 0 and 1. The limit as p approaches 0 equals negative infinity, similarly the limit as p approaches 1 equals infinity. If we don't make this adjustment and just equally spaced q.grid from 0 to 1 it would both the graph by making our x axis way too large and it would be unreadable.

The code qt(q.grid, df = df[j]) (in my code qt(q_grid,df_grid[i])) computes the quantiles of the t distribution using the probability inputs of q.grid with a different degree of freedom from the vector df_grid with each loop.

Problem 5)

The best fitting distribution is the t distribution with degrees of freedom equal to 3. The data closely follows the fitted line with only slight deviation at the tails. The other choices of degrees of freedom have clear issues in the tails with many fatter and longer than the theoretical distribution.

Problem 6)

```r
library("fGarch")
```

```
NOTE: Packages 'fBasics', 'timeDate', and 'timeSeries' are no longer
attached to the search() path when 'fGarch' is attached.

If needed attach them yourself in your R script by e.g.,
        require("timeSeries")
```

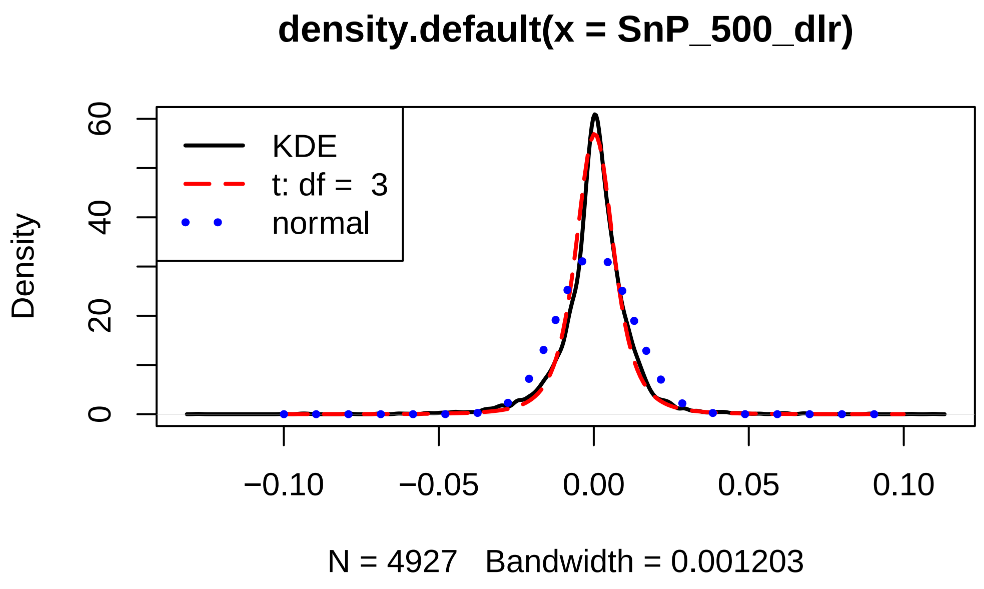
```

Attaching package: 'fGarch'

The following object is masked from 'package:TTR':

    volatility
```

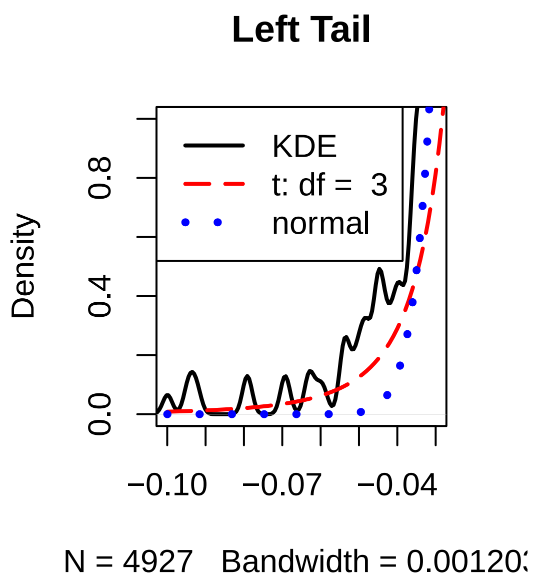
```r
x=seq(-0.1, 0.1,by = 0.001)
par(mfrow = c(1, 1))
df = 3
mad_t = mad(SnP_500_dlr,
            constant = sqrt(df / (df- 2)) / qt(0.75, df))
plot(density(SnP_500_dlr), lwd = 2, ylim = c(0, 60))
lines(x, dstd(x, mean = mean(SnP_500_dlr), sd = mad_t, nu = df),
 lty = 5, lwd = 2, col = "red")
lines(x, dnorm(x, mean = mean(SnP_500_dlr), sd = sd(SnP_500_dlr)),
 lty = 3, lwd = 4, col = "blue")
legend("topleft", c("KDE", paste("t: df = ",df), "normal"),
 lwd = c(2, 2, 4), lty = c(1, 5, 3),
 col = c("black", "red", "blue"))
```

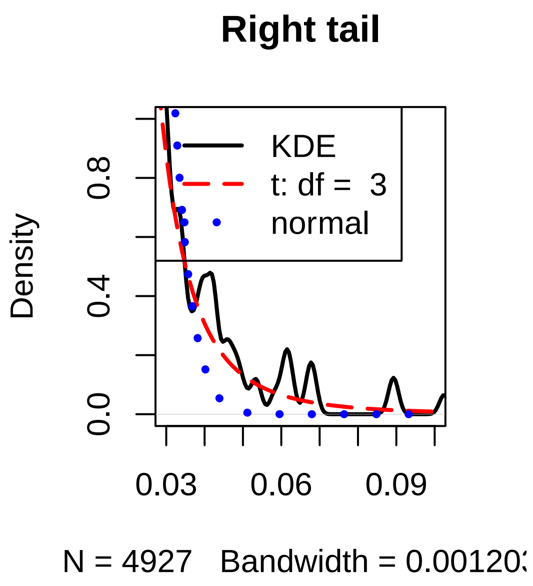


```
par(mfrow = c(1,2))

plot(density(SnP_500_dlr), lwd = 2, ylim = c(0, 1), xlim = c(-.1, -0.03),
     main = "Left Tail")
lines(x, dstd(x, mean = mean(SnP_500_dlr), sd = mad_t, nu = df),
 lty = 5, lwd = 2, col = "red")
lines(x, dnorm(x, mean = mean(SnP_500_dlr), sd = sd(SnP_500_dlr)),
 lty = 3, lwd = 4, col = "blue")
legend("topleft", c("KDE", paste("t: df = ",df), "normal"),
 lwd = c(2, 2, 4), lty = c(1, 5, 3),
 col = c("black", "red", "blue"))

plot(density(SnP_500_dlr), lwd = 2, ylim = c(0, 1), xlim = c(0.03, 0.1),
     main = "Right tail")
lines(x, dstd(x, mean = mean(SnP_500_dlr), sd = mad_t, nu = df),
 lty = 5, lwd = 2, col = "red")
lines(x, dnorm(x, mean = mean(SnP_500_dlr), sd = sd(SnP_500_dlr)),
 lty = 3, lwd = 4, col = "blue")
legend("topleft", c("KDE", paste("t: df = ",df), "normal"),
 lwd = c(2, 2, 4), lty = c(1, 5, 3),
 col = c("black", "red", "blue"))
```





The t distribution with degrees of freedom of 3 does a great job of fitting the return data. Looking at the center of the distribution we can see how closely it follows the peak of the density estimate. In the tails, we are very zoomed in so we can see many peaks and valleys of the KDE but the t distribution does pretty well here too.

The normal distribution on the other hand does a very poor job. The peak is far lower than the KDE and the density dissipates in the tail far earlier than the return data.