# Time Series HW1

35/50

B082040005 高念慈

2023-02-24

## 1.

8

Use the package quantmod of R to download and plot the daily prices
for the following stocks from January 01, 2020 to February 10, 2023.

(1) Apple
(2) Intel
(3) Microsoft

In addition, you must add the main title and axis labels to the time plot. -2

```
# from 正常，to 少一天
getSymbols("AAPL",from="2020-01-01",to='2023-02-11')
```

```
## [1] "AAPL"
```

```
getSymbols("INTC",from="2020-01-01",to='2023-02-11')
```

```
## [1] "INTC"
```

```
getSymbols("MSFT",from="2020-01-01",to='2023-02-11')
```

```
## [1] "MSFT"
```

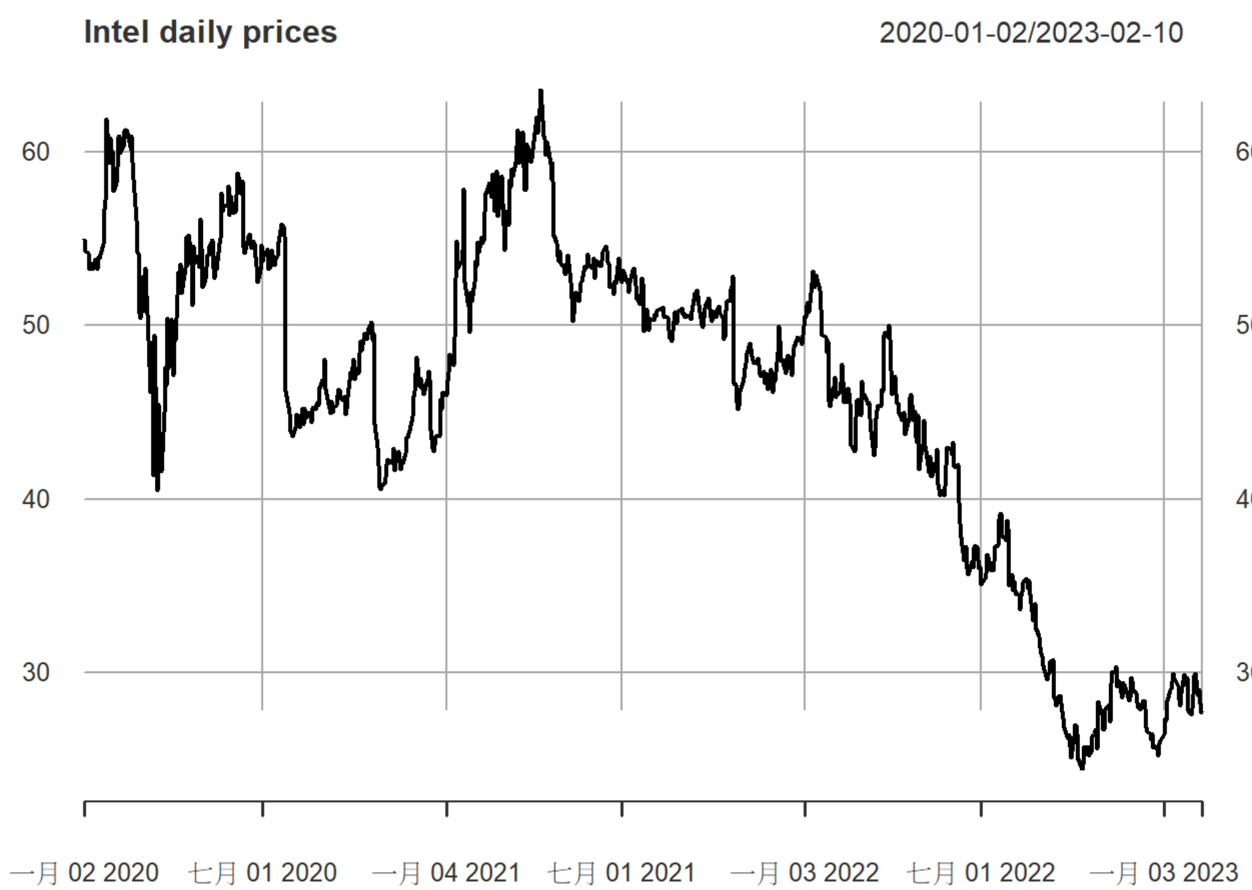
## 取出 adjust price

```
AAPL_adjust = AAPL[,6]
INTC_adjust = INTC[,6]
MSFT_adjust = MSFT[,6]
```

## main title and plot

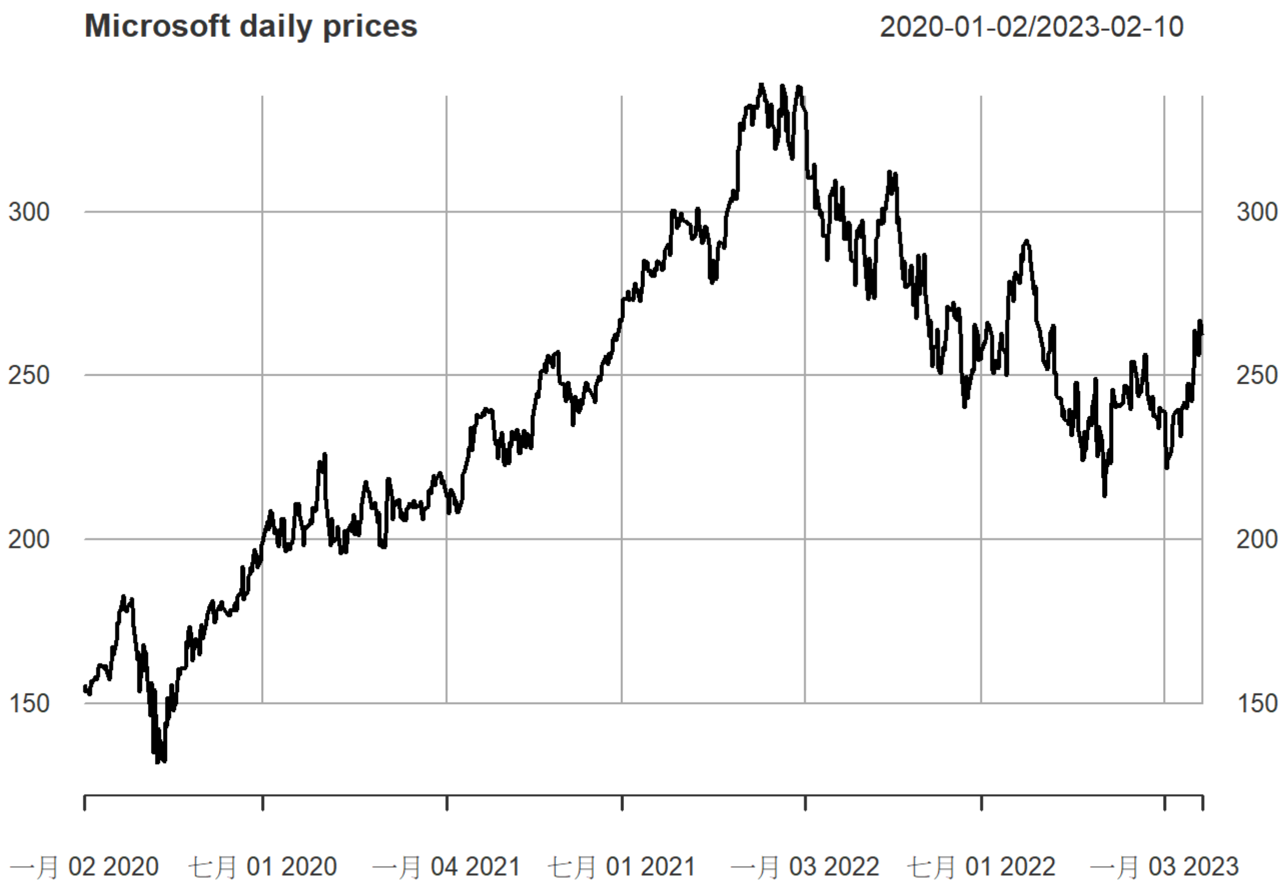
```
plot(AAPL_adjust['2020-01-01/2023-02-10'],main = 'Apple daily prices')
```


```
plot(INTC_adjust['2020-01-01/2023-02-10'],main = 'Intel daily prices')
```



```
plot(MSFT_adjust['2020-01-01/2023-02-10'],main = 'Microsoft daily prices')
```

0




2. 7.

Consider the daily stock returns of American Express (AXP), Caterpillar (CAT), and Starbucks (SBUX) from January 01, 2020 to February 10, 2023. The price data can be obtained by using R package quantmod.

```
getSymbols("AXP",from="2020-01-01",to='2023-02-11')
```

```
## [1] "AXP"
```

```
getSymbols("CAT",from="2020-01-01",to='2023-02-11')
```

```
## [1] "CAT"
```

```
getSymbols("SBUX",from="2020-01-01",to='2023-02-11')
```

```
## [1] "SBUX"
```

a. Compute the simple returns in percentages. Compute the sample mean, standard deviation, skewness, excess kurtosis, minimum, and maximum of the percentage simple returns.

# 取出 adjust price

```
AXP_adjust = AXP[,6]
CAT_adjust = CAT[,6]
SBUX_adjust = SBUX[,6]
```

- $R_t = e^{r_t} - 1$
- simple returns : $R_t$
- log returns : $r_t$

# the simple returns in percentages

## WAY 1

```
# [exp(Log Return)-1]*100

AXP_simple_returns = 100*(exp(diff(log(AXP_adjust)))-1)
head(AXP_simple_returns)
```

```
##            AXP.Adjusted
## 2020-01-02           NA
## 2020-01-03   -0.9932381
## 2020-01-06   -0.4334054
## 2020-01-07   -0.5239364
## 2020-01-08    1.7259576
## 2020-01-09    1.8081831
```

```
CAT_simple_returns = 100*(exp(diff(log(CAT_adjust)))-1)
head(CAT_simple_returns)
```

```
##            CAT.Adjusted
## 2020-01-02           NA
## 2020-01-03  -1.38840896
## 2020-01-06  -0.06740345
## 2020-01-07  -1.32126785
## 2020-01-08   0.88808131
## 2020-01-09  -0.25052996
```

```
SBUX_simple_returns = 100*(exp(diff(log(SBUX_adjust)))-1)
head(SBUX_simple_returns)
```

```
##            SBUX.Adjusted
## 2020-01-02            NA
## 2020-01-03    -0.5819709
## 2020-01-06    -0.7880322
## 2020-01-07    -0.3063619
## 2020-01-08     1.1609414
## 2020-01-09     1.8564328
```

## WAY2

$R_t = \frac{P_t - P_{t-1}}{P_{t-1}}$

第一项会是NA，位置会对不起来

```
n = nrow(AXP_adjust)

AXP_simple_returns2 = 100*(diff(AXP_adjust)/AXP_adjust[1:n-1])
head(AXP_simple_returns2)
```

```
##            AXP.Adjusted
## 2020-01-02           NA
## 2020-01-03    -1.003202
## 2020-01-06    -0.435292
## 2020-01-07    -0.526696
## 2020-01-08     1.696674
## 2020-01-09     1.776069
```

```
CAT_simple_returns2 = 100*(diff(CAT_adjust)/CAT_adjust[1:n-1])
head(CAT_simple_returns2)
```

```
##            CAT.Adjusted
## 2020-01-02           NA
## 2020-01-03  -1.40795716
## 2020-01-06  -0.06744891
## 2020-01-07  -1.33895908
## 2020-01-08   0.88026385
## 2020-01-09  -0.25115919
```

```
SBUX_simple_returns2 = 100*(diff(SBUX_adjust)/SBUX_adjust[1:n-1])
head(SBUX_simple_returns2)
```

```
##            SBUX.Adjusted
## 2020-01-02            NA
## 2020-01-03    -0.5853777
## 2020-01-06    -0.7942915
## 2020-01-07    -0.3073033
## 2020-01-08     1.1476183
## 2020-01-09     1.8225975
```

## sample basicStats of the percentage simple returns.

- 下面用WAY 2 X. -2.
- mean
- standard deviation
- skewness
- excess kurtosis
- minimum
- maximum

simple_return 0

simple_return2 X.

```
# AXP
basicStats(AXP_simple_returns2)
```

```
##              AXP.Adjusted
## nobs           783.000000
## NAs              1.000000
## Minimum        -17.396608
## Maximum         17.953614
## 1. Quartile     -1.287114
## 3. Quartile      1.220390
## Mean             0.011450
## Median           0.061959
## Sum              8.954196
## SE Mean          0.099891
## LCL Mean        -0.184636
## UCL Mean         0.207536
## Variance         7.802915
## Stdev            2.793370
## Skewness         0.109500
## Kurtosis         9.892407
```

```
mean(AXP_simple_returns2,na.rm=T)           # sample mean
```

```
## [1] 0.01145038
```

```
sqrt(var(AXP_simple_returns2,na.rm=T))      # standard deviation
```

```
##               AXP.Adjusted
## AXP.Adjusted       2.79337
```

```
skewness(AXP_simple_returns2,na.rm=T)       # skewness
```

```
## [1] 0.1095005
## attr(,"method")
## [1] "moment"
```

```
kurtosis(AXP_simple_returns2,na.rm=T)       # excess kurtosis
```

```
## [1] 9.892407
## attr(,"method")
## [1] "excess"
```

```
min(AXP_simple_returns2,na.rm=T)            # minimum
```

```
## [1] -17.39661
```

```
max(AXP_simple_returns2,na.rm=T)            # maximum
```

```
## [1] 17.95361
```

```
# CAT
basicStats(CAT_simple_returns2)
```

```
##             CAT.Adjusted
## nobs          783.000000
## NAs             1.000000
## Minimum       -16.661876
## Maximum         9.364524
## 1. Quartile    -1.073313
## 3. Quartile     1.213824
## Mean            0.048126
## Median          0.092790
## Sum            37.634180
## SE Mean         0.080052
## LCL Mean       -0.109017
## UCL Mean        0.205268
## Variance        5.011297
## Stdev           2.238593
## Skewness       -0.702338
## Kurtosis        5.616326
```

```
mean(CAT_simple_returns2,na.rm=T)           # sample mean
```

```
## [1] 0.04812555
```

```
sqrt(var(CAT_simple_returns2,na.rm=T))      # standard deviation
```

```
##              CAT.Adjusted
## CAT.Adjusted     2.238593
```

```
skewness(CAT_simple_returns2,na.rm=T)       # skewness
```

```
## [1] -0.7023379
## attr(,"method")
## [1] "moment"
```

```
kurtosis(CAT_simple_returns2,na.rm=T)       # excess kurtosis
```

```
## [1] 5.616326
## attr(,"method")
## [1] "excess"
```

```
min(CAT_simple_returns2,na.rm=T)            # minimum
```

```
## [1] -16.66188
```

```
max(CAT_simple_returns2,na.rm=T)            # maximum
```

```
## [1] 9.364524
```

```
# SBUX
basicStats(SBUX_simple_returns2)
```

```
##             SBUX.Adjusted
## nobs           783.000000
## NAs              1.000000
## Minimum        -19.337754
## Maximum         12.839093
## 1. Quartile     -0.996005
## 3. Quartile      1.100722
## Mean             0.005188
## Median           0.033780
## Sum              4.056891
## SE Mean          0.080342
## LCL Mean        -0.152523
## UCL Mean         0.162899
## Variance         5.047625
## Stdev            2.246692
## Skewness        -0.571561
## Kurtosis        10.505403
```

```
mean(SBUX_simple_returns2,na.rm=T)           # sample mean
```

```
## [1] 0.00518784
```

```
sqrt(var(SBUX_simple_returns2,na.rm=T))      # standard deviation
```

```
##               SBUX.Adjusted
## SBUX.Adjusted      2.246692
```

```
skewness(SBUX_simple_returns2,na.rm=T)       # skewness
```

```
## [1] -0.5715607
## attr(,"method")
## [1] "moment"
```

```
kurtosis(SBUX_simple_returns2,na.rm=T)       # excess kurtosis
```

```
## [1] 10.5054
## attr(,"method")
## [1] "excess"
```

```
min(SBUX_simple_returns2,na.rm=T)            # minimum
```

```
## [1] -19.33775
```

```
max(SBUX_simple_returns2,na.rm=T)           # maximum
```

```
## [1] 12.83909
```

b. Transform the simple returns to log returns.

- $r_t = ln(\frac{R_t}{100} + 1) \times 100$
- simple returns : $R_t$
- log returns : $r_t$
- **USE WAY 1**

```
AXP_log_returns = (log((AXP_simple_returns/100) + 1))*100
head(AXP_log_returns)
```

```
##            AXP.Adjusted
## 2020-01-02           NA
## 2020-01-03   -0.9982036
## 2020-01-06   -0.4343473
## 2020-01-07   -0.5253138
## 2020-01-08    1.7112322
## 2020-01-09    1.7920299
```

```
CAT_log_returns = (log((CAT_simple_returns/100) + 1))*100
head(CAT_log_returns)
```

```
##            CAT.Adjusted
## 2020-01-02           NA
## 2020-01-03  -1.39813751
## 2020-01-06  -0.06742617
## 2020-01-07  -1.33007425
## 2020-01-08   0.88416106
## 2020-01-09  -0.25084432
```

```
SBUX_log_returns = (log((SBUX_simple_returns/100) + 1))*100
head(SBUX_log_returns)
```

```
##            SBUX.Adjusted
## 2020-01-02            NA
## 2020-01-03    -0.5836710
## 2020-01-06    -0.7911536
## 2020-01-07    -0.3068321
## 2020-01-08     1.1542542
## 2020-01-09     1.8394115
```

- **USE WAY 2**

```
AXP_log_returns2 = (log((AXP_simple_returns2/100) + 1))*100
head(AXP_log_returns2)
```

```
##            AXP.Adjusted
## 2020-01-02           NA
## 2020-01-03   -1.0082683
## 2020-01-06   -0.4362421
## 2020-01-07   -0.5280879
## 2020-01-08    1.6824410
## 2020-01-09    1.7604807
```

```
CAT_log_returns2 = (log((CAT_simple_returns2/100) + 1))*100
head(CAT_log_returns2)
```

```
##            CAT.Adjusted
## 2020-01-02           NA
## 2020-01-03  -1.41796291
## 2020-01-06  -0.06747167
## 2020-01-07  -1.34800397
## 2020-01-08   0.87641211
## 2020-01-09  -0.25147513
```

```
SBUX_log_returns2 = (log((SBUX_simple_returns2/100) + 1))*100
head(SBUX_log_returns2)
```

```
##            SBUX.Adjusted
## 2020-01-02            NA
## 2020-01-03    -0.5870977
## 2020-01-06    -0.7974628
## 2020-01-07    -0.3077765
## 2020-01-08     1.1410831
## 2020-01-09     1.8061873
```

- **Log Return**

```
# Log Return

a = 100*((diff(log(AXP_adjust))))
head(a)
```

```
##            AXP.Adjusted
## 2020-01-02           NA
## 2020-01-03   -0.9982036
## 2020-01-06   -0.4343473
## 2020-01-07   -0.5253138
## 2020-01-08    1.7112322
## 2020-01-09    1.7920299
```

```
b = 100*((diff(log(CAT_adjust))))
head(b)
```

```
##            CAT.Adjusted
## 2020-01-02           NA
## 2020-01-03  -1.39813751
## 2020-01-06  -0.06742617
## 2020-01-07  -1.33007425
## 2020-01-08   0.88416106
## 2020-01-09  -0.25084432
```

```
c = 100*((diff(log(SBUX_adjust))))
head(c)
```

```
##            SBUX.Adjusted
## 2020-01-02            NA
## 2020-01-03    -0.5836710
## 2020-01-06    -0.7911536
## 2020-01-07    -0.3068321
## 2020-01-08     1.1542542
## 2020-01-09     1.8394115
```

c. Express the log returns in percentages. Compute the sample mean, standard deviation, skewness, excess kurtosis, minimum, and maximum of the percentage log returns.

- 下面用 WAY 2 X - 1.
- mean
- standard deviation
- skewness
- excess kurtosis
- minimum
- maximum

```
# AXP
basicStats(AXP_log_returns2)
```

```
##             AXP.Adjusted
## nobs          783.000000
## NAs             1.000000
## Minimum       -19.111945
## Maximum        16.512126
## 1. Quartile    -1.295470
## 3. Quartile     1.213003
## Mean           -0.027622
## Median          0.061940
## Sum           -21.600508
## SE Mean         0.100176
## LCL Mean       -0.224269
## UCL Mean        0.169025
## Variance        7.847626
## Stdev           2.801362
## Skewness       -0.386399
## Kurtosis        9.980155
```

```
mean(AXP_log_returns2,na.rm=T)           # sample mean
```

```
## [1] -0.02762213
```

```
sqrt(var(AXP_log_returns2,na.rm=T))      # standard deviation
```

```
##              AXP.Adjusted
## AXP.Adjusted     2.801362
```

```
skewness(AXP_log_returns2,na.rm=T)       # skewness
```

```
## [1] -0.386399
## attr(,"method")
## [1] "moment"
```

```
kurtosis(AXP_log_returns2,na.rm=T)       # excess kurtosis
```

```
## [1] 9.980155
## attr(,"method")
## [1] "excess"
```

```
min(AXP_log_returns2,na.rm=T)            # minimum
```

```
## [1] -19.11194
```

```
max(AXP_log_returns2,na.rm=T)            # maximum
```

```
## [1] 16.51213
```

```
# CAT
basicStats(CAT_log_returns2)
```

```
##              CAT.Adjusted
## nobs           783.000000
## NAs              1.000000
## Minimum        -18.226407
## Maximum          8.951637
## 1. Quartile     -1.079115
## 3. Quartile      1.206516
## Mean             0.022793
## Median           0.092747
## Sum             17.824114
## SE Mean          0.080807
## LCL Mean        -0.135832
## UCL Mean         0.181418
## Variance         5.106286
## Stdev            2.259709
## Skewness        -0.963726
## Kurtosis         7.113793
```

```
mean(CAT_log_returns2,na.rm=T)           # sample mean
```

```
## [1] 0.02279298
```

```
sqrt(var(CAT_log_returns2,na.rm=T))      # standard deviation
```

```
##              CAT.Adjusted
## CAT.Adjusted     2.259709
```

```
skewness(CAT_log_returns2,na.rm=T)       # skewness
```

```
## [1] -0.963726
## attr(,"method")
## [1] "moment"
```

```
kurtosis(CAT_log_returns2,na.rm=T)       # excess kurtosis
```

```
## [1] 7.113793
## attr(,"method")
## [1] "excess"
```

```
min(CAT_log_returns2,na.rm=T)            # minimum
```

```
## [1] -18.22641
```

```
max(CAT_log_returns2,na.rm=T)            # maximum
```

```
## [1] 8.951637
```

```
# SBUX
basicStats(SBUX_log_returns2)
```

```
##             SBUX.Adjusted
## nobs           783.000000
## NAs              1.000000
## Minimum        -21.489956
## Maximum         12.079266
## 1. Quartile     -1.000998
## 3. Quartile      1.094709
## Mean            -0.020325
## Median           0.033775
## Sum            -15.894179
## SE Mean          0.081121
## LCL Mean        -0.179567
## UCL Mean         0.138917
## Variance         5.146105
## Stdev            2.268503
## Skewness        -1.023354
## Kurtosis        13.253596
```

```
mean(SBUX_log_returns2,na.rm=T)           # sample mean
```

```
## [1] -0.02032504
```

```
sqrt(var(SBUX_log_returns2,na.rm=T))      # standard deviation
```

```
##               SBUX.Adjusted
## SBUX.Adjusted      2.268503
```

```
skewness(SBUX_log_returns2,na.rm=T)       # skewness
```

```
## [1] -1.023354
## attr(,"method")
## [1] "moment"
```

```
kurtosis(SBUX_log_returns2,na.rm=T)       # excess kurtosis
```

```
## [1] 13.2536
## attr(,"method")
## [1] "excess"
```

```
min(SBUX_log_returns2,na.rm=T)            # minimum
```

```
## [1] -21.48996
```

```
max(SBUX_log_returns2,na.rm=T)            # maximum
```

```
## [1] 12.07927
```

# 3. 10

Consider the monthly stock returns for General Motors (GM), CRSP value-weighted index (VW), CRSP equalweighted index (EW), and S&P composite index from January 1975 to December 2008. The returns of the indexes include dividend distributions.
Data file is m-gm3dx7508.txt (date, gm, vw, ew, sp).

```
da = read.table("https://faculty.chicagobooth.edu/-/media/faculty/ruey-s-tsay/teaching/fts3/m
-gm3dx7508.txt",header = T)
head(da)
```

```
##       date       gm       vw       ew       sp
## 1 19750131 0.252033 0.141600 0.299260 0.122812
## 2 19750228 0.028571 0.058411 0.053918 0.059886
## 3 19750331 0.054487 0.030191 0.081497 0.021694
## 4 19750430 0.045593 0.046497 0.031093 0.047265
## 5 19750530 0.037209 0.055140 0.072876 0.044101
## 6 19750630 0.107955 0.051473 0.071792 0.044323
```

a. Compute the simple returns in percentages. Compute the sample mean, standard deviation, skewness, excess kurtosis, minimum, and maximum of the percentage simple returns.

```
da = cbind(da[,"date"],(da[,c("gm","vw","ew","sp")])*100)
names(da)[1] = "data"
head(da)
```

```
##       data      gm      vw      ew      sp
## 1 19750131 25.2033 14.1600 29.9260 12.2812
## 2 19750228  2.8571  5.8411  5.3918  5.9886
## 3 19750331  5.4487  3.0191  8.1497  2.1694
## 4 19750430  4.5593  4.6497  3.1093  4.7265
## 5 19750530  3.7209  5.5140  7.2876  4.4101
## 6 19750630 10.7955  5.1473  7.1792  4.4323
```

- mean
- standard deviation
- skewness
- excess kurtosis
- minimum
- maximum

```
basicStats(da[,c("gm","vw","ew","sp")])
```

```
##                     gm         vw         ew         sp
## nobs          408.000000 408.000000 408.000000 408.000000
## NAs             0.000000   0.000000   0.000000   0.000000
## Minimum       -38.931300 -22.536300 -27.224800 -21.763000
## Maximum        27.661900  14.160000  29.926000  13.176700
## 1. Quartile    -4.348825  -1.583500  -1.684150  -1.762400
## 3. Quartile     5.450150   3.995300   4.564425   3.598425
## Mean            0.556755   1.011799   1.331385   0.730084
## Median          0.678100   1.387950   1.617200   1.003550
## Sum           227.156000 412.813800 543.204900 297.874400
## SE Mean         0.459067   0.223153   0.277038   0.215849
## LCL Mean       -0.345684   0.573122   0.786780   0.305767
## UCL Mean        1.459194   1.450475   1.875989   1.154402
## Variance       85.983038  20.317313  31.314081  19.008981
## Stdev           9.272704   4.507473   5.595899   4.359929
## Skewness       -0.383475  -0.742662  -0.300123  -0.570545
## Kurtosis        2.048076   2.666032   4.333664   2.268600
```

```
colMeans(da[,c("gm","vw","ew","sp")])           # sample mean
```

```
##        gm        vw        ew        sp
## 0.5567549 1.0117985 1.3313846 0.7300843
```

```
apply(da[,-1],2,mean)                           # sample mean
```

```
##        gm        vw        ew        sp
## 0.5567549 1.0117985 1.3313846 0.7300843
```

```
sqrt(apply(da[,-1],2,var))                      # standard deviation
```

```
##       gm       vw       ew       sp
## 9.272704 4.507473 5.595899 4.359929
```

```
apply(da[,-1],2,skewness)                       # skewness
```

```
##         gm         vw         ew         sp
## -0.3834748 -0.7426620 -0.3001231 -0.5705447
```

```
apply(da[,-1],2,kurtosis)                       # excess kurtosis
```

```
##       gm       vw       ew       sp
## 2.048076 2.666032 4.333664 2.268600
```

```
apply(da[,-1],2,min)                            # minimum
```

```
##       gm       vw       ew       sp
## -38.9313 -22.5363 -27.2248 -21.7630
```

```
apply(da[,-1],2,max)                                    # maximum
```

```
##      gm      vw      ew      sp
## 27.6619 14.1600 29.9260 13.1767
```

b. Transform the simple returns to log returns.

```
a = da[,c("gm","vw","ew","sp")]
log_returns = (log(a/100 + 1))*100
head(log_returns)
```

X .

```
##          gm        vw        ew        sp
## 1 22.476863 13.243079 26.179487 11.583625
## 2  2.817046  5.676873  5.251465  5.816136
## 3  5.305439  2.974422  7.834619  2.146203
## 4  4.458419  4.544840  3.061940  4.618200
## 5  3.653345  5.367346  7.034289  4.315623
## 6 10.251597  5.019204  6.933201  4.336883
```

c. Express the log returns in percentages. Compute the sample mean, standard deviation, skewness, excess kurtosis, minimum, and maximum of the percentage log returns.

- mean
- standard deviation
- skewness
- excess kurtosis
- minimum
- maximum

```
basicStats(log_returns)
```

```
##                     gm         vw         ew         sp
## nobs        408.000000 408.000000 408.000000 408.000000
## NAs           0.000000   0.000000   0.000000   0.000000
## Minimum     -49.317073 -25.536075 -31.779495 -24.542750
## Maximum      24.421518  13.243079  26.179487  12.378013
## 1. Quartile  -4.446221  -1.596172  -1.698497  -1.778115
## 3. Quartile   5.306814   3.917552   4.463320   3.535194
## Mean          0.110182   0.904567   1.166997   0.631937
## Median        0.675811   1.378403   1.604263   0.998548
## Sum          44.954256 369.063338 476.134773 257.830396
## SE Mean       0.474789   0.225783   0.278528   0.217942
## LCL Mean     -0.823164   0.460721   0.619465   0.203505
## UCL Mean      1.043528   1.348413   1.714529   1.060369
## Variance     91.973391  20.798939  31.651673  19.379422
## Stdev         9.590276   4.560585   5.625982   4.402206
## Skewness     -1.023664  -1.051001  -0.836133  -0.854843
## Kurtosis      4.020752   3.937548   5.242452   3.334693
```

```
colMeans(log_returns[,c("gm","vw","ew","sp")],na.rm=TRUE)       # sample mean
```

```
##        gm        vw        ew        sp
## 0.1101820 0.9045670 1.1669970 0.6319372
```

```
apply(log_returns,2,mean,na.rm=TRUE)                              # sample mean
```

```
##        gm        vw        ew        sp
## 0.1101820 0.9045670 1.1669970 0.6319372
```

```
sqrt(apply(log_returns,2,var,na.rm=TRUE))                         # standard deviation
```

```
##       gm       vw       ew       sp
## 9.590276 4.560585 5.625982 4.402206
```

```
apply(log_returns,2,skewness,na.rm=TRUE)                          # skewness
```

```
##         gm         vw         ew         sp
## -1.0236641 -1.0510007 -0.8361335 -0.8548433
```

```
apply(log_returns,2,kurtosis,na.rm=TRUE)                          # excess kurtosis
```

```
##       gm       vw       ew       sp
## 4.020752 3.937548 5.242452 3.334693
```

```
apply(log_returns,2,min,na.rm=TRUE)                               # minimum
```

```
##        gm        vw        ew        sp
## -49.31707 -25.53607 -31.77949 -24.54275
```

```
apply(log_returns,2,max,na.rm=TRUE)                               # maximum
```

```
##       gm       vw       ew       sp
## 24.42152 13.24308 26.17949 12.37801
```

# 4. 2

Consider the monthly stock returns of S&P composite index from
January 1975 to December 2008 in Exercise 3. Answer the following questions:

```
newda = da[,c("data",'sp')]
head(newda)
```

```
##        data      sp
## 1 19750131 12.2812
## 2 19750228  5.9886
## 3 19750331  2.1694
## 4 19750430  4.7265
## 5 19750530  4.4101
## 6 19750630  4.4323
```

a. What is the average annual log return over the data span? -5.

$R \approx \frac{1}{k} \sum_{j=0}^{k-1} R_{t-j}$ X 準確值. simple return → log return

```
k = 2008-1974
average_annual_return = sum(newda[,"sp"])/k
average_annual_log_return = (log(average_annual_return/100 + 1))
average_annual_log_return
```

→ calculate average annual log return.

```
## [1] 0.08398274
```

b. Assume that there were no transaction costs. If one invested $1.00 on the S&P composite index at the beginning of 1975, what was the value of the investment at the end of 2008? -3

```
1*(1+average_annual_return/100)**k
```

```
## [1] 17.38162
```

8

# 5.

Daily foreign exchange rates (spot rates) can be obtained from the Federal Reserve Bank in Chicago. The data are the noon buying rates in New York City certified by the Federal Reserve Bank of New York.

Consider the exchange rates between the U.S. dollar and the Canadian dollar, euro, U.K. pound, and the Japanese yen from January 4, 2000, to March 27, 2009. The data are also on the Web. (d-caus.txt, d-usuk.txt, d-jpus.txt and d-useu.txt)

```
caus = read.table("https://faculty.chicagobooth.edu/-/media/faculty/ruey-s-tsay/teaching/fts
3/d-caus.txt",header = T)
head(caus)
```

```
##   year mon day   rate
## 1 2000   1   4 1.4518
## 2 2000   1   5 1.4518
## 3 2000   1   6 1.4571
## 4 2000   1   7 1.4505
## 5 2000   1  10 1.4568
## 6 2000   1  11 1.4570
```

```
useu = read.table("https://faculty.chicagobooth.edu/-/media/faculty/ruey-s-tsay/teaching/fts
3/d-useu.txt",header = T)
head(useu)
```

```
##   Date Mon Day  Value
## 1 2000   1   4 1.0309
## 2 2000   1   5 1.0335
## 3 2000   1   6 1.0324
## 4 2000   1   7 1.0294
## 5 2000   1  10 1.0252
## 6 2000   1  11 1.0322
```

```
usuk = read.table("https://faculty.chicagobooth.edu/-/media/faculty/ruey-s-tsay/teaching/fts
3/d-usuk.txt",header = T)
head(usuk)
```

```
##   date mon day  value
## 1 2000   1   4 1.6370
## 2 2000   1   5 1.6415
## 3 2000   1   6 1.6475
## 4 2000   1   7 1.6384
## 5 2000   1  10 1.6374
## 6 2000   1  11 1.6480
```

```
jpus = read.table("https://faculty.chicagobooth.edu/-/media/faculty/ruey-s-tsay/teaching/fts
3/d-jpus.txt",header = T)
head(jpus)
```

```
##   year mon day  value
## 1 2000   1   4 103.09
## 2 2000   1   5 103.77
## 3 2000   1   6 105.19
## 4 2000   1   7 105.17
## 5 2000   1  10 105.28
## 6 2000   1  11 106.09
```

a. Compute the daily log return of each exchange rate.

- **Log Return**

```
# Log Return

caus_log_return = data.frame(((diff(log(caus[,"rate"])))))
names(caus_log_return) = "caus_log_return"
head(caus_log_return)
```

```
##   caus_log_return
## 1    0.0000000000
## 2    0.0036439932
## 3   -0.0045398345
## 4    0.0043339249
## 5    0.0001372778
## 6   -0.0013736266
```

```
useu_log_return = data.frame(((diff(log(useu[,"Value"])))))
names(useu_log_return) = "useu_log_return"
head(useu_log_return)
```

```
##   useu_log_return
## 1     0.002518893
## 2    -0.001064911
## 3    -0.002910081
## 4    -0.004088393
## 5     0.006804731
## 6    -0.003980008
```

```
usuk_log_return = data.frame(((diff(log(usuk[,"value"])))))
names(usuk_log_return) = "usuk_log_return"
head(usuk_log_return)
```

```
##   usuk_log_return
## 1    0.0027451596
## 2    0.0036485294
## 3   -0.0055388315
## 4   -0.0006105379
## 5    0.0064528135
## 6   -0.0009106087
```

```
jpus_log_return = data.frame(((diff(log(usuk[,"value"])))))
names(jpus_log_return) = "jpus_log_return"
head(jpus_log_return)
```

-2

```
##   jpus_log_return
## 1    0.0027451596
## 2    0.0036485294
## 3   -0.0055388315
## 4   -0.0006105379
## 5    0.0064528135
## 6   -0.0009106087
```

b. Compute the sample mean, standard deviation, skewness, excess kurtosis, minimum, and maximum of the log returns of each exchange rate

```
finaldata = cbind(caus_log_return,useu_log_return,usuk_log_return,jpus_log_return)
head(finaldata)
```

```
##   caus_log_return useu_log_return usuk_log_return jpus_log_return
## 1    0.0000000000     0.002518893    0.0027451596    0.0027451596
## 2    0.0036439932    -0.001064911    0.0036485294    0.0036485294
## 3   -0.0045398345    -0.002910081   -0.0055388315   -0.0055388315
## 4    0.0043339249    -0.004088393   -0.0006105379   -0.0006105379
## 5    0.0001372778     0.006804731    0.0064528135    0.0064528135
## 6   -0.0013736266    -0.003980008   -0.0009106087   -0.0009106087
```

```
basicStats(finaldata)
```

```
##             caus_log_return useu_log_return usuk_log_return jpus_log_return
## nobs            2322.000000     2322.000000     2322.000000     2322.000000
## NAs                0.000000        0.000000        0.000000        0.000000
## Minimum           -0.050716       -0.030031       -0.049662       -0.049662
## Maximum            0.038070        0.046208        0.044349        0.044349
## 1. Quartile       -0.003097       -0.003397       -0.003158       -0.003158
## 3. Quartile        0.002816        0.003795        0.003300        0.003300
## Mean              -0.000068        0.000110       -0.000057       -0.000057
## Median            -0.000070        0.000075        0.000105        0.000105
## Sum               -0.158338        0.255198       -0.133444       -0.133444
## SE Mean            0.000122        0.000136        0.000128        0.000128
## LCL Mean          -0.000307       -0.000156       -0.000309       -0.000309
## UCL Mean           0.000171        0.000376        0.000194        0.000194
## Variance           0.000035        0.000043        0.000038        0.000038
## Stdev              0.005876        0.006539        0.006177        0.006177
## Skewness          -0.238885        0.125932       -0.394828       -0.394828
## Kurtosis           8.245334        2.804500        7.061154        7.061154
```

```
apply(finaldata,2,mean,na.rm=TRUE)                          # sample mean
```

```
## caus_log_return useu_log_return usuk_log_return jpus_log_return
##   -6.819042e-05    1.099043e-04   -5.746948e-05   -5.746948e-05
```

```
sqrt(apply(finaldata,2,var,na.rm=TRUE))                     # standard deviation
```

```
## caus_log_return useu_log_return usuk_log_return jpus_log_return
##     0.005876476     0.006539431     0.006176620     0.006176620
```

```
apply(finaldata,2,skewness,na.rm=TRUE)                      # skewness
```

```
## caus_log_return useu_log_return usuk_log_return jpus_log_return
##      -0.2388845       0.1259317      -0.3948278      -0.3948278
```

```
apply(finaldata,2,kurtosis,na.rm=TRUE)                      # excess kurtosis
```

```
## caus_log_return useu_log_return usuk_log_return jpus_log_return
##        8.245334        2.804500        7.061154        7.061154
```

```r
apply(finaldata,2,min,na.rm=TRUE)                                # minimum
```

```
## caus_log_return useu_log_return usuk_log_return jpus_log_return
##     -0.05071599     -0.03003101     -0.04966250     -0.04966250
```

```r
apply(finaldata,2,max,na.rm=TRUE)                                # maximum
```

```
## caus_log_return useu_log_return usuk_log_return jpus_log_return
##      0.03806962      0.04620792      0.04434858      0.04434858
```