BUS 41000 R Usage Showcase

Richard Chen

March 29, 2016

Note

We will use the free statistics package R, which can be downloaded from http://www.r-project.org. For working with R, I strongly recommend the free software RStudio, which is available for Windows, Mac, and Linux, from http://www.rstudio.com. This document is prepared by R Markdown. This is a convenient tool for creating reports that have embeded R output.
This tool may be useful for preparing your final project. In order to create this file yourself, download hw01_starter.Rmd and pres Knit PDF. RStudio may ask you to install additional packages. To start learning about R Markdown, go here: http://rmarkdown.rstudio.com/.

In the following, R codes are followed by R output. For any R function which you feel confused about, please utilize the help function in R for detailed instructions. For example, function `cor()` help use to compute correlation (matrices), if you would like to know the details of its usage, input the R command `help(cor)`.

R Code Example for Question 1

The data in this example is from the file "CountryMonthlyReturns2.csv". We care about the monthly returns from Japan and Canada. First of all, we should read the data into R.

However, before we input the required data, we should know and specify the working directory first, all the line but the last is R comment, uncomment the last line (deleting the # sign), and put your preferred location as the working directory. You can also check the directory by `getwd()`.

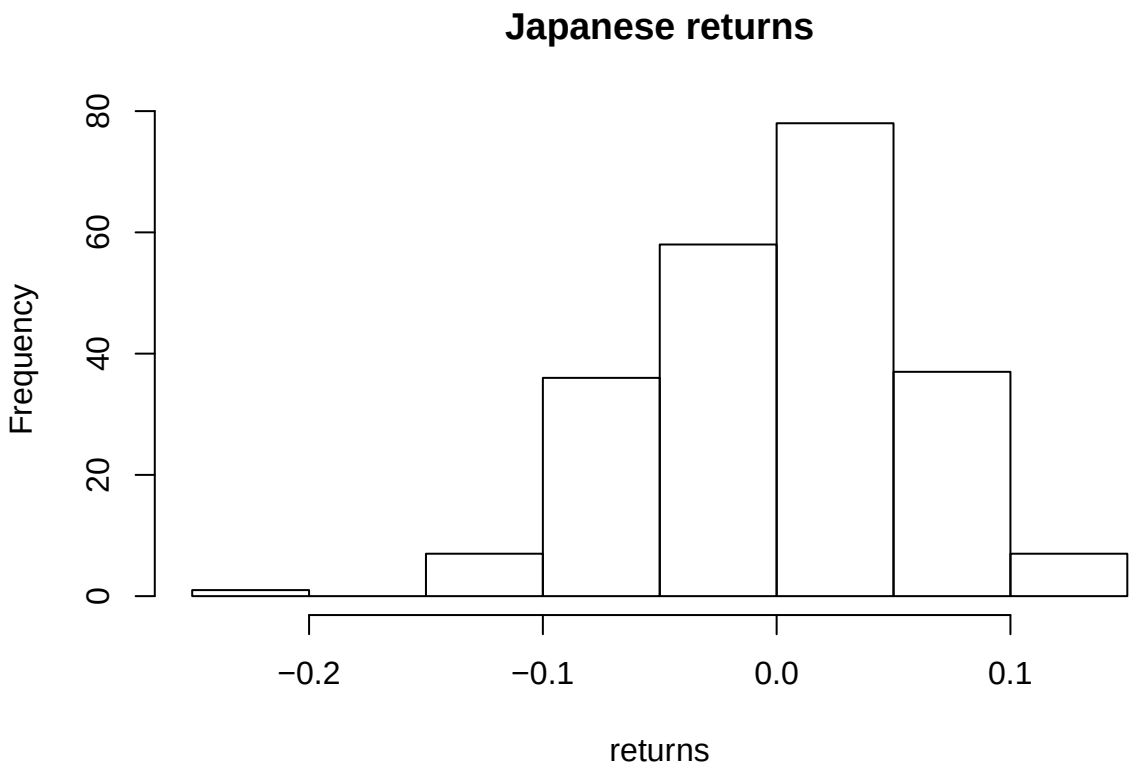
```
################################
#   Country Monthly Returns -- 2
################################

# Monthly return data on country's broadband portfolios from Jan 1996 till Aug 2014.

# set working directory
# you should change the folder to a location on your own computer
# setwd("input your prefered location here")
```

Now we are ready to download and read the data

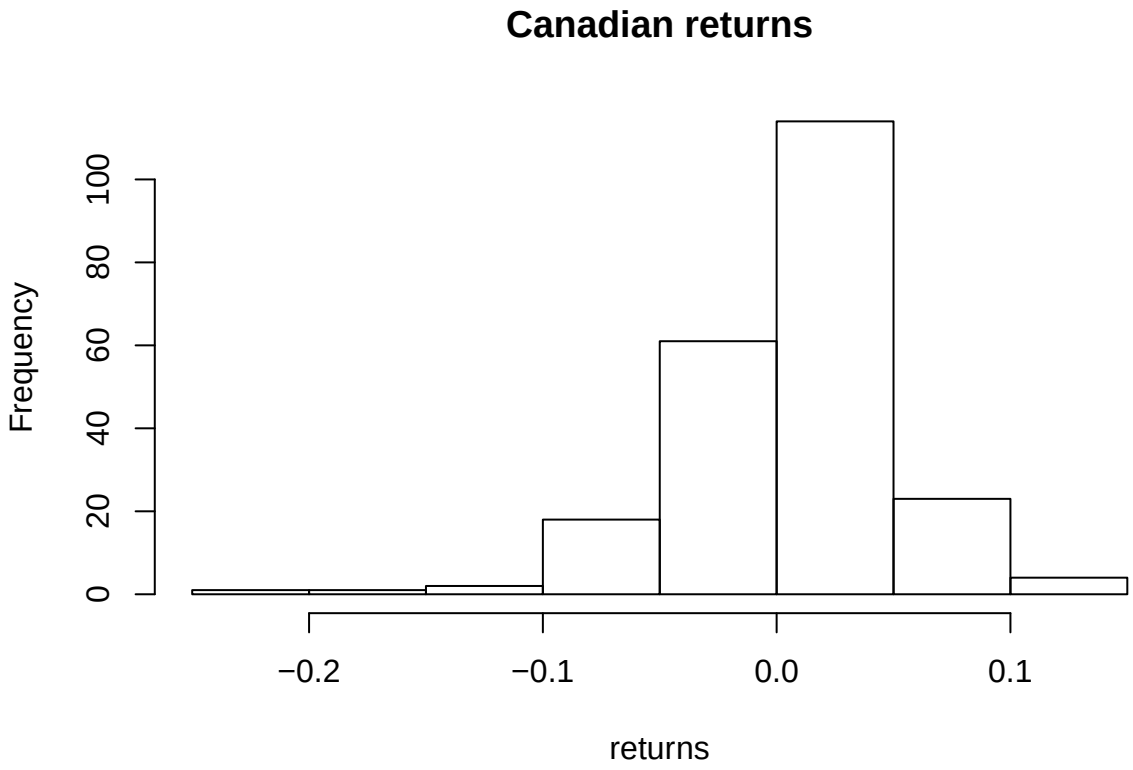
```
# download data to the current working folder
download.file("https://github.com/mlakolar/BUS41000/raw/master/data/CountryMonthlyReturns2.csv",
              destfile="CountryMonthlyReturns2.csv")

countryReturn_df = read.csv("CountryMonthlyReturns2.csv")
```

As an initial check, let's see the histogram of some variables:

```r
# plot histrograms
hist(countryReturn_df$Japan,
     breaks = 10, # this parameter controls number of bins
     xlab = "returns", ylab = "Frequency", main="Japanese returns")
```



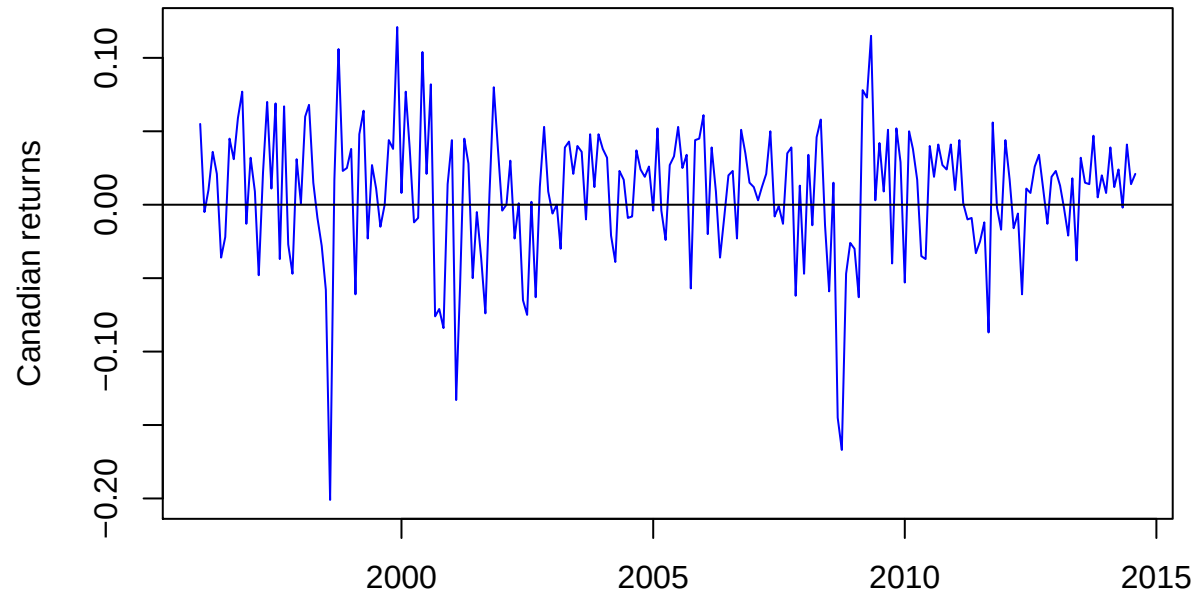

```r
hist(countryReturn_df$Canada,
     breaks = 10,
     xlab = "returns", ylab = "Frequency", main="Canadian returns")
```



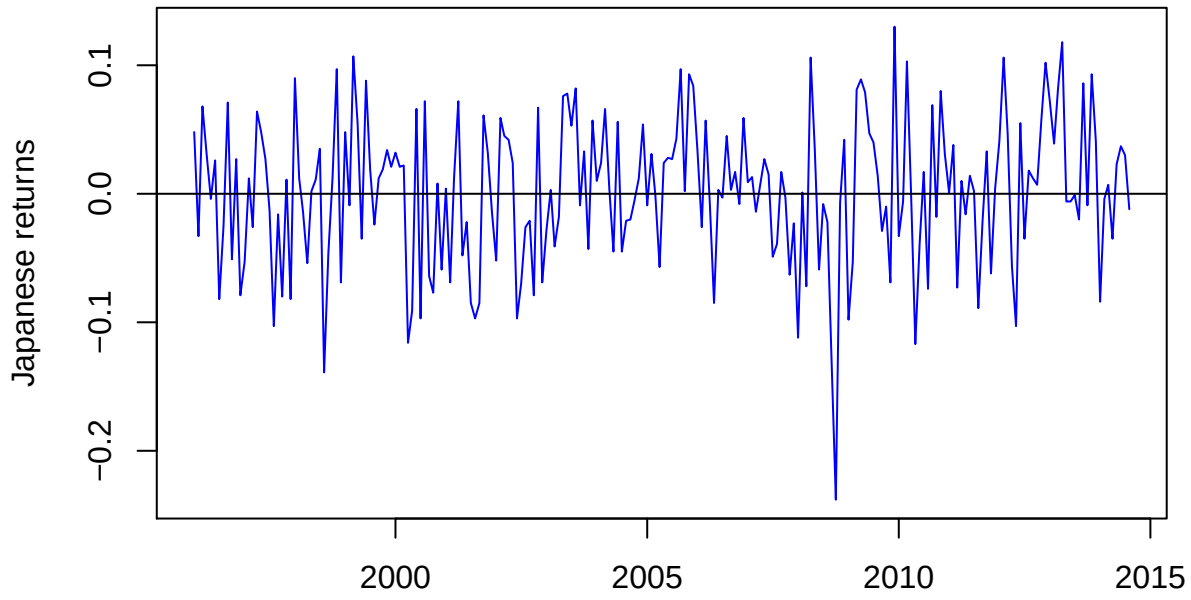

```
# to save image, click on export and "Save as Image"
```

We can also see some temporal trend via plotting the time series:

```
plot(x=as.Date(paste0(countryReturn_df$Date, "-01"), format = "%m/%d/%Y"),
     y=countryReturn_df$Canada, type="l", col="blue",
     xlab="", ylab="Canadian returns", main = "")
abline(h=0)
```



```
plot(x=as.Date(paste0(countryReturn_df$Date, "-01"), format = "%m/%d/%Y"),
     y=countryReturn_df$Japan, type="l", col="blue",
     xlab="", ylab="Japanese returns", main = "")
abline(h=0)
```



We can also compute the mean and standard deviations, as an numberically measure of data fecture“:

```
mean(countryReturn_df$Canada); sd(countryReturn_df$Canada)
```

```
## [1] 0.00825
```

```
## [1] 0.04469578
```

```
mean(countryReturn_df$Japan); sd(countryReturn_df$Japan)
```

```
## [1] 0.00153125
```

```
## [1] 0.05726141
```

According to the empirical rule, 95% of the data approximately locate in the interval $[\mu - 2s, \mu + 2s]$, we can approximate the intervals which contain about 95% of the data of the variables:

```
c(mean(countryReturn_df$Japan)-2*sd(countryReturn_df$Japan),mean(countryReturn_df$Japan)+2*sd(countryRe
```

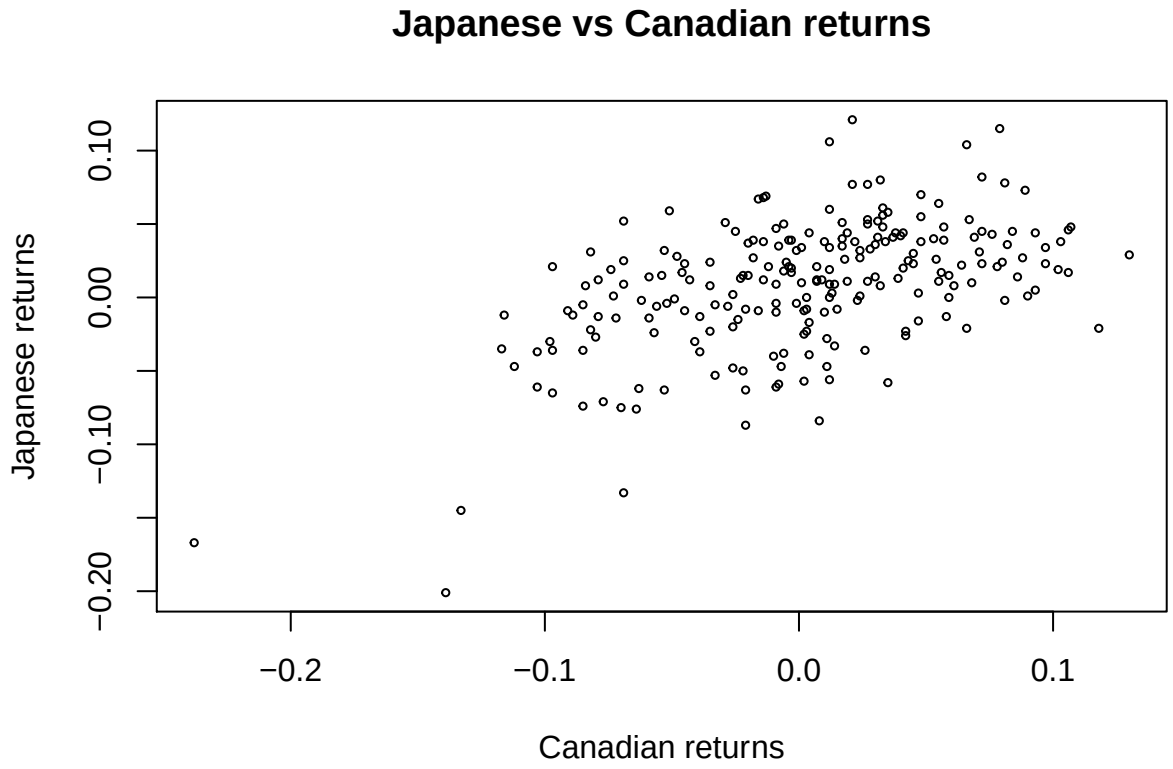
```
## [1] -0.1129916  0.1160541
```

```
c(mean(countryReturn_df$Canada)-2*sd(countryReturn_df$Canada),mean(countryReturn_df$Canada)+2*sd(countr
```

```
## [1] -0.08114157  0.09764157
```

Scatter plot of the two variables:

```r
plot(x=countryReturn_df$Japan, y=countryReturn_df$Canada,
     type="p", col="black",
     cex = 0.5,                                    # controls size of points
     xlab="Canadian returns", ylab="Japanese returns", main = "Japanese vs Canadian returns")
```



The covariance and correlation between the two series can be computed by:

```r
cov(countryReturn_df$Japan,countryReturn_df$Canada)
```

```
## [1] 0.001367988
```

```r
cor(countryReturn_df$Japan,countryReturn_df$Canada)
```

```
## [1] 0.5345073
```

We can use the build-in function "cor()" to compute the correlation matrix of all the variables in the data set:

```r
cor(countryReturn_df[,-1]) # the first column is the date, irrelavent for the corrrelation matrix
```

```
##             Australia   Austria   Belgium    Canada   Denmark   Finland
## Australia   1.0000000 0.6704776 0.6438713 0.6797236 0.6180243 0.5608437
## Austria     0.6704776 1.0000000 0.7411166 0.6650339 0.6790396 0.4605711
## Belgium     0.6438713 0.7411166 1.0000000 0.6009331 0.7189554 0.5421499
## Canada      0.6797236 0.6650339 0.6009331 1.0000000 0.6736044 0.6212961
## Denmark     0.6180243 0.6790396 0.7189554 0.6736044 1.0000000 0.5862118
## Finland     0.5608437 0.4605711 0.5421499 0.6212961 0.5862118 1.0000000
```

```
## France      0.6573431 0.6538039 0.8075733 0.6837532 0.7230550 0.7384811
## Germany     0.6327517 0.6318698 0.7489770 0.6621978 0.7249538 0.6877437
## HongKong    0.6258725 0.5811087 0.4985763 0.6786325 0.5195418 0.4833354
## Ireland     0.6642234 0.6369331 0.6981749 0.5910711 0.7184052 0.5702124
## Japan       0.5849133 0.5280348 0.4379802 0.5345073 0.4613467 0.4643288
## Netherlands 0.5311288 0.6152085 0.7265127 0.5475539 0.6443467 0.5307925
## NewZealand  0.6328014 0.5669714 0.4907362 0.5012801 0.5257935 0.4277547
## Spain       0.6129359 0.5857867 0.6784857 0.5981944 0.5995776 0.5842721
## Sweden      0.5979439 0.5446140 0.6403899 0.6295725 0.6996791 0.7488425
## Switzerland 0.5861421 0.5882713 0.7404581 0.5646645 0.6545910 0.5606724
## UK          0.7303060 0.6957456 0.7610382 0.7014349 0.6973744 0.6533517
## USA         0.7081417 0.6313730 0.7281446 0.7877786 0.6866123 0.6880405
## Brazil      0.5639818 0.4670540 0.4221480 0.5804489 0.4710060 0.5119054
## Chile       0.4743731 0.4491174 0.4206435 0.5285505 0.4665362 0.3986968
## Mexico      0.6006315 0.5396844 0.4777814 0.6875743 0.4883464 0.4832811
## Peru        0.4422776 0.4953483 0.3395889 0.4789480 0.3505954 0.2369585
## South.Korea 0.5202035 0.4177055 0.3907107 0.4681153 0.4288943 0.4297035
##                France   Germany  HongKong   Ireland     Japan Netherlands
## Australia   0.6573431 0.6327517 0.6258725 0.6642234 0.5849133   0.5311288
## Austria     0.6538039 0.6318698 0.5811087 0.6369331 0.5280348   0.6152085
## Belgium     0.8075733 0.7489770 0.4985763 0.6981749 0.4379802   0.7265127
## Canada      0.6837532 0.6621978 0.6786325 0.5910711 0.5345073   0.5475539
## Denmark     0.7230550 0.7249538 0.5195418 0.7184052 0.4613467   0.6443467
## Finland     0.7384811 0.6877437 0.4833354 0.5702124 0.4643288   0.5307925
## France      1.0000000 0.9057753 0.5739701 0.6800795 0.5444738   0.7077971
## Germany     0.9057753 1.0000000 0.5848281 0.6602274 0.5232040   0.7190000
## HongKong    0.5739701 0.5848281 1.0000000 0.4332814 0.5018352   0.5200426
## Ireland     0.6800795 0.6602274 0.4332814 1.0000000 0.4880897   0.5899933
## Japan       0.5444738 0.5232040 0.5018352 0.4880897 1.0000000   0.4173539
## Netherlands 0.7077971 0.7190000 0.5200426 0.5899933 0.4173539   1.0000000
## NewZealand  0.4968183 0.4824245 0.4739738 0.4747102 0.4375893   0.3858850
## Spain       0.8067275 0.7370606 0.5792854 0.5998081 0.4866818   0.6254792
## Sweden      0.8176037 0.8304672 0.5973980 0.6065644 0.5358427   0.6712783
## Switzerland 0.7989745 0.7446520 0.4805557 0.6410246 0.4958760   0.6662079
## UK          0.8190962 0.7867908 0.6322654 0.7103416 0.5259886   0.7012243
## USA         0.8011831 0.7913985 0.6591336 0.7113608 0.5839363   0.6227216
## Brazil      0.5472689 0.5078994 0.5939429 0.4344350 0.4260748   0.3594638
## Chile       0.4565501 0.4381567 0.5379214 0.4207402 0.3688103   0.3508425
## Mexico      0.5440546 0.5808514 0.6339704 0.4881024 0.4368920   0.4609820
## Peru        0.2914877 0.3172167 0.3834366 0.3241497 0.3590441   0.3035452
## South.Korea 0.4433931 0.4382308 0.4993137 0.3937804 0.5149592   0.3862860
##             NewZealand     Spain    Sweden Switzerland        UK       USA
## Australia    0.6328014 0.6129359 0.5979439   0.5861421 0.7303060 0.7081417
## Austria      0.5669714 0.5857867 0.5446140   0.5882713 0.6957456 0.6313730
## Belgium      0.4907362 0.6784857 0.6403899   0.7404581 0.7610382 0.7281446
## Canada       0.5012801 0.5981944 0.6295725   0.5646645 0.7014349 0.7877786
## Denmark      0.5257935 0.5995776 0.6996791   0.6545910 0.6973744 0.6866123
## Finland      0.4277547 0.5842721 0.7488425   0.5606724 0.6533517 0.6880405
## France       0.4968183 0.8067275 0.8176037   0.7989745 0.8190962 0.8011831
## Germany      0.4824245 0.7370606 0.8304672   0.7446520 0.7867908 0.7913985
## HongKong     0.4739738 0.5792854 0.5973980   0.4805557 0.6322654 0.6591336
## Ireland      0.4747102 0.5998081 0.6065644   0.6410246 0.7103416 0.7113608
## Japan        0.4375893 0.4866818 0.5358427   0.4958760 0.5259886 0.5839363
## Netherlands  0.3858850 0.6254792 0.6712783   0.6662079 0.7012243 0.6227216
```

```
## NewZealand   1.0000000 0.4814932 0.4573488   0.5763765 0.5427861 0.5421729
## Spain        0.4814932 1.0000000 0.7310355   0.6659338 0.7307110 0.6943745
## Sweden       0.4573488 0.7310355 1.0000000   0.6816989 0.7262147 0.7226737
## Switzerland  0.5763765 0.6659338 0.6816989   1.0000000 0.7413779 0.7089105
## UK           0.5427861 0.7307110 0.7262147   0.7413779 1.0000000 0.8255826
## USA          0.5421729 0.6943745 0.7226737   0.7089105 0.8255826 1.0000000
## Brazil       0.4103215 0.5444987 0.5161613   0.4271179 0.5353068 0.5175980
## Chile        0.4333813 0.4311748 0.4594414   0.4407258 0.4826515 0.5167675
## Mexico       0.4531750 0.5486624 0.5120858   0.4770645 0.5764844 0.6660728
## Peru         0.2808114 0.3066046 0.3269390   0.2577390 0.3538175 0.3838020
## South.Korea  0.4499432 0.4423668 0.4483643   0.3841054 0.5128874 0.4922427
##                Brazil     Chile    Mexico      Peru South.Korea
## Australia   0.5639818 0.4743731 0.6006315 0.4422776   0.5202035
## Austria     0.4670540 0.4491174 0.5396844 0.4953483   0.4177055
## Belgium     0.4221480 0.4206435 0.4777814 0.3395889   0.3907107
## Canada      0.5804489 0.5285505 0.6875743 0.4789480   0.4681153
## Denmark     0.4710060 0.4665362 0.4883464 0.3505954   0.4288943
## Finland     0.5119054 0.3986968 0.4832811 0.2369585   0.4297035
## France      0.5472689 0.4565501 0.5440546 0.2914877   0.4433931
## Germany     0.5078994 0.4381567 0.5808514 0.3172167   0.4382308
## HongKong    0.5939429 0.5379214 0.6339704 0.3834366   0.4993137
## Ireland     0.4344350 0.4207402 0.4881024 0.3241497   0.3937804
## Japan       0.4260748 0.3688103 0.4368920 0.3590441   0.5149592
## Netherlands 0.3594638 0.3508425 0.4609820 0.3035452   0.3862860
## NewZealand  0.4103215 0.4333813 0.4531750 0.2808114   0.4499432
## Spain       0.5444987 0.4311748 0.5486624 0.3066046   0.4423668
## Sweden      0.5161613 0.4594414 0.5120858 0.3269390   0.4483643
## Switzerland 0.4271179 0.4407258 0.4770645 0.2577390   0.3841054
## UK          0.5353068 0.4826515 0.5764844 0.3538175   0.5128874
## USA         0.5175980 0.5167675 0.6660728 0.3838020   0.4922427
## Brazil      1.0000000 0.5457366 0.6110619 0.4133186   0.4242121
## Chile       0.5457366 1.0000000 0.5076630 0.4587706   0.4135685
## Mexico      0.6110619 0.5076630 1.0000000 0.4247362   0.4236402
## Peru        0.4133186 0.4587706 0.4247362 1.0000000   0.3150030
## South.Korea 0.4242121 0.4135685 0.4236402 0.3150030   1.0000000
```

Example Code for Question 4

First prepare the data:

```
library(fImport)
```

```
## Loading required package: timeDate
```

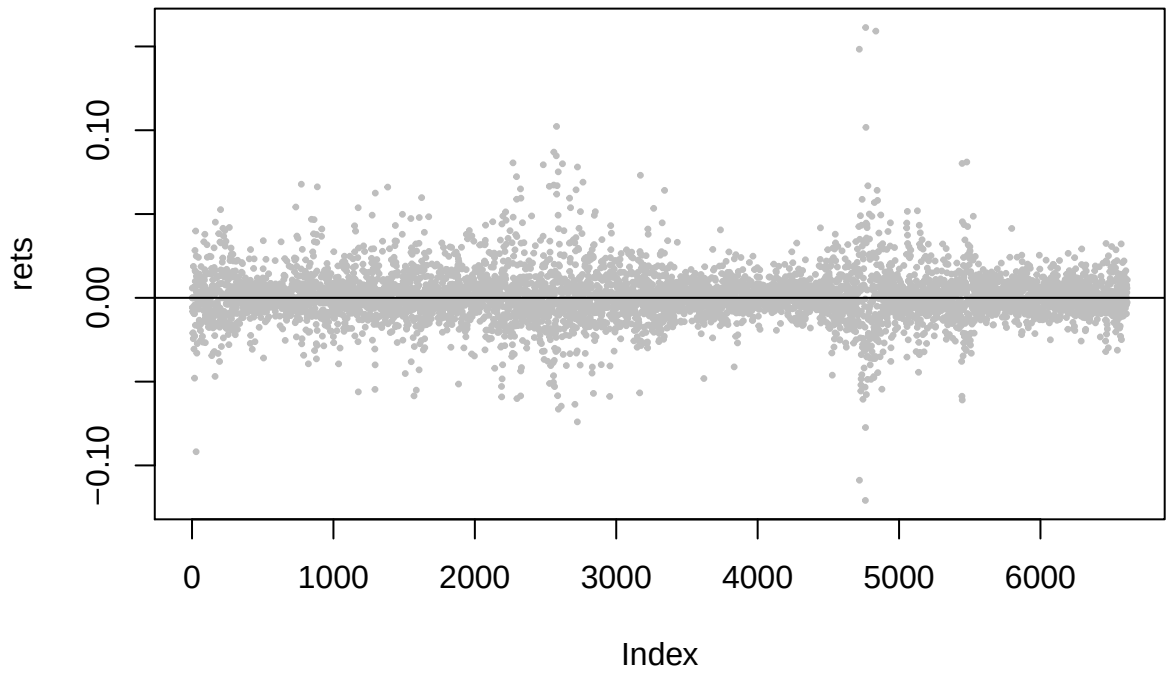
```
## Loading required package: timeSeries
```

```
Y=yahooSeries("BRK-A", from="1990-01-01")
y=rev(Y$'BRK-A.Adj.Close')
n=length(y)
rets=y[-1]/y[-n]-1
summary(rets)
```

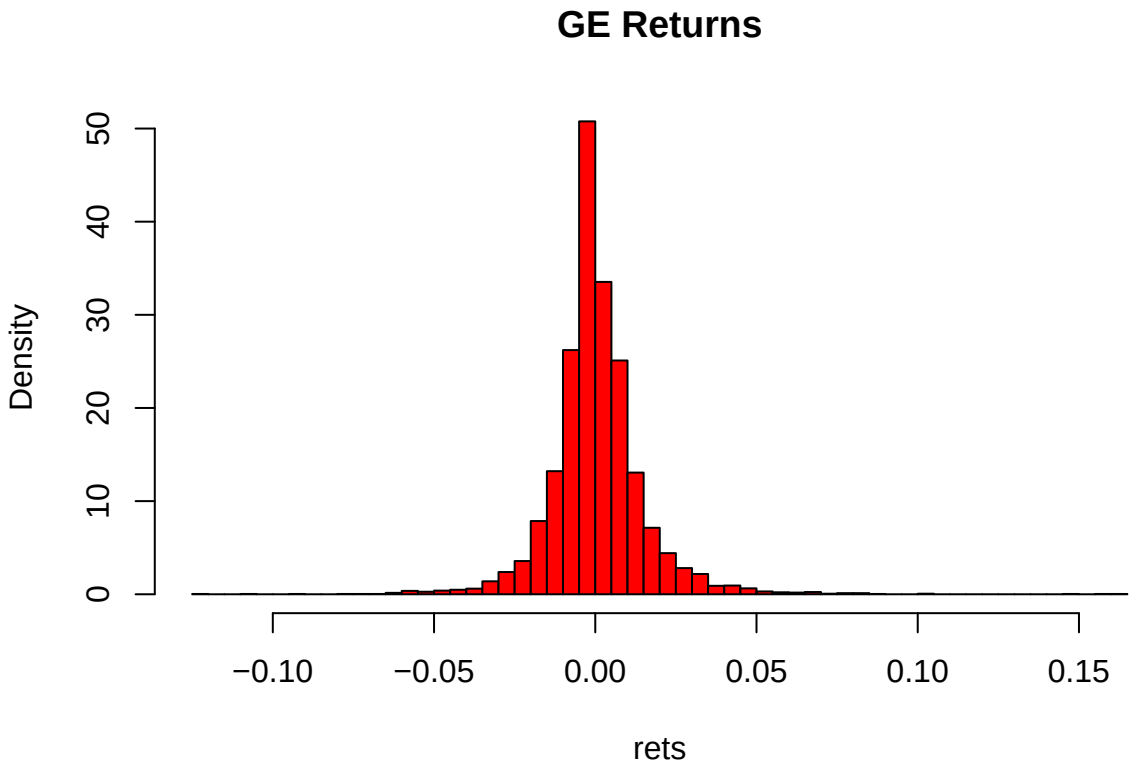
```
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## -0.1209000 -0.0059430  0.0000000  0.0005931  0.0063800  0.1613000
```

Then visualize the data by time series plot and histogram:

```
plot(rets,pch=20,col=24,cex=0.5)
abline(0,0)
```




```
hist(rets,breaks=50,freq=FALSE,main="GE Returns",col="red")
```




```
# find the highest return
iMax = which(rets == max(rets))        # index of the highest return
rets[iMax]                             # largest return
```

```
## [1] 0.1612903
```

```
# find the lowest return
iMin = which(rets == min(rets))        # index of the lowest return
rets[iMin]                             # lowest return
```

```
## [1] -0.1208791
```