ECON90033 – QUANTITATIVE ANALYSIS OF FINANCE I

TUTORIAL 2

Download the t2e1, t2e2, and t2e3 Excel data files from the subject website and save them to your computer or USB flash drive. Read this handout and complete the tutorial exercises before your tutorial class so that you can ask for help during the tutorial if necessary.

Financial Asset Prices and Returns

Exercise 1 (HMPY, p. 22, Ex 2)

The *t2e1.xlsx Excel* file contains monthly observations for the period April 1990 to July 2004 on the equity price of Exxon, General Electric, IBM, Microsoft, and Walmart, together with the price of gold.

a) Plot the price indices and comment on the results.

Launch *RStudio*, create a new project and script, and name them *t2e1*. Import the data from the *t2e1Excel* file, attach it to your project, and save it as *t2e1.RData*.

```
attach(t2e1)
```

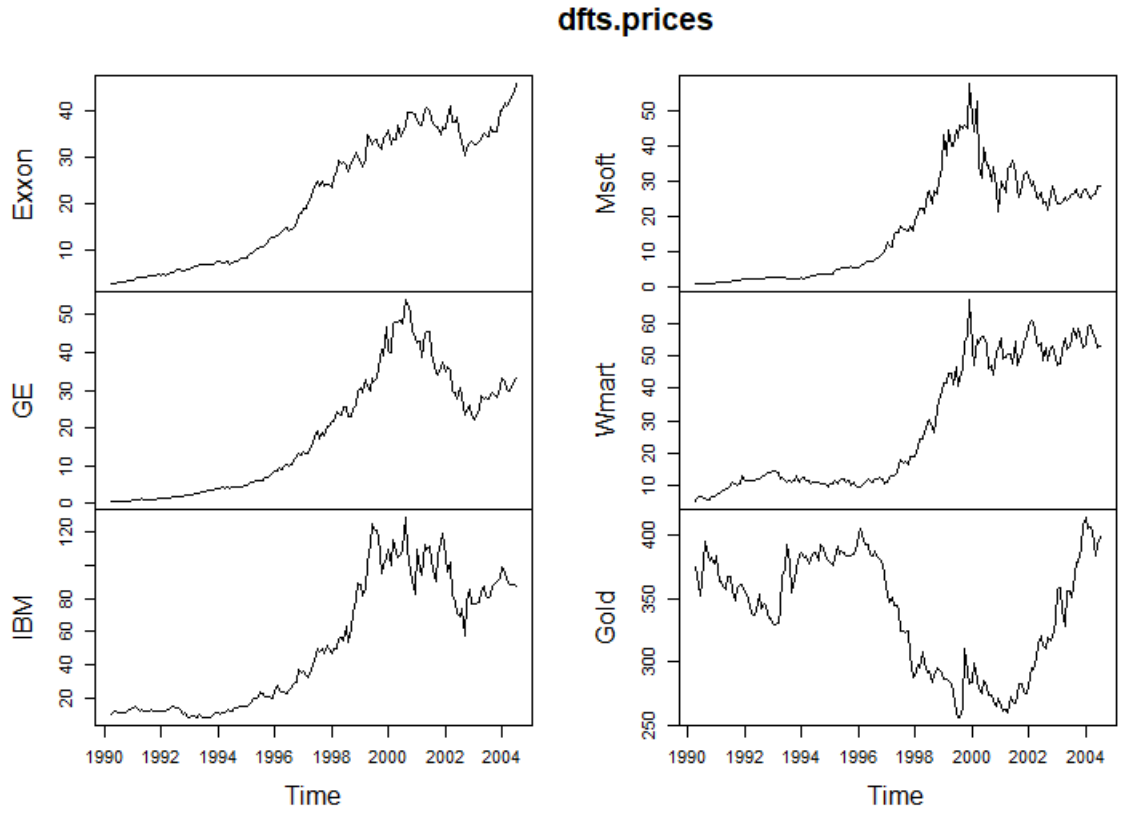
Next we need to create *ts* objects from the six price indices and plot them. We could do so by plugging each series one by one into the *ts()* function and then plot the *ts* object, as we did with the GDP in Exercise 1 of tutorial 1. We can, however, save time by combining the series in a time series data frame and plot them all at once. The

```
dfts.prices = ts(data.frame(Exxon, GE, IBM, Msoft, Wmart, Gold),
                 start = c(1990, 4), end = c(2004, 7), frequency = 12)
plot.ts(dfts.prices)
```

commands produce the plot on the next page. It shows that the equity price of Exxon, General Electric, IBM, Microsoft, and Walmart grew almost exponentially till about year 2000. After 2000, Exxon, GE, IBM, and Msoft first sharply dropped till about 2002-03 and then Exxon recovered and even surpassed its previous high, GE and IBM recovered partially, while Msoft fluctuated around its relatively lower 1998-99 level. Wmart had a milder drop at around 2000 and then fluctuated just like Msoft, but stayed relatively high.

Compared to the equity price of Exxon, General Electric, IBM, Microsoft, and Walmart, the price of Gold exhibits a different pattern. Between 1990 and 1996 it fluctuated a lot, but by 1996 it returned and even surpassed its 1990-91 level. After that, however, it dropped

sharply till almost 2000, while the other five price indices increased, and since 2001 Gold returned to its previous high.



b) Compute simple and logarithmic returns to each of the assets. For each asset plot the two returns series and comment on any differences.

The simple (net) return (R_t) is the proportional (percentage) change in price (P_t) of an asset,

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

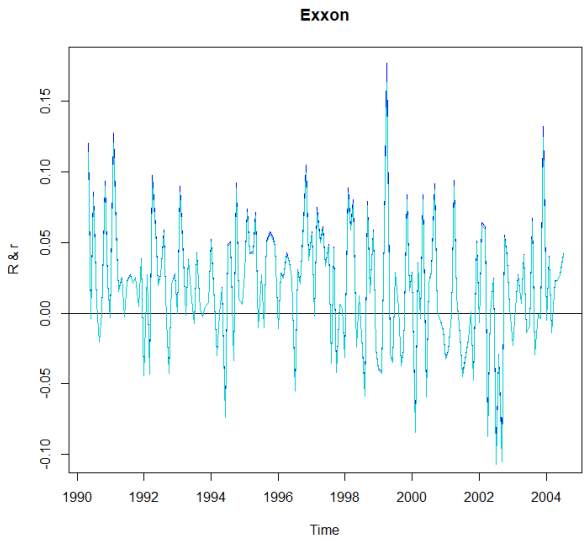
while the one-period logarithmic (log, in brief) return is

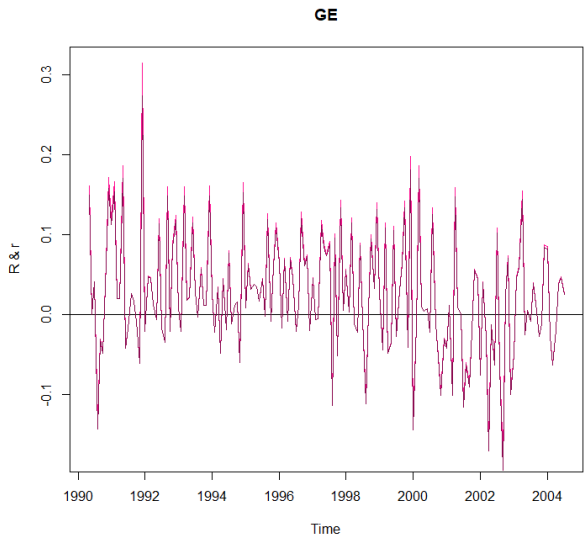
$$r_t = \ln(1 + R_t) = \ln P_t - \ln P_{t-1} = \Delta \ln P_t$$

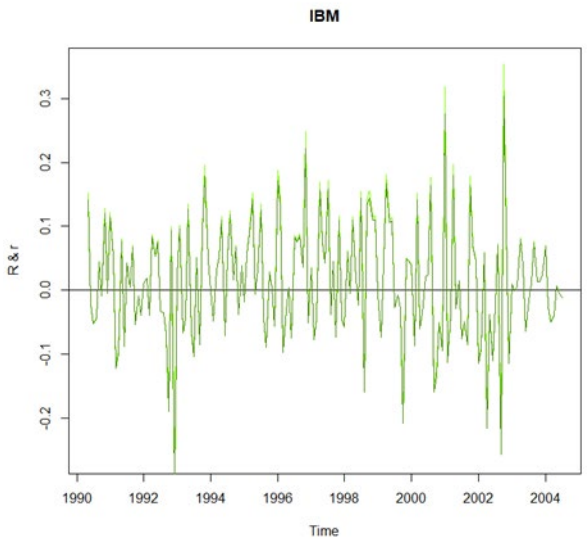
For the equity price of Exxon, these series can be created and plotted by executing the following commands:

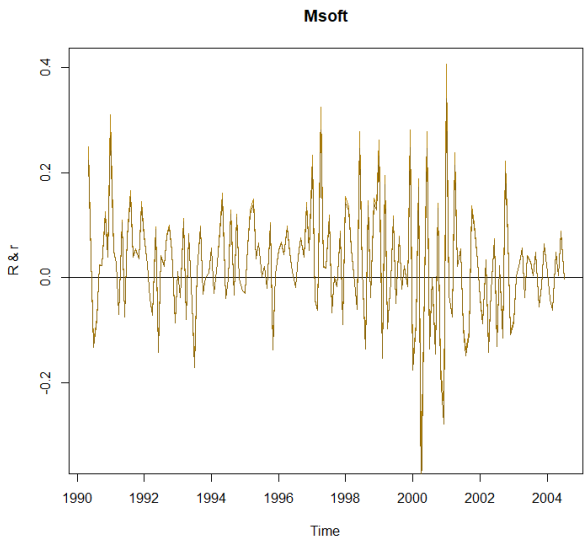
```
Exxon = ts(Exxon, start = c(1990, 4), end = c(2004, 7), frequency = 12)
R_Exxon = Exxon/lag(Exxon,-1) - 1
r_Exxon = diff(log(Exxon), 1)
plot.ts(R_Exxon, ylab = "R & r", main = “Exxon”, col = "blue")
abline(h = 0)
lines(r_Exxon, col = "cyan3")
```

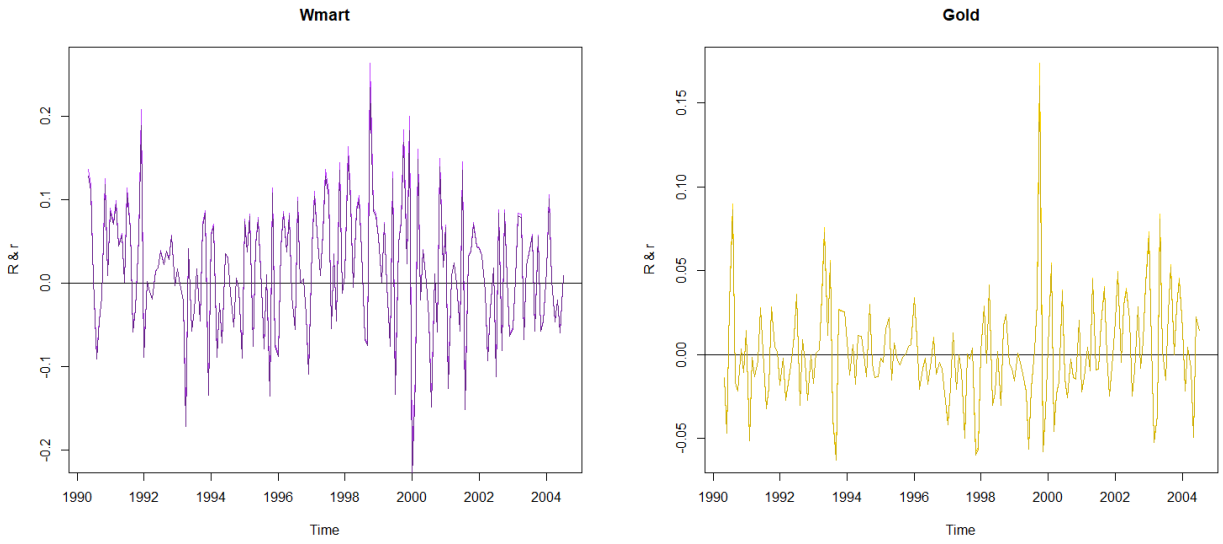
The other plots can be obtained similarly. You should get the following time series plots.











On each plot the two series are visually almost indistinguishable. This is due to the fact that for small x in general, $\ln(1+x) \approx x$, so

$$r_t = \ln(1 + R_t) \approx R_t$$

Otherwise, comparing these plots to each other, you can see that the range of the vertical axis on the first and sixth plots is narrower than on the other four plots. This suggests that the equity prices of Exxon and Gold are less volatile than the equity prices of General Electric, IBM, Microsoft and Walmart.

c) Assume that you hold each of the stocks in an equal-weighted portfolio. Compute the portfolio returns in both simple and logarithmic form for the first 7 months of 2004.

Before we perform the required tasks, let's see the price, the return and the log return of a portfolio, in general.

The price of a portfolio in time t, $P_{P,t}$, is

$$P_{P,t} = \sum_{i=1}^{N} w_{it} P_{it} \quad , \quad \sum_{i=1}^{N} w_{it} = 1$$

where N is the number of stocks in the portfolio, P_{it} is the price of stock i in time t, and w_{it} is the weight of stock i in the portfolio in time t.

It can be shown[1], that the portfolio one-period rate of return, R_{it}, is the weighted average of the returns to the assets in the portfolio,

[1] See pages 15-16 in the prescribed textbook (HMPY).

$$R_{P,t} = \sum_{i=1}^{N} w_{it} R_{it}$$

and the one-period log return to the portfolio is

$$r_{P,t} = \ln\left(\sum_{i=1}^{N} w_{it} e^{r_{it}} \right)$$

Hence, strictly speaking, the log return to the portfolio is not equal to the weighted average of the log returns to the assets in the portfolio. Yet, in practice, when r_{it}'s are relatively small, the one-period log return of the portfolio is often approximated with it, i.e.[2],

$$r_{P,t} \approx \sum_{i=1}^{N} w_{it} r_{it}$$

Once the one-period simple returns and log returns of the portfolio are known, temporal aggregation can be performed just like in the case of a single asset. Namely, the multi-period return and log return of the portfolio are

$$R_{P,t}(k) = \prod_{j=0}^{k-1} (1 + R_{P,t-j}) - 1$$

$$r_{P,t}(k) = \sum_{i=0}^{k-1} r_{P,t-i}$$

Returning to our example, the six equities are supposed to have the same weight in the portfolio, so each weight is w_{it} = 1/6, i = 1, 2, …, 6. Hence, the one-period simple and log returns of the portfolio (*R_P* and *r_P*) can be calculated by executing the following commands[3]:

```
R_P = (R_Exxon + R_GE + R_IBM + R_Msoft + R_Wmart + R_Gold) / 6

r_P = log((exp(R_Exxon) + exp(R_GE) + exp(R_IBM) + exp(R_Msoft)
          + exp(R_Wmart) + exp(R_Gold)) / 6)
```

Their values for the first 7 months of 2004 are returned by the following command:

```
print(window(cbind(R_P, r_P), start = c(2004, 1), end = c(2004, 7)))
```

[2] Note, however, that when there are many small returns and/or the holding period is longer, this approximation can be quite inaccurate.

[3] These commands assume that the simple and log returns of each asset were already obtained in part (b). If you have the returns only for Exxon, calculate them for the other five equities first.

```
                      R_P          r_P
Jan 2004  0.0319718698  0.032540143
Feb 2004  0.0052016942  0.006576937
Mar 2004 -0.0291653125 -0.028760586
Apr 2004 -0.0069056763 -0.006354734
May 2004  0.0007299416  0.001131467
Jun 2004  0.0205187520  0.021537639
Jul 2004  0.0130144422  0.013174949
```

In the previous command there are two useful R functions, *cbind()* and *window()*.

cbind() is the column bind function. It is used for merging two or more data frames together given that the number of rows in the data frames are equal.

window(x, …, start = , end =) is a generic function which extracts the subset of the object x observed between the times start and end.

cbind(R_P, r_P), merges the two portfolio return series and *window(…., start = c(2004, 1), end = c(2004, 7))* extracts the observations for January, …, July of 2004.

Turning to the multi-period returns, the log return of the portfolio for the first 7 months of 2004 is just the sum of the seven simple log returns (*r_P*) above,

```
r_Pk = sum(window(r_P, start = c(2004, 1), end = c(2004, 7)))
print(r_Pk)
```

```
0.03984582
```

To get the simple return of the portfolio (*R_P*) for the first 7 months of 2004, instead of the *sum()* function, we need to use the *prod()* function.

prod() returns the product of all the values present in its arguments. If there is only a single argument, this function computes the multiplicative output of its individual elements.

The

```
R_Pk = prod(window(1 + R_P, start = c(2004, 1), end = c(2004, 7))) - 1
print(R_Pk)
```

commands return

```
0.0346903
```

Hence, for the first 7 months of 2004 the simple and log returns of the portfolio are about 0.0347 and 0.0398.

Save your *R* code and quit *RStudio*.

Statistical Properties of Financial Data

Exercise 2 (HMPY, p. 44, Ex 1)

The *t2e2.xlsx Excel* file contains monthly observations on US equity prices, dividends, earnings, consumer price index, and interest rate for the period January 1900 to September 2016.

a) Plot the equity price (*PRICE*) over time and interpret its time series properties.

Launch *RStudio*, create a new project and script, and name them *t2e2*. Import the data from the *t2e2* Excel file and attach it to your project.

```
attach(t2e2)
```

Create *ts* object from *PRICE* and plot it.

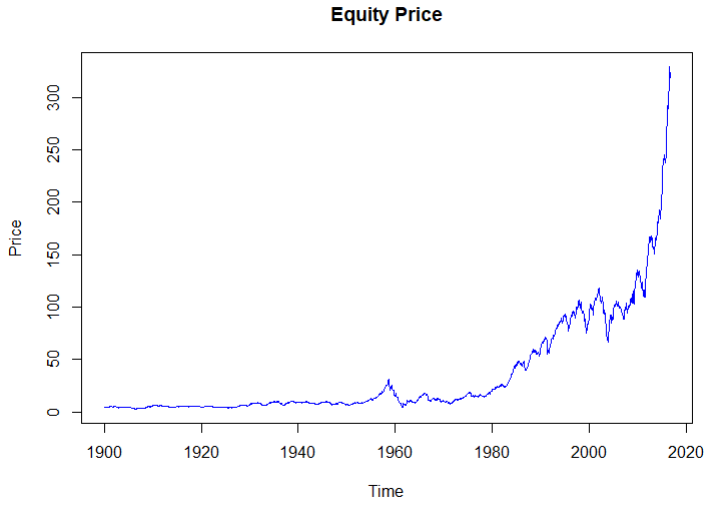
```
PRICE = ts(PRICE, start = c(1900, 1), end = c(2016, 9), frequency = 12)
plot.ts(PRICE, ylab = "Price", main = "Equity Price", col = "blue")
```

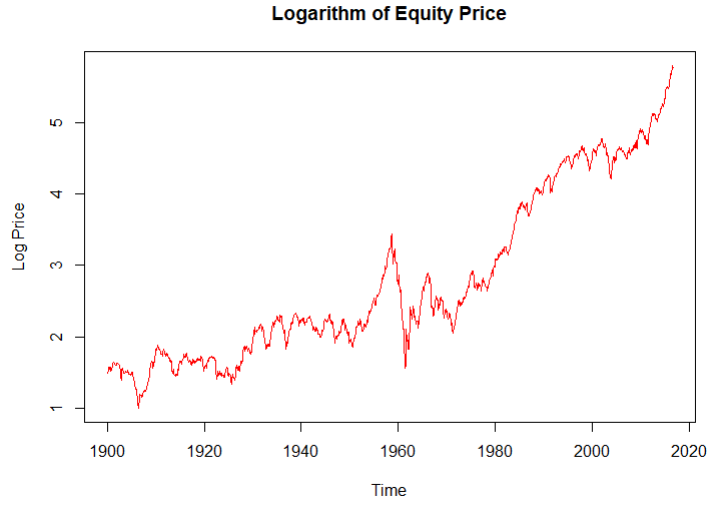
produce the plot displayed on the next page. It shows that apart from some minor fluctuations and a bigger one around 1960, the equity price did not change much up until 1980. After that, however, it grew almost exponentially, though with some major interruptions around 2000.

b) Plot the natural logarithm of the equity price over time and interpret its time series properties.

```
LNPRICE = log(PRICE)
plot.ts(LNPRICE, ylab = "Log Price", main = "Logarithm of Equity Price", col = "red")
```

produce the second plot on the next page. As you can see, while the equity price appears to have an exponential trend, its logarithm seems to fluctuate around a linear trend.





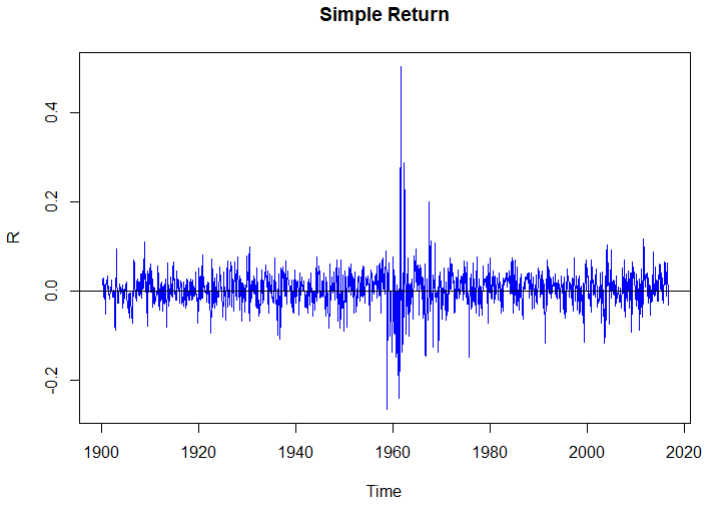
c) Calculate and plot the simple return and the log return on equities over time and interpret their time series properties.

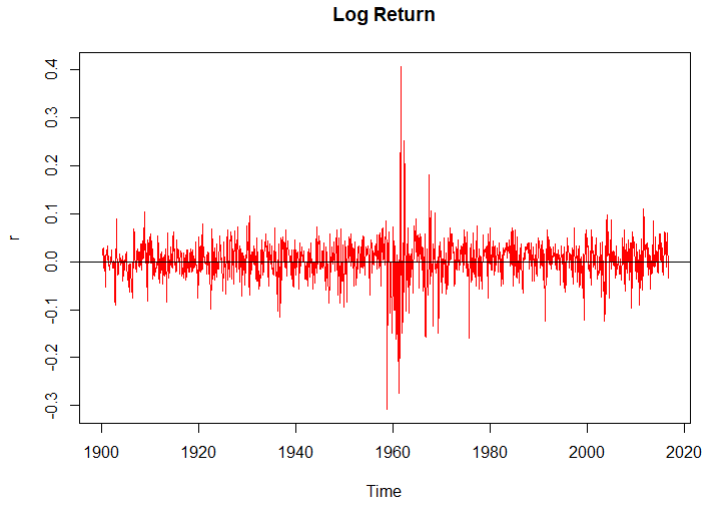
The returns can be calculated and plotted just like in the previous exercise. Executing the

```
R = PRICE/lag(PRICE,-1) - 1
plot.ts(R, ylab = "R", main = "Simple Return", col = "blue")
abline(h = 0)

r = diff(log(PRICE), 1)
plot.ts(r, ylab = "r", main = "Log Return", col = "red")
abline(h = 0)
```

commands, you should get the following plots:





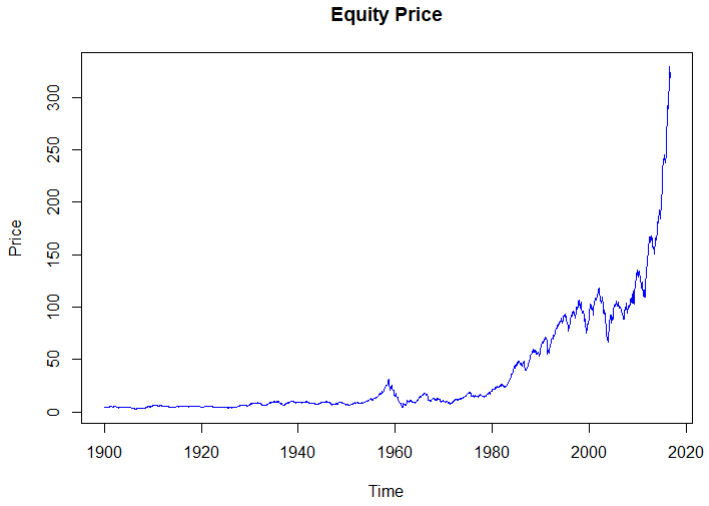
As in Exercise 1, the two plots look very similar. However, as R and r have been displayed separately, by comparing the scales on the vertical axes, it is also visible this time that the largest and smallest values of R are further from zero than those of r.[4]

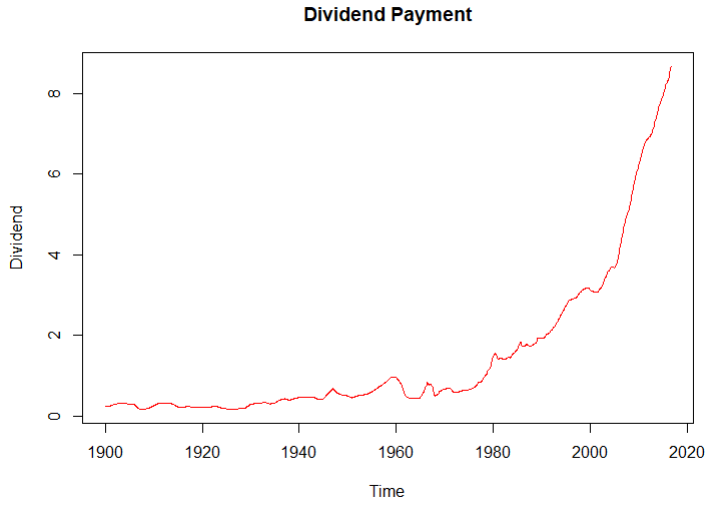
[4] This is because the $\ln(1+x) \approx x$ approximation becomes less and less accurate as x departs from zero.

d) Plot *PRICE* and *DIVIDEND* series and compare their time series properties.

plot.ts(PRICE, ylab = "Price", main = "Equity Price", col = "blue")
plot.ts(DIVIDEND, ylab = "Dividend", main = "Dividend Payment", col = "red")

produce the following plots.





PRICE and *DIVIDEND* appear to follow similar exponential time paths, but *PRICE* is clearly more volatile than *DIVIDEND*.

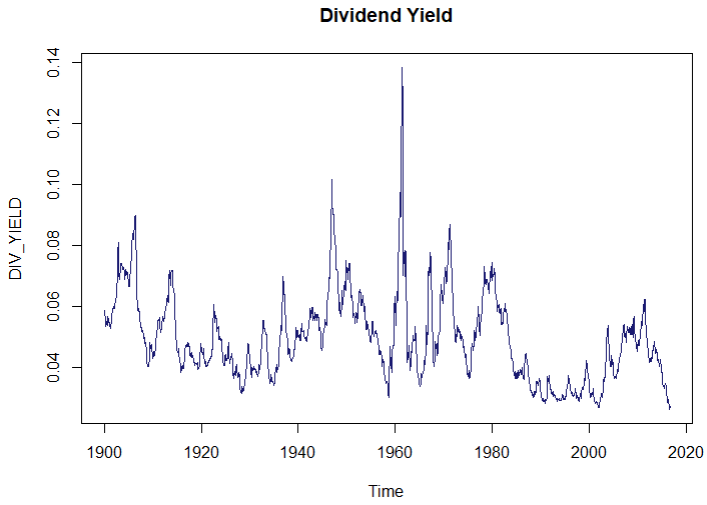
e) Compute the dividend yield. Plot this series and comment on the plot. How does this plot compare to the plots of equity prices and dividend payments in part (d)?

The dividend yield, also known as the dividend–price ratio, of a share is the dividend of a share divided its price.

Execute

```
DIVIDEND = ts(DIVIDEND, start = c(1900, 1), end = c(2016, 9), frequency = 12)
DIV_YIELD = DIVIDEND/PRICE
plot.ts(DIV_YIELD, col = "midnightblue")
```

to get



The dividend yield seems to fluctuate with varying volatility around its historical mean (0.049), or maybe around a slowly declining more or less linear trend. It also appears to be the most volatile in the middle of the sample period, between 1940 and 1980.

Compared to the graphs of equity prices and dividend payments, the major difference is that while *PRICE* and *DIVIDEND* are clearly not stationary in their means, *DIV_YIELD* might be.

f) The present value model predicts a linear relationship between the logarithm of equity prices and the logarithm of dividends. Use a scatter diagram to verify this prediction and comment on the plot.

Let's see first where this linear relationship comes from.

The present value model is based on the idea that individuals are willing to defer consumption to reap future benefits, granted that the value of an investment today is worth the present value of expected future benefits. In terms of equity price and dividend payments this means that in equilibrium the price of an equity (P) is equal to the discounted future stream of dividend payments (D), i.e.,

$$P_t = E\left[\frac{D_{t+1}}{1+\delta_t} + \frac{D_{t+2}}{(1+\delta_t)^2} + \frac{D_{t+3}}{(1+\delta_t)^3} + \ldots \mid \Omega_t\right] = E_t\left[\frac{D_{t+1}}{1+\delta_t} + \frac{D_{t+2}}{(1+\delta_t)^2} + \frac{D_{t+3}}{(1+\delta_t)^3} + \ldots\right]$$

where Ω_t is the information set at time t, i.e., the set of all information available for the investor at time t,

δ_t is the discount rate at time t, and

$E_t[\cdot]$ is the expectations operator conditional on information at time t, i.e. a short-hand notation for $E[\ldots \mid \Omega_t]$.

Assuming that the conditional expected value of all future dividend payments is equal to the current dividend payment, i.e.,

$$E_t\left[D_{t+k}\right] = D_t \quad , \quad k = 1, 2, \ldots$$

the present value model is equivalent to

$$P_t = E_t\left[\frac{D_t}{1+\delta_t} + \frac{D_t}{(1+\delta_t)^2} + \frac{D_t}{(1+\delta_t)^3} + \ldots\right] = D_t \sum_{k=1}^{\infty} \frac{1}{(1+\delta_t)^k} = \frac{D_t}{1+\delta_t} \sum_{k=0}^{\infty} \frac{1}{(1+\delta_t)^k}$$

$$= \frac{D_t}{1+\delta_t} \frac{1}{1-\frac{1}{1+\delta_t}} = \frac{D_t}{1+\delta_t} \frac{1+\delta_t}{\delta_t} = \frac{D_t}{\delta_t}$$

From this, the dividend-price ratio is

$$\frac{D_t}{P_t} = \delta_t$$

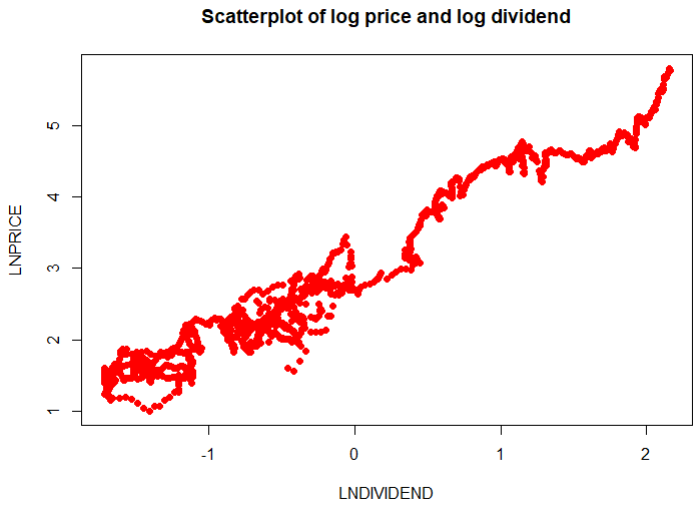
Rearranging this equality, taking the logarithm of both sides, and assuming that the discount rate is kept constant ($\delta_t = \delta$), we get the following linear relationship between the logarithms of P_t and D_t:

$$\ln P_t = -\ln \delta + \ln D_t$$

Execute now the following commands

```
LNDIVIDEND = log(DIVIDEND)
plot(LNDIVIDEND, LNPRICE, main = "Scatterplot of log price and log dividend",
    col = "red", pch = 19)
```

to get this scatterplot:



It shows that the (*LNDIVIDEND*; *LNPRICE*) points are scattered around a straight line, as predicted by the present value model (based on the assumption that the current dividend payment is the best predictor of all future dividend payments).

g) Calculate the sample mean, standard deviation, skewness and kurtosis of the returns on US equities. Interpret the statistics.[5]

The simple and log returns on US equities (*R*, *r*) were calculated in part (c), so now we just have to find the required statistics. We could get them one by one, but it is more convenient to obtain them at once by executing

```
library(pastecs)
stat.desc(cbind(R, r), basic = FALSE, desc = TRUE, norm = TRUE)
```

The results are shown on the next page. The first thing that you can observe is that the simple return and the log return have slightly different descriptive statistics, so $R_t \approx r_t$ is indeed an approximation, though a reasonable accurate one.

To simplify the task, let's focus on the statistics for *R*; the statistics for *r* could be interpreted similarly.

[5] Whenever you are asked to interpret some descriptive statistics, make sure that your answers are precise and comprehensible for the readers even if they do not know the question itself.

	R	r
median	4.966479e-03	4.954187e-03
mean	3.924287e-03	3.052568e-03
SE.mean	1.118448e-03	1.112431e-03
CI.mean.0.95	2.194017e-03	2.182213e-03
var	1.751297e-03	1.732504e-03
std.dev	4.184850e-02	4.162335e-02
coef.var	1.066398e+01	1.363552e+01
skewness	7.326611e-01	-2.608898e-01
skew.2SE	5.601781e+00	-1.994711e+00
kurtosis	1.857342e+01	1.197136e+01
kurt.2SE	7.105480e+01	4.579785e+01
normtest.W	8.915805e-01	9.015078e-01
normtest.p	2.825889e-30	4.347293e-29

The sample mean is about 0.00392, so the average monthly return on US equities is about 0.392%.

The sample standard deviation is about 0.04185. This means that the 'average' deviation of the return on US equities from its sample mean is about 0.04185, i.e., 4.185 percentage point.

The sample skewness statistic is about 0.73266. This point estimate is positive, suggesting that return on US equities skewed to the right. Whether this point estimate is small or large cannot be told without its standard error. Note, however, that *skew.2SE*, which is the point estimate divided by two standard errors, is 5.6018 > 1, so skewness is significantly different from zero.

The sample excess kurtosis statistic is about 18.5734. This point estimate is positive, so this sample of returns on US equities is leptokurtic, i.e., it has a higher peak, thinner bell shape and heavier tails, thus a higher probability of extreme outlier values, than a normal distribution with the same mean and standard deviation. Note also, that *kurt.2SE*, which is the point estimate divided by two standard errors, is 71.0548 > 1, so excess kurtosis is significantly different from zero.

Save your *R* code and quit *RStudio*.

In the first two exercises of this tutorials you worked with monthly observations as in business, commerce, economics and finance variables are typically observed and recorded at discrete and regularly spaced points in time, like every month, every quarter or every year. Some data at higher frequencies, such as at daily and hourly frequencies, however, are also available. In fact, more recently, observations taken at even finer scales have become available. An example is provided by the next exercise.

Exercise 3 (HMPY, p. 45, Ex 3)

The *t2e3.xlsx Excel* file contains high-frequency data (1-second intervals) giving a snapshot of the trades recorded for the American Airlines (AMR) trades between 09:30 and 16:00 on 1 August 2006. There are nine variables.

hour: hour at which the data are recorded.
minute: minute at which the data are recorded.
second: second at which the data are recorded.
time: time variable effectively starts at 9.30am and ends at 4.00pm with the time interval being one second.
x: a binary variable giving a one if a trade has occurred, and 0 otherwise.
n: a counter variable which increases by one when a trade occurs.
d: the duration time in seconds between the latest two trades.
firm: firm news dummy variable.
macro: macro news dummy variable.

At this stage ignore the last two variables. To understand the other variables, consider the following table that shows the subset of the data from 9:42:00 (row 722) to 9:42:13 (row 735).

	A	B	C	D	E	F	G
1	hour	minute	second	time	x	n	d
722	9	42	0	34920	0	116	1
723	9	42	1	34921	0	116	1
724	9	42	2	34922	0	116	1
725	9	42	3	34923	0	116	1
726	9	42	4	34924	1	117	11
727	9	42	5	34925	1	118	1
728	9	42	6	34926	0	118	1
729	9	42	7	34927	0	118	1
730	9	42	8	34928	0	118	1
731	9	42	9	34929	0	118	1
732	9	42	10	34930	0	118	1
733	9	42	11	34931	1	119	6
734	9	42	12	34932	1	120	1
735	9	42	13	34933	0	120	1

In row 722, *hour* = 9, *minute* = 42, *second* = 0, so *time* = 9*60*60 + 42*60 = 34920 (seconds). In the first second, i.e., in the [9:42:00 ; 9:42:01) interval, which is closed from below and open from above, *x* = 0, so there wasn't any trade of AMR shares. Prior to this second, there were already *n* = 116 AMR transactions that day. Finally, *d* = 1 means that the duration time between

the latest two trades, which were at 9:41:52 and 9:41:53, was 1 second.[6]

Moving on to the next four rows, you can see that the next transaction occurred at 9:42:04 (row 726), so in that second x = 1, n increase from 116 to 117, and d = 11 means that the duration since the previous transaction at 9:41:53 was 11 seconds.

In the next second, at 9:42:05, there was again an ARM transaction, so x = 1, n = 118, d = 1.

Launch *RStudio*, create a new project and script, and name them *t2e3*. Import the data from the *Data* sheet of the *t2e3* Excel file and attach it to your project.

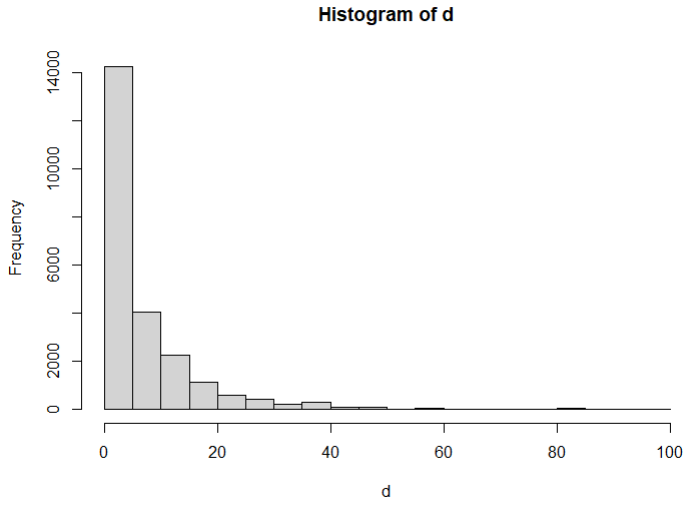
attach(t2e3)

Save the data as *t2e3.RData*. Although we have again time series data, in this exercise it is unnecessary to create *ts* objects.

Use a histogram to graph the empirical distribution of the duration times.

A basic histogram is returned by

hist(d)



This histogram illustrates that duration time is one of the variables that are definitely not normally distributed. Instead, the frequency is the highest in the first category (bin) and it drops in a seemingly exponential way.

We can get a better picture of the empirical distribution by making the class intervals narrower and depicting the relative frequency distribution instead of the frequency distribution.

[6] You cannot see these transactions in this table, but you can verify them in the Excel spreadsheet.

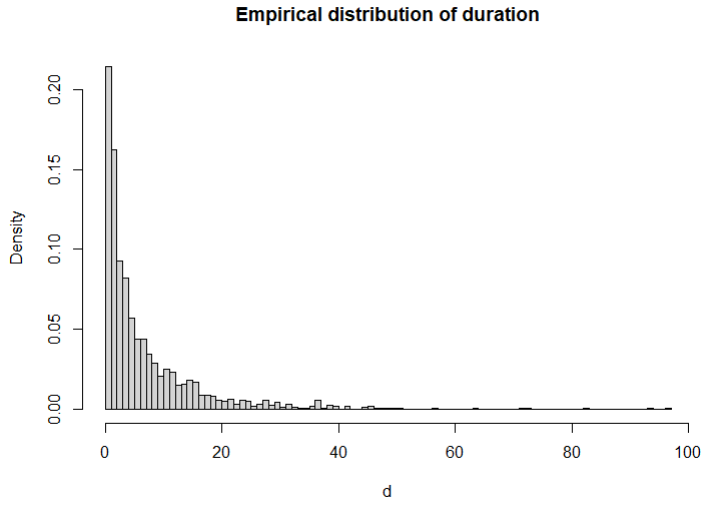
The number of bins can be controlled by the breaks argument of the hist() function.

Whether a histogram illustrates a frequency distribution, or a relative frequency distribution is determined by the freq argument of the hist() function. It is a logical variable, equal to TRUE (default) for frequencies or FALSE for relative frequencies.

The

hist(d, breaks = 100, freq = FALSE, main = "Empirical distribution of duration")

command produces the following plot:



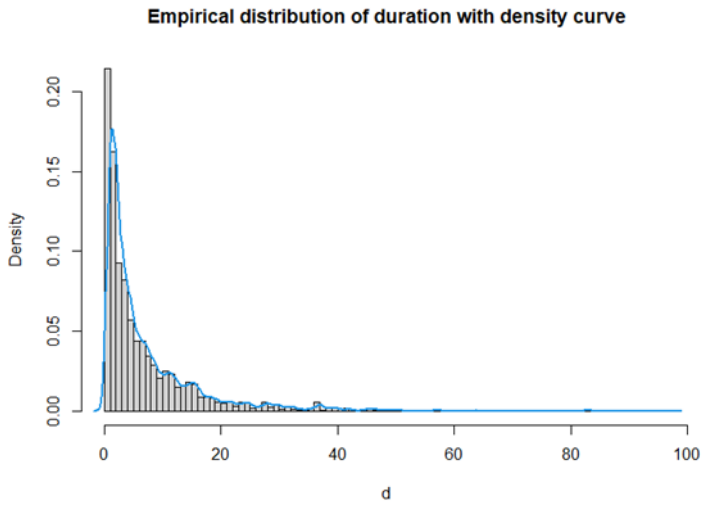
As a further refinement, we can superimpose the density curve of the data using the *density()* function. The

hist(d, breaks = 100, freq = FALSE,
main = "Empirical distribution of duration with density curve")
lines(density(d), col = 4, lwd = 2)

commands produce the plot shown on the next page. It confirms that the empirical distribution of the duration times between successive transactions is very different from the normal distribution, it is rather similar to an exponential distribution.

A continuous random variable *X* is said to have an exponential distribution with parameter $\lambda > 0$ if its probability density function is

$$f(x) = \begin{cases} \lambda e^{-\lambda x} & for\ x > 0 \\ 0 & otherwise \end{cases}$$



The expected value and standard deviation of an exponential random variable *X* are equal,

$$\mu_X = \sigma_X = \frac{1}{\lambda}$$

Hence, one way to check whether duration times are indeed, more or less, exponentially distributed is to compare the sample mean and sample standard deviation of *d*.

mean(d)
sd(d)

return

```
> mean(d)
[1] 7.269647
> sd(d)
[1] 9.244672
```

The difference between these sample mean and sample standard deviation is about 1.975. This is more than 20% of the sample standard deviation and more than 27% of the sample mean, so it is quite substantial, suggesting that the population mean and standard deviation might be different, and thus the population of durations might not be exponentially distributed.

A formal test for the null hypothesis H_0: *d* is distributed exponentially against the alternative hypothesis H_A: *d* is not distributed exponentially is the Lillefors test for exponentiality. We do not discuss the details of this test, but it can be performed easily with the *gofTest()* function of the *EnvStats* package of *R*.

Install the *EnvStats* package and run

```
library(EnvStats)
test_exp_lillie = gofTest(d, test = "lillie", distribution = "exp")
test_exp_lillie$statistic
test_exp_lillie$p.value
```

You should get the following test statistic and *p*-value:

```
> test_exp_lillie$statistic
        D
0.1306629
> test_exp_lillie$p.value
[1] 0
```

The *p*-value is practically zero, so the null hypothesis of exponential distribution is rejected by the Lillefors test.[7]

Save your *R* code and quit *RStudio*.

[7] It is worth to mention that *gofTest()* can perform other tests as well. If you replace *lillie* in the previous commands with *chisq*, *ad*, or *cvm*, and re-execute them, you can see that the chi-square, Anderson-Darling, and Cramer-von Mises tests lead to the same conclusion.