Working with the xlsx package solutions (part-2)

Below are the solutions to these exercises on working with the xlsx package.

```
####################
#                  #
#    Exercise 1    #
#                  #
####################
# Install if necessary
# install.packages("xlsx", dependencies = TRUE)
# Load:
require(xlsx)
```

```
## Warning: package 'rJava' was built under R version 3.3.2
```

```
####################
#                  #
#    Exercise 2    #
#                  #
####################
# Make sure to have the the read working directory set.
commuting   <-   read.xlsx2("theme_11_small_areas.xlsx",
sheetIndex  =  1,
                           colClasses = c(rep("character", 9),
rep("integer", 32)))
```

```
####################
#                  #
#    Exercise 3    #
#                  #
####################
names(commuting)                                              <-
gsub("Population_Aged_5_Over_By_|To_Work_School_College_",
                          "",
```

```
                          names(commuting))


####################
#                  #
#    Exercise 4    #
#                  #
####################
commuting_ed <- aggregate(. ~ Electoral.Division.Name,
                          commuting[,c(5, 10:41)],
                          FUN = sum)


####################
#                  #
#    Exercise 5    #
#                  #
####################
wb <- createWorkbook(type = "xlsx")


####################
#                  #
#    Exercise 6    #
#                  #
####################
for (i in 1:3) {
  sheetname <- paste0("sheet", i)
  assign(sheetname, createSheet(wb, sheetName = sheetname))
}


####################
#                  #
#    Exercise 7    #
#                  #
####################
commut_prop <- data.frame(
                  Divison   =
                    commuting_ed$Electoral.Division.Name,
                  prop_foot =
```

```
                    commuting_ed$Means_Of_Travel_On_Foot_2011
/
                   commuting_ed$Means_Of_Travel_Total_2011,
                 prop_cycl =
                    commuting_ed$Means_Of_Travel_Bicycle_2011
/
                   commuting_ed$Means_Of_Travel_Total_2011,
                 prop_earl =
commuting_ed$Time_Leaving_Home_To_Travel_Before_0630_2011 /
commuting_ed$Time_Leaving_Home_To_Travel_Total_2011)

commut_prop[, -1] <- round(commut_prop[, -1], 2)


####################
#                  #
#    Exercise 8    #
#                  #
####################
category <- c("prop_foot", "prop_cycl", "prop_earl")

for (i in 1:3) {
  addDataFrame(
    commut_prop[order(commut_prop[, category[i]], decreasing =
TRUE), ][1:5,],
    get(paste0("sheet", i)),
    startRow = 2L,
    startColumn = 1L,
    row.names = FALSE)
}


####################
#                  #
#    Exercise 9    #
#                  #
####################
font <- Font(wb, color = "blue", heightInPoints = 14, isBold =
TRUE)
border <- Border(color = "black", position = "BOTTOM", pen =
"BORDER_THIN")
```

```
for (i in 1:3) {
  title <- CellBlock(get(paste0("sheet", i)), 1, 1, 1, 1)
  abovet <- CellBlock(get(paste0("sheet", i)), 1, 1, 1, 4)
  CB.setRowData(title,
                paste("Top 5 electoral divisions in Dublin
area",
                      "in the category of",
                      substr(category[i], 6, 9)),
                1)
  # Ok, the title is not that great (nor pretty), but you get
the idea 
  CB.setFont(title, font, 1, 1)
  CB.setBorder(abovet, border, 1, 1:4)
}


####################
#                  #
#    Exercise 10   #
#                  #
####################
saveWorkbook(wb, "filename.xlsx") # setwd() first
```

Volatility modelling in R (Part-1) solutions

Below are the solutions to these exercises on volatility modelling.

```
###############
#             #
# Exercise 1  #
#             #
###############
```

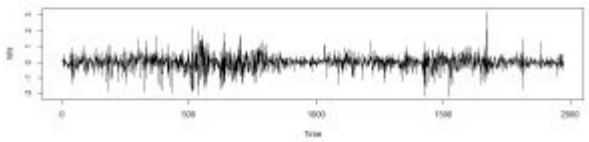
```
library(rugarch)
```

```
## Warning: package 'rugarch' was built under R version 3.3.3
```

```
data("dmbp")
```

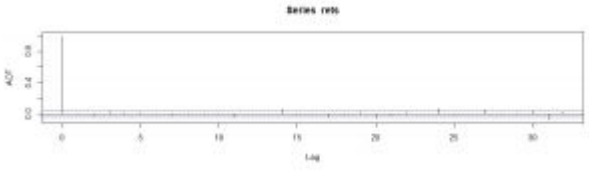
```
###############
#             #
# Exercise 2  #
#             #
###############
```

```
rets <- ts(dmbp$V1)
plot(rets)
```



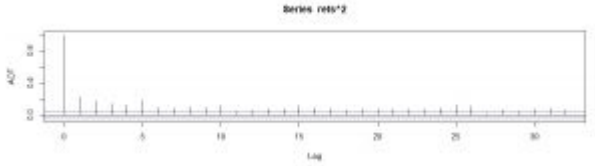
```
###############
#             #
# Exercise 3  #
#             #
###############
```

```
acf(rets)
```



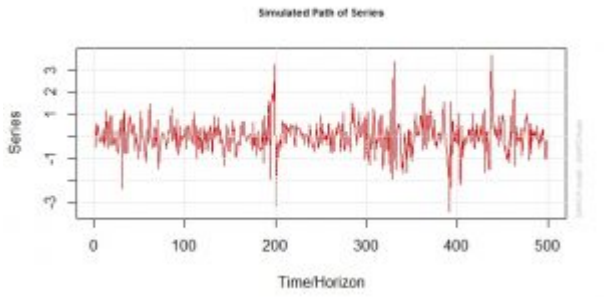

```
###############
#             #
# Exercise 4  #
#             #
###############
```

```
acf(rets^2)
```

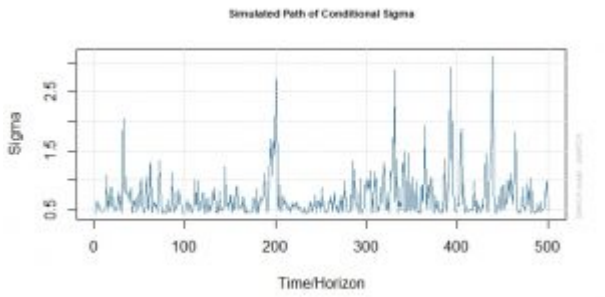


```
###############
#             #
# Exercise 5  #
#             #
###############

arch1.mod      =      ugarchspec(variance.model      =
list(garchOrder=c(1,)),
                                                mean.model  =
list(armaOrder=c(,)),
                                          fixed.pars=list(mu=,
omega=0.2, alpha1=0.7))
arch1.mod

##
## *----------------------------------*
## *       GARCH Model Spec           *
## *----------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model          : sGARCH(1,0)
## Variance Targeting   : FALSE
##
## Conditional Mean Dynamics
## -----------------------------------
## Mean Model           : ARFIMA(0,0,0)
## Include Mean         : TRUE
## GARCH-in-Mean                : FALSE
##
## Conditional Distribution
## -----------------------------------
## Distribution :  norm
## Includes Skew        :  FALSE
## Includes Shape       :  FALSE
## Includes Lambda      :  FALSE
```

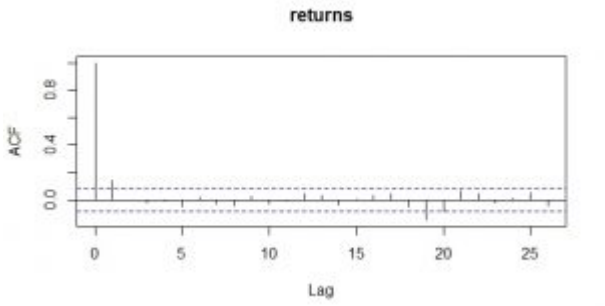
```
###############
#             #
# Exercise 6  #
#             #
###############

arch1.sim = ugarchpath(arch1.mod,n.sim=500)

###############
#             #
# Exercise 7  #
#             #
###############

plot(arch1.sim, which=2)
```



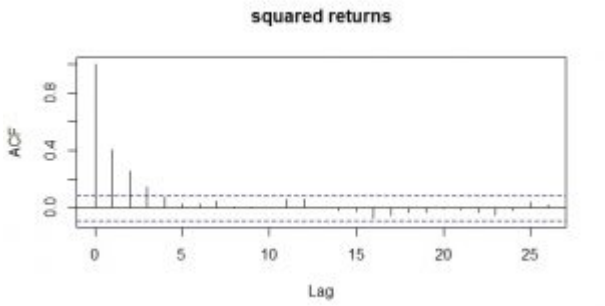

```
plot(arch1.sim, which=1)
```




```
###############
#             #
# Exercise 8  #
#             #
###############

acf(arch1.sim@path$seriesSim, main="returns")
```




```
acf(arch1.sim@path$seriesSim^2, main="squared returns")
```




```
###############
#             #
# Exercise 9  #
#             #
###############

Box.test(arch1.sim@path$seriesSim^2, type = "Ljung-Box", lag =
12)

##
##      Box-Ljung test
##
## data:  arch1.sim@path$seriesSim^2
## X-squared = 132.71, df = 12, p-value < 2.2e-16

################
#             #
# Exercise 10  #
#             #
################

Box.test(rets^2, type = "Ljung-Box", lag = 12)
```

```
##
##      Box-Ljung test
##
## data:  rets^2
## X-squared = 407.84, df = 12, p-value < 2.2e-16
```

Data Visualization with googleVis solutions part 4

Below are the solutions to these exercises on visualizations with googleVis.

```
####################
#                  #
#    Exercise 1    #
#                  #
####################

library(googleVis)
df=data.frame(name=c("James", "Curry", "Harden"),
              Pts=c(20,23,34),
              Rbs=c(13,7,9))
LineCD2 <- gvisLineChart(df, "name", c("Pts","Rbs"),
                         options=list(
series="[{color:'green',targetAxisIndex: 0},
{color:'yellow',targetAxisIndex:1}]",
                                      vAxes="[{title:'Pts'},
{title:'Rbs'}]"
                         ))
plot(LineCD2)

####################
#                  #
#    Exercise 2    #
#                  #
```

```
####################

library(googleVis)
df=data.frame(name=c("James", "Curry", "Harden"),
              Pts=c(20,23,34),
              Rbs=c(13,7,9))
LineCD2 <- gvisLineChart(df, "name", c("Pts","Rbs"),
                         options=list(
series="[{color:'green',targetAxisIndex: 0, lineWidth: 3},
{color:'yellow',targetAxisIndex:1,lineWidth: 6}]",
                                      vAxes="[{title:'Pts'},
{title:'Rbs'}]"
                         ))
plot(LineCD2)

####################
#                  #
#    Exercise 3    #
#                  #
####################

library(googleVis)
df=data.frame(name=c("James", "Curry", "Harden"),
              Pts=c(20,23,34),
              Rbs=c(13,7,9))
LineCD2 <- gvisLineChart(df, "name", c("Pts","Rbs"),
                         options=list(
series="[{color:'green',targetAxisIndex: 0, lineWidth: 3,
                           lineDashStyle: [14, 2, 2, 7]},
{color:'yellow',targetAxisIndex:1,lineWidth: 6,
                           lineDashStyle: [10, 2]}]",
                                      vAxes="[{title:'Pts'},
{title:'Rbs'}]"
                         ))
plot(LineCD2)

####################
#                  #
#    Exercise 4    #
#                  #
####################
```

```
library(googleVis)
ScatterCD <- gvisScatterChart(cars,
                              options=list(
                                legend="none",
pointSize=3,lineWidth=2,pointShape='square',
                                                   title="Cars",
vAxis="{title:'speed'}",
                                hAxis="{title:'dist'}",
                                width=600, height=300))
plot(ScatterCD)


####################
#                  #
#    Exercise 5    #
#                  #
####################

library(googleVis)
ScatterCD <- gvisScatterChart(cars,
                              options=list(
                                legend="none",
pointSize=7,lineWidth=2,pointShape='triangle',
                                                   title="Cars",
vAxis="{title:'speed'}",
                                hAxis="{title:'dist'}",
                                width=600, height=300))
plot(ScatterCD)

####################
#                  #
#    Exercise 6    #
#                  #
####################

library(googleVis)
ScatterCD <- gvisScatterChart(cars,
                              options=list(gvis.editor="Edit",
                                legend="none",
pointSize=7,lineWidth=2,pointShape='triangle',
                                                   title="Cars",
```

```
vAxis="{title:'speed'}",
                                hAxis="{title:'dist'}",
                                width=600, height=300))
plot(ScatterCD)

####################
#                  #
#    Exercise 7    #
#                  #
####################

library(googleVis)
df=data.frame(name=c("James", "Curry", "Harden"),
              Pts=c(20,23,34),
              Rbs=c(13,7,9))
LineCD2 <- gvisLineChart(df, "name", c("Pts","Rbs"),
options=list(backgroundColor="lightblue",
series="[{color:'green',targetAxisIndex: 0, lineWidth: 3,
                           lineDashStyle: [14, 2, 2, 7]},
{color:'yellow',targetAxisIndex:1,lineWidth: 6,
                           lineDashStyle: [10, 2]}]",
                                    vAxes="[{title:'Pts'},
{title:'Rbs'}]"
                         ))
plot(LineCD2)

####################
#                  #
#    Exercise 8    #
#                  #
####################

library(googleVis)
df=data.frame(name=c("James", "Curry", "Harden"),
              Pts=c(20,23,34),
              Rbs=c(13,7,9))
LineCD2 <- gvisLineChart(df, "name", c("Pts","Rbs"),
options=list(backgroundColor="lightblue",
                                       title="Line Chart",
titleTextStyle="{color:'blue',
                                             fontName:'Courier',
```

```
                                        fontSize:16}",
series="[{color:'green',targetAxisIndex: 0, lineWidth: 3,
                          lineDashStyle: [14, 2, 2, 7]},
{color:'yellow',targetAxisIndex:1,lineWidth: 6,
                          lineDashStyle: [10, 2]}]",
                                   vAxes="[{title:'Pts'},
{title:'Rbs'}]"
                         ))
plot(LineCD2)

####################
#                  #
#    Exercise 9    #
#                  #
####################

library(googleVis)
df=data.frame(name=c("James", "Curry", "Harden"),
              Pts=c(20,23,34),
              Rbs=c(13,7,9))
LineCD2 <- gvisLineChart(df, "name", c("Pts","Rbs"),
options=list(backgroundColor="lightblue",
                                        title="Line Chart",
titleTextStyle="{color:'blue',
                                              fontName:'Courier',
                                        fontSize:16}",
                                        curveType="function",
series="[{color:'green',targetAxisIndex: 0, lineWidth: 3,
                          lineDashStyle: [14, 2, 2, 7]},
{color:'yellow',targetAxisIndex:1,lineWidth: 6,
                          lineDashStyle: [10, 2]}]",
                                   vAxes="[{title:'Pts'},
{title:'Rbs'}]"
                         ))
plot(LineCD2)

####################
#                  #
#    Exercise 10   #
#                  #
####################
```

```
library(googleVis)
df=data.frame(name=c("James", "Curry", "Harden"),
              Pts=c(20,23,34),
              Rbs=c(13,7,9))
LineCD2 <- gvisLineChart(df, "name", c("Pts","Rbs"),
options=list(backgroundColor="lightblue",
                                    title="Line Chart",
titleTextStyle="{color:'blue',
                                          fontName:'Courier',
                                    fontSize:16}",
                                    curveType="function",
                                    legend="bottom",
vAxis="{gridlines:{color:'red', count:4}}",
                                      hAxis="{title:'Name',
titleTextStyle:{color:'orange'}}",
series="[{color:'green',targetAxisIndex: 0, lineWidth: 3,
                           lineDashStyle: [14, 2, 2, 7]},
{color:'yellow',targetAxisIndex:1,lineWidth: 6,
                           lineDashStyle: [10, 2]}]",
                                    vAxes="[{title:'Pts'},
{title:'Rbs'}]"
                          ))
plot(LineCD2)
```

Data wrangling : Reshaping Solution

Below are the solutions to these exercises on data reshaping.

```
####################
#                  #
#    Exercise 1    #
#                  #
####################
```

```
str(data)
```

```
## 'data.frame':        153 obs. of  3 variables:
##  $ Temp : int  67 72 74 62 56 66 65 59 61 69 ...
##  $ Month: int  5 5 5 5 5 5 5 5 5 5 ...
##  $ Day  : int  1 2 3 4 5 6 7 8 9 10 ...
```

```
####################
#                  #
#    Exercise 2    #
#                  #
####################
```

```
long2wide <- data %>% spread(Month, Temp); long2wide
```

```
##    Day  5  6  7  8  9
## 1    1 67 78 84 81 91
## 2    2 72 74 85 81 92
## 3    3 74 67 81 82 93
## 4    4 62 84 84 86 93
## 5    5 56 85 83 85 87
## 6    6 66 79 83 87 84
## 7    7 65 82 88 89 80
## 8    8 59 87 92 90 78
## 9    9 61 90 92 90 75
## 10  10 69 87 89 92 73
## 11  11 74 93 82 86 81
## 12  12 69 92 73 86 76
## 13  13 66 82 81 82 77
## 14  14 68 80 91 80 71
## 15  15 58 79 80 79 71
## 16  16 64 77 81 77 78
## 17  17 66 72 82 79 67
## 18  18 57 65 84 76 76
## 19  19 68 73 87 78 68
## 20  20 62 76 85 78 82
## 21  21 59 77 74 77 64
## 22  22 73 76 81 72 71
## 23  23 61 76 82 75 81
## 24  24 61 76 86 79 69
## 25  25 57 75 85 81 63
```

```
## 26  26 58 78 82 86 70
## 27  27 57 73 86 88 77
## 28  28 67 80 88 97 75
## 29  29 81 77 86 94 76
## 30  30 79 83 83 96 68
## 31  31 76 NA 81 94 NA

####################
#                  #
#    Exercise 3    #
#                  #
####################

back2long <- long2wide %>% gather(Month, Temp, 2:6); back2long

##     Day Month Temp
## 1     1     5   67
## 2     2     5   72
## 3     3     5   74
## 4     4     5   62
## 5     5     5   56
## 6     6     5   66
## 7     7     5   65
## 8     8     5   59
## 9     9     5   61
## 10   10     5   69
## 11   11     5   74
## 12   12     5   69
## 13   13     5   66
## 14   14     5   68
## 15   15     5   58
## 16   16     5   64
## 17   17     5   66
## 18   18     5   57
## 19   19     5   68
## 20   20     5   62
## 21   21     5   59
## 22   22     5   73
## 23   23     5   61
## 24   24     5   61
## 25   25     5   57
```

```
## 26   26     5   58
## 27   27     5   57
## 28   28     5   67
## 29   29     5   81
## 30   30     5   79
## 31   31     5   76
## 32    1     6   78
## 33    2     6   74
## 34    3     6   67
## 35    4     6   84
## 36    5     6   85
## 37    6     6   79
## 38    7     6   82
## 39    8     6   87
## 40    9     6   90
## 41   10     6   87
## 42   11     6   93
## 43   12     6   92
## 44   13     6   82
## 45   14     6   80
## 46   15     6   79
## 47   16     6   77
## 48   17     6   72
## 49   18     6   65
## 50   19     6   73
## 51   20     6   76
## 52   21     6   77
## 53   22     6   76
## 54   23     6   76
## 55   24     6   76
## 56   25     6   75
## 57   26     6   78
## 58   27     6   73
## 59   28     6   80
## 60   29     6   77
## 61   30     6   83
## 62   31     6   NA
## 63    1     7   84
## 64    2     7   85
## 65    3     7   81
## 66    4     7   84
```

```
## 67    5     7   83
## 68    6     7   83
## 69    7     7   88
## 70    8     7   92
## 71    9     7   92
## 72   10     7   89
## 73   11     7   82
## 74   12     7   73
## 75   13     7   81
## 76   14     7   91
## 77   15     7   80
## 78   16     7   81
## 79   17     7   82
## 80   18     7   84
## 81   19     7   87
## 82   20     7   85
## 83   21     7   74
## 84   22     7   81
## 85   23     7   82
## 86   24     7   86
## 87   25     7   85
## 88   26     7   82
## 89   27     7   86
## 90   28     7   88
## 91   29     7   86
## 92   30     7   83
## 93   31     7   81
## 94    1     8   81
## 95    2     8   81
## 96    3     8   82
## 97    4     8   86
## 98    5     8   85
## 99    6     8   87
## 100   7     8   89
## 101   8     8   90
## 102   9     8   90
## 103  10     8   92
## 104  11     8   86
## 105  12     8   86
## 106  13     8   82
## 107  14     8   80
```

```
## 108  15     8   79
## 109  16     8   77
## 110  17     8   79
## 111  18     8   76
## 112  19     8   78
## 113  20     8   78
## 114  21     8   77
## 115  22     8   72
## 116  23     8   75
## 117  24     8   79
## 118  25     8   81
## 119  26     8   86
## 120  27     8   88
## 121  28     8   97
## 122  29     8   94
## 123  30     8   96
## 124  31     8   94
## 125   1     9   91
## 126   2     9   92
## 127   3     9   93
## 128   4     9   93
## 129   5     9   87
## 130   6     9   84
## 131   7     9   80
## 132   8     9   78
## 133   9     9   75
## 134  10     9   73
## 135  11     9   81
## 136  12     9   76
## 137  13     9   77
## 138  14     9   71
## 139  15     9   71
## 140  16     9   78
## 141  17     9   67
## 142  18     9   76
## 143  19     9   68
## 144  20     9   82
## 145  21     9   64
## 146  22     9   71
## 147  23     9   81
## 148  24     9   69
```

```
## 149  25     9   63
## 150  26     9   70
## 151  27     9   77
## 152  28     9   75
## 153  29     9   76
## 154  30     9   68
## 155  31     9   NA

####################
#                  #
#    Exercise 4    #
#                  #
####################

back2long <- long2wide %>% gather(Month, Temp, "5", "6", "7",
"8", "9"); back2long

##     Day Month Temp
## 1     1     5   67
## 2     2     5   72
## 3     3     5   74
## 4     4     5   62
## 5     5     5   56
## 6     6     5   66
## 7     7     5   65
## 8     8     5   59
## 9     9     5   61
## 10   10     5   69
## 11   11     5   74
## 12   12     5   69
## 13   13     5   66
## 14   14     5   68
## 15   15     5   58
## 16   16     5   64
## 17   17     5   66
## 18   18     5   57
## 19   19     5   68
## 20   20     5   62
## 21   21     5   59
## 22   22     5   73
## 23   23     5   61
```

```
## 24   24     5   61
## 25   25     5   57
## 26   26     5   58
## 27   27     5   57
## 28   28     5   67
## 29   29     5   81
## 30   30     5   79
## 31   31     5   76
## 32    1     6   78
## 33    2     6   74
## 34    3     6   67
## 35    4     6   84
## 36    5     6   85
## 37    6     6   79
## 38    7     6   82
## 39    8     6   87
## 40    9     6   90
## 41   10     6   87
## 42   11     6   93
## 43   12     6   92
## 44   13     6   82
## 45   14     6   80
## 46   15     6   79
## 47   16     6   77
## 48   17     6   72
## 49   18     6   65
## 50   19     6   73
## 51   20     6   76
## 52   21     6   77
## 53   22     6   76
## 54   23     6   76
## 55   24     6   76
## 56   25     6   75
## 57   26     6   78
## 58   27     6   73
## 59   28     6   80
## 60   29     6   77
## 61   30     6   83
## 62   31     6   NA
## 63    1     7   84
## 64    2     7   85
```

```
## 65    3     7   81
## 66    4     7   84
## 67    5     7   83
## 68    6     7   83
## 69    7     7   88
## 70    8     7   92
## 71    9     7   92
## 72   10     7   89
## 73   11     7   82
## 74   12     7   73
## 75   13     7   81
## 76   14     7   91
## 77   15     7   80
## 78   16     7   81
## 79   17     7   82
## 80   18     7   84
## 81   19     7   87
## 82   20     7   85
## 83   21     7   74
## 84   22     7   81
## 85   23     7   82
## 86   24     7   86
## 87   25     7   85
## 88   26     7   82
## 89   27     7   86
## 90   28     7   88
## 91   29     7   86
## 92   30     7   83
## 93   31     7   81
## 94    1     8   81
## 95    2     8   81
## 96    3     8   82
## 97    4     8   86
## 98    5     8   85
## 99    6     8   87
## 100   7     8   89
## 101   8     8   90
## 102   9     8   90
## 103  10     8   92
## 104  11     8   86
## 105  12     8   86
```

```
## 106  13     8   82
## 107  14     8   80
## 108  15     8   79
## 109  16     8   77
## 110  17     8   79
## 111  18     8   76
## 112  19     8   78
## 113  20     8   78
## 114  21     8   77
## 115  22     8   72
## 116  23     8   75
## 117  24     8   79
## 118  25     8   81
## 119  26     8   86
## 120  27     8   88
## 121  28     8   97
## 122  29     8   94
## 123  30     8   96
## 124  31     8   94
## 125   1     9   91
## 126   2     9   92
## 127   3     9   93
## 128   4     9   93
## 129   5     9   87
## 130   6     9   84
## 131   7     9   80
## 132   8     9   78
## 133   9     9   75
## 134  10     9   73
## 135  11     9   81
## 136  12     9   76
## 137  13     9   77
## 138  14     9   71
## 139  15     9   71
## 140  16     9   78
## 141  17     9   67
## 142  18     9   76
## 143  19     9   68
## 144  20     9   82
## 145  21     9   64
## 146  22     9   71
```

```
## 147  23     9   81
## 148  24     9   69
## 149  25     9   63
## 150  26     9   70
## 151  27     9   77
## 152  28     9   75
## 153  29     9   76
## 154  30     9   68
## 155  31     9   NA

####################
#                  #
#    Exercise 5    #
#                  #
####################

back2long <- long2wide %>% gather(Month, Temp, -Day);
back2long

##     Day Month Temp
## 1     1     5   67
## 2     2     5   72
## 3     3     5   74
## 4     4     5   62
## 5     5     5   56
## 6     6     5   66
## 7     7     5   65
## 8     8     5   59
## 9     9     5   61
## 10   10     5   69
## 11   11     5   74
## 12   12     5   69
## 13   13     5   66
## 14   14     5   68
## 15   15     5   58
## 16   16     5   64
## 17   17     5   66
## 18   18     5   57
## 19   19     5   68
## 20   20     5   62
## 21   21     5   59
```

```
## 22   22     5   73
## 23   23     5   61
## 24   24     5   61
## 25   25     5   57
## 26   26     5   58
## 27   27     5   57
## 28   28     5   67
## 29   29     5   81
## 30   30     5   79
## 31   31     5   76
## 32    1     6   78
## 33    2     6   74
## 34    3     6   67
## 35    4     6   84
## 36    5     6   85
## 37    6     6   79
## 38    7     6   82
## 39    8     6   87
## 40    9     6   90
## 41   10     6   87
## 42   11     6   93
## 43   12     6   92
## 44   13     6   82
## 45   14     6   80
## 46   15     6   79
## 47   16     6   77
## 48   17     6   72
## 49   18     6   65
## 50   19     6   73
## 51   20     6   76
## 52   21     6   77
## 53   22     6   76
## 54   23     6   76
## 55   24     6   76
## 56   25     6   75
## 57   26     6   78
## 58   27     6   73
## 59   28     6   80
## 60   29     6   77
## 61   30     6   83
## 62   31     6   NA
```

```
## 63    1     7   84
## 64    2     7   85
## 65    3     7   81
## 66    4     7   84
## 67    5     7   83
## 68    6     7   83
## 69    7     7   88
## 70    8     7   92
## 71    9     7   92
## 72   10     7   89
## 73   11     7   82
## 74   12     7   73
## 75   13     7   81
## 76   14     7   91
## 77   15     7   80
## 78   16     7   81
## 79   17     7   82
## 80   18     7   84
## 81   19     7   87
## 82   20     7   85
## 83   21     7   74
## 84   22     7   81
## 85   23     7   82
## 86   24     7   86
## 87   25     7   85
## 88   26     7   82
## 89   27     7   86
## 90   28     7   88
## 91   29     7   86
## 92   30     7   83
## 93   31     7   81
## 94    1     8   81
## 95    2     8   81
## 96    3     8   82
## 97    4     8   86
## 98    5     8   85
## 99    6     8   87
## 100   7     8   89
## 101   8     8   90
## 102   9     8   90
## 103  10     8   92
```

```
## 104  11     8   86
## 105  12     8   86
## 106  13     8   82
## 107  14     8   80
## 108  15     8   79
## 109  16     8   77
## 110  17     8   79
## 111  18     8   76
## 112  19     8   78
## 113  20     8   78
## 114  21     8   77
## 115  22     8   72
## 116  23     8   75
## 117  24     8   79
## 118  25     8   81
## 119  26     8   86
## 120  27     8   88
## 121  28     8   97
## 122  29     8   94
## 123  30     8   96
## 124  31     8   94
## 125   1     9   91
## 126   2     9   92
## 127   3     9   93
## 128   4     9   93
## 129   5     9   87
## 130   6     9   84
## 131   7     9   80
## 132   8     9   78
## 133   9     9   75
## 134  10     9   73
## 135  11     9   81
## 136  12     9   76
## 137  13     9   77
## 138  14     9   71
## 139  15     9   71
## 140  16     9   78
## 141  17     9   67
## 142  18     9   76
## 143  19     9   68
## 144  20     9   82
```

```
## 145  21     9   64
## 146  22     9   71
## 147  23     9   81
## 148  24     9   69
## 149  25     9   63
## 150  26     9   70
## 151  27     9   77
## 152  28     9   75
## 153  29     9   76
## 154  30     9   68
## 155  31     9   NA

####################
#                  #
#    Exercise 6    #
#                  #
####################

back2long_unite <- back2long %>% unite(col = "Date", c(Day,
Month), sep = "-"); back2long_unite

##     Date Temp
## 1    1-5   67
## 2    2-5   72
## 3    3-5   74
## 4    4-5   62
## 5    5-5   56
## 6    6-5   66
## 7    7-5   65
## 8    8-5   59
## 9    9-5   61
## 10  10-5   69
## 11  11-5   74
## 12  12-5   69
## 13  13-5   66
## 14  14-5   68
## 15  15-5   58
## 16  16-5   64
## 17  17-5   66
## 18  18-5   57
## 19  19-5   68
```

```
## 20  20-5   62
## 21  21-5   59
## 22  22-5   73
## 23  23-5   61
## 24  24-5   61
## 25  25-5   57
## 26  26-5   58
## 27  27-5   57
## 28  28-5   67
## 29  29-5   81
## 30  30-5   79
## 31  31-5   76
## 32   1-6   78
## 33   2-6   74
## 34   3-6   67
## 35   4-6   84
## 36   5-6   85
## 37   6-6   79
## 38   7-6   82
## 39   8-6   87
## 40   9-6   90
## 41  10-6   87
## 42  11-6   93
## 43  12-6   92
## 44  13-6   82
## 45  14-6   80
## 46  15-6   79
## 47  16-6   77
## 48  17-6   72
## 49  18-6   65
## 50  19-6   73
## 51  20-6   76
## 52  21-6   77
## 53  22-6   76
## 54  23-6   76
## 55  24-6   76
## 56  25-6   75
## 57  26-6   78
## 58  27-6   73
## 59  28-6   80
## 60  29-6   77
```

```
## 61  30-6   83
## 62  31-6   NA
## 63   1-7   84
## 64   2-7   85
## 65   3-7   81
## 66   4-7   84
## 67   5-7   83
## 68   6-7   83
## 69   7-7   88
## 70   8-7   92
## 71   9-7   92
## 72  10-7   89
## 73  11-7   82
## 74  12-7   73
## 75  13-7   81
## 76  14-7   91
## 77  15-7   80
## 78  16-7   81
## 79  17-7   82
## 80  18-7   84
## 81  19-7   87
## 82  20-7   85
## 83  21-7   74
## 84  22-7   81
## 85  23-7   82
## 86  24-7   86
## 87  25-7   85
## 88  26-7   82
## 89  27-7   86
## 90  28-7   88
## 91  29-7   86
## 92  30-7   83
## 93  31-7   81
## 94   1-8   81
## 95   2-8   81
## 96   3-8   82
## 97   4-8   86
## 98   5-8   85
## 99   6-8   87
## 100  7-8   89
## 101  8-8   90
```

```
## 102  9-8   90
## 103 10-8   92
## 104 11-8   86
## 105 12-8   86
## 106 13-8   82
## 107 14-8   80
## 108 15-8   79
## 109 16-8   77
## 110 17-8   79
## 111 18-8   76
## 112 19-8   78
## 113 20-8   78
## 114 21-8   77
## 115 22-8   72
## 116 23-8   75
## 117 24-8   79
## 118 25-8   81
## 119 26-8   86
## 120 27-8   88
## 121 28-8   97
## 122 29-8   94
## 123 30-8   96
## 124 31-8   94
## 125  1-9   91
## 126  2-9   92
## 127  3-9   93
## 128  4-9   93
## 129  5-9   87
## 130  6-9   84
## 131  7-9   80
## 132  8-9   78
## 133  9-9   75
## 134 10-9   73
## 135 11-9   81
## 136 12-9   76
## 137 13-9   77
## 138 14-9   71
## 139 15-9   71
## 140 16-9   78
## 141 17-9   67
## 142 18-9   76
```

```
## 143 19-9   68
## 144 20-9   82
## 145 21-9   64
## 146 22-9   71
## 147 23-9   81
## 148 24-9   69
## 149 25-9   63
## 150 26-9   70
## 151 27-9   77
## 152 28-9   75
## 153 29-9   76
## 154 30-9   68
## 155 31-9   NA

####################
#                  #
#    Exercise 7    #
#                  #
####################

back2long_separate <- back2long_unite %>% separate(col = Date,
into = c("Day", "Month")) ; back2long_separate

##     Day Month Temp
## 1     1     5   67
## 2     2     5   72
## 3     3     5   74
## 4     4     5   62
## 5     5     5   56
## 6     6     5   66
## 7     7     5   65
## 8     8     5   59
## 9     9     5   61
## 10   10     5   69
## 11   11     5   74
## 12   12     5   69
## 13   13     5   66
## 14   14     5   68
## 15   15     5   58
## 16   16     5   64
## 17   17     5   66
```

```
## 18   18     5   57
## 19   19     5   68
## 20   20     5   62
## 21   21     5   59
## 22   22     5   73
## 23   23     5   61
## 24   24     5   61
## 25   25     5   57
## 26   26     5   58
## 27   27     5   57
## 28   28     5   67
## 29   29     5   81
## 30   30     5   79
## 31   31     5   76
## 32    1     6   78
## 33    2     6   74
## 34    3     6   67
## 35    4     6   84
## 36    5     6   85
## 37    6     6   79
## 38    7     6   82
## 39    8     6   87
## 40    9     6   90
## 41   10     6   87
## 42   11     6   93
## 43   12     6   92
## 44   13     6   82
## 45   14     6   80
## 46   15     6   79
## 47   16     6   77
## 48   17     6   72
## 49   18     6   65
## 50   19     6   73
## 51   20     6   76
## 52   21     6   77
## 53   22     6   76
## 54   23     6   76
## 55   24     6   76
## 56   25     6   75
## 57   26     6   78
## 58   27     6   73
```

```
## 59   28     6   80
## 60   29     6   77
## 61   30     6   83
## 62   31     6   NA
## 63    1     7   84
## 64    2     7   85
## 65    3     7   81
## 66    4     7   84
## 67    5     7   83
## 68    6     7   83
## 69    7     7   88
## 70    8     7   92
## 71    9     7   92
## 72   10     7   89
## 73   11     7   82
## 74   12     7   73
## 75   13     7   81
## 76   14     7   91
## 77   15     7   80
## 78   16     7   81
## 79   17     7   82
## 80   18     7   84
## 81   19     7   87
## 82   20     7   85
## 83   21     7   74
## 84   22     7   81
## 85   23     7   82
## 86   24     7   86
## 87   25     7   85
## 88   26     7   82
## 89   27     7   86
## 90   28     7   88
## 91   29     7   86
## 92   30     7   83
## 93   31     7   81
## 94    1     8   81
## 95    2     8   81
## 96    3     8   82
## 97    4     8   86
## 98    5     8   85
## 99    6     8   87
```

```
## 100   7     8   89
## 101   8     8   90
## 102   9     8   90
## 103  10     8   92
## 104  11     8   86
## 105  12     8   86
## 106  13     8   82
## 107  14     8   80
## 108  15     8   79
## 109  16     8   77
## 110  17     8   79
## 111  18     8   76
## 112  19     8   78
## 113  20     8   78
## 114  21     8   77
## 115  22     8   72
## 116  23     8   75
## 117  24     8   79
## 118  25     8   81
## 119  26     8   86
## 120  27     8   88
## 121  28     8   97
## 122  29     8   94
## 123  30     8   96
## 124  31     8   94
## 125   1     9   91
## 126   2     9   92
## 127   3     9   93
## 128   4     9   93
## 129   5     9   87
## 130   6     9   84
## 131   7     9   80
## 132   8     9   78
## 133   9     9   75
## 134  10     9   73
## 135  11     9   81
## 136  12     9   76
## 137  13     9   77
## 138  14     9   71
## 139  15     9   71
## 140  16     9   78
```

```
## 141  17     9   67
## 142  18     9   76
## 143  19     9   68
## 144  20     9   82
## 145  21     9   64
## 146  22     9   71
## 147  23     9   81
## 148  24     9   69
## 149  25     9   63
## 150  26     9   70
## 151  27     9   77
## 152  28     9   75
## 153  29     9   76
## 154  30     9   68
## 155  31     9   NA

####################
#                  #
#    Exercise 8    #
#                  #
####################

back2long_na <- back2long %>% replace_na(replace = list(Temp =
"unknown")) ; back2long_na

##     Day Month    Temp
## 1     1     5      67
## 2     2     5      72
## 3     3     5      74
## 4     4     5      62
## 5     5     5      56
## 6     6     5      66
## 7     7     5      65
## 8     8     5      59
## 9     9     5      61
## 10   10     5      69
## 11   11     5      74
## 12   12     5      69
## 13   13     5      66
## 14   14     5      68
## 15   15     5      58
```

```
## 16   16     5      64
## 17   17     5      66
## 18   18     5      57
## 19   19     5      68
## 20   20     5      62
## 21   21     5      59
## 22   22     5      73
## 23   23     5      61
## 24   24     5      61
## 25   25     5      57
## 26   26     5      58
## 27   27     5      57
## 28   28     5      67
## 29   29     5      81
## 30   30     5      79
## 31   31     5      76
## 32    1     6      78
## 33    2     6      74
## 34    3     6      67
## 35    4     6      84
## 36    5     6      85
## 37    6     6      79
## 38    7     6      82
## 39    8     6      87
## 40    9     6      90
## 41   10     6      87
## 42   11     6      93
## 43   12     6      92
## 44   13     6      82
## 45   14     6      80
## 46   15     6      79
## 47   16     6      77
## 48   17     6      72
## 49   18     6      65
## 50   19     6      73
## 51   20     6      76
## 52   21     6      77
## 53   22     6      76
## 54   23     6      76
## 55   24     6      76
## 56   25     6      75
```

```
## 57   26     6      78
## 58   27     6      73
## 59   28     6      80
## 60   29     6      77
## 61   30     6      83
## 62   31     6 unknown
## 63    1     7      84
## 64    2     7      85
## 65    3     7      81
## 66    4     7      84
## 67    5     7      83
## 68    6     7      83
## 69    7     7      88
## 70    8     7      92
## 71    9     7      92
## 72   10     7      89
## 73   11     7      82
## 74   12     7      73
## 75   13     7      81
## 76   14     7      91
## 77   15     7      80
## 78   16     7      81
## 79   17     7      82
## 80   18     7      84
## 81   19     7      87
## 82   20     7      85
## 83   21     7      74
## 84   22     7      81
## 85   23     7      82
## 86   24     7      86
## 87   25     7      85
## 88   26     7      82
## 89   27     7      86
## 90   28     7      88
## 91   29     7      86
## 92   30     7      83
## 93   31     7      81
## 94    1     8      81
## 95    2     8      81
## 96    3     8      82
## 97    4     8      86
```

```
## 98    5     8      85
## 99    6     8      87
## 100   7     8      89
## 101   8     8      90
## 102   9     8      90
## 103  10     8      92
## 104  11     8      86
## 105  12     8      86
## 106  13     8      82
## 107  14     8      80
## 108  15     8      79
## 109  16     8      77
## 110  17     8      79
## 111  18     8      76
## 112  19     8      78
## 113  20     8      78
## 114  21     8      77
## 115  22     8      72
## 116  23     8      75
## 117  24     8      79
## 118  25     8      81
## 119  26     8      86
## 120  27     8      88
## 121  28     8      97
## 122  29     8      94
## 123  30     8      96
## 124  31     8      94
## 125   1     9      91
## 126   2     9      92
## 127   3     9      93
## 128   4     9      93
## 129   5     9      87
## 130   6     9      84
## 131   7     9      80
## 132   8     9      78
## 133   9     9      75
## 134  10     9      73
## 135  11     9      81
## 136  12     9      76
## 137  13     9      77
## 138  14     9      71
```

```
## 139  15     9      71
## 140  16     9      78
## 141  17     9      67
## 142  18     9      76
## 143  19     9      68
## 144  20     9      82
## 145  21     9      64
## 146  22     9      71
## 147  23     9      81
## 148  24     9      69
## 149  25     9      63
## 150  26     9      70
## 151  27     9      77
## 152  28     9      75
## 153  29     9      76
## 154  30     9      68
## 155  31     9 unknown

####################
#                  #
#    Exercise 9    #
#                  #
####################

back2long_na <- back2long_na %>% fill(year) ; back2long_na

##     Day Month    Temp year
## 1     1     5      67 2015
## 2     2     5      72 2015
## 3     3     5      74 2015
## 4     4     5      62 2015
## 5     5     5      56 2015
## 6     6     5      66 2015
## 7     7     5      65 2015
## 8     8     5      59 2015
## 9     9     5      61 2015
## 10   10     5      69 2015
## 11   11     5      74 2015
## 12   12     5      69 2015
## 13   13     5      66 2015
## 14   14     5      68 2015
```

```
## 15   15     5      58 2015
## 16   16     5      64 2015
## 17   17     5      66 2015
## 18   18     5      57 2015
## 19   19     5      68 2015
## 20   20     5      62 2015
## 21   21     5      59 2015
## 22   22     5      73 2015
## 23   23     5      61 2015
## 24   24     5      61 2015
## 25   25     5      57 2015
## 26   26     5      58 2015
## 27   27     5      57 2015
## 28   28     5      67 2015
## 29   29     5      81 2015
## 30   30     5      79 2015
## 31   31     5      76 2015
## 32    1     6      78 2015
## 33    2     6      74 2015
## 34    3     6      67 2015
## 35    4     6      84 2015
## 36    5     6      85 2015
## 37    6     6      79 2015
## 38    7     6      82 2015
## 39    8     6      87 2015
## 40    9     6      90 2015
## 41   10     6      87 2015
## 42   11     6      93 2015
## 43   12     6      92 2015
## 44   13     6      82 2015
## 45   14     6      80 2015
## 46   15     6      79 2015
## 47   16     6      77 2015
## 48   17     6      72 2015
## 49   18     6      65 2015
## 50   19     6      73 2015
## 51   20     6      76 2016
## 52   21     6      77 2016
## 53   22     6      76 2016
## 54   23     6      76 2016
## 55   24     6      76 2016
```

```
## 56   25     6      75 2016
## 57   26     6      78 2016
## 58   27     6      73 2016
## 59   28     6      80 2016
## 60   29     6      77 2016
## 61   30     6      83 2016
## 62   31     6 unknown 2016
## 63    1     7      84 2016
## 64    2     7      85 2016
## 65    3     7      81 2016
## 66    4     7      84 2016
## 67    5     7      83 2016
## 68    6     7      83 2016
## 69    7     7      88 2016
## 70    8     7      92 2016
## 71    9     7      92 2016
## 72   10     7      89 2016
## 73   11     7      82 2016
## 74   12     7      73 2016
## 75   13     7      81 2016
## 76   14     7      91 2016
## 77   15     7      80 2016
## 78   16     7      81 2016
## 79   17     7      82 2016
## 80   18     7      84 2016
## 81   19     7      87 2016
## 82   20     7      85 2016
## 83   21     7      74 2016
## 84   22     7      81 2016
## 85   23     7      82 2016
## 86   24     7      86 2016
## 87   25     7      85 2016
## 88   26     7      82 2016
## 89   27     7      86 2016
## 90   28     7      88 2016
## 91   29     7      86 2016
## 92   30     7      83 2016
## 93   31     7      81 2016
## 94    1     8      81 2016
## 95    2     8      81 2016
## 96    3     8      82 2016
```

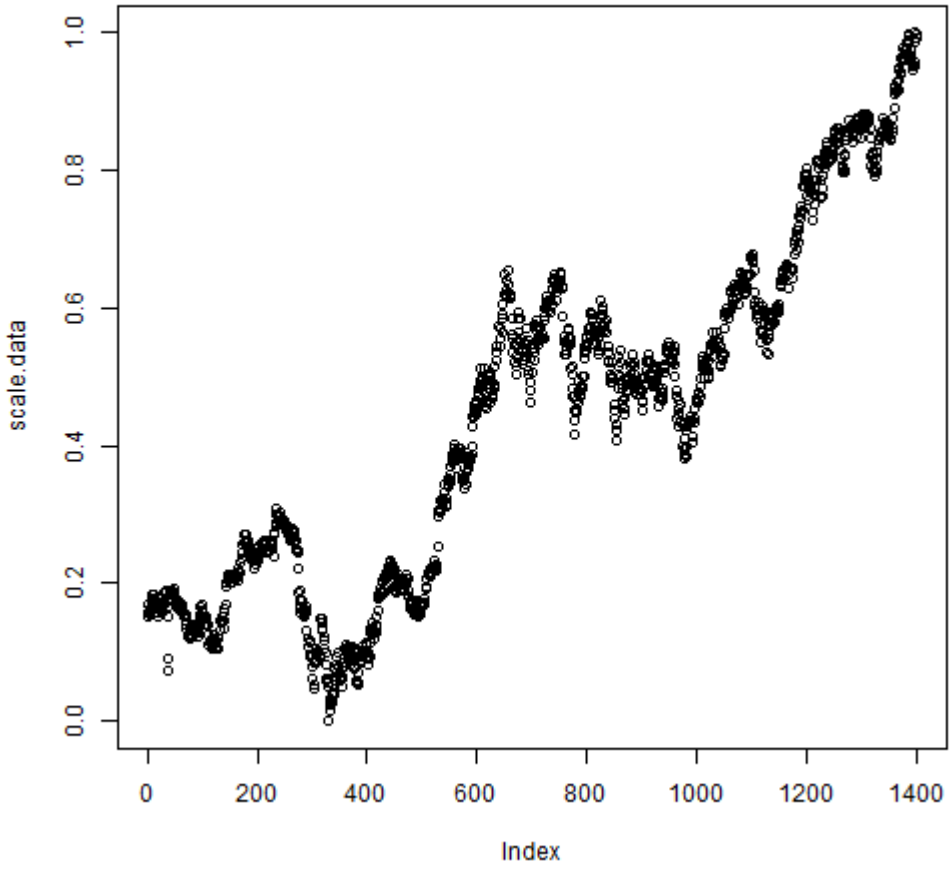
```
## 97    4     8      86 2016
## 98    5     8      85 2016
## 99    6     8      87 2016
## 100   7     8      89 2016
## 101   8     8      90 2016
## 102   9     8      90 2016
## 103  10     8      92 2017
## 104  11     8      86 2017
## 105  12     8      86 2017
## 106  13     8      82 2017
## 107  14     8      80 2017
## 108  15     8      79 2017
## 109  16     8      77 2017
## 110  17     8      79 2017
## 111  18     8      76 2017
## 112  19     8      78 2017
## 113  20     8      78 2017
## 114  21     8      77 2017
## 115  22     8      72 2017
## 116  23     8      75 2017
## 117  24     8      79 2017
## 118  25     8      81 2017
## 119  26     8      86 2017
## 120  27     8      88 2017
## 121  28     8      97 2017
## 122  29     8      94 2017
## 123  30     8      96 2017
## 124  31     8      94 2017
## 125   1     9      91 2017
## 126   2     9      92 2017
## 127   3     9      93 2017
## 128   4     9      93 2017
## 129   5     9      87 2017
## 130   6     9      84 2017
## 131   7     9      80 2017
## 132   8     9      78 2017
## 133   9     9      75 2017
## 134  10     9      73 2017
## 135  11     9      81 2017
## 136  12     9      76 2017
## 137  13     9      77 2017
```

```
## 138  14     9      71 2017
## 139  15     9      71 2017
## 140  16     9      78 2017
## 141  17     9      67 2017
## 142  18     9      76 2017
## 143  19     9      68 2017
## 144  20     9      82 2017
## 145  21     9      64 2017
## 146  22     9      71 2017
## 147  23     9      81 2017
## 148  24     9      69 2017
## 149  25     9      63 2017
## 150  26     9      70 2017
## 151  27     9      77 2017
## 152  28     9      75 2017
## 153  29     9      76 2017
## 154  30     9      68 2017
## 155  31     9 unknown 2017

####################
#                  #
#    Exercise 10   #
#                  #
####################

back2long_na %$% extract_numeric(Temp)

##   [1] 67 72 74 62 56 66 65 59 61 69 74 69 66 68 58 64 66 57
68 62 59 73 61
##  [24] 61 57 58 57 67 81 79 76 78 74 67 84 85 79 82 87 90 87
93 92 82 80 79
##  [47] 77 72 65 73 76 77 76 76 76 75 78 73 80 77 83 NA 84 85
81 84 83 83 88
##  [70] 92 92 89 82 73 81 91 80 81 82 84 87 85 74 81 82 86 85
82 86 88 86 83
##  [93] 81 81 81 82 86 85 87 89 90 90 92 86 86 82 80 79 77 79
76 78 78 77 72
## [116] 75 79 81 86 88 97 94 96 94 91 92 93 93 87 84 80 78 75
73 81 76 77 71
## [139] 71 78 67 76 68 82 64 71 81 69 63 70 77 75 76 68 NA
```

Neural networks Solutions (Part-3)

Below are the solutions to these exercises on neural networks.

```
####################
#                  #
#    Exercise 1    #
#                  #
####################
library(datasets)
str(EuStockMarkets)

##  Time-Series [1:1860, 1:4] from 1991 to 1999: 1629 1614
1607 1621 1618 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : NULL
##   ..$ : chr [1:4] "DAX" "SMI" "CAC" "FTSE"

data<-EuStockMarkets[,1]
df.data<-as.data.frame(t(matrix(data, 1)))

####################
#                  #
#    Exercise 2    #
#                  #
####################
scale.0.1<-function(x){
     return   ((x-min(x,na.rm=TRUE))/(max(x,na.rm=TRUE)-
min(x,na.rm=TRUE)))
}

scale.data<-scale.0.1(df.data[1:1400,])
plot(scale.data)
```

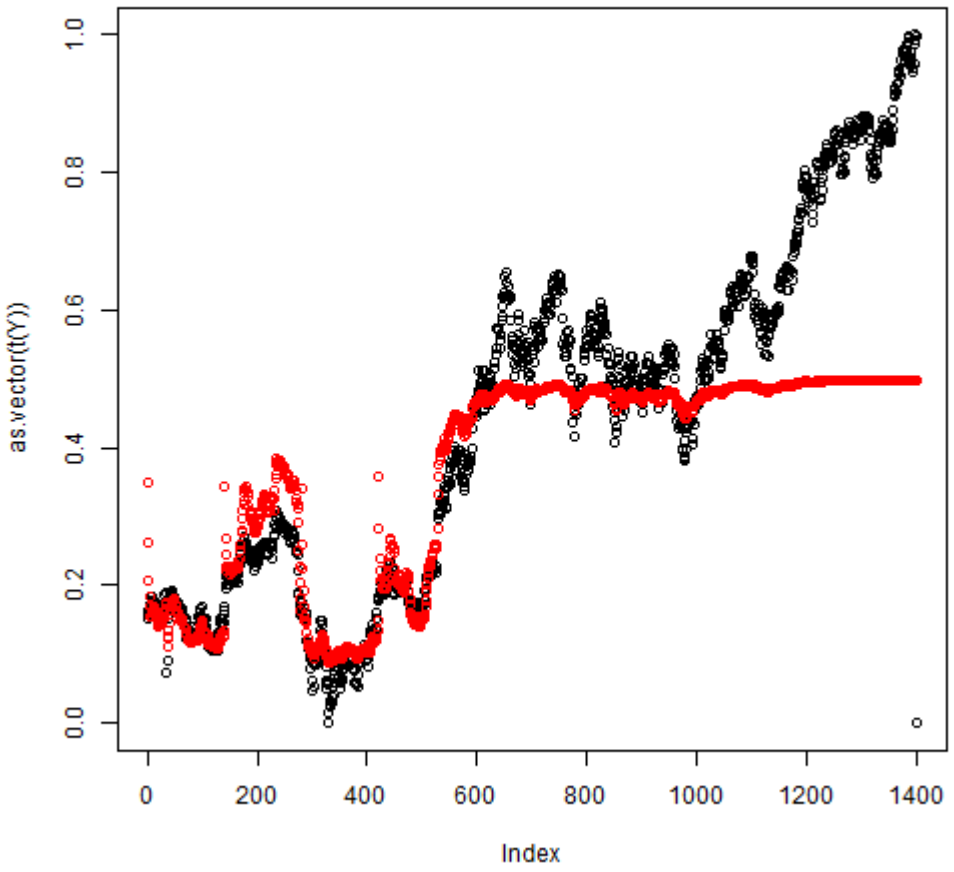



```
sum(is.na(scale.data))
```

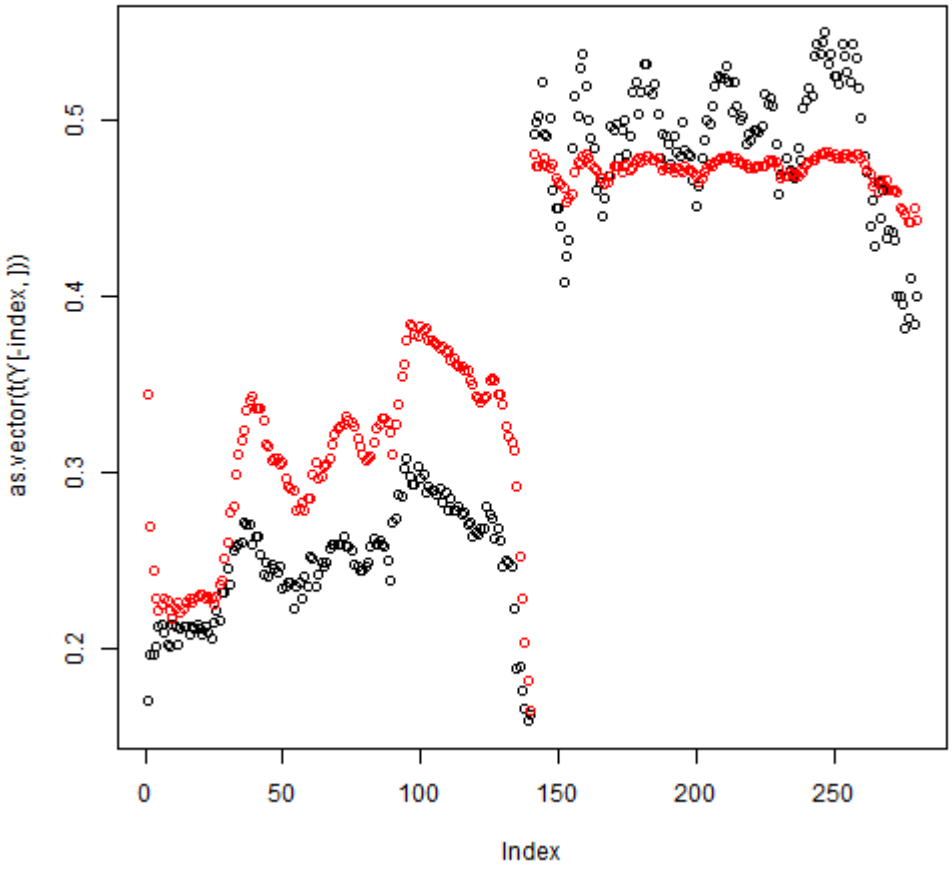
```
## [1] 0
```

```
#####################
#                   #
#    Exercise 3     #
#                   #
#####################
#10 samples of 140 observations
X<-matrix(scale.data[1:1400],nrow=140)
Y<-matrix(c(scale.data[2:1400],),nrow=140)

X <- t(X)
Y <- t(Y)

#####################
#                   #
#    Exercise 4     #
```

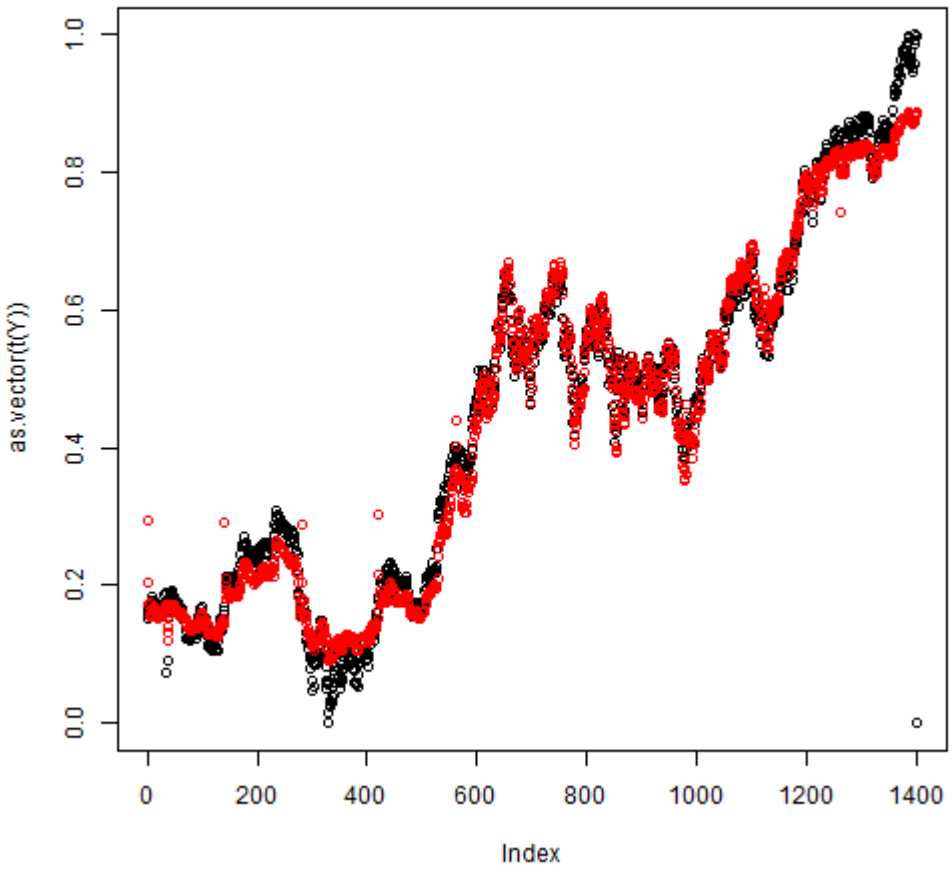
```
#                    #
####################
library(rnn)
set.seed(42)
index<-sample(1:10,8,replace=FALSE)
model.stock.1hl<-trainr(Y[index,],X[index,],learningrate   =
0.01, hidden_dim = 1,numepochs = 500)

####################
#                    #
#    Exercise 5    #
#                    #
####################
pred.1hl <- predictr(model.stock.1hl, X)

plot(as.vector(t(Y)))
points(as.vector(t(pred.1hl)),col='red')
```



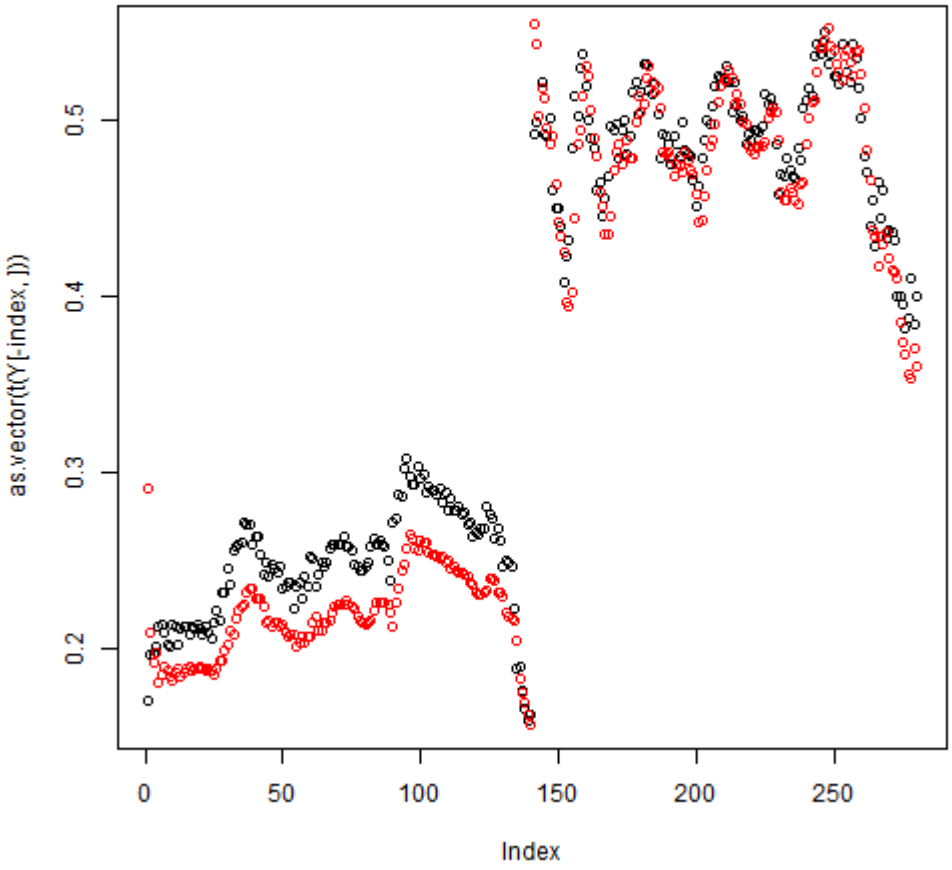

```
plot(as.vector(t(Y[-index,])))
points(as.vector(t(pred.1hl[-index,])),col='red')
```




```
#####################
#                   #
#    Exercise 6     #
#                   #
#####################
model.stock.10hl<-trainr(Y[index,],X[index,],learningrate  =
0.01, hidden_dim = 10,numepochs = 500)

pred.10hl <- predictr(model.stock.10hl, X)

plot(as.vector(t(Y)))
points(as.vector(t(pred.10hl)),col='red')
```



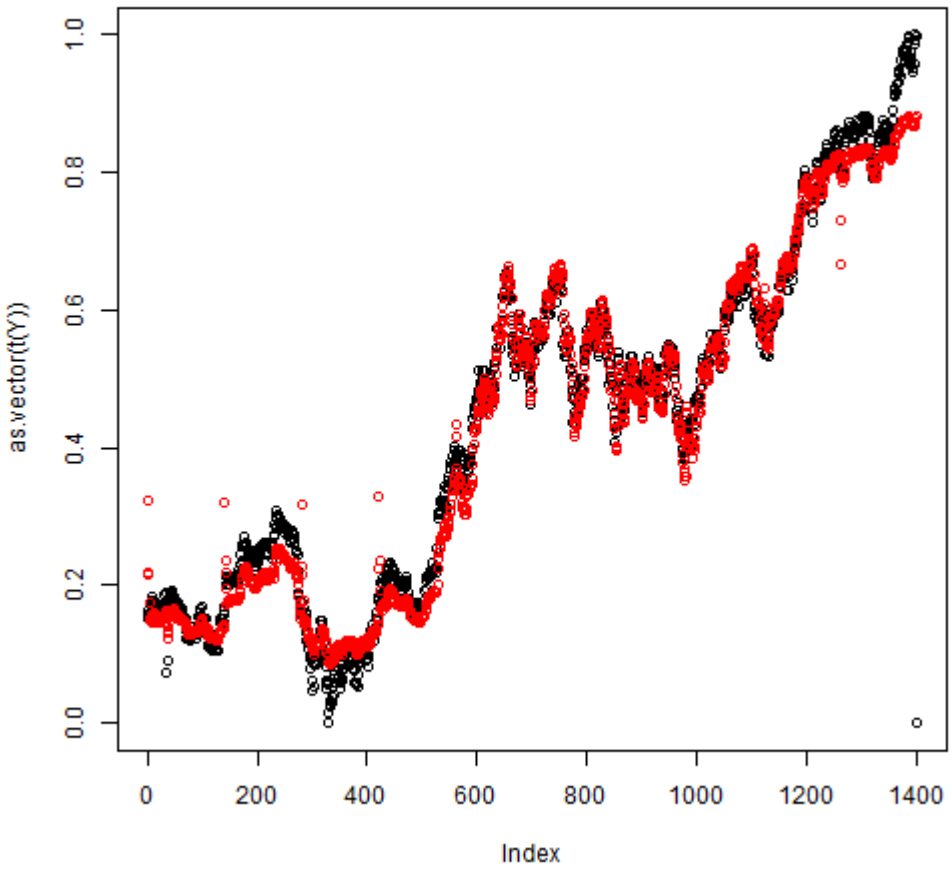

```
plot(as.vector(t(Y[-index,])))
points(as.vector(t(pred.10hl[-index,])),col='red')
```



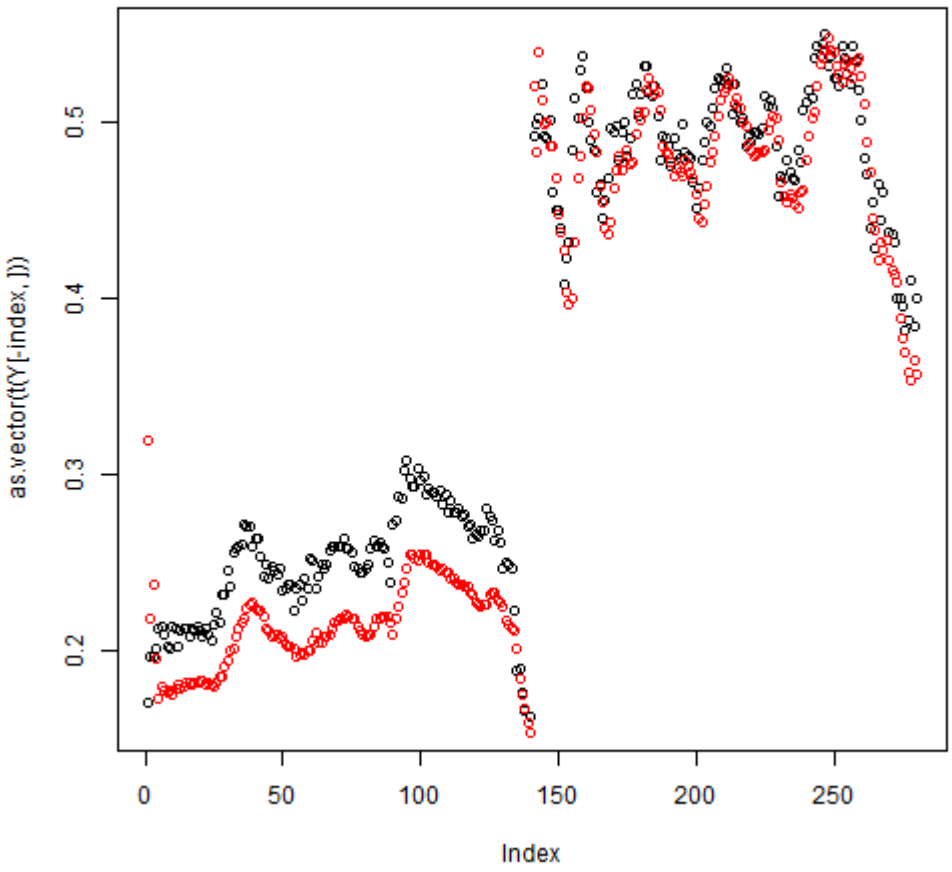

```
rmse.10.hl<-sqrt((sum(Y[-index,]-pred.10hl[-
index,])^2)/(nrow(Y[-index,])*ncol(Y[-index,])))
rmse.10.hl
```

```
## [1] 0.3016284
```

```
#####################
#                   #
#    Exercise 7     #
#                   #
#####################
model.stock.momentum<-trainr(Y[index,],X[index,],learningrate
= 0.01, hidden_dim = 20,momentum=0.7, numepochs = 250)

pred.momentum <- predictr(model.stock.momentum, X)

plot(as.vector(t(Y)))
points(as.vector(t(pred.momentum)),col='red')
```



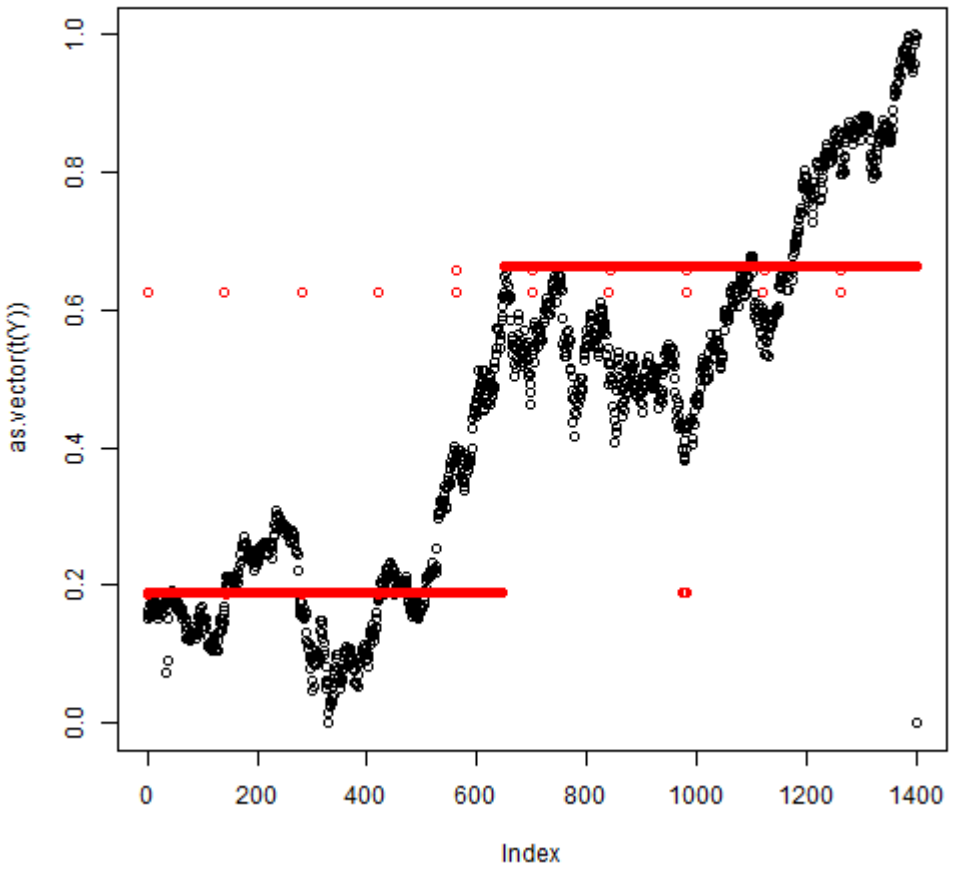

```
plot(as.vector(t(Y[-index,])))
points(as.vector(t(pred.momentum[-index,])),col='red')
```



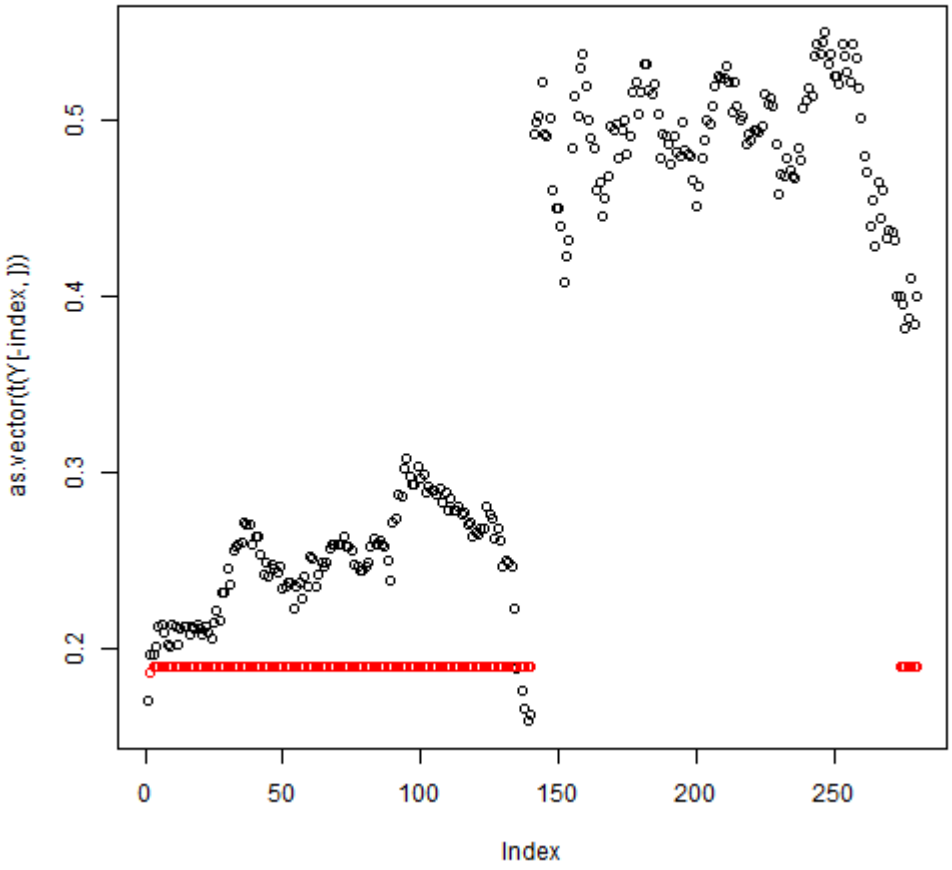

```
rmse.momentum<-sqrt((sum(Y[-index,]-pred.momentum[-
index,])^2)/(nrow(Y[-index,])*ncol(Y[-index,])))
rmse.momentum
```

```
## [1] 0.365263
```

```
#####################
#                   #
#    Exercise 8     #
#                   #
#####################
model.stock.LSTM<-trainr(Y[index,],X[index,],learningrate   =
0.01, hidden_dim = 20,network_type = "lstm", numepochs = 500)

pred.LSTM <- predictr(model.stock.LSTM, X)

plot(as.vector(t(Y)))
points(as.vector(t(pred.LSTM)),col='red')
```

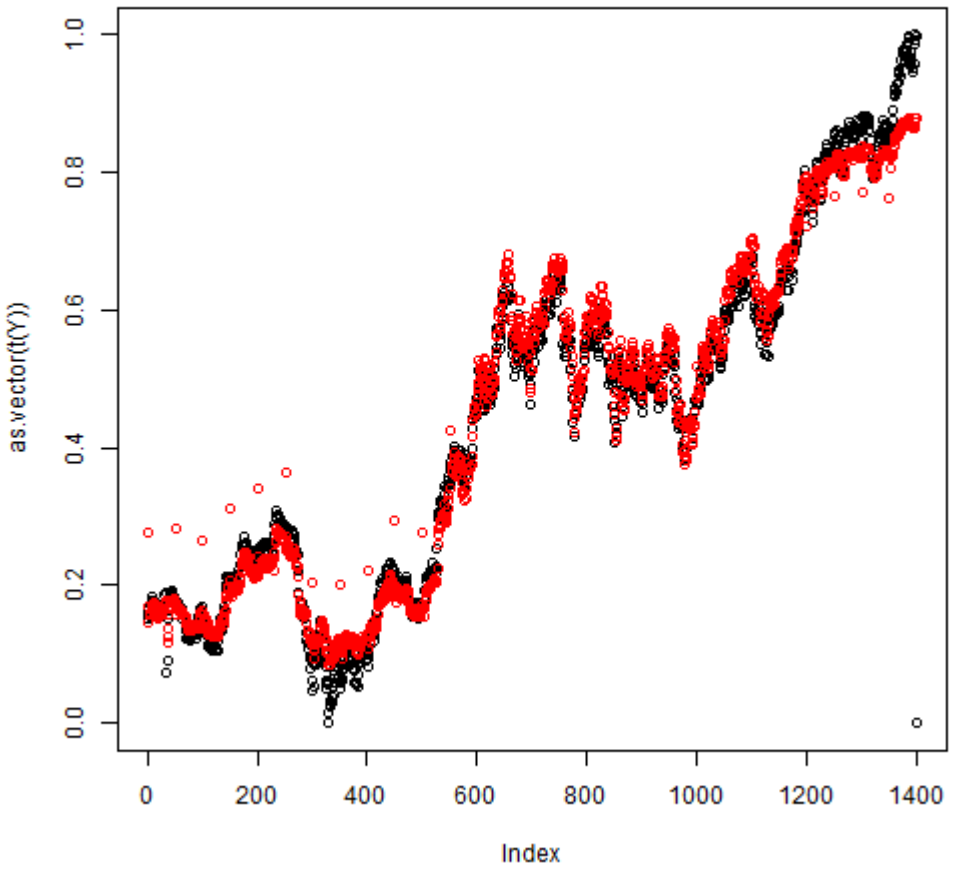



```
plot(as.vector(t(Y[-index,])))
points(as.vector(t(pred.LSTM[-index,])),col='red')
```

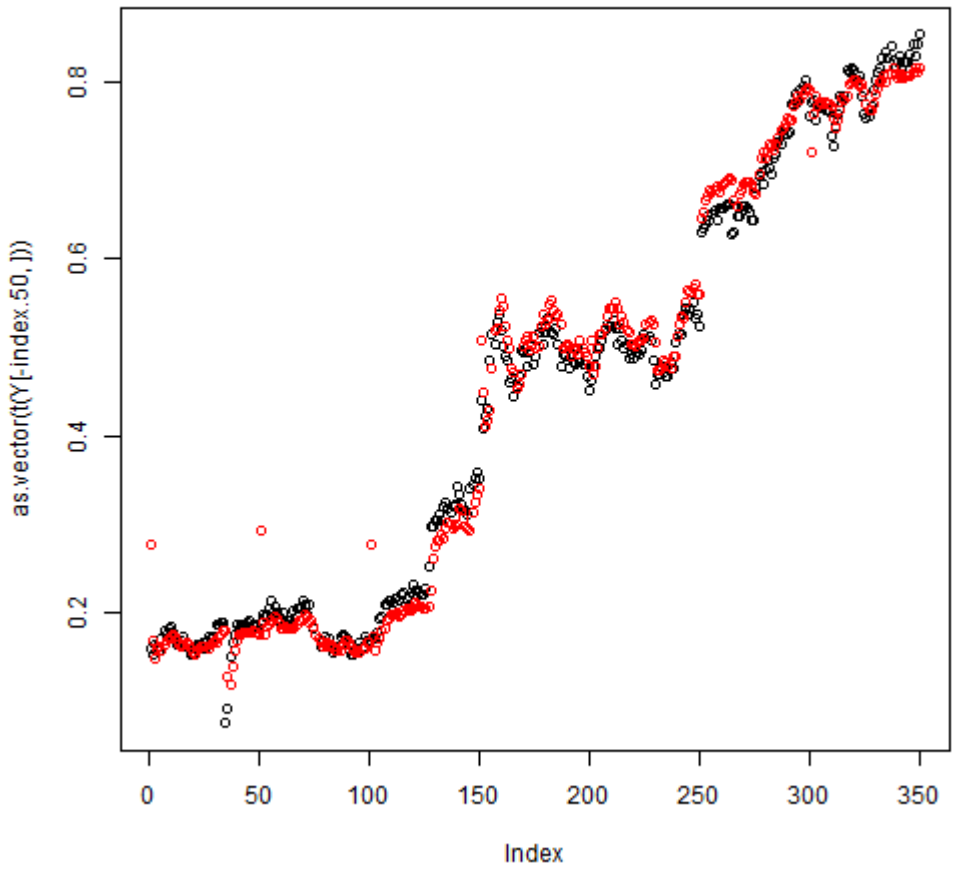



```
rmse.LSTM<-sqrt((sum(Y[-index,]-pred.LSTM[-
index,])^2)/(nrow(Y[-index,])*ncol(Y[-index,])))
rmse.LSTM

## [1] 0.8217892

#####################
#                   #
#    Exercise 9     #
#                   #
#####################
#28 samples of 50 observations
X<-matrix(scale.data[1:1400],nrow=50)
Y<-matrix(c(scale.data[2:1400],),nrow=50)

X <- t(X)
Y <- t(Y)

set.seed(42)
```

```
index.50<-sample(1:28,21,replace=FALSE)

model.stock.50<-trainr(Y[index.50,],X[index.50,],learningrate
= 0.01, hidden_dim = 10,numepochs = 500)

pred.50 <- predictr(model.stock.50, X)

plot(as.vector(t(Y)))
points(as.vector(t(pred.50)),col='red')
```

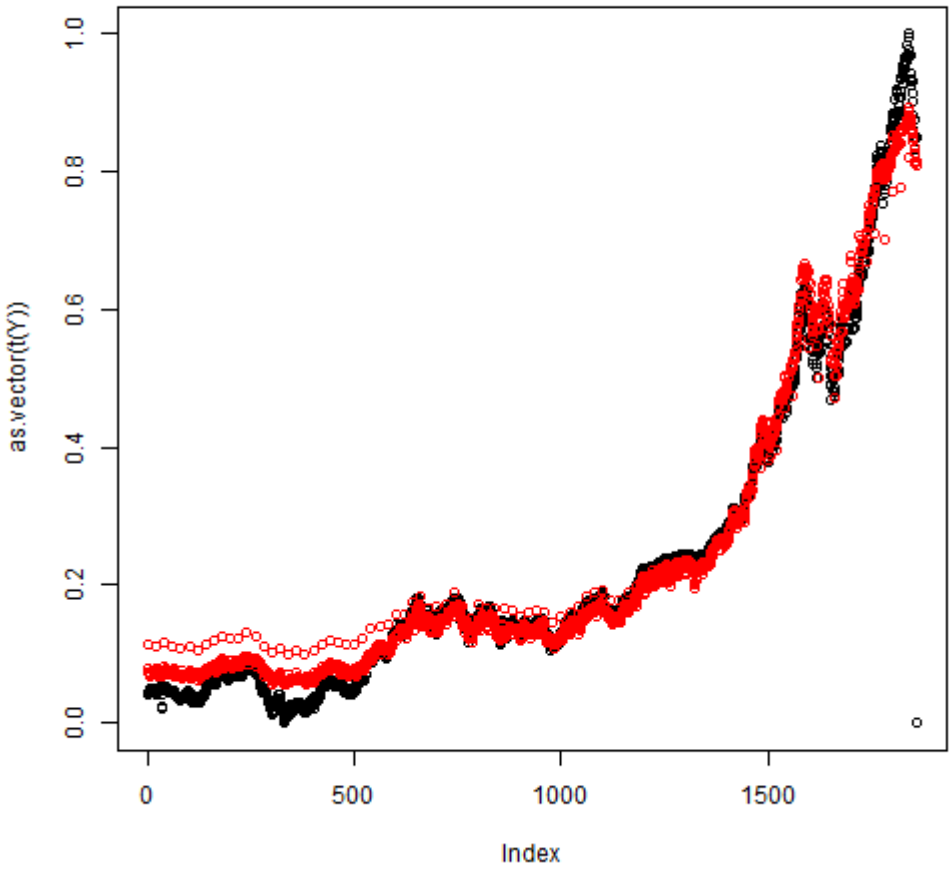



```
plot(as.vector(t(Y[-index.50,])))
points(as.vector(t(pred.50[-index.50,])),col='red')
```

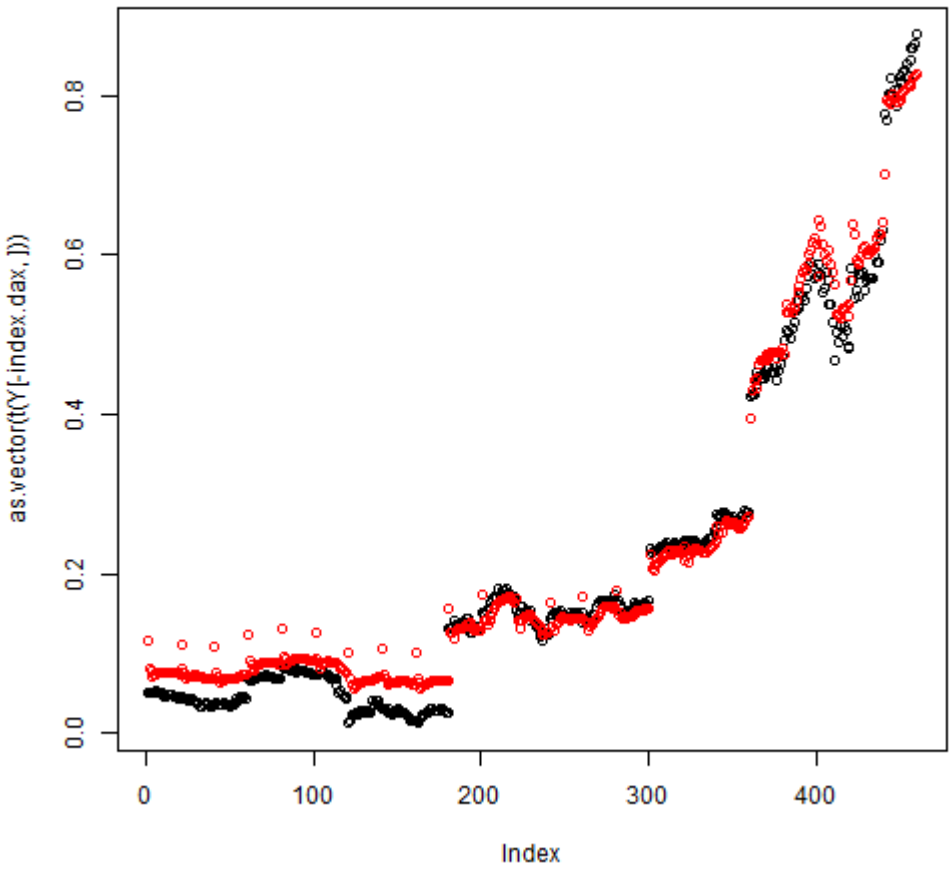



```
rmse.50<-sqrt((sum(Y[-index.50,]-pred.50[-
index.50,])^2)/(nrow(Y[-index.50,])*ncol(Y[-index.50,])))
rmse.50

## [1] 0.02927242

#####################
#                   #
#    Exercise 10    #
#                   #
#####################
dax.data<-scale.0.1(df.data)

#93 samples of 20 observations
X<-matrix(dax.data[1:1860,],nrow=20)
Y<-matrix(c(dax.data[2:1860,],),nrow=20)

X <- t(X)
Y <- t(Y)
```

```
set.seed(42)
index.dax<-sample(1:93,70,replace=FALSE)

model.dax<-trainr(Y[index.dax,],X[index.dax,],learningrate  =
0.01, hidden_dim = c(10,10),momentum=0.1,numepochs = 500)

pred.dax <- predictr(model.dax, X)

plot(as.vector(t(Y)))
points(as.vector(t(pred.dax)),col='red')
```




```
plot(as.vector(t(Y[-index.dax,])))
points(as.vector(t(pred.dax[-index.dax,])),col='red')
```




```
rmse.dax<-sqrt((sum(Y[-index.dax,]-pred.dax[-index.dax,])^2)/(nrow(Y[-index.dax,])*ncol(Y[-index.dax,])))
rmse.dax
```

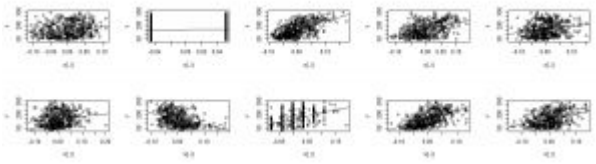
```
## [1] 0.2719791
```

Ridge regression in R solutions

Below are the solutions to these exercises on ridge regression.

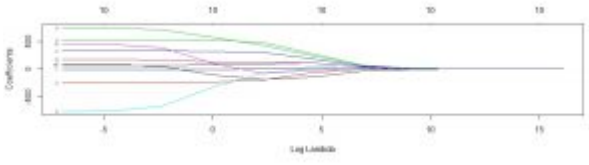
```
################
```

```
#             #
# Exercise 1  #
#             #
###############

library(lars)
library(glmnet)

## Warning: package 'glmnet' was built under R version 3.3.3

## Warning: package 'foreach' was built under R version 3.3.3

data(diabetes)
attach(diabetes)
set.seed(1234)

par(mfrow=c(2,5))
for(i in 1:10){
  plot(x[,i], y)
  abline(lm(y~x[,i]))
}
```



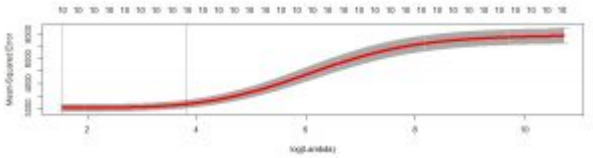
```
layout(1)

model_ols <- lm(y ~ x)
summary(model_ols)

##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -155.829  -38.534   -0.227   37.806  151.355
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  152.133      2.576  59.061  < 2e-16 ***
## xage         -10.012     59.749  -0.168 0.867000
## xsex        -239.819     61.222  -3.917 0.000104 ***
## xbmi         519.840     66.534   7.813 4.30e-14 ***
## xmap         324.390     65.422   4.958 1.02e-06 ***
## xtc         -792.184    416.684  -1.901 0.057947 .
## xldl         476.746    339.035   1.406 0.160389
## xhdl         101.045    212.533   0.475 0.634721
## xtch         177.064    161.476   1.097 0.273456
## xltg         751.279    171.902   4.370 1.56e-05 ***
## xglu          67.625     65.984   1.025 0.305998
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '
' 1
##
## Residual standard error: 54.15 on 431 degrees of freedom
## Multiple R-squared:  0.5177, Adjusted R-squared:  0.5066
## F-statistic: 46.27 on 10 and 431 DF,  p-value: < 2.2e-16

###############
#             #
# Exercise 2  #
#             #
###############

lambdas <- 10^seq(7, -3)
model_ridge <- glmnet(x, y, alpha = , lambda = lambdas)
plot.glmnet(model_ridge, xvar = "lambda", label = TRUE)
```



```
###############
#             #
# Exercise 3  #
#             #
###############
```

```
cv_fit <- cv.glmnet(x=x, y=y, alpha = , nlambda = 1000)
plot.cv.glmnet(cv_fit)
```



```
cv_fit$lambda.min
```

```
## [1] 4.685654
```

```
###############
#             #
# Exercise 4  #
#             #
###############
```

```
fit <- glmnet(x=x, y=y, alpha = , lambda=cv_fit$lambda.min)
fit$beta
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##              s0
## age   -1.776857
## sex -218.078518
## bmi  503.649515
## map  309.268175
## tc  -116.815832
## ldl  -51.664808
## hdl -181.472588
## tch  113.468602
## ltg  470.871230
## glu   80.969337
```

```
###############
#             #
# Exercise 5  #
#             #
###############
```

```
fit <- glmnet(x=x, y=y, alpha = , lambda=cv_fit$lambda.1se)
fit$beta
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##              s0
## age   22.463959
## sex -120.242431
## bmi  366.769888
## map  235.675894
## tc    -9.896795
## ldl  -52.093095
## hdl -170.482275
## tch  121.536669
## ltg  313.810759
## glu  112.152681
```

```
###############
#             #
# Exercise 6  #
#             #
###############
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.3.3
```

```
intrain <- createDataPartition(y=diabetes$y,
                               p = 0.8,
                               list = FALSE)
training <- diabetes[intrain,]
testing <- diabetes[-intrain,]
```

```
###############
#             #
# Exercise 7  #
#             #
###############
```

```
cv_ridge <- cv.glmnet(x=training$x, y=training$y,
```

```
                      alpha = , nlambda = 1000)
ridge_reg <- glmnet(x=training$x, y=training$y,
                    alpha = , lambda=cv_ridge$lambda.min)
ridge_reg$beta
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##              s0
## age   38.25965
## sex -209.67238
## bmi  529.69156
## map  341.55293
## tc  -102.08181
## ldl  -70.38056
## hdl -141.87799
## tch  102.70460
## ltg  489.04852
## glu   52.72637
```

```
ridge_reg <- glmnet(x=training$x, y=training$y,
                    alpha = , lambda=cv_ridge$lambda.1se)
ridge_reg$beta
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##               s0
## age   49.941787
## sex -127.389221
## bmi  399.264021
## map  272.565206
## tc    -5.586767
## ldl  -66.919061
## hdl -151.119495
## tch  104.071028
## ltg  339.155470
## glu   96.613412
```

```
###############
#             #
# Exercise 8  #
#             #
###############
```

```
ridge_reg <- glmnet(x=training$x, y=training$y,
                    alpha = , lambda=cv_ridge$lambda.min)
ridge_pred <- predict.glmnet(ridge_reg,
                             s = cv_ridge$lambda.min, newx =
testing$x)
sd((ridge_pred - testing$y)^2)/sqrt(length(testing$y))
```

```
## [1] 415.6961
```

```
ridge_reg <- glmnet(x=training$x, y=training$y,
                    alpha = , lambda=cv_ridge$lambda.1se)
ridge_pred <- predict.glmnet(ridge_reg,
                             s = cv_ridge$lambda.1se, newx =
testing$x)
sd((ridge_pred - testing$y)^2)/sqrt(length(testing$y))
```

```
## [1] 394.3651
```

```
# lower prediction error with higher lambda

###############
#             #
# Exercise 9  #
#             #
###############

ols_reg <- lm(y ~ x, data = training)
summary(ols_reg)
```

```
##
## Call:
## lm(formula = y ~ x, data = training)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -154.669  -41.299    1.594   38.940  151.834
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  153.601      2.922  52.575  < 2e-16 ***
```

```
## xage          33.104     68.732   0.482  0.63036
## xsex        -228.818     69.589  -3.288  0.00111 **
## xbmi         546.497     77.994   7.007 1.29e-11 ***
## xmap         359.381     74.686   4.812 2.24e-06 ***
## xtc         -721.463    447.808  -1.611  0.10808
## xldl         403.276    362.512   1.112  0.26672
## xhdl         128.830    232.186   0.555  0.57935
## xtch         177.948    180.552   0.986  0.32503
## xltg         748.765    185.718   4.032 6.82e-05 ***
## xglu          34.245     77.094   0.444  0.65718
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '
' 1
##
## Residual standard error: 54.92 on 344 degrees of freedom
## Multiple R-squared:  0.5142, Adjusted R-squared:  0.5001
## F-statistic: 36.42 on 10 and 344 DF,  p-value: < 2.2e-16

################
#              #
# Exercise 10  #
#              #
################

ols_pred <- predict(ols_reg, newdata=testing$x, type =
"response")
sd((ols_pred - testing$y)^2)/sqrt(length(testing$y))

## [1] 419.758

# least squares prediction error is higher.
```

Manipulate Biological Data

Using Biostrings Package Solutions(Part 4)

Below are the solutions to these exercises on Manipulate Biological Data Using Biostrings Package.

```
####################
#                  #
#    Exercise 1    #
#                  #
####################

library(Biostrings)
RNA1 <-RNAString("AAGAAAACCAUGAUGGAAGCCAG")
palindrome<-findPalindromes(RNA1)
print(palindrome)

##   Views on a 23-letter RNAString subject
## subject: AAGAAAACCAUGAUGGAAGCCAG
## views:
##     start end width
## [1]     8  16     9 [CCAUGAUGG]

####################
#                  #
#    Exercise 2    #
#                  #
####################

library(Biostrings)
DNA1 <- DNAString("AAGAAAACCATGATGGAAGCCAG")
palindrome<-findPalindromes(DNA1)
print(palindrome)

##   Views on a 23-letter DNAString subject
## subject: AAGAAAACCATGATGGAAGCCAG
## views:
##     start end width
```

```
## [1]     8  16     9 [CCATGATGG]

####################
#                  #
#    Exercise 3    #
#                  #
####################

library(Biostrings)
DNA1 <- DNAString("AAGAAAACCATGATGGAAGCCAG")
print(dinucleotideFrequency(DNA1))

## AA AC AG AT CA CC CG CT GA GC GG GT TA TC TG TT
##  5  1  3  2  2  2  0  0  3  1  1  0  0  0  2  0

####################
#                  #
#    Exercise 4    #
#                  #
####################

library(Biostrings)
RNA1 <-RNAString("AUAGUAGAUGCGGCGCGCUAGAG")
print(dinucleotideFrequency(RNA1))

## AA AC AG AU CA CC CG CU GA GC GG GU UA UC UG UU
##  0  0  4  2  0  0  3  1  2  4  1  1  3  0  1  0

####################
#                  #
#    Exercise 5    #
#                  #
####################

library(Biostrings)
DNA1 <- DNAString("AAGAAAACCATGATGGAAGCCAG")
print(oligonucleotideFrequency(DNA1,4))

## AAAA AAAC AAAG AAAT AACA AACC AACG AACT AAGA AAGC AAGG AAGT
AATA AATC AATG
```

```
##    1    1    0    0    0    1    0    0    1    1    0    0
0    0    0
## AATT ACAA ACAC ACAG ACAT ACCA ACCC ACCG ACCT ACGA ACGC ACGG
ACGT ACTA ACTC
##    0    0    0    0    0    1    0    0    0    0    0    0
0    0    0
## ACTG ACTT AGAA AGAC AGAG AGAT AGCA AGCC AGCG AGCT AGGA AGGC
AGGG AGGT AGTA
##    0    0    1    0    0    0    0    1    0    0    0    0
0    0    0
## AGTC AGTG AGTT ATAA ATAC ATAG ATAT ATCA ATCC ATCG ATCT ATGA
ATGC ATGG ATGT
##    0    0    0    0    0    0    0    0    0    0    0    1
0    1    0
## ATTA ATTC ATTG ATTT CAAA CAAC CAAG CAAT CACA CACC CACG CACT
CAGA CAGC CAGG
##    0    0    0    0    0    0    0    0    0    0    0    0
0    0    0
## CAGT CATA CATC CATG CATT CCAA CCAC CCAG CCAT CCCA CCCC CCCG
CCCT CCGA CCGC
##    0    0    0    1    0    0    0    1    1    0    0    0
0    0    0
## CCGG CCGT CCTA CCTC CCTG CCTT CGAA CGAC CGAG CGAT CGCA CGCC
CGCG CGCT CGGA
##    0    0    0    0    0    0    0    0    0    0    0    0
0    0    0
## CGGC CGGG CGGT CGTA CGTC CGTG CGTT CTAA CTAC CTAG CTAT CTCA
CTCC CTCG CTCT
##    0    0    0    0    0    0    0    0    0    0    0    0
0    0    0
## CTGA CTGC CTGG CTGT CTTA CTTC CTTG CTTT GAAA GAAC GAAG GAAT
GACA GACC GACG
##    0    0    0    0    0    0    0    0    1    0    1    0
0    0    0
## GACT GAGA GAGC GAGG GAGT GATA GATC GATG GATT GCAA GCAC GCAG
GCAT GCCA GCCC
##    0    0    0    0    0    0    0    1    0    0    0    0
0    1    0
## GCCG GCCT GCGA GCGC GCGG GCGT GCTA GCTC GCTG GCTT GGAA GGAC
GGAG GGAT GGCA
##    0    0    0    0    0    0    0    0    0    0    1    0
```

```
0    0    0
## GGCC GGCG GGCT GGGA GGGC GGGG GGGT GGTA GGTC GGTG GGTT GTAA
GTAC GTAG GTAT
##    0    0    0    0    0    0    0    0    0    0    0    0
0    0    0
## GTCA GTCC GTCG GTCT GTGA GTGC GTGG GTGT GTTA GTTC GTTG GTTT
TAAA TAAC TAAG
##    0    0    0    0    0    0    0    0    0    0    0    0
0    0    0
## TAAT TACA TACC TACG TACT TAGA TAGC TAGG TAGT TATA TATC TATG
TATT TCAA TCAC
##    0    0    0    0    0    0    0    0    0    0    0    0
0    0    0
## TCAG TCAT TCCA TCCC TCCG TCCT TCGA TCGC TCGG TCGT TCTA TCTC
TCTG TCTT TGAA
##    0    0    0    0    0    0    0    0    0    0    0    0
0    0    0
## TGAC TGAG TGAT TGCA TGCC TGCG TGCT TGGA TGGC TGGG TGGT TGTA
TGTC TGTG TGTT
##    0    0    1    0    0    0    0    1    0    0    0    0
0    0    0
## TTAA TTAC TTAG TTAT TTCA TTCC TTCG TTCT TTGA TTGC TTGG TTGT
TTTA TTTC TTTG
##    0    0    0    0    0    0    0    0    0    0    0    0
0    0    0
## TTTT
##    0

######################
#                    #
#    Exercise 6      #
#                    #
######################

library(Biostrings)
RNA1 <-RNAString("AUAGUAGAUGCGGCGCGCUAGAG")
print(oligonucleotideFrequency(RNA1,4))

## AAAA AAAC AAAG AAAU AACA AACC AACG AACU AAGA AAGC AAGG AAGU
AAUA AAUC AAUG
##    0    0    0    0    0    0    0    0    0    0    0    0
```

```
0    0    0
## AAUU ACAA ACAC ACAG ACAU ACCA ACCC ACCG ACCU ACGA ACGC ACGG
ACGU ACUA ACUC
##    0    0    0    0    0    0    0    0    0    0    0    0
0    0    0
## ACUG ACUU AGAA AGAC AGAG AGAU AGCA AGCC AGCG AGCU AGGA AGGC
AGGG AGGU AGUA
##    0    0    0    0    1    1    0    0    0    0    0    0
0    0    1
## AGUC AGUG AGUU AUAA AUAC AUAG AUAU AUCA AUCC AUCG AUCU AUGA
AUGC AUGG AUGU
##    0    0    0    0    0    1    0    0    0    0    0    0
1    0    0
## AUUA AUUC AUUG AUUU CAAA CAAC CAAG CAAU CACA CACC CACG CACU
CAGA CAGC CAGG
##    0    0    0    0    0    0    0    0    0    0    0    0
0    0    0
## CAGU CAUA CAUC CAUG CAUU CCAA CCAC CCAG CCAU CCCA CCCC CCCG
CCCU CCGA CCGC
##    0    0    0    0    0    0    0    0    0    0    0    0
0    0    0
## CCGG CCGU CCUA CCUC CCUG CCUU CGAA CGAC CGAG CGAU CGCA CGCC
CGCG CGCU CGGA
##    0    0    0    0    0    0    0    0    0    0    0    0
1    1    0
## CGGC CGGG CGGU CGUA CGUC CGUG CGUU CUAA CUAC CUAG CUAU CUCA
CUCC CUCG CUCU
##    1    0    0    0    0    0    0    0    0    1    0    0
0    0    0
## CUGA CUGC CUGG CUGU CUUA CUUC CUUG CUUU GAAA GAAC GAAG GAAU
GACA GACC GACG
##    0    0    0    0    0    0    0    0    0    0    0    0
0    0    0
## GACU GAGA GAGC GAGG GAGU GAUA GAUC GAUG GAUU GCAA GCAC GCAG
GCAU GCCA GCCC
##    0    0    0    0    0    0    0    1    0    0    0    0
0    0    0
## GCCG GCCU GCGA GCGC GCGG GCGU GCUA GCUC GCUG GCUU GGAA GGAC
GGAG GGAU GGCA
##    0    0    0    2    1    0    1    0    0    0    0    0
0    0    0
```

```
## GGCC GGCG GGCU GGGA GGGC GGGG GGGU GGUA GGUC GGUG GGUU GUAA
GUAC GUAG GUAU
##    0    1    0    0    0    0    0    0    0    0    0    0
0    1    0
## GUCA GUCC GUCG GUCU GUGA GUGC GUGG GUGU GUUA GUUC GUUG GUUU
UAAA UAAC UAAG
##    0    0    0    0    0    0    0    0    0    0    0    0
0    0    0
## UAAU UACA UACC UACG UACU UAGA UAGC UAGG UAGU UAUA UAUC UAUG
UAUU UCAA UCAC
##    0    0    0    0    0    2    0    0    1    0    0    0
0    0    0
## UCAG UCAU UCCA UCCC UCCG UCCU UCGA UCGC UCGG UCGU UCUA UCUC
UCUG UCUU UGAA
##    0    0    0    0    0    0    0    0    0    0    0    0
0    0    0
## UGAC UGAG UGAU UGCA UGCC UGCG UGCU UGGA UGGC UGGG UGGU UGUA
UGUC UGUG UGUU
##    0    0    0    0    0    1    0    0    0    0    0    0
0    0    0
## UUAA UUAC UUAG UUAU UUCA UUCC UUCG UUCU UUGA UUGC UUGG UUGU
UUUA UUUC UUUG
##    0    0    0    0    0    0    0    0    0    0    0    0
0    0    0
## UUUU
##    0

####################
#                  #
#    Exercise 7    #
#                  #
####################

library(Biostrings)
DNA1 <- DNAString("AAGAAAACCATGATGGAAGCCAG")
print(trinucleotideFrequency(DNA1))

## AAA AAC AAG AAT ACA ACC ACG ACT AGA AGC AGG AGT ATA ATC ATG
ATT CAA CAC
##   2   1   2   0   0   1   0   0   1   1   0   0   0   0   2
0   0   0
```

```
## CAG CAT CCA CCC CCG CCT CGA CGC CGG CGT CTA CTC CTG CTT GAA
GAC GAG GAT
##   1   1   2   0   0   0   0   0   0   0   0   0   0   2
0   0   1
## GCA GCC GCG GCT GGA GGC GGG GGT GTA GTC GTG GTT TAA TAC TAG
TAT TCA TCC
##   0   1   0   0   1   0   0   0   0   0   0   0   0   0
0   0   0
## TCG TCT TGA TGC TGG TGT TTA TTC TTG TTT
##   0   0   1   0   1   0   0   0   0   0

####################
#                  #
#    Exercise 8    #
#                  #
####################

library(Biostrings)
RNA1 <-RNAString("AUAGUAGAUGCGGCGCGCUAGAG")
print(trinucleotideFrequency(RNA1))

## AAA AAC AAG AAU ACA ACC ACG ACU AGA AGC AGG AGU AUA AUC AUG
AUU CAA CAC
##   0   0   0   0   0   0   0   0   2   0   0   1   1   0   1
0   0   0
## CAG CAU CCA CCC CCG CCU CGA CGC CGG CGU CUA CUC CUG CUU GAA
GAC GAG GAU
##   0   0   0   0   0   0   0   2   1   0   1   0   0   0   0
0   1   1
## GCA GCC GCG GCU GGA GGC GGG GGU GUA GUC GUG GUU UAA UAC UAG
UAU UCA UCC
##   0   0   3   1   0   1   0   0   1   0   0   0   0   0   3
0   0   0
## UCG UCU UGA UGC UGG UGU UUA UUC UUG UUU
##   0   0   0   1   0   0   0   0   0   0

####################
#                  #
#    Exercise 9    #
#                  #
####################
```

```
library(Biostrings)
print(alphabet(AAString()))
```

```
##  [1] "A" "R" "N" "D" "C" "Q" "E" "G" "H" "I" "L" "K" "M"
"F" "P" "S" "T"
## [18] "W" "Y" "V" "U" "O" "B" "J" "Z" "X" "*" "-" "+" "."
```

```
####################
#                  #
#    Exercise 10    #
#                  #
####################
```

```
library(Biostrings)
Amino1 <- AAString("MVVVDFFV")
print(length(Amino1))
```

```
## [1] 8
```

```
print(alphabetFrequency(Amino1))
```

```
##     A     R     N     D     C     Q     E     G     H     I
L     K
##     0     0     0     1     0     0     0     0     0     0
0     0
##     M     F     P     S     T     W     Y     V     U     O
B     J
##     1     2     0     0     0     0     0     4     0     0
0     0
##     Z     X     *     -     +     . other
##     0     0     0     0     0     0     0
```

Data visualization with googleVis solutions part 3

Below are the solutions to these exercises on visualizations with googleVis.

```
####################
#                  #
#    Exercise 1    #
#                  #
####################

library(googleVis)
ScatterC <- gvisScatterChart(cars)

####################
#                  #
#    Exercise 2    #
#                  #
####################

library(googleVis)
ScatterC <- gvisScatterChart(cars)
plot(ScatterC)

####################
#                  #
#    Exercise 3    #
#                  #
####################

library(googleVis)
ScatterC <- gvisScatterChart(cars,
                             options=list(
                                          title="Cars",
vAxis="{title:'speed'}",
                               hAxis="{title:'dist'}" ))
plot(ScatterC)
```

```
####################
#                  #
#    Exercise 4    #
#                  #
####################

library(googleVis)
ScatterC <- gvisScatterChart(cars,
                             options=list(
                                          title="Cars",
vAxis="{title:'speed'}",
                               hAxis="{title:'dist'}",
                               width=600, height=300))
plot(ScatterC)

####################
#                  #
#    Exercise 5    #
#                  #
####################

library(googleVis)

ScatterC <- gvisScatterChart(cars,
                             options=list(
                               legend="none",
                                          title="Cars",
vAxis="{title:'speed'}",
                               hAxis="{title:'dist'}",
                               width=600, height=300))
plot(ScatterC)

####################
#                  #
#    Exercise 6    #
#                  #
####################

library(googleVis)
ScatterC <- gvisScatterChart(cars,
                             options=list(
```

```
                                legend="none",
                                pointSize=3,lineWidth=2,
                                              title="Cars",
vAxis="{title:'speed'}",
                                hAxis="{title:'dist'}",
                                width=600, height=300))
plot(ScatterC)

####################
#                  #
#    Exercise 7    #
#                  #
####################

library(googleVis)
BubbleC <- gvisBubbleChart(Fruits)

####################
#                  #
#    Exercise 8    #
#                  #
####################

library(googleVis)
BubbleC <- gvisBubbleChart(Fruits)
plot(BubbleC)

####################
#                  #
#    Exercise 9    #
#                  #
####################

library(googleVis)
head(Fruits)
BubbleC <- gvisBubbleChart(Fruits,idvar="Fruit",
                           xvar="Sales", yvar="Expenses",
                           colorvar="Year", sizevar="Profit")
plot(BubbleC)

####################
```

```
#                   #
#    Exercise 10    #
#                   #
#####################

library(googleVis)
BubbleC <- gvisBubbleChart(Fruits,idvar="Fruit",
                           xvar="Sales", yvar="Expenses",
                           colorvar="Year", sizevar="Profit",
                           options=list(
                                     hAxis='{minValue:70,
maxValue:130}',
                                     vAxis='{minValue:50,
maxValue:100}'))
plot(BubbleC)
```

Using the xlsx package to create an Excel file solutions

Below are the solutions to these exercises on creating an Excel file using the xlsx package.

```
####################
#                  #
#    Exercise 1    #
#                  #
####################
install.packages("xlsx", dependencies = TRUE)
require(xlsx)

####################
#                  #
#    Exercise 2    #
```

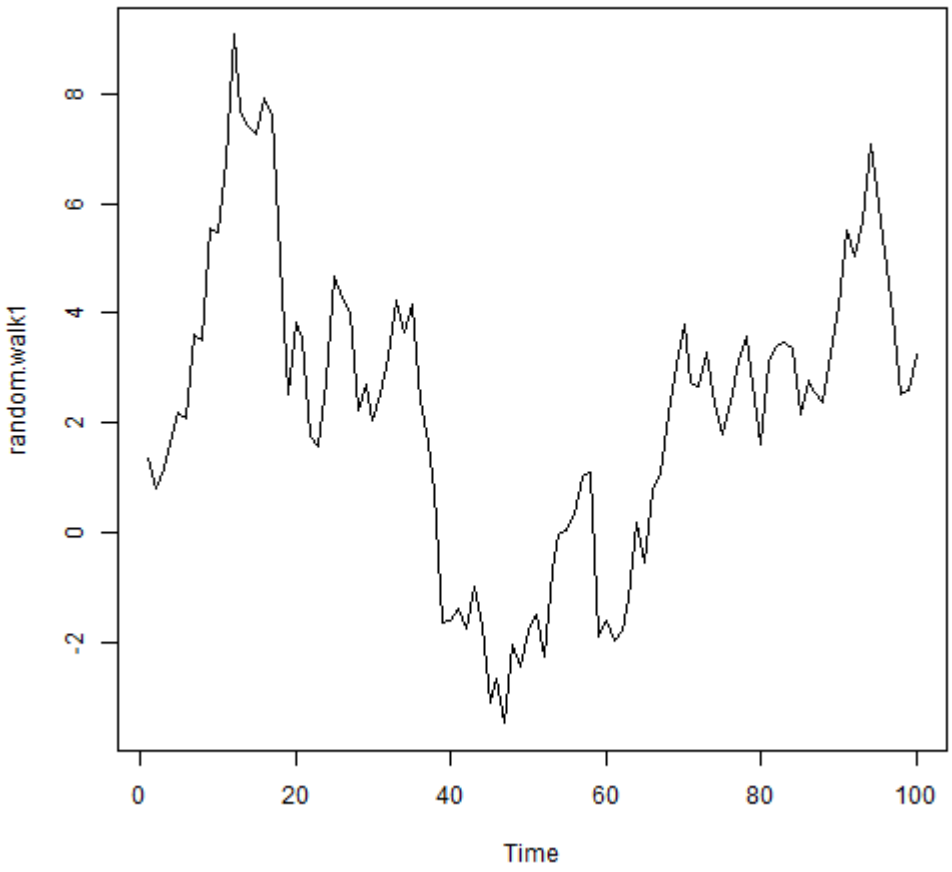
```
#                   #
#####################
wb <- createWorkbook(type = "xlsx")


#####################
#                   #
#    Exercise 3     #
#                   #
#####################
sheet1 <- createSheet(wb, sheetName = "iris")


#####################
#                   #
#    Exercise 4     #
#                   #
#####################
addDataFrame(iris, sheet1, startRow = 1L, startColumn = 1L,
row.names = FALSE)


#####################
#                   #
#    Exercise 5     #
#                   #
#####################
createFreezePane(sheet1, 2, 1, startRow = 2, startColumn = 1)


#####################
#                   #
#    Exercise 6     #
#                   #
#####################
setColumnWidth(sheet1, colIndex = 1:5, colWidth = 12)


#####################
#                   #
#    Exercise 7     #
```

```
#                   #
#####################
font <- Font(wb, isBold = TRUE)
head <- CellBlock(sheet1, 1, 1, 1, 5, create = FALSE)
CB.setFont(head, font, 1, 1:5)


#####################
#                   #
#    Exercise 8     #
#                   #
#####################
table1 <- tapply(iris$Sepal.Length, iris$Species, mean)
table1 <- as.data.frame(t(table1))
sheet2 <- createSheet(wb, sheetName = "pw")
addDataFrame(table1, sheet2, startRow = 2L, startColumn = 1L,
row.names = FALSE)


#####################
#                   #
#    Exercise 9     #
#                   #
#####################
title <- CellBlock(sheet2, 1, 1, 1, 1)
CB.setRowData(title, "Petal width mean by species", 1)
addMergedRegion(sheet2, 1, 1, 1, 3)


#####################
#                   #
#    Exercise 10    #
#                   #
#####################
saveWorkbook(wb, "filename.xlsx") # setwd() first
```

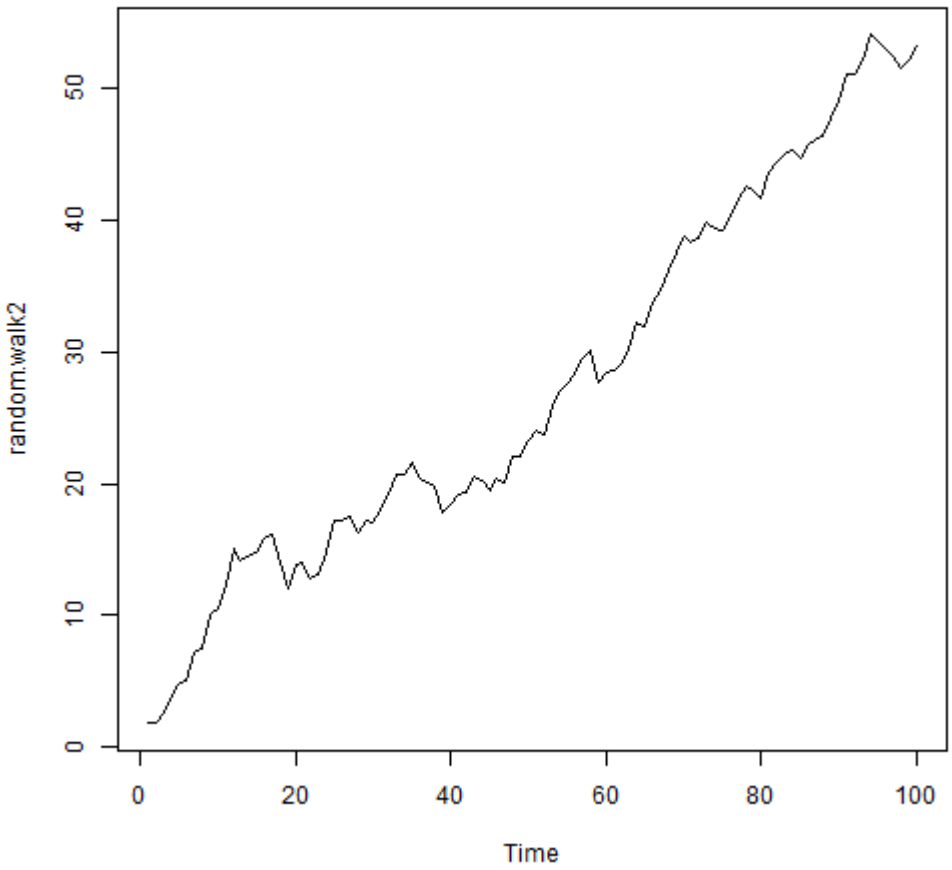
Forecasting for small business Solutions (Part-2)

Below are the solutions to these exercises on time series forecasting.

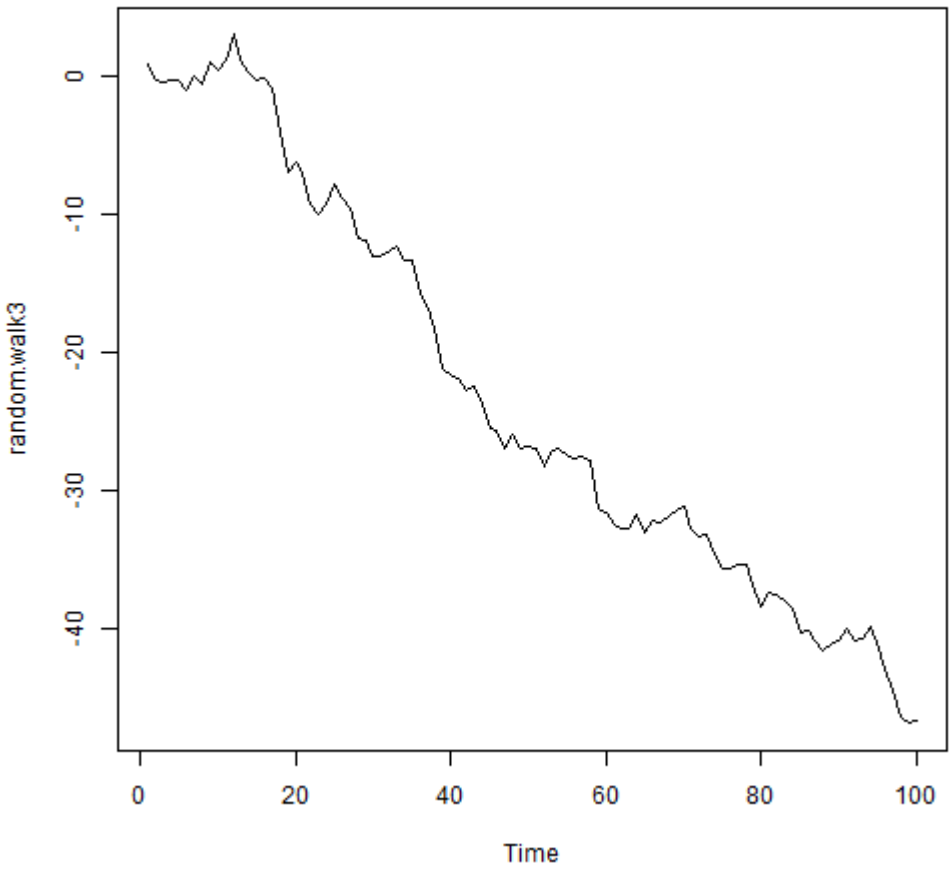
```
####################
#                  #
#    Exercise 1    #
#                  #
####################
set.seed(42)
random<-rnorm(100, mean = , sd = 1)
ts1<-cumsum(random)
random.walk1<-as.ts(ts1,start=1, end=100, frequency = 1)

####################
#                  #
#    Exercise 2    #
#                  #
####################
ts.plot(random.walk1)
```



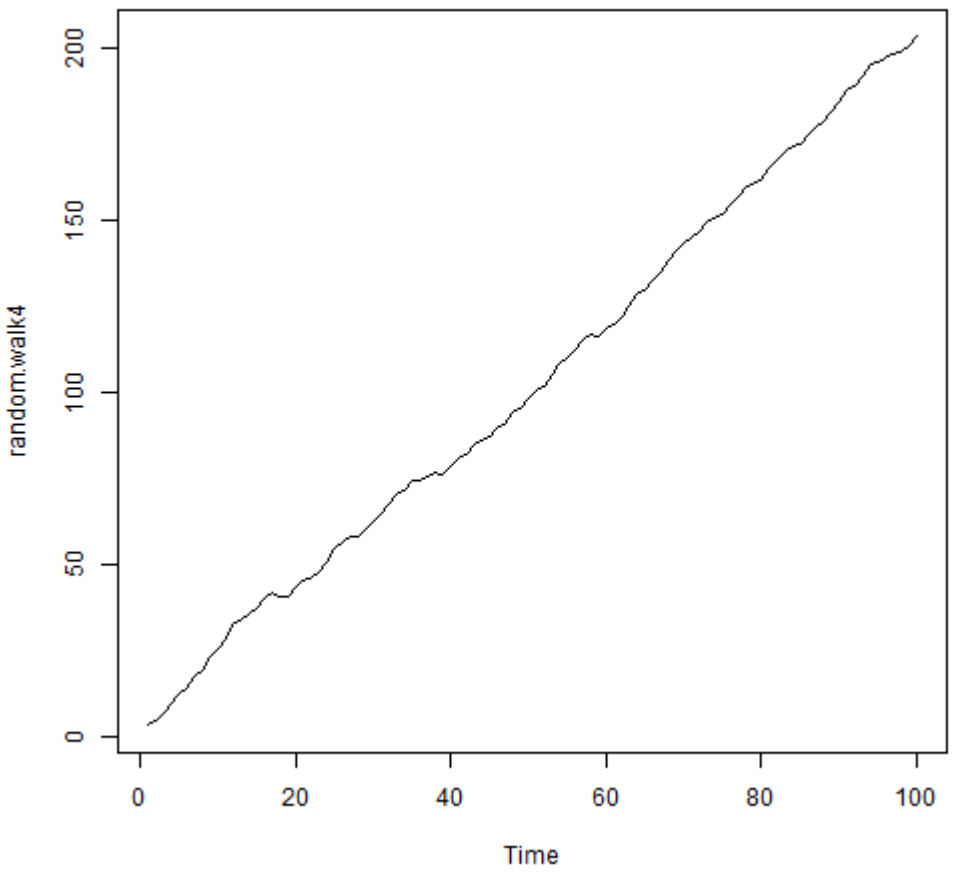

```
####################
#                  #
#    Exercise 3    #
#                  #
####################
set.seed(42)
random<-rnorm(100, mean = 0.5, sd = 1)
ts2<-cumsum(random)
random.walk2<-as.ts(ts2,start=1, end=100, frequency = 1)
ts.plot(random.walk2)
```



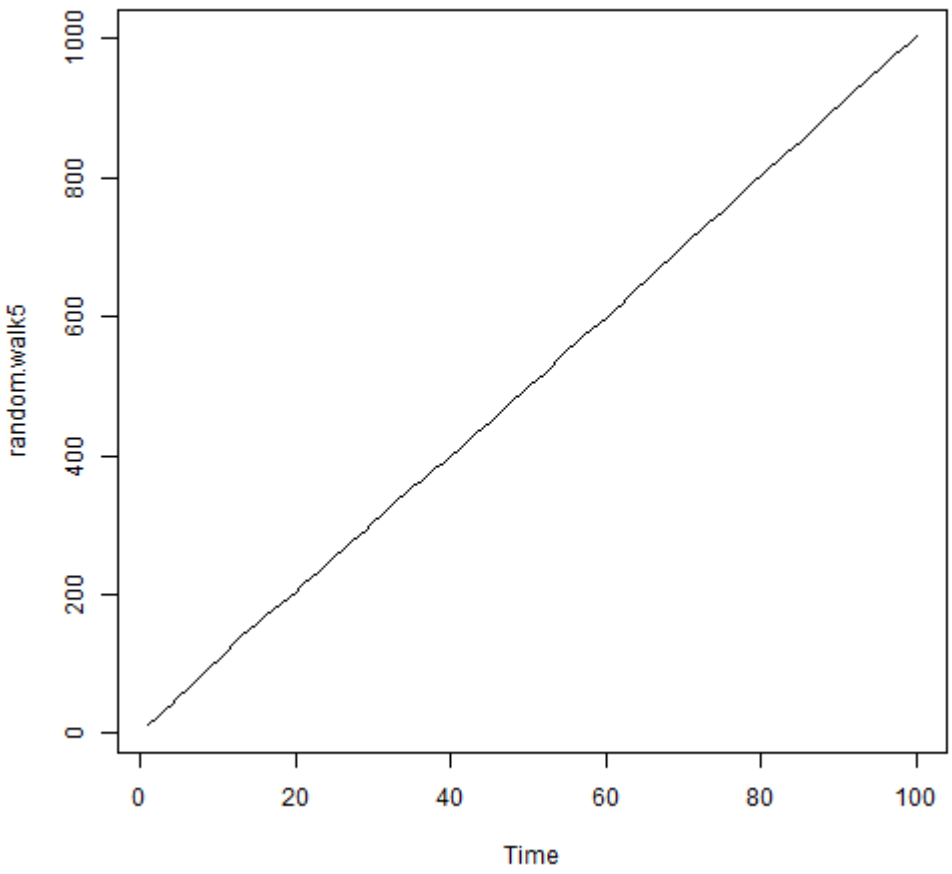

```
####################
#                  #
#    Exercise 4    #
#                  #
####################
set.seed(42)
random<-rnorm(100, mean = -0.5, sd = 1)
ts3<-cumsum(random)
random.walk3<-as.ts(ts3,start=1, end=100, frequency = 1)
ts.plot(random.walk3)
```



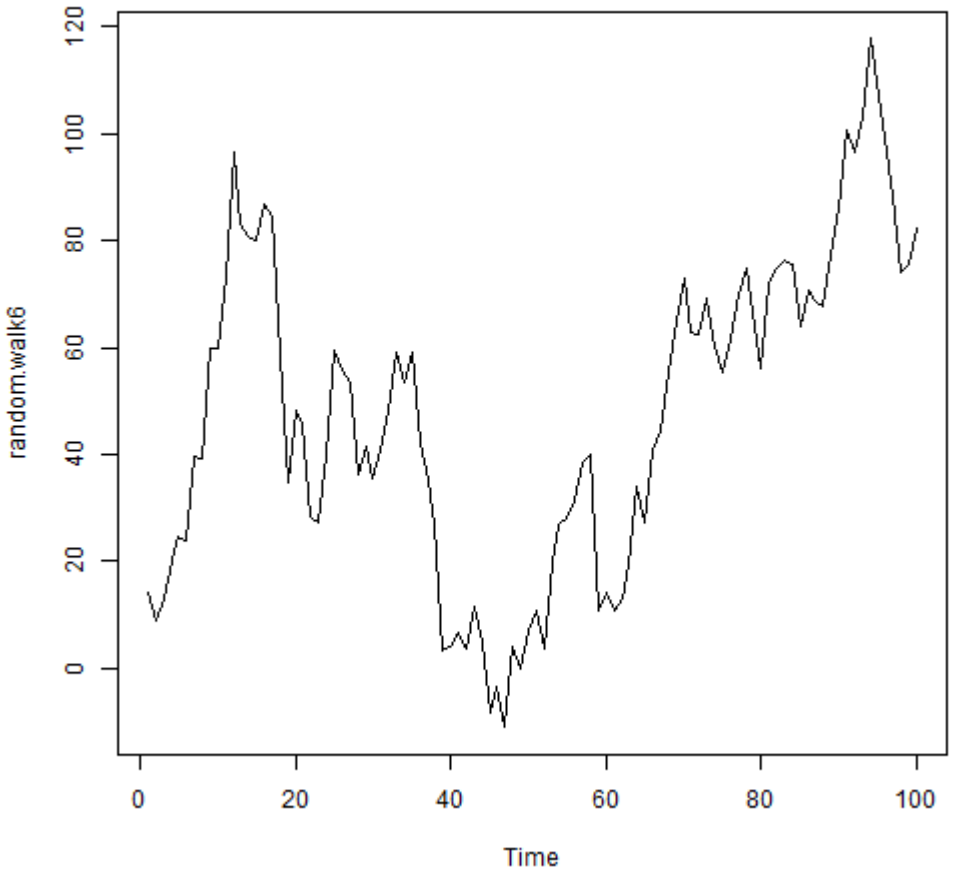

```
set.seed(42)
random<-rnorm(100, mean = 2, sd = 1)
ts4<-cumsum(random)
random.walk4<-as.ts(ts4,start=1, end=100, frequency = 1)
ts.plot(random.walk4)
```



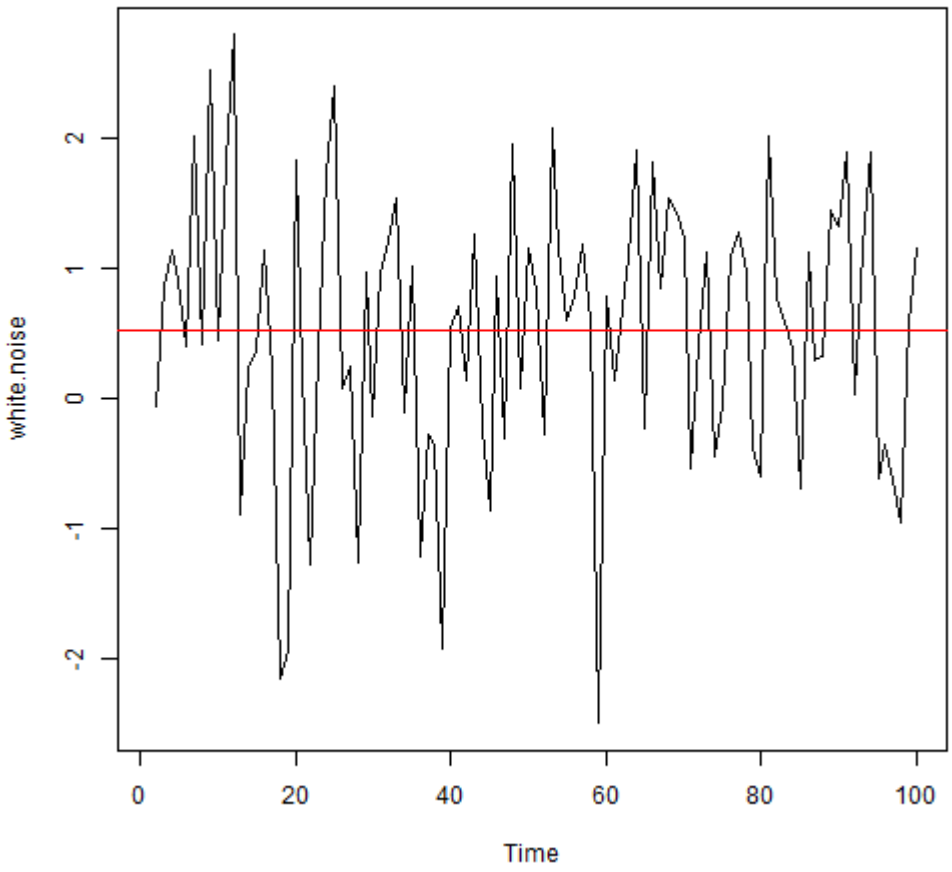

```
set.seed(42)
random<-rnorm(100, mean = 10, sd = 1)
ts5<-cumsum(random)
random.walk5<-as.ts(ts5,start=1, end=100, frequency = 1)
ts.plot(random.walk5)
```



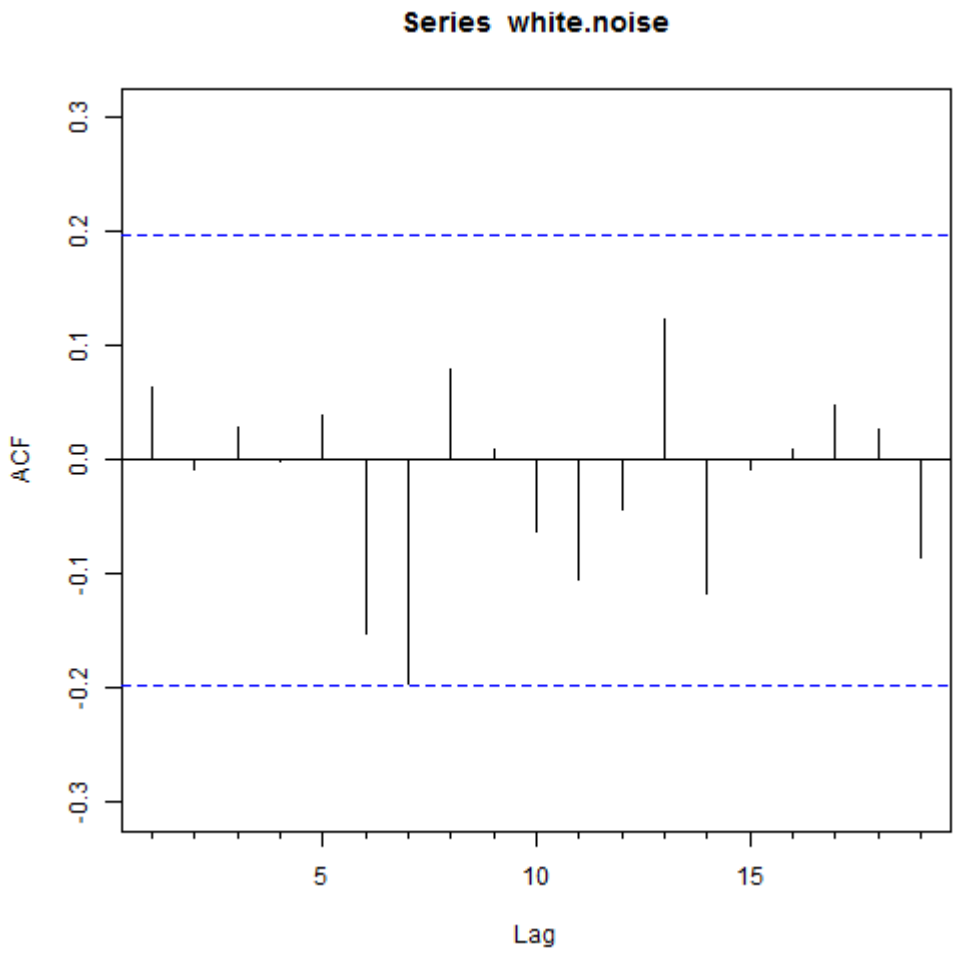

```
set.seed(42)
random<-rnorm(100, mean = 0.5, sd = 10)
ts6<-cumsum(random)
random.walk6<-as.ts(ts6,start=1, end=100, frequency = 1)
ts.plot(random.walk6)
```




```
####################
#                  #
#    Exercise 5    #
#                  #
####################
set.seed(42)
white.noise<-  c(NA,random.walk2[2:length(random.walk1)]  -
random.walk2[1:(length(random.walk2)-1)])
white.noise<-
as.ts(white.noise,start=1,end(length(white.noise)),frequency=1
)
plot(white.noise)
abline(a=mean(white.noise,na.rm=TRUE),b=, col="red")
```




```
#####################
#                   #
#    Exercise 6     #
#                   #
#####################
library("forecast")
white.noise<-white.noise[!is.na(white.noise)]
Acf(white.noise)
```




```
#####################
#                   #
#    Exercise 7     #
#                   #
#####################
Box.test(white.noise, lag=7, type="Ljung-Box")

##
##      Box-Ljung test
##
## data:  white.noise
## X-squared = 7.3137, df = 7, p-value = 0.397

library("tseries")
kpss.test(white.noise, null="Level")

## Warning in kpss.test(white.noise, null = "Level"): p-value
greater than
## printed p-value
```

```
##
##      KPSS Test for Level Stationarity
##
## data:  white.noise
## KPSS Level = 0.071342, Truncation lag parameter = 2, p-
value = 0.1
```

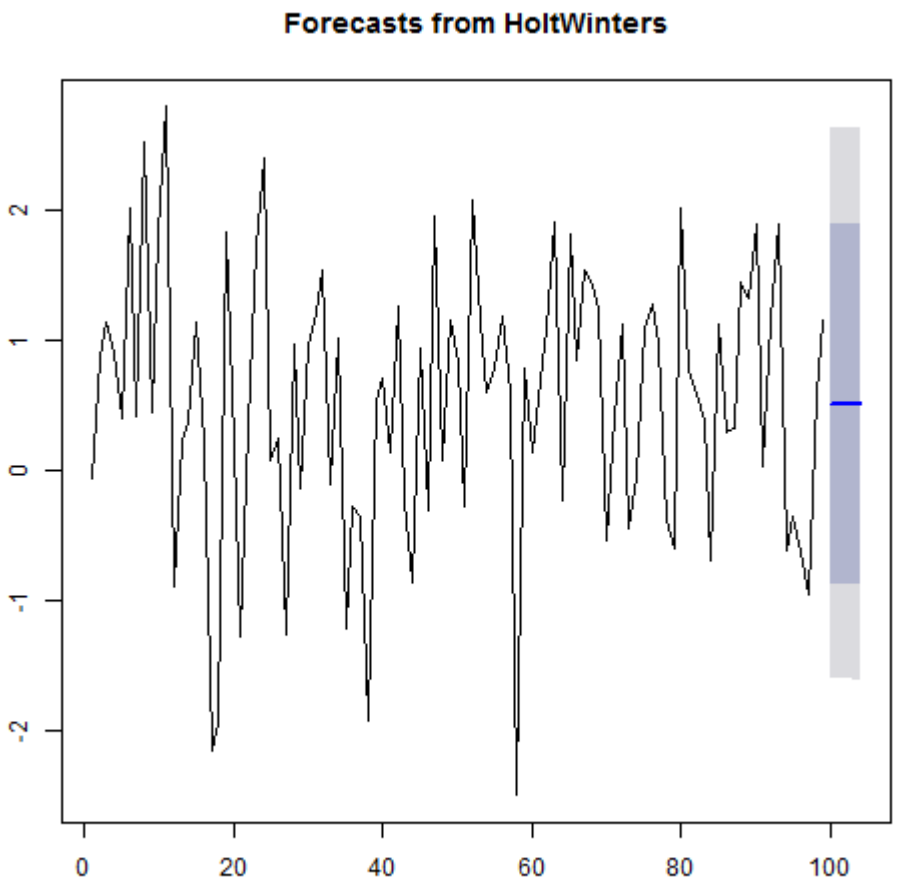
```
adf.test(white.noise, alternative="stationary")
```

```
##
##      Augmented Dickey-Fuller Test
##
## data:  white.noise
## Dickey-Fuller = -3.8697, Lag order = 4, p-value = 0.01841
## alternative hypothesis: stationary
```

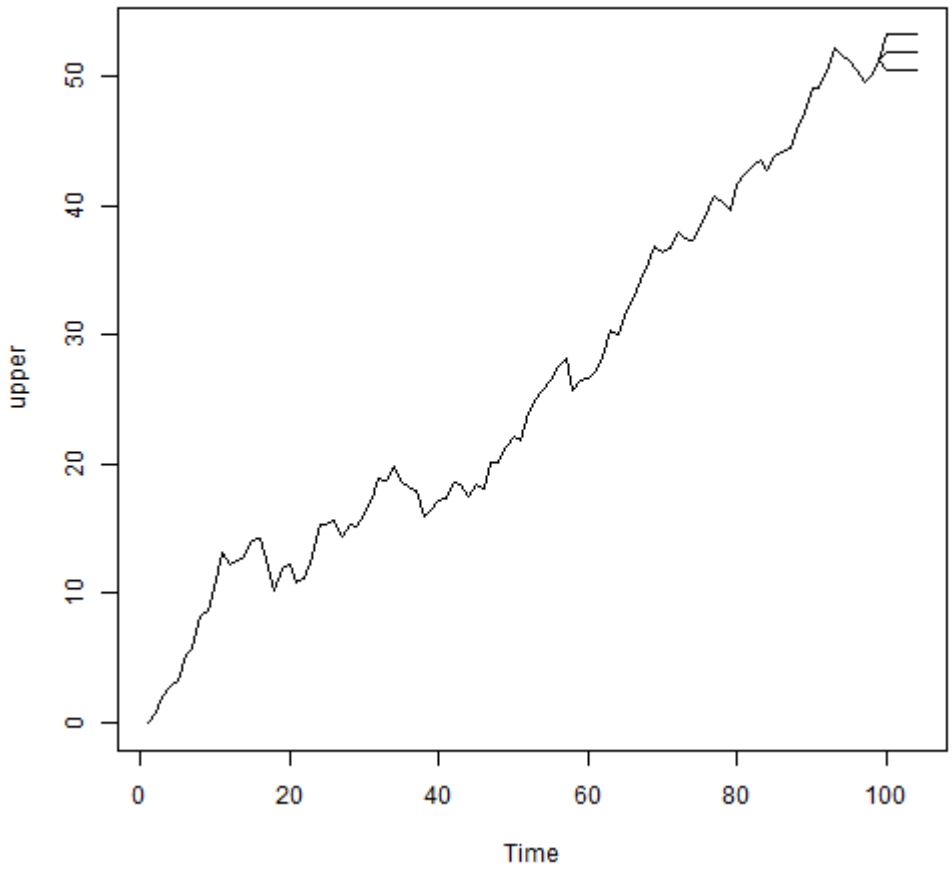
```
####################
#                  #
#    Exercise 8    #
#                  #
####################
prediction  <-  forecast.HoltWinters(HoltWinters(white.noise,
beta=FALSE, gamma=FALSE, l.start=white.noise[1]), h=5)
prediction
```

```
##     Point Forecast      Lo 80    Hi 80     Lo 95    Hi 95
## 100      0.5100705 -0.8687548 1.888896 -1.598661 2.618802
## 101      0.5100705 -0.8702338 1.890375 -1.600923 2.621064
## 102      0.5100705 -0.8717112 1.891852 -1.603182 2.623323
## 103      0.5100705 -0.8731871 1.893328 -1.605439 2.625580
## 104      0.5100705 -0.8746613 1.894802 -1.607694 2.627835
```

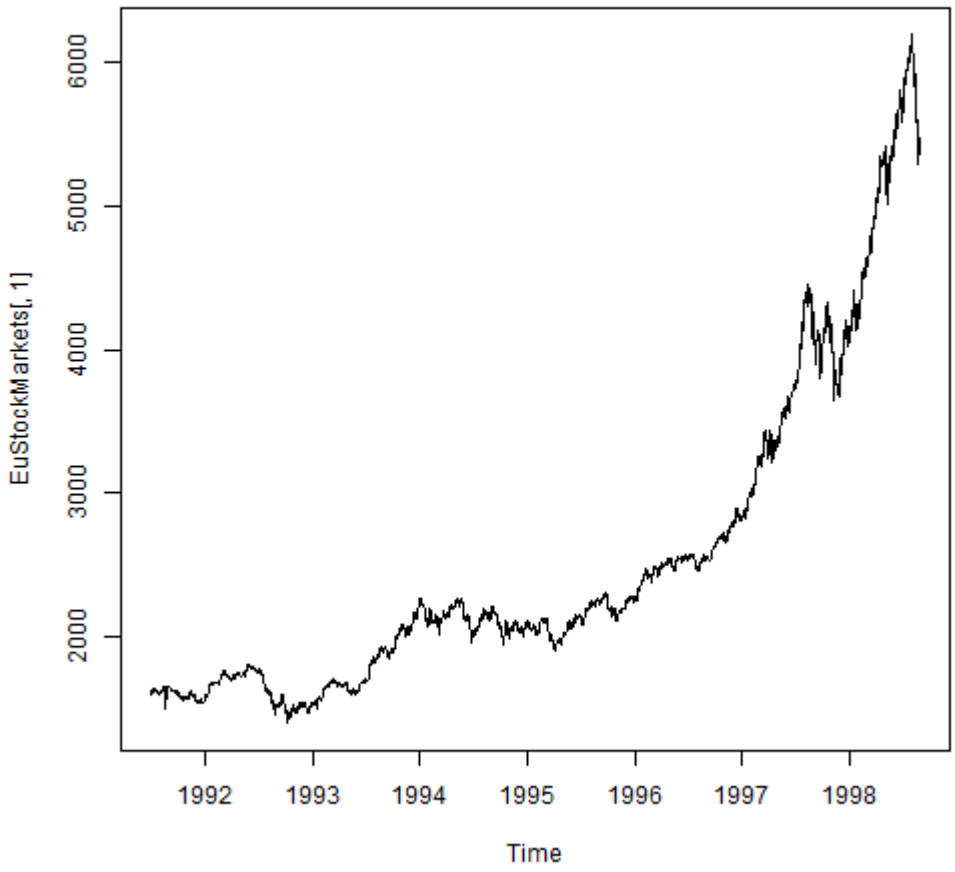
```
plot.forecast(prediction)
```



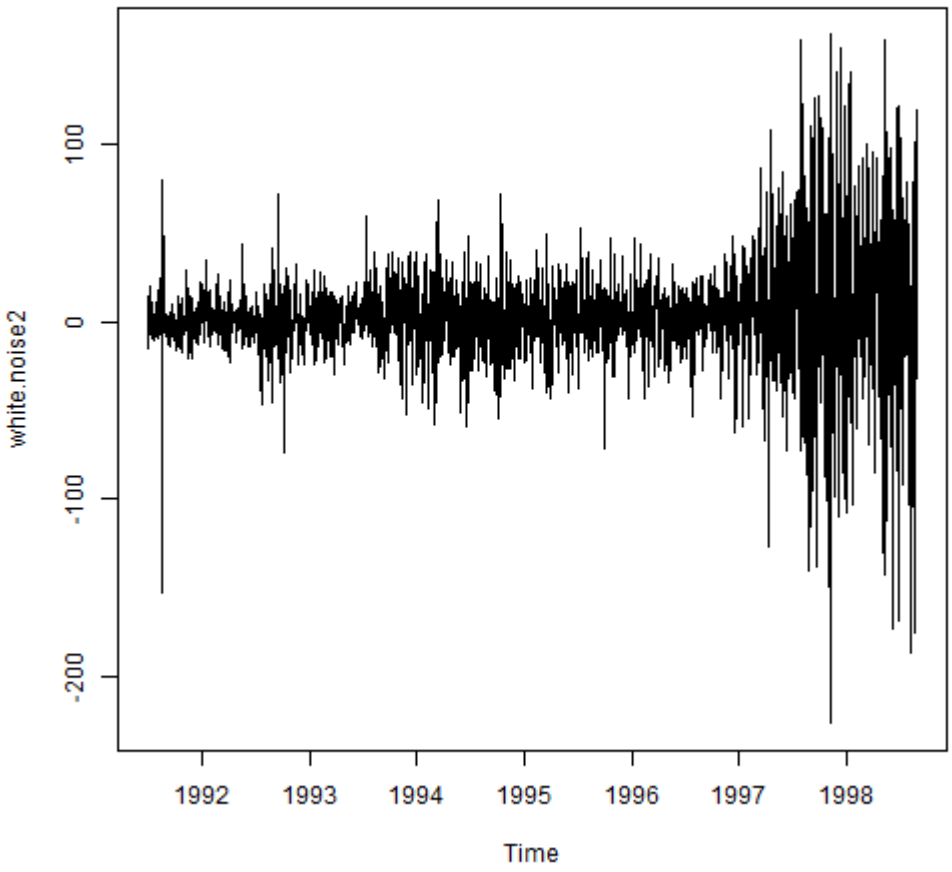

```
#####################
#                   #
#    Exercise 9     #
#                   #
#####################
rw<- cumsum(c(prediction$x,prediction$mean))
rw[101:104]<-rep(rw[100],4)
lower<-rw
lower[100:104]<- rep(rw[99],5)+prediction$lower[,1]
upper<-rw
upper[100:104]<- rep(rw[99],5)+prediction$upper[,1]
ts.plot(upper)
lines(lower)
lines(rw)
```




```
#####################
#                   #
#    Exercise 10    #
#                   #
#####################
data(EuStockMarkets)
plot(EuStockMarkets[,1])
```




```
white.noise2<-diff(EuStockMarkets[,1],lag=1)
plot(white.noise2)
```


white.noise2
100
0
-100
-200
1992
1993
1994
1995
1996
1997
1998
Time