SRT411_A0_AotaoXu

Aotao Xu

February 2, 2017

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

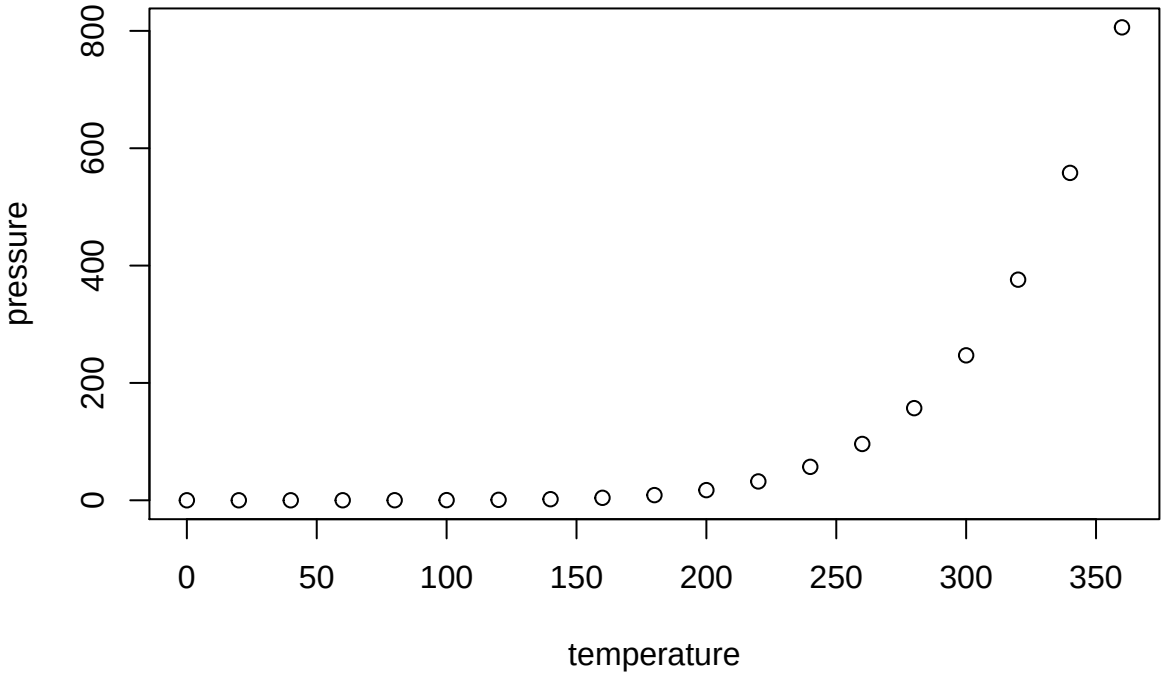
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed           dist       
##  Min.   : 4.0   Min.   :  2.00  
##  1st Qu.:12.0   1st Qu.: 26.00  
##  Median :15.0   Median : 36.00  
##  Mean   :15.4   Mean   : 42.98  
##  3rd Qu.:19.0   3rd Qu.: 56.00  
##  Max.   :25.0   Max.   :120.00
```

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

Todo_1_3.1

Compute the difference between 2014 and the year you started at this university and divide this by the difference between 2014 and the year you were born.Multiply this with 100 to get the percentage of your life you have spent at this university.Use brackets if you need them.

```
(2017-2014)/(2017-1994)*100
```

```
## [1] 13.04348
```

Todo_2_3.2

Repeat the previous ToDo, but with several steps in between. You can give the variables any name you want, but the name has to start with a letter.

```
Year = 2017
Birth = 1994
(Year -2014)/(2014 - Birth)*100
```

```
## [1] 15
```

Todo_3_3.4

Compute the sum of 4, 5, 8 and 11 by first combining them into a vector and then using the function sum.

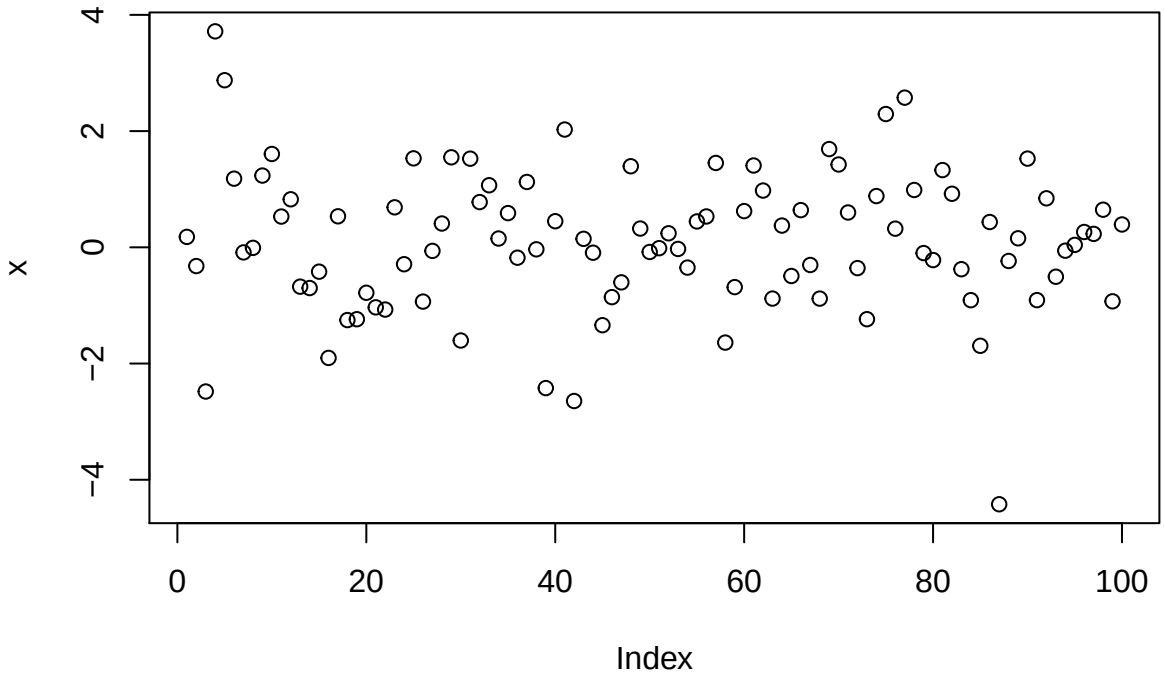
```
sum( 4 + 5 + 8 + 11 )
```

```
## [1] 28
```

Todo_4_3.5

Plot 100 normal random numbers.

```
x = rnorm(100)
plot(x)
```



##

Todo_5_4.0 Find help for the sqrt function.

```
help(rnorm)
```

Todo_6

Make a file called firstscript.R containing R-code that generates 100 random numbers and plots them, and run this script several times same as todo5 ## Todo_7_6.1 Put the numbers 31 to 60 in a vector named P and in a matrix with 6 rows and 5 columns named Q . Tip: use the function seq . Look at the different ways scalars, vectors and matrices are denoted in the workspace window.

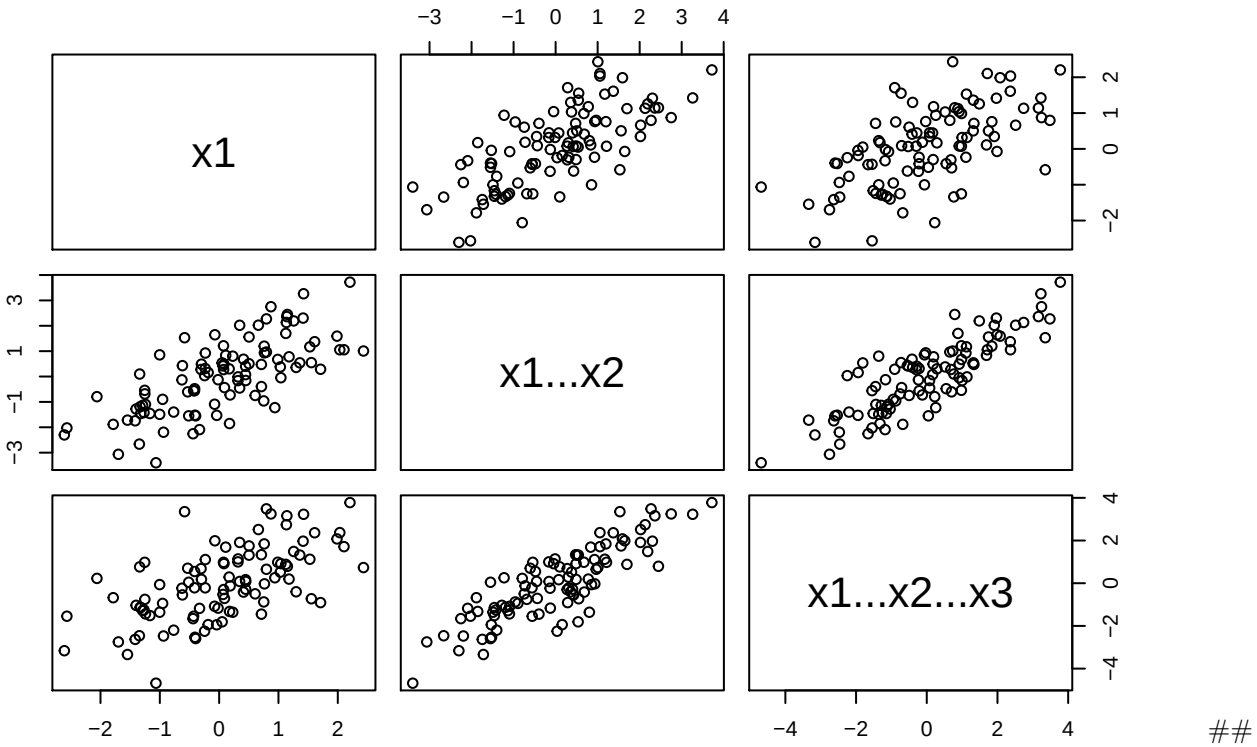
```
Q = matrix(31:60,6,5)
Q
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]   31   37   43   49   55
## [2,]   32   38   44   50   56
## [3,]   33   39   45   51   57
## [4,]   34   40   46   52   58
## [5,]   35   41   47   53   59
## [6,]   36   42   48   54   60
```

Todo_8_6.3

Make a script file which constructs three random normal vectors of length 100. Call these vectors x1,x2 and x3.Make a data frame called t with three columns (called a,b and c) containing respectively x1,x1+x2 and x1+x2+x3.Call the following functions for this data frame:plot(t) and sd(t).Can you understand the results? Rerun this script a few times.

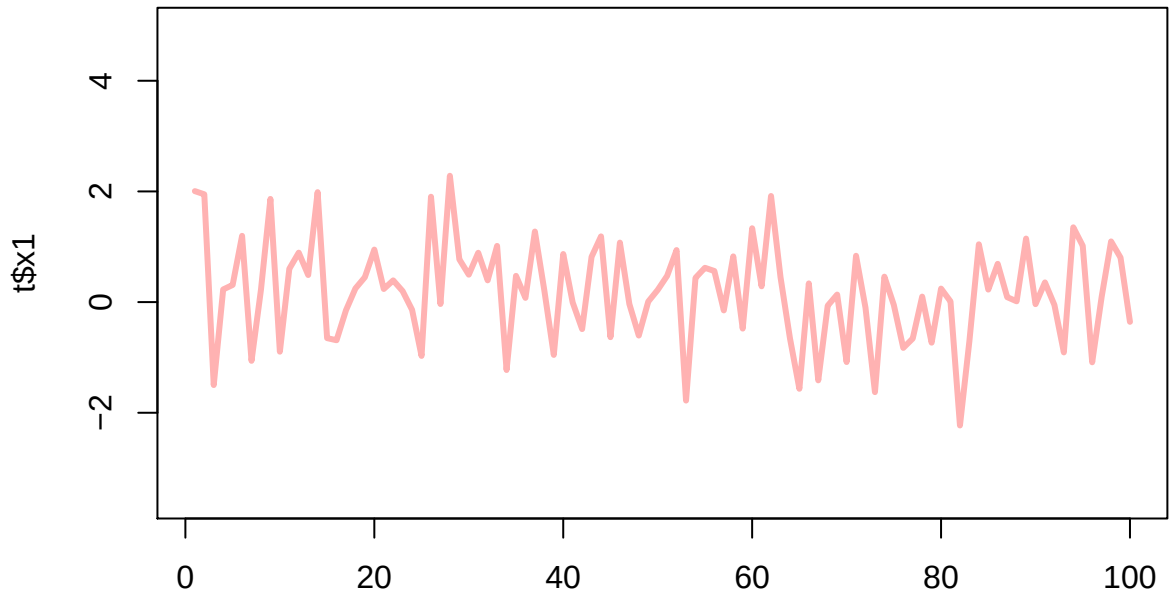
```
x1 = rnorm(100)
x2 = rnorm(100)
x3 = rnorm(100)
t = data.frame(x1,x1+x2,x1+x2+x3)
plot(t)
```



##

Todo_9_7.0 Add these lines to the script file of the previous section. Try to find out, either by experimenting or by using the help, what the meaning is of rgb, the last argument of rgb,lwd,pch,cex.

```
x1 = rnorm(100)
x2 = rnorm(100)
x3 = rnorm(100)
t = data.frame(x1,x1+x2,x1+x2+x3)
plot(t$x1, type="l", ylim=range(t), lwd=3, col=rgb(1,0,0,0.3))
lines(t$b, type="s", lwd=2, col=rgb(0.3,0.4,0.3,0.9))
points(t$c, pch=20, cex=4, col=rgb(0,0,1,0.3))
```



##

Todo_10_8.0 Make a file called tst1.txt in Notepad from the example in Figure4 and store it in your working directory. Write a script to read it, to multiply the column called g by 5 and to store it as tst2.txt.

```
d = read.table(file = "/home/axu10/Desktop/SRT411/tst1.txt", header = TRUE)
d1 = data.frame(a = c(1,2,4,8,16,32),
                g = c(2,4,8,16,32,64),
                x = c(3,6,12,24,48,96))
d1
```

```
##    a  g  x
## 1  1  2  3
## 2  2  4  6
## 3  4  8 12
## 4  8 16 24
## 5 16 32 48
## 6 32 64 96
```

```
write.table(d1$g*5, file = "/home/axu10/Desktop/SRT411/tst2.txt",
            row.names = FALSE)
d1
```

```
##    a  g  x
## 1  1  2  3
## 2  2  4  6
## 3  4  8 12
## 4  8 16 24
## 5 16 32 48
## 6 32 64 96
```

Todo_11_9.0

Compute the mean of the square root of a vector of 100 random numbers. What happens?

```r
v=c(rnorm(100))
mean(sqrt(v))
```

```
## Warning in sqrt(v): NaNs produced
```

```
## [1] NaN
```

Clearly, output for todo 10 is "NaNs produced", this is because some negative value is appear when we generate 100 random number. In order to solve this problem, we can use script as below

```r
v=c(runif(n=100, min = 0,max = 1))
v1=sqrt(v)
v1
```

```
##   [1] 0.77049964 0.99359312 0.35842641 0.31593238 0.56634372 0.91631068
##   [7] 0.58574914 0.87609573 0.14634193 0.86224726 0.88132902 0.67214610
##  [13] 0.77163060 0.92293196 0.50145495 0.77538031 0.95989013 0.83782392
##  [19] 0.43208669 0.99409711 0.76994297 0.53733210 0.61697271 0.85359546
##  [25] 0.56763742 0.07237582 0.87110189 0.92610843 0.43405601 0.83439925
##  [31] 0.44093086 0.59634700 0.47057030 0.91356349 0.42346496 0.58245167
##  [37] 0.32787632 0.84309359 0.79343483 0.76437095 0.72739362 0.74269440
##  [43] 0.20295330 0.94933253 0.13091622 0.68515985 0.57377444 0.28745935
##  [49] 0.62513811 0.42077760 0.91424281 0.91233380 0.82438801 0.58439830
##  [55] 0.69184364 0.94509379 0.41190922 0.90537194 0.99139744 0.54456105
##  [61] 0.88254799 0.63300522 0.72534877 0.74513190 0.66768339 0.41044943
##  [67] 0.44052340 0.84188066 0.95050675 0.84210931 0.55085516 0.09075487
##  [73] 0.93255812 0.82189004 0.48898893 0.80520023 0.56500263 0.95605101
##  [79] 0.95796676 0.43561614 0.27646581 0.85226494 0.84591703 0.96017913
##  [85] 0.85766664 0.60125610 0.52470692 0.84513942 0.88063453 0.97804827
##  [91] 0.97864327 0.41025339 0.94844414 0.41493722 0.88237497 0.63569430
##  [97] 0.19432089 0.93663652 0.40790758 0.53963312
```

```r
v2=mean(v1)
v2
```

```
## [1] 0.6756627
```

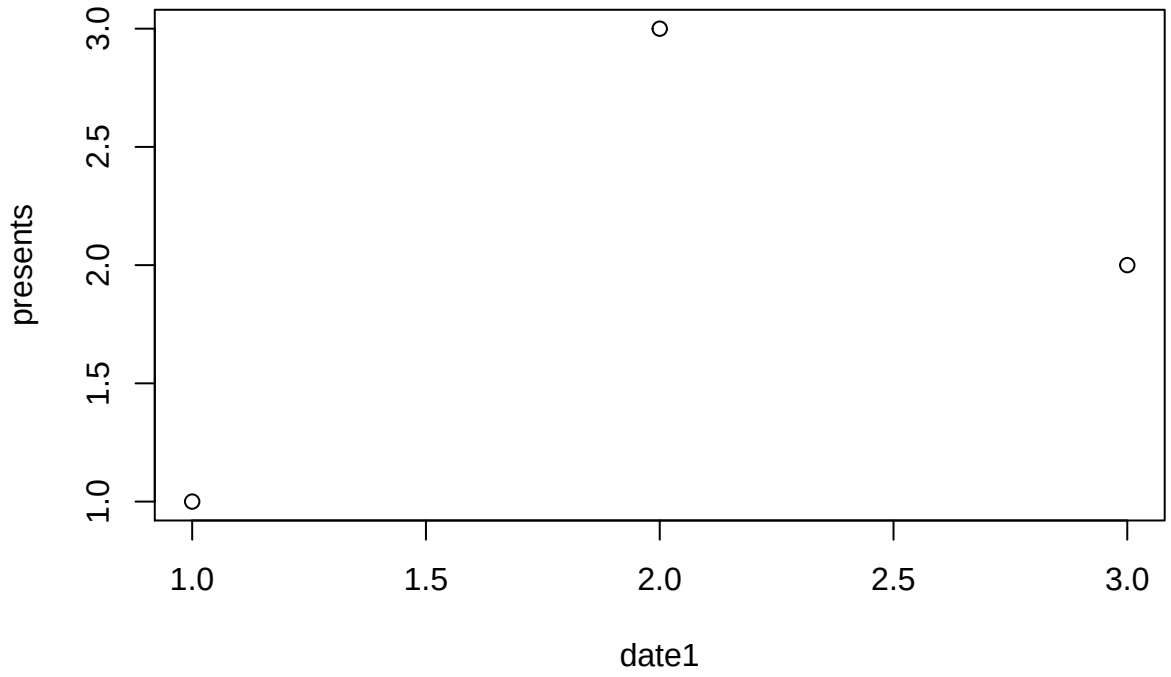
Todo_12_10.2

Make a graph with on the x-axis: today, Sinterklaas 2014 and your next birthday and on the y-axis the number of presents you expect on each of these days. Tip: make two vectors first.

```r
date1=strptime(c("20170202","20171225","20170514"),format = "%Y%m%d")

d<- data.frame(date1=c("20170202","20171225","20170514"),presents=c("3","3","4"))
##d<- data.frame(date1,presents=c("2","3","4"))
x<-d$date1
y<-d$prensents

plot(x,y,xlab="date1",ylab="presents")
```



##

Todo_13_11.2 Make a vector from 1 to 100.Make a for-loop which runs through the whole vector.Multiply the elements which are smaller than 5 and larger than 90 with 10 and the other elements with 0.1.

```
s=c()
for(i in 1:100)
 {if (i<5 | i >90)
   {s[i]=i * 10
   }
  else{
   s[i]=i*0.1
   }
}
s
```

```
##   [1]   10.0   20.0   30.0   40.0    0.5    0.6    0.7    0.8    0.9    1.0
##  [11]    1.1    1.2    1.3    1.4    1.5    1.6    1.7    1.8    1.9    2.0
##  [21]    2.1    2.2    2.3    2.4    2.5    2.6    2.7    2.8    2.9    3.0
##  [31]    3.1    3.2    3.3    3.4    3.5    3.6    3.7    3.8    3.9    4.0
##  [41]    4.1    4.2    4.3    4.4    4.5    4.6    4.7    4.8    4.9    5.0
##  [51]    5.1    5.2    5.3    5.4    5.5    5.6    5.7    5.8    5.9    6.0
##  [61]    6.1    6.2    6.3    6.4    6.5    6.6    6.7    6.8    6.9    7.0
##  [71]    7.1    7.2    7.3    7.4    7.5    7.6    7.7    7.8    7.9    8.0
##  [81]    8.1    8.2    8.3    8.4    8.5    8.6    8.7    8.8    8.9    9.0
##  [91]  910.0  920.0  930.0  940.0  950.0  960.0  970.0  980.0  990.0 1000.0
```

Todo_14_11.3

Write a function for the previous ToDo,so that you can feed it any vector you like (as argument).Use a for-loop in the function to do the computation with each element.Use the standard R function length in the specification of the counter.

```
fun1=function(arg1,arg2)
{
s=c()
for(i in arg1:arg2)
 {if (i<5 | i >90)
   {s[i]=i * 10
   }
  else{
   s[i]=i*0.1
   }
}
s
}
fun1(arg1=4,arg2=65)
```

```
##  [1]   NA   NA   NA 40.0  0.5  0.6  0.7  0.8  0.9  1.0  1.1  1.2  1.3  1.4
## [15]  1.5  1.6  1.7  1.8  1.9  2.0  2.1  2.2  2.3  2.4  2.5  2.6  2.7  2.8
## [29]  2.9  3.0  3.1  3.2  3.3  3.4  3.5  3.6  3.7  3.8  3.9  4.0  4.1  4.2
## [43]  4.3  4.4  4.5  4.6  4.7  4.8  4.9  5.0  5.1  5.2  5.3  5.4  5.5  5.6
## [57]  5.7  5.8  5.9  6.0  6.1  6.2  6.3  6.4  6.5
```