ifgi Institute for Geoinformatics University of Münster
MSc Geospatial Technologies
Spatial Data Science with R
Violeta Ana Luz Sosa León
500435

Assignment 1

The documentation used as a reference in the one published in [R Documentation Org](#) site. Same displayed in the console after running the command for it.

1. Use `rstudio` to write an R script in which you generate a sample of size 1000 from the standard normal distribution (`rnorm`) with mean 50 standard deviation 4, plot a histogram of that sample, and store the graph as pdf in your working directory.

```
###################################
##  ifgi                         ##
##  Spatial Data Science with R  ##
##  Violeta Sosa                  ##
##  Script No. 1                 ##
###################################

library(ggplot2)
setwd("C:\\Users\\Public")
#### Point 1 ####

#Useful to know args from the function rnorm
#?rnorm

#Now we know better how to use it
#Directly
sample<-rnorm(1000,mean=50,sd=4)

#Using variables in case we want to change it later
#size<-1000
#xmean<-50
#xsd<-4
#rnorm(size,mean=xmean,sd=xsd)

# #Ploting Basic Histogram for Sample
sample
#?hist
hist(sample)
plot1<-hist(sample,
     main="Histogram for Sample with Mean 50 and SD 4",
     xlab="Sample generated rnorm function",
     border="darkgreen",
     col="chartreuse4")


#Ploting Histogram for Sample using ggplot
sampledf<-data.frame(sampleplot=sample)
sampledf

#Define arguments for the plotting function
#?geom_histogram
#?labs
ggplot(sampledf, aes(x=sampleplot)) +
geom_histogram(bins=60,col="chartreuse4",fill="darkgreen",alpha=.2)+labs(title="Histo
gram Plot for Sample with mean=40 std=4",x="Sample generated", y = "Count")
#name for the document
#?ggsave
ggsave("samplePDF2.pdf")
#Changes in the bin size for printing
#?stat_bin
```

ifgi Institute for Geoinformatics University of Münster
MSc Geospatial Technologies
Spatial Data Science with R
Violeta Ana Luz Sosa León
500435

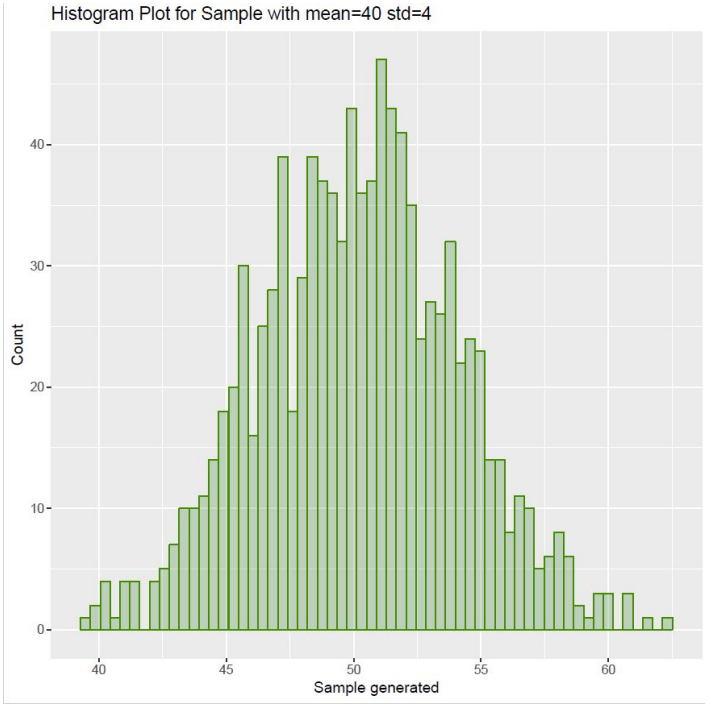


Figure 1. Histogram Plot generated by ggplot2

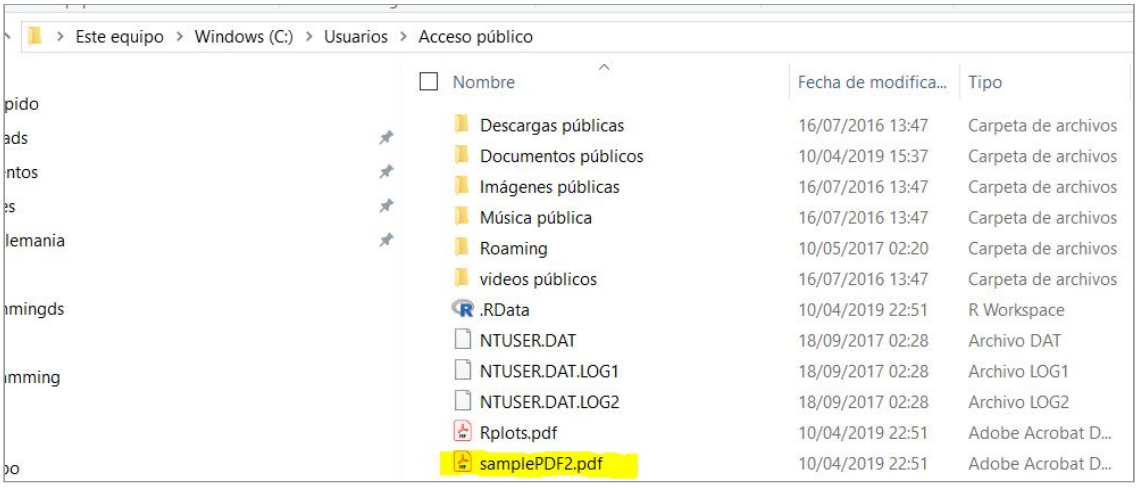


Figure 2. PDF in the working directory

2. Try to execute that script with R (i.e., without Rstudio)

(i) using `source()` and

```
> getwd()
[1] "C:/Users/Violet/Documents"
> source("C:\\Users\\Violet\\Documents\\Master\\Europa\\Clases
Alemania\\R\\Scripts\\programmingds\\ass1_script1.R")
Saving 6.6 x 6.59 in image
```

(ii) in batch mode (`R CMD BATCH file.R`)

For this particular case, using the Windows cmd shell:

```
C:\>cd \

C:\>"C:\Program Files\R\R-3.5.1\bin\R.exe" CMD BATCH
"C:\Users\Violet\Documents\Master\Europa\Clases
Alemania\R\Scripts\programmingds\ass1_script1.R"

C:\>cd \Users\Public

C:\Users\Public>DIR
 El volumen de la unidad C es Windows
 El número de serie del volumen es: 0867-CECC

 Directorio de C:\Users\Public

10/04/2019  22:51    <DIR>          .
10/04/2019  22:51    <DIR>          ..
10/04/2019  22:51            10,628 .RData
10/04/2019  15:37    <DIR>          Documents
16/07/2016  13:47    <DIR>          Downloads
16/07/2016  13:47    <DIR>          Music
18/09/2017  02:28             8,192 NTUSER.DAT
18/09/2017  02:28             8,192 NTUSER.DAT.LOG1
18/09/2017  02:28                 0 NTUSER.DAT.LOG2
16/07/2016  13:47    <DIR>          Pictures
10/05/2017  02:20    <DIR>          Roaming
10/04/2019  22:51             7,038 Rplots.pdf
10/04/2019  22:51             5,266 samplePDF2.pdf
16/07/2016  13:47    <DIR>          Videos
               6 archivos         39,316 bytes
               8 dirs  53,850,599,424 bytes libres
```

3. Explain the difference between a `character array`

```
> x = c("first", "second", "third")
> x
[1] "first" "second" "third"
```

and a factor

```
> y = factor(x)
> y
[1] first second third Levels: first second third
```

The difference between a character array and a factor is that character arrays are nominal and may contain repetitive values. On the other hand, factors are categorical values that identify unique and predefined values.

and explain why

```
> x == y
[1] TRUE TRUE TRUE
```

but

```
> identical(x, y)
[1] FALSE
```

For **x==y** the code will compare element by element for each position per vector.

Following the documentation "*(...) a factor can only be compared to another factor with an identical set of levels (not necessarily in the same ordering) or a character vector (....) the general dispatch mechanism precludes comparing ordered and unordered factors*".[1]

For the identical function **identical(x,y)** the script will check strictly for the data class and type of object.

```
#By using the functions class and type of we can check the
#objects are different

> typeof(x)
[1] "character"

> typeof(y)
[1] "integer"

> class(x)
[1] "character"

> class(y)
[1] "factor"
```

A factor is considered an integer type of object by definition "*factor returns an object of class "factor" which has a set of integer codes the length of x with a "levels" attribute of mode character and unique*"[2] according to R documentation.

[1] *R Documentation. Factor function. From base v3.5.3 by R-core R-core@R-project.org*

[2] *R Documentation. Factor function. From base v3.5.3 by R-core R-core@R-project.org*

ifgi Institute for Geoinformatics University of Münster
MSc Geospatial Technologies
Spatial Data Science with R
Violeta Ana Luz Sosa León
500435

4. Explain the difference between `NA` and `NaN`

`NA stands for Not Available/Missing Values. It is a reserved position in a vector indicating that the value is not yet available, assigned or calculated.`

`NaN stands for Not a Number. It is a derived/calculated value from a computational procedure indicating the impossibility of the result. In this scope, complex numbers are an example.`

`To get detailed information from R documentation, run:`

```
> ?NA
> ?NaN
```

5. Give two ways to create the sequence 1, 4, 7, 10, … with 555 elements,

Complete script without output:

```
> 1+3*0:554
> length(1+3*0:554)
> ?seq
> ?seq.int
> seq(by=3,length.out=555)
> length(1+3*0:554)
> ?seq
> ?seq.int
> seq(by=3,length.out=555)
> #Verify the length
> length(seq(by=3,length.out=555))
> #Documentation points out this is faster. Not to much
differences, our sequence is not that long
> seq.int(by=3,length.out = 555)
> #Verify the length
> length(seq.int(by=3,length.out = 555))
```

ifgi Institute for Geoinformatics University of Münster
MSc Geospatial Technologies
Spatial Data Science with R
Violeta Ana Luz Sosa León
500435

Complete script with output:

one using :

```
> 1+3*0:554
  [1]    1    4    7   10   13   16   19   22   25   28   31   34   37   40   43
46   49   52   55
 [20]   58   61   64   67   70   73   76   79   82   85   88   91   94   97  100
103  106  109  112
 [39]  115  118  121  124  127  130  133  136  139  142  145  148  151  154  157
160  163  166  169
 [58]  172  175  178  181  184  187  190  193  196  199  202  205  208  211  214
217  220  223  226
 [77]  229  232  235  238  241  244  247  250  253  256  259  262  265  268  271
274  277  280  283
 [96]  286  289  292  295  298  301  304  307  310  313  316  319  322  325  328
331  334  337  340
[115]  343  346  349  352  355  358  361  364  367  370  373  376  379  382  385
388  391  394  397
[134]  400  403  406  409  412  415  418  421  424  427  430  433  436  439  442
445  448  451  454
[153]  457  460  463  466  469  472  475  478  481  484  487  490  493  496  499
502  505  508  511
[172]  514  517  520  523  526  529  532  535  538  541  544  547  550  553  556
559  562  565  568
[191]  571  574  577  580  583  586  589  592  595  598  601  604  607  610  613
616  619  622  625
[210]  628  631  634  637  640  643  646  649  652  655  658  661  664  667  670
673  676  679  682
[229]  685  688  691  694  697  700  703  706  709  712  715  718  721  724  727
730  733  736  739
[248]  742  745  748  751  754  757  760  763  766  769  772  775  778  781  784
787  790  793  796
[267]  799  802  805  808  811  814  817  820  823  826  829  832  835  838  841
844  847  850  853
[286]  856  859  862  865  868  871  874  877  880  883  886  889  892  895  898
901  904  907  910
[305]  913  916  919  922  925  928  931  934  937  940  943  946  949  952  955
958  961  964  967
[324]  970  973  976  979  982  985  988  991  994  997 1000 1003 1006 1009 1012
1015 1018 1021 1024
[343] 1027 1030 1033 1036 1039 1042 1045 1048 1051 1054 1057 1060 1063 1066 1069
1072 1075 1078 1081
[362] 1084 1087 1090 1093 1096 1099 1102 1105 1108 1111 1114 1117 1120 1123 1126
1129 1132 1135 1138
[381] 1141 1144 1147 1150 1153 1156 1159 1162 1165 1168 1171 1174 1177 1180 1183
1186 1189 1192 1195
[400] 1198 1201 1204 1207 1210 1213 1216 1219 1222 1225 1228 1231 1234 1237 1240
1243 1246 1249 1252
[419] 1255 1258 1261 1264 1267 1270 1273 1276 1279 1282 1285 1288 1291 1294 1297
1300 1303 1306 1309
[438] 1312 1315 1318 1321 1324 1327 1330 1333 1336 1339 1342 1345 1348 1351 1354
1357 1360 1363 1366
[457] 1369 1372 1375 1378 1381 1384 1387 1390 1393 1396 1399 1402 1405 1408 1411
1414 1417 1420 1423
[476] 1426 1429 1432 1435 1438 1441 1444 1447 1450 1453 1456 1459 1462 1465 1468
1471 1474 1477 1480
[495] 1483 1486 1489 1492 1495 1498 1501 1504 1507 1510 1513 1516 1519 1522 1525
1528 1531 1534 1537
[514] 1540 1543 1546 1549 1552 1555 1558 1561 1564 1567 1570 1573 1576 1579 1582
1585 1588 1591 1594
[533] 1597 1600 1603 1606 1609 1612 1615 1618 1621 1624 1627 1630 1633 1636 1639
1642 1645 1648 1651
[552] 1654 1657 1660 1663
> length(1+3*0:554)
```

```
[1] 555
```

and the other using `seq`

```
> ?seq
> ?seq.int
> seq(by=3,length.out=555)
  [1]    1    4    7   10   13   16   19   22   25   28   31   34   37   40   43
46   49   52   55
 [20]   58   61   64   67   70   73   76   79   82   85   88   91   94   97  100
103  106  109  112
 [39]  115  118  121  124  127  130  133  136  139  142  145  148  151  154  157
160  163  166  169
 [58]  172  175  178  181  184  187  190  193  196  199  202  205  208  211  214
217  220  223  226
 [77]  229  232  235  238  241  244  247  250  253  256  259  262  265  268  271
274  277  280  283
 [96]  286  289  292  295  298  301  304  307  310  313  316  319  322  325  328
331  334  337  340
[115]  343  346  349  352  355  358  361  364  367  370  373  376  379  382  385
388  391  394  397
[134]  400  403  406  409  412  415  418  421  424  427  430  433  436  439  442
445  448  451  454
[153]  457  460  463  466  469  472  475  478  481  484  487  490  493  496  499
502  505  508  511
[172]  514  517  520  523  526  529  532  535  538  541  544  547  550  553  556
559  562  565  568
[191]  571  574  577  580  583  586  589  592  595  598  601  604  607  610  613
616  619  622  625
[210]  628  631  634  637  640  643  646  649  652  655  658  661  664  667  670
673  676  679  682
[229]  685  688  691  694  697  700  703  706  709  712  715  718  721  724  727
730  733  736  739
[248]  742  745  748  751  754  757  760  763  766  769  772  775  778  781  784
787  790  793  796
[267]  799  802  805  808  811  814  817  820  823  826  829  832  835  838  841
844  847  850  853
[286]  856  859  862  865  868  871  874  877  880  883  886  889  892  895  898
901  904  907  910
[305]  913  916  919  922  925  928  931  934  937  940  943  946  949  952  955
958  961  964  967
[324]  970  973  976  979  982  985  988  991  994  997 1000 1003 1006 1009 1012
1015 1018 1021 1024
[343] 1027 1030 1033 1036 1039 1042 1045 1048 1051 1054 1057 1060 1063 1066 1069
1072 1075 1078 1081
[362] 1084 1087 1090 1093 1096 1099 1102 1105 1108 1111 1114 1117 1120 1123 1126
1129 1132 1135 1138
[381] 1141 1144 1147 1150 1153 1156 1159 1162 1165 1168 1171 1174 1177 1180 1183
1186 1189 1192 1195
[400] 1198 1201 1204 1207 1210 1213 1216 1219 1222 1225 1228 1231 1234 1237 1240
1243 1246 1249 1252
[419] 1255 1258 1261 1264 1267 1270 1273 1276 1279 1282 1285 1288 1291 1294 1297
1300 1303 1306 1309
[438] 1312 1315 1318 1321 1324 1327 1330 1333 1336 1339 1342 1345 1348 1351 1354
1357 1360 1363 1366
[457] 1369 1372 1375 1378 1381 1384 1387 1390 1393 1396 1399 1402 1405 1408 1411
1414 1417 1420 1423
[476] 1426 1429 1432 1435 1438 1441 1444 1447 1450 1453 1456 1459 1462 1465 1468
1471 1474 1477 1480
[495] 1483 1486 1489 1492 1495 1498 1501 1504 1507 1510 1513 1516 1519 1522 1525
1528 1531 1534 1537
[514] 1540 1543 1546 1549 1552 1555 1558 1561 1564 1567 1570 1573 1576 1579 1582
1585 1588 1591 1594
[533] 1597 1600 1603 1606 1609 1612 1615 1618 1621 1624 1627 1630 1633 1636 1639
```

```
1642 1645 1648 1651
[552] 1654 1657 1660 1663
> #Verify the length
> length(seq(by=3,length.out=555))
[1] 555
> #Documentation points out this is faster. Not to much
differences, our sequence is not that long
> seq.int(by=3,length.out = 555)
  [1]    1    4    7   10   13   16   19   22   25   28   31   34   37   40   43
46   49   52   55
 [20]   58   61   64   67   70   73   76   79   82   85   88   91   94   97  100
103  106  109  112
 [39]  115  118  121  124  127  130  133  136  139  142  145  148  151  154  157
160  163  166  169
 [58]  172  175  178  181  184  187  190  193  196  199  202  205  208  211  214
217  220  223  226
 [77]  229  232  235  238  241  244  247  250  253  256  259  262  265  268  271
274  277  280  283
 [96]  286  289  292  295  298  301  304  307  310  313  316  319  322  325  328
331  334  337  340
[115]  343  346  349  352  355  358  361  364  367  370  373  376  379  382  385
388  391  394  397
[134]  400  403  406  409  412  415  418  421  424  427  430  433  436  439  442
445  448  451  454
[153]  457  460  463  466  469  472  475  478  481  484  487  490  493  496  499
502  505  508  511
[172]  514  517  520  523  526  529  532  535  538  541  544  547  550  553  556
559  562  565  568
[191]  571  574  577  580  583  586  589  592  595  598  601  604  607  610  613
616  619  622  625
[210]  628  631  634  637  640  643  646  649  652  655  658  661  664  667  670
673  676  679  682
[229]  685  688  691  694  697  700  703  706  709  712  715  718  721  724  727
730  733  736  739
[248]  742  745  748  751  754  757  760  763  766  769  772  775  778  781  784
787  790  793  796
[267]  799  802  805  808  811  814  817  820  823  826  829  832  835  838  841
844  847  850  853
[286]  856  859  862  865  868  871  874  877  880  883  886  889  892  895  898
901  904  907  910
[305]  913  916  919  922  925  928  931  934  937  940  943  946  949  952  955
958  961  964  967
[324]  970  973  976  979  982  985  988  991  994  997 1000 1003 1006 1009 1012
1015 1018 1021 1024
[343] 1027 1030 1033 1036 1039 1042 1045 1048 1051 1054 1057 1060 1063 1066 1069
1072 1075 1078 1081
[362] 1084 1087 1090 1093 1096 1099 1102 1105 1108 1111 1114 1117 1120 1123 1126
1129 1132 1135 1138
[381] 1141 1144 1147 1150 1153 1156 1159 1162 1165 1168 1171 1174 1177 1180 1183
1186 1189 1192 1195
[400] 1198 1201 1204 1207 1210 1213 1216 1219 1222 1225 1228 1231 1234 1237 1240
1243 1246 1249 1252
[419] 1255 1258 1261 1264 1267 1270 1273 1276 1279 1282 1285 1288 1291 1294 1297
1300 1303 1306 1309
[438] 1312 1315 1318 1321 1324 1327 1330 1333 1336 1339 1342 1345 1348 1351 1354
1357 1360 1363 1366
[457] 1369 1372 1375 1378 1381 1384 1387 1390 1393 1396 1399 1402 1405 1408 1411
1414 1417 1420 1423
[476] 1426 1429 1432 1435 1438 1441 1444 1447 1450 1453 1456 1459 1462 1465 1468
1471 1474 1477 1480
[495] 1483 1486 1489 1492 1495 1498 1501 1504 1507 1510 1513 1516 1519 1522 1525
1528 1531 1534 1537
[514] 1540 1543 1546 1549 1552 1555 1558 1561 1564 1567 1570 1573 1576 1579 1582
1585 1588 1591 1594
[533] 1597 1600 1603 1606 1609 1612 1615 1618 1621 1624 1627 1630 1633 1636 1639
1642 1645 1648 1651
```

```
[552] 1654 1657 1660 1663

> #Verify the length
> length(seq.int(by=3,length.out = 555))
[1] 555
```

6. Explain what a `data.frame` is

A data.frame is a two-dimensional object (columns and rows), a collection of "*values of type numeric, character or logical.*"[3] One of many uses of data.frames is to storage observations as rows and particular characteristics of those observations as columns. For this, it is necessary for each column vector to have the same length. The data inside each column usually have the same data type but, different columns can have different data types.

7. Explain the correspondences and differences between `list` and `data.frame` objects

Correspondences

- Both can store different data types objects such as lists, sequences, strings, booleans.
- Both can be handled by the $ operator to get data from the selected vector.

Differences

- In data.frames the length of each column must be the same but Lists' objects can have different lengths
- Data.frames cannot store other data.frames, but List can store data.frames and other lists
- List are considered recursive vectors, one vector can contain other vectors and be independent each from another. On the other side, each one of the vectors in data.frame is related to another value in other columns/vectors. As an illustrative example:

[3] *Introduction to R. Chapter 6. List. Datacamp Course*

```
> lists<-list(4, "string", c(4.5,77,88.99,66),c(FALSE,
TRUE),1:10)
> lists
[[1]]
[1] 4

[[2]]
[1] "string"

[[3]]
[1]  4.50 77.00 88.99 66.00

[[4]]
[1] FALSE  TRUE

[[5]]
 [1]  1  2  3  4  5  6  7  8  9 10

> datafe <- data.frame(a=4:23, b="string",
c=c(4.5,77,88.99,66),d=c(FALSE, TRUE,TRUE,TRUE))
> datafe
    a      b     c     d
1   4 string  4.50 FALSE
2   5 string 77.00  TRUE
3   6 string 88.99  TRUE
4   7 string 66.00  TRUE
5   8 string  4.50 FALSE
6   9 string 77.00  TRUE
7  10 string 88.99  TRUE
8  11 string 66.00  TRUE
9  12 string  4.50 FALSE
10 13 string 77.00  TRUE
11 14 string 88.99  TRUE
12 15 string 66.00  TRUE
13 16 string  4.50 FALSE
14 17 string 77.00  TRUE
15 18 string 88.99  TRUE
16 19 string 66.00  TRUE
17 20 string  4.50 FALSE
18 21 string 77.00  TRUE
19 22 string 88.99  TRUE
20 23 string 66.00  TRUE
```

We can identify an independent definition between the Lists' objects whereas in the data.frame there is a corresponding value defined by the maximum length value from the vector collection.

References

H. Wickham. Chapter 2: Plot Geoms. ggplot2: Elegant Graphics for Data Analysis. April 17, 2017, pp.24 [Link]

H. Wickham. Chapter 8: Themes. ggplot2: Elegant Graphics for Data Analysis. April 17, 2017,pp.191,192 [Link]

K. Willems. Introduction to R: How to Make a Histogram with ggplot2. Datacamp Community Tutorials. March 11th, 2019. [Link]

R Core Team and contributors worldwide. The R Base Package. Version R 3.7.0. 2019 [Link]

Robert I. Kabacoff, Ph.D. Batch Processing. *Quick R by Data Camp*. 2017 [Link]

W. N. Venables, D. M. Smith, and the R Core Team. *Appendix B Invoking R.* An Introduction to R Notes on R: A Programming Environment for Data Analysis and Graphics Version 3.5.3 (2019-03-11), pp. 90 [Link]

Appendix: R Script

```
###################################
##  ifgi                         ##
##  Spatial Data Science with R  ##
##  Violeta Sosa                 ##
##  Script No. 1                 ##
###################################

library(ggplot2)
setwd("C:\\Users\\Public")
#### Exercise 1 ####

#Useful to know args from the function rnorm
#?rnorm

#Now we know better how to use it
#Directly
sample<-rnorm(1000,mean=50,sd=4)

#Using variables in case we want to change it later
#size<-1000
#xmean<-50
#xsd<-4
#rnorm(size,mean=xmean,sd=xsd)

# #Ploting Basic Histogram for Sample
sample
#?hist
hist(sample)
plot1<-hist(sample,
     main="Histogram for Sample with Mean 50 and SD 4",
     xlab="Sample generated rnorm function",
     border="darkgreen",
     col="chartreuse4")


#Ploting Histogram for Sample using ggplot
sampledf<-data.frame(sampleplot=sample)
sampledf

#Define arguments for the plotting function
#?geom_histogram
#?labs
ggplot(sampledf, aes(x=sampleplot)) +
geom_histogram(bins=60,col="chartreuse4",fill="darkgreen",alpha=.2)+labs(title="Hist
ogram Plot for Sample with mean=40 std=4",x="Sample generated", y = "Count")
#name for the document
#?ggsave
ggsave("samplePDF2.pdf")
#Changes in the bin size for printing
#?stat_bin




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

x = c("first", "second", "third")
z = c("second", "first", "third")
x
```

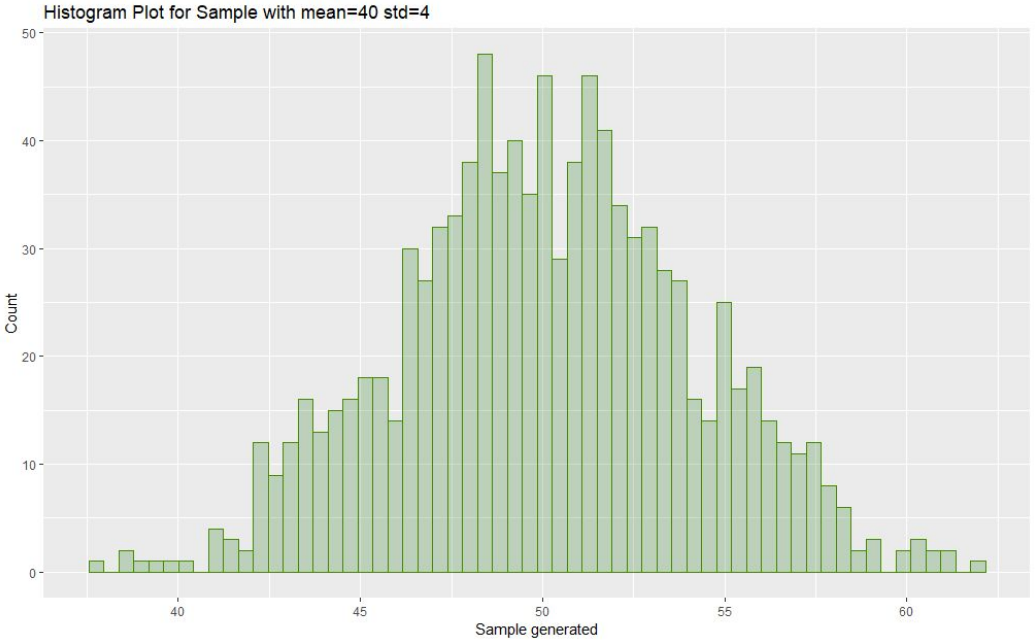
ifgi Institute for Geoinformatics University of Münster
MSc Geospatial Technologies
Spatial Data Science with R
Violeta Ana Luz Sosa León
500435

```
#Using different functions
typeof(x)

y=factor(x)
y
k=factor(z)
typeof(y)

?class
?type
mode(x)
mode(y)
?factor
z
z==k

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

#the sequence increments by multiples of 3: 3,6,9,12,15...

1+3*0:554
length(1+3*0:554)


?seq
?seq.int

seq(by=3,length.out=555)
#Verify the length
length(seq(by=3,length.out=555))
#Documentation points out this is faster. Not to much differences 'cause our
sequence is not that long
seq.int(by=3,length.out = 555)
#Verify the length
length(seq.int(by=3,length.out = 555))




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

lists<-list(4, "string", c(4.5,77,88.99,66),c(FALSE, TRUE),1:10)
lists

# Why here sometimes is ok to have a seq and create the list but others the error
# Error in data.frame(4, "string", c(4.5, 77, 88.99, 66), c(FALSE, TRUE),  :
#         arguments imply differing number of rows: 1, 4, 2, 10
# cannot be handled? Because of the list length? Should not it fill out the next
values from the start?

datafe <- data.frame(a=4:23, b="string", c=c(4.5,77,88.99,66),d=c(FALSE,
TRUE,TRUE,TRUE))
datafe
```

ifgi Institute for Geoinformatics University of Münster
MSc Geospatial Technologies
Spatial Data Science with R
Violeta Ana Luz Sosa León
500435

Appendix: Histogram Plots

ggplot2 library



Native histogram plot