Lab 5: Import, Select, Output Data and Data frame

Introduction and Objective:

In Lab 4, we took a glance at how to import data into R. In today's lab, we are going to continue our exploration of importing data and selecting data. Furthermore, we are going to learn the data frame and how to generate a data frame (data table), and then save it for future use.

All the R command lines are in **bold**; all the notes are following a #; all the R results directly follow the R codes/command lines and are not in bold or following a #.

Files you need for today's lab:

1) Mouse_DNA_methylation_small_table.txt
2) Mouse_DNA_methylation_small_table.csv
3) MidtermGradeSheet.csv

setwd("C:/R")

If you are using a Mac set the working directory as we learned in the first lab.

Extracting data from a sample table "Mouse_DNA_methylation_small_table"

I. # In Lab 4, we learned how to use "read.csv()" to extract all the data from a csv file. Now, we are going to use "read.table()" to import all the data from a text file.

a<-read.table("Mouse_DNA_methylation_small_table.txt")

View(a)

The function "View()" is for showing the actual data table. When you type this function, make sure the first letter is an upper case "V".

dim(a)

[1] 31 13

The function "dim()" is for showing the dimension of a data table.

The first number, here it is 31, tells you how many rows there are in this data table. The second number, which is 13 in this case, tells you how many columns there are in this data table.

head(a)

The function “head()” gives you the header and the first six rows of the data table. So based on these six rows, you can have an idea of what this data looks like:

```
        V1       V2       V3       V4       V5       V6       V7
1 Name_GeneA GF_Mouse1 GF_Mouse2 GF_Mouse3 GF_Mouse4 GF_Mouse5
GF_Mouse6
2   Fam189a1     0.97     0.78     1.25     0.76      1.1     1.92
3       Ncf1        5     5.24     2.81     5.93     2.76     3.15
4   Arhgap33    32.88    40.63    34.43    39.97     36.9    34.71
5   Fam189a1      1.1     0.59     0.63     2.48     0.97     1.65
6    L3mbtl4     0.67     1.16     0.73     1.53     0.55     0.91
        V8        V9       V10       V11       V12       V13
1 SPF_Mouse1 SPF_Mouse2 SPF_Mouse3 SPF_Mouse4 SPF_Mouse5 SPF_Mouse6
2      4.09      4.46      4.34       4.7      4.67      4.14
3     17.46     15.05     16.82     19.46     16.24      16.9
4     57.74     57.42      61.3     56.47     57.87     59.39
5      6.69      7.86      7.08      5.97      7.56      7.82
6        9.69      8.52       7.9     11.22      12.1      10.1
```

All these V2, V3, etc., are the variable names given by R, as R does not know what variable name is for each of the variables. If we don’t use the argument header=TRUE to specify the variables when we import data into R using read.table(), by default, R will generate a new row that has V#s as the variable names.

Try the command lines as follows:

```
a<-read.table("Mouse_DNA_methylation_small_table.txt", header=TRUE)
head(a)
        Name_GeneA GF_Mouse1 GF_Mouse2 GF_Mouse3 GF_Mouse4 GF_Mouse5
GF_Mouse6 SPF_Mouse1 SPF_Mouse2
      1   Fam189a1      0.97      0.78      1.25      0.76      1.10      1.92       4.09       4.46
      2       Ncf1      5.00      5.24      2.81      5.93      2.76      3.15      17.46      15.05
      3   Arhgap33     32.88     40.63     34.43     39.97     36.90     34.71      57.74      57.42
      4   Fam189a1      1.10      0.59      0.63      2.48      0.97      1.65       6.69       7.86
      5    L3mbtl4      0.67      1.16      0.73      1.53      0.55      0.91       9.69       8.52
      6     Nedd4l     95.06     93.81     95.88     91.41     92.25     90.24      66.57      63.09
```

	SPF_Mouse3	SPF_Mouse4	SPF_Mouse5	SPF_Mouse6
1	4.34	4.70	4.67	4.14
2	16.82	19.46	16.24	16.90
3	61.30	56.47	57.87	59.39
4	7.08	5.97	7.56	7.82
5	7.90	11.22	12.10	10.10
6	64.13	74.25	63.48	61.67

From the above results, you can see the automatically generated variables names like V1, V2, V3 … are all disappeared. Instead, R recognized the first row in the data table as the row for variable names.

II. # We learned how to use "read.csv()" or "read.table()" to import all the data either from a csv data file or from a text file. Now let's take a look at how to selectively extract data using "data.frame()".

A data frame in R is a format of data with all the variables as the columns, and all the columns have the same length. This important concept was also adopted by many other data formats can be processed by R. For example, in the future, when we talk about tibble and the R package tidyverse, you will see tibble, a popular format of data presentation, adopted the idea of the conventional data frame, meaning it also uses variables as the columns, and all the columns have the same length.

```
b<-data.frame(a[2,2:6],a[3,7:13])
```

What does the above command line mean?

It means, you are using data.frame() to select the numbers in the table "a" from column 2 to column 6 in row 2, and then combine them with the values from column 7 to column 13 in row 3 of table a.

Challenge Question 1: How many numbers are selected by this command line?

If you don't know, you can simply type "b" to see which numbers and how many have been selected:

b

```
   V2   V3   V4   V5  V6   V7    V8    V9   V10   V11   V12  V13
2 0.97 0.78 1.25 0.76 1.1 3.15 17.46 15.05 16.82 19.46 16.24 16.9
```

Instead of using "read.table()" and "data.frame()", you can also do this by using "read.csv()" followed by "data.frame()".

Try and see the difference:

a<-read.csv("Mouse_DNA_methylation_small_table.csv")

dim(a)

```
[1] 30 13
```

head(a)

```
  Name_GeneA GF_Mouse1 GF_Mouse2 GF_Mouse3 GF_Mouse4 GF_Mouse5
GF_Mouse6
1   Fam189a1      0.97      0.78      1.25      0.76      1.10      1.92
2       Ncf1      5.00      5.24      2.81      5.93      2.76      3.15
3   Arhgap33     32.88     40.63     34.43     39.97     36.90     34.71
4   Fam189a1      1.10      0.59      0.63      2.48      0.97      1.65
5    L3mbtl4      0.67      1.16      0.73      1.53      0.55      0.91
6     Nedd4l     95.06     93.81     95.88     91.41     92.25     90.24
  SPF_Mouse1 SPF_Mouse2 SPF_Mouse3 SPF_Mouse4 SPF_Mouse5 SPF_Mouse6
1       4.09       4.46       4.34       4.70       4.67       4.14
2      17.46      15.05      16.82      19.46      16.24      16.90
3      57.74      57.42      61.30      56.47      57.87      59.39
4       6.69       7.86       7.08       5.97       7.56       7.82
5       9.69       8.52       7.90      11.22      12.10      10.10
6      66.57      63.09      64.13      74.25      63.48      61.67
```

III. # The different conventional ways R allows us to select data.

1) # You can select data by telling R which columns and rows you'd like to choose:

a2<-a[,c(3,9)]

Notice that there is nothing before the comma, which means we do not specify which rows to select but select all the rows.

dim(a2)

[1] 30 2

head(a2)

```
  GF_Mouse2 SPF_Mouse2
1      0.78       4.46
2      5.24      15.05
3     40.63      57.42
4      0.59       7.86
5      1.16       8.52
6     93.81      63.09
```

2) # You can also tell R the names of the variables:

a1<-a[,c("GF_Mouse2", "SPF_Mouse2")]

dim(a1)

[1] 30 2

head(a1)

```
  GF_Mouse2 SPF_Mouse2
1      0.78       4.46
2      5.24      15.05
3     40.63      57.42
4      0.59       7.86
5      1.16       8.52
6     93.81      63.09
```

3) # Use function "subset()" to conditionally select data from a data frame.

For example, we select the data, only if the gene has more than 40% DNA methylation in Germ-free mouse #2:

a4<-subset(a,GF_Mouse2>40)

dim(a4)

[1] 13 13

head(a4)

```
  Name_GeneA GF_Mouse1 GF_Mouse2 GF_Mouse3 GF_Mouse4 GF_Mouse5
GF_Mouse6
```

```
3    Arhgap33    32.88    40.63    34.43    39.97    36.90    34.71
6     Nedd4l    95.06    93.81    95.88    91.41    92.25    90.24
10     Esm1    70.72    61.86    67.73    56.94    64.84    60.93
14      Txk    57.92    46.21    60.42    46.49    50.51    57.16
15   Cbfa2t3    55.85    61.20    51.83    55.62    57.15    54.01
16    Sh2b3    52.70    60.07    54.31    57.92    59.40    57.67
  SPF_Mouse1 SPF_Mouse2 SPF_Mouse3 SPF_Mouse4 SPF_Mouse5 SPF_Mouse6
3     57.74     57.42     61.30     56.47     57.87     59.39
6     66.57     63.09     64.13     74.25     63.48     61.67
10    35.91     28.62     31.96     38.97     29.60     30.70
14    14.47     18.73     22.47     20.11     10.13     16.89
15    12.40     16.84     27.45     29.80     20.40     16.28
16    84.30     77.17     75.73     76.70     84.34     83.64
```

4) # If we want to further narrow down the data, for instance, we are interested in the DNA methylation of the genes in SPF mouse #3, #4, and #5, when the DNA methylation of these genes have more than 40% methylation in Germ-free mouse #2:

```
a5<-subset(a,GF_Mouse2>40,
select=c("SPF_Mouse3","SPF_Mouse4","SPF_Mouse5"))
dim(a5)
[1] 13  3

head(a5)
  SPF_Mouse3 SPF_Mouse4 SPF_Mouse5
3     61.30     56.47     57.87
6     64.13     74.25     63.48
10    31.96     38.97     29.60
14    22.47     20.11     10.13
15    27.45     29.80     20.40
16    75.73     76.70     84.34
```

5) # The above result does not show the gene names. If we desire to know the gene names, then we need to include the variable name corresponding to the column for the gene names and can type:

```
a6<-subset(a,GF_Mouse2>40,
select=c("Name_GeneA","SPF_Mouse3","SPF_Mouse4","SPF_Mouse5"))
```

dim(a6)

[1] 13 4

head(a6)

	Name_GeneA	SPF_Mouse3	SPF_Mouse4	SPF_Mouse5
3	Arhgap33	61.30	56.47	57.87
6	Nedd4l	64.13	74.25	63.48
10	Esm1	31.96	38.97	29.60
14	Txk	22.47	20.11	10.13
15	Cbfa2t3	27.45	29.80	20.40
16	Sh2b3	75.73	76.70	84.34

IV. # After you generate a data frame (data table), you may want to save it for future use. Now, I am showing you how to save the created data frame.

We can save the data table by either using the function “write.table()” or “write.csv()”.

write.table(a6,"GF2_40%asCutOff_3SPF.txt",sep="\t",row.names=FALSE)

The argument sep=”\t” means “separating 2 data values with a tab”. You can also specify how to separate the data values, such as using commas.

Challenge Question 2: Where in your computer can you find this table you just created? Where is your saved data frame (data table)? Which folder has this table been saved to?

You can type the following command line and see R does read the table.

b1<-read.table("GF2_40%asCutOff_3SPF.txt")

b1

	V1	V2	V3	V4
1	Name_GeneA	SPF_Mouse3	SPF_Mouse4	SPF_Mouse5
2	Arhgap33	61.3	56.47	57.87
3	Nedd4l	64.13	74.25	63.48
4	Esm1	31.96	38.97	29.6
5	Txk	22.47	20.11	10.13
6	Cbfa2t3	27.45	29.8	20.4

7	Sh2b3	75.73	76.7	84.34
8	Hk2	44.78	50.84	29.87
9	Mxra7	79.78	81.4	75.21
10	Bahcc1	33.78	45.46	41.53
11	Evpl	57.69	56.99	56.31
12	H2-Eb1	78.84	82.79	84.82
13	Mfsd1	71.75	78.14	68.63
14	Rfx3	92.48	89.34	90.08

If you want to export the original row numbers, then use row.name=TRUE (tell R to keep these row numbers):

write.table(a6,"GF2_40%asCutOff_3SPF.txt",sep="\t",row.names=TRUE)

b1<-read.table("GF2_40%asCutOff_3SPF.txt")

b1

	Name_GeneA	SPF_Mouse3	SPF_Mouse4	SPF_Mouse5
3	Arhgap33	61.30	56.47	57.87
6	Nedd4l	64.13	74.25	63.48
10	Esm1	31.96	38.97	29.60
14	Txk	22.47	20.11	10.13
15	Cbfa2t3	27.45	29.80	20.40
16	Sh2b3	75.73	76.70	84.34
17	Hk2	44.78	50.84	29.87
20	Mxra7	79.78	81.40	75.21
21	Bahcc1	33.78	45.46	41.53
25	Evpl	57.69	56.99	56.31
26	H2-Eb1	78.84	82.79	84.82
28	Mfsd1	71.75	78.14	68.63
29	Rfx3	92.48	89.34	90.08

You can create a data frame and save it using write.csv() as well:

write.csv(a6,"GF2_40%asCutOff_3SPF.csv",row.names=T)

V. A small project using data.frame

We had our first lecture exam. We got a table (not for this year) that shows the students' IDs (faked ones) and the scores. Now, we would like to know the rank of the scores, and then generate a new table having a column showing the rank.

```
a<-read.csv("MidtermGradeSheet.csv")

mode(a)
[1] "list"

dim(a)
[1] 37  2

head(a)
   Student.ID Midterm
1          AE    81.0
2          AN    78.5
3          AS    82.0
4          CA    94.0
5          CF    77.0
6          DN    87.0

b<-rank(a[,2])

b
 [1] 15.5 13.0 19.0 35.0 11.5 24.0 37.0  8.0 10.0  5.0 14.0 36.0 17.5  2.0 33.0
[16] 25.0 27.0 20.0 30.0 11.5 22.0 22.0 17.5  9.0  4.0  7.0 34.0 28.5 32.0 22.0
[31] 15.5 28.5  3.0  1.0 26.0  6.0 31.0
```

Firstly, you may notice that the rank by default is ascending, meaning the highest score has the highest number in rank. Secondly, the rank number is the average of places. For example, we had two students who scored 81, and the rank of them showing here is 15.5, which is not an integer. This actually shows the places of these two students are 15 and 16 (they took the places of 15 and 16, but their scores are a tie, it is hard to say who is taking the place of 15.)

Challenge Question 3: If you have three students who scored the same grade, what the rank number would be for these 3 students. For example, if there are 3 students who scored 81, what the rank number would be for these students? Will it be 15.34? (Hint: the rank number is the average.)

Challenge Question 4: How can you get the rank in descending order?

You can try this in R by yourself (Modify the data and change one number to 81, and then do the rank again. You will see it.)

Now, you make the rank as a column of your new data frame:

b1<-data.frame(b)
b1

```
    b
1  15.5
2  13.0
3  19.0
4  35.0
5  11.5
6  24.0
7  37.0
8   8.0
9  10.0
10  5.0
11 14.0
12 36.0
13 17.5
14  2.0
15 33.0
16 25.0
17 27.0
18 20.0
19 30.0
20 11.5
21 22.0
22 22.0
23 17.5
24  9.0
25  4.0
26  7.0
27 34.0
28 28.5
29 32.0
30 22.0
31 15.5
32 28.5
```

33 3.0
34 1.0
35 26.0
36 6.0
37 31.0

The column name of the above result is “b”. If you want to change the column name from “b” to “rank” to make more sense to you, you can type the following command line:

colnames(b1)<-c("rank")

The function “colnames()” allows you to name the columns in a data frame.

b1

rank
1 15.5
2 13.0
3 19.0
4 35.0
5 11.5
6 24.0
7 37.0
8   8.0
9 10.0
10 5.0
11 14.0
12 36.0
13 17.5
14 2.0
15 33.0
16 25.0
17 27.0
18 20.0
19 30.0
20 11.5
21 22.0
22 22.0
23 17.5
24 9.0
25 4.0
26 7.0

27 34.0
28 28.5
29 32.0
30 22.0
31 15.5
32 28.5
33 3.0
34 1.0
35 26.0
36 6.0
37 31.0

You can also use “rownames()” to change the names of the rows in a data frame.

Once you change the column name from “b” to “rank”, you may want to assemble your new data frame to include the student IDs and their corresponding rank by using the function “data.frame()”.

c<- data.frame(a,b1)

dim(c)

[1] 37 3

head(c)

	Student.ID	Midterm	rank
1	AE	81.0	15.5
2	AN	78.5	13.0
3	AS	82.0	19.0
4	CA	94.0	35.0
5	CF	77.0	11.5
6	DN	87.0	24.0

Thus, you generated a new data frame with a column named as “rank”. You may want to save this table, by typing the following command:

write.csv(c,"MidtermWithRank.csv")

Answer keys to the challenge questions:

Challenge question 1: 12
Challenge question 2: In the folder, which you designated as the working directory.

Challenge question 3:

Actually, it is not 15.34. The three students took the places of 15, 16, and 17. Since the rank number is the average of the places, the rank number for these students will be (15+16+17)/3=16

Challenge question 4:

The easiest way is to put a minus before the data you want to rank, for example, rank(-a)