Course: R Language in Computational Probability and Statistics

Lecture 5: Data import/export. Creating own functions

Lecturer: Nataliia Kruhlova

Імпорт/експорт даних у форматі *.Rdata

Іноді необхідно зберегти певні результати роботи програми у форматі R, наприклад, для подальшого обміну змінними з іншими користувачами. Це можна виконати за допомогою функції **save()**.

Приклад.

```
>value<-seq(-10,10,by=2)
>save(value,file="D:/R/Result1.Rdata")
```

Ми спочатку створили послідовність цілих парних чисел **value**, а потім ці дані зберегли у файл Result1.Rdata у каталозі R на диску D. Увага! Символ / застосовується для запису шляху до файлу.

Команда **save()** буде зберігати об'єкти у форматі, який можна прочитати лише середовищем R, текстові редактори будуть зчитувати файл як послідовність незрозумілих символів. Щоб завантажити файл у внутрішньому форматі, використовують функцію **load()**.

Приклад.

```
>value_new<-load(file="D:/R/Result1.Rdata")
```

Функція **load** у об'єкт **value_new** записує дані із змінної **value**, оскільки об'єкти у внутрішньому форматі зберігаються зі своїми ідентифікаторами. Якщо необхідно в одному файлі зберегти декілька змінних, то їх подають в якості параметрів у функцію **save()** через кому**.**

Функція **file.choose()**, запускає вікно вибору файлу з внутрішнього сховища комп'ютера, після чого користувач самостійно вибирає потрібний файл із теки Щоб завантажити об'єкт із файлу, який обирається вручну, потрібно виконати команду: **load(file=file.choose()).**

Імпорт/експорт текстових табличних даних

Практично будь-яка програма для статистичного аналізу даних може створювати та зчитувати файли даних у формі текстових таблиць. Ці файли можна легко прочитати у стандартних текстових редакторах, тому їх використовують для обміну статистичними даними між програмами.

Припустимо, що таблиця збережена у текстовому файлі.

```
      Name Функан Ймовірність Фізика Філософія
1  Fediyko     57         60     64       77
2 Petrenko     63         58     60       88
3  Vasylko     67         54     78       66
```

Щоб завантажити текстову таблицю, використовується функція **read.table().**

Приклад. Прочитайте текстову таблицю, що записана у файлі **D:/R/Some_table.tx**t.

```
>tab1<-read.table(file=”D:/R/Some_table.txt”,header=T
)
>tab1
Name Функан Ймовірність Фізика Філософія
1  Fediyko     57         60     64       77
2 Petrenko     63         58     60       88
3  Vasylko     67         54     78       66
```

В цьому прикладі ми у новий набір даних **tabl** завантажили таблицю із текстового файлу. Параметр **header=T** в перший рядок фрейму помістить імена змінних. Якщо ми хочемо деякий стовпчик таблиці зробити іменами рядків набору даних, то задається параметр **row.names**, в якому вказується назва стовпчика імен або його номер.

Приклад.

```
> tab2<-read.table(file="D:/R/Some_table.txt",header=T,row.names="Name")
```

```
> tab2
        row.names Функан Ймовірність Фізика Філософія
Fediyko         1     57         60     64       77
Petrenko        2     63         58     60       88
Vasylko         3     67         54     78       66
```

Для запису текстової таблиці використовується функція **write.table()**:
Приклад.

```
> write.table(tab2,file="D:/R/Result_some.txt")
```

За замовчуванням, у першому рядку записуються імена змінних набору даних, у першому стовпчику зберігаються назви рядків. Якщо задати значення параметрів **row.names=F**,**col.names=F**, то перший рядок і перший стовпчик не будуть вважатися іменами даних.
Функція **read.table** автоматично визначає кількість стовпчиків за кількістю імен у першому рядку набору даних. Тип змінних залежить від формату даних у стовпчику. Наприклад, якщо більшість елементів стовпчика мають числовий формат, а хоча б один елемент буде символьного типу, то весь стовпчик збережеться як змінна символьного типу. Якщо при наборі даних у таблиці виявляться рядочки з різною кількістю елементів, то це призведе до помилки зчитування файлу. Символьні рядки, що містять пробіли між словами, мають записуватись у лапках. Наприклад:

Ім’я	Зріст	Абонемент
Ольга	169	F
"Вася Пупкін"	183	F
Ірина	159	T
"Микола К."	194	F

Мова програмування R може експортувати та імпортувати дані у форматі **csv (comma separated values).** В цьому форматі окремі елементи змінної розділяються комами, а дробова частина числа відділяється від цілої крапкою. Цей формат даних дуже поширений для обміну між різними статистичними програмами. Для запису файлу у цьому форматі використовується функція **write.csv**, а для зчитування **- read.csv.** Ці функції мають синтаксис схожий із функціями для запису та зчитування текстових таблиць, а відмінність полягає лише в значеннях параметрів за умовчанням для регулювання розділових символів. В R реалізована можливість змінювати значення цих параметрів для запису або зчитування таблиць нестандартного формату. Параметр **dec** відповідає за зміну символа, що відділяє цілу частину числа від дробової.

Подібним до **csv** є формат **csv2,** в якому ціла частина числа відділяється від дробової комою, а дані відокремлюються між собою крапкою з комою. Функція **read.csv2()** призначена для зчитування даних цього формату, а функція

write.csv2() використовується для запису таких даних. Як відомо, дані у форматі **csv, csv2** можна зберегти у програмі **EXCEL.**

У функції **read.table** параметр **file** може бути не тільки іменем файлу на комп'ютері. Значення параметру **file="clipboard**", дозволяє зчитати таблицю із буферу обміну операційної системи.

Приклад. Створіть в **EXCEL** таблицю, виділіть її частину та скопіюйте у буфер обміну

```
> tab3<-read.table(file="clipboard",dec=",")
> tab3
        V1          V2    V3   V4   V5   V6   V7   V8   V9
1 Владислав Віталійович 10.46 3.75 5.31 3.22 1.88 3.79 5.09
2    Андрій    Юрійович  7.50 3.10 4.46 2.31 1.55 3.19 3.86
3    Нікіта  Русланович  8.54 3.71 3.45 2.63 1.86 2.46 4.48
4 Володимир  Михайлович  9.09 4.29 6.06 2.80 2.15 4.33 4.94
```

Щоб прочитати таблицю з інтернету, ми маємо вказати у відповідній команді для зчитування URL даних. Але дані мають бути у форматі txt, csv, csv2, xlsx.

Приклад. Зчитайте таблицю з інтернет-ресурсу [1] https://sdmx.data.unicef.org/ws/public/sdmxapi/rest/data/UNICEF,GENDER,1.0/.GN_UNMPLY...?format=csv. Виведіть перших 5 рядків і 5 стовпчиків набору даних.

```
> tab4<-read.csv("https://sdmx.data.unicef.org/ws/public/sdmxapi/rest/data/UNIC
EF,GENDER,1.0/.GN_UNMPLY...?format=csv")
> tab4[1:5,1:5]
  REF_AREA Geographic.area INDICATOR                 Indicator SEX
1      AFG     Afghanistan GN_UNMPLY Labour force unemployment rate   M
2      AFG     Afghanistan GN_UNMPLY Labour force unemployment rate  _T
3      AFG     Afghanistan GN_UNMPLY Labour force unemployment rate   M
4      AFG     Afghanistan GN_UNMPLY Labour force unemployment rate   F
5      AFG     Afghanistan GN_UNMPLY Labour force unemployment rate   M
```

Експорт та імпорт даних із файлів із розширенням **.xlsx** можна здійснювати командами **read.xlsx(), write.xlsx(),** попередньо під'єднавши пакет **xlsx**. Аналогічні функції для роботи з даними такого формату є у пакеті **openxlsx: read.xlsx, write.xlsx.**

Приклад.

```
> library(openxlsx)
> write.xlsx(tab4[1:5,1:5],"D:/R/Tab_new.xlsx")
> read.xlsx("D:/R/Tab_new.xlsx")
  REF_AREA Geographic.area INDICATOR                  Indicator SEX
1      AFG     Afghanistan GN_UNMPLY Labour force unemployment rate   M
2      AFG     Afghanistan GN_UNMPLY Labour force unemployment rate  _T
3      AFG     Afghanistan GN_UNMPLY Labour force unemployment rate   M
4      AFG     Afghanistan GN_UNMPLY Labour force unemployment rate   F
5      AFG     Afghanistan GN_UNMPLY Labour force unemployment rate   M
```

Створення власних функцій

Створення у **R** власних функцій передбачає створення нового об'єкту типу **function,** який може бути привласнений новій змінній. Загальний синтаксис має вигляд:

Some_function<-function(Параметр 1, Параметр 2,…)
{ тіло функції
Результат}.

В тілі функції перераховуються команди, що будуть виконуватися над локальними параметрами функції.

Значенням функції буде результат останньої команди в тілі функції. Викликаючи функцію, фактичні значення параметрів заміняють формальні параметри в тілі функції.

Приклад. Потрібно створити функцію, яка рахує площу трикутника за формулою Герона. Аргументами функції будуть довжини сторін трикутника.

```
> geron<-function(A,B,C){
+   p<-(A+B+C)/2
+   s<-sqrt(p*(p-A)*(p-B)*(p-C)
)
+   s
+ }
> geron(2,3,4)
[1] 2.904738
```

Результатом функції може бути число, логічна змінна, список чи вектор.

Приклад. Потрібно створити функцію, яка приймає в якості вхідних параметрів два цілих числа m і n, а вертає вектор, що містить всі дроби виду {i/m, i = 0, 1, ..., m} і {j/n, j = 0, 1, ..., n}. Вектор не повинен містити повторів і має бути впорядкований в порядку спадання.

```
>fractions <- function(m, n) {
+     m1<-(0:m)/m
+     n1<-(0:n)/n
+     return(sort(unique(c(m1,n1)),decreasing=T))
+ }
>fractions(5,8)
 [1] 1.000 0.875 0.800 0.750 0.625 0.600 0.500 0.400 0.375 0.250 0.200 0.12
5
[13] 0.000
```

Приклад. Якщо вектор має велику довжину, то важко оцінити, які він містить елементи, і скільки разів вони повторюються. В цьому випадку зручно виводити таблицю частот елементів. Нехай x – вектор, елементами якого є цілі числа. Напишіть функцію, яка поверне матрицю з такими рядками: в першому рядку перераховані всі різні елементи вектору, впорядковані в порядку зростання, в другому рядку вказано кількість повторів елементів.

```
>elements <- function(x) {
+     n<-table(x)
+     return(rbind(as.numeric(names(n)),unname(n)))
+ }
> x <- c(5, 2, 7, 7, 7, 2, 0, 0)
>elements(x)
     [,1] [,2] [,3] [,4]
[1,]    0    2    5    7
[2,]    2    2    1    3
```

Приклад. Напишіть функцію, яка приймає на вхід вектор, всі елементи якого є цілими числами, і число, а повертає вектор індексів елементів вектору, які найближче до вказаного числа (модуль різниці між числами найменший). Якщо таких елементів декілька, то потрібно вказати всі позиції. Індекси вивести в порядку зростання.

```
>closest <- function(v, n) {
+     return(which(abs(v-n)==min(abs(v-n))))
+ }
>
> v <- c(5, 2, 7, 7, 7, 2, 0, 0)
> n<-1
>closest(v,n)
[1] 2 6 7 8
```

Можна задати параметри їх фактичним значенням і змінювати порядок параметрів.

Приклад.

```
>closest(n=5,c(7,-4,3,8,-2,0))
[1] 1 3
```

Якщо певні параметри(опції) вже задано, то **R** дозволяє задавати лише необхідні параметри, викликаючи функцію.

Приклад.

```
> closest <- function(v, n=6) {return(which(abs(v-n)==min(abs(v-n))))}
> closest(c(-4:4))
[1] 9
```

Параметрами функції можуть бути і вирази.

Приклад.

```
> closest <- function(v,n=mean(v)) {return(which(abs(v-n)==min(abs(v-n))))}
> closest(c(0,-3,4,8,2,5,6,-2))
[1] 5
```

Якщо попередній приклад сформулювати таким чином: потрібно вивести індекси елементів, що знаходяться найближче до середнього значення вектору, то функцію можна переписати так, що вона буде залежати тільки від одного параметра, а інший стане внутрішньою змінною.

Приклад.

```
> closest <- function(v) {n<-mean(v);return(which(abs(v-n)==min(abs(v-n))))}
> closest(c(-2,3,4,7,3,8,7,4,-8))
[1] 2 5
```

Зауважимо, що **n** є локальною змінною і виклик функції її не змінює.

Приклад.

```
> n
[1] 100
> mean(c(-2,3,4,7,3,8,7,4,-8))
[1] 2.88888
```

У цьому прикладі **n** - глобальна змінна.

Для зміни глобальної змінної використовується глобальне привласнення <<-.

Приклад.

```
> n<-100
> closest <- function(v) {n<<-mean(v);return(which(abs(v-n)==min(abs(v-n))))}
> closest(c(-2,3,4,7,3,8,7,4,-8))
[1] 2 5
> n
[1] 2.888889
```

Але глобальне привласнення краще не використовувати.

Список джерел

1) The Humanitarian Data Exchange. https://data.humdata.org/