

Lesson 9: Read and write files in R

Modesto

2022-10-25

Contents

Wellcome & Disclaimer	1
Data input & output in R	1
Working directory	2
Reading/writing data in R	3
Basic Data Management in R	4
Explore a dataframe	4
Change and add variables	9
Dealing with <i>NAs</i>	9
References	11
Short exercises	12

Wellcome & Disclaimer

This site contains the materials for the *Coding tools for Biochemistry & Molecular Biology* (Herramientas de Programación para Bioquímica y Biología Molecular) course of fall 2022 in the Bachelor's Degree in Biochemistry @UAM. This materials are the basis for GitHub-pages-based website that can be accessed here. Detailed academic information about the course contents, dates and assessment only can be found at the UAM Moodle site.

All this material is open access and it is shared under CC BY-NC license.

Data input & output in R

As you already know, launching R starts an interactive session with input from the keyboard and output to the screen. If you are using small datasets, you can directly define and introduce your data in the Console, as you did in the examples before. Additionally, you can define your objects and introduce your data interactively with the functions `scan()` and `readline()` as in the following examples. Regarding the output, you can just call the object by its name or use the function `print()`, which displays on the screen the contents of its argument object. Although, seldom used, you can also edit the contents of your objects using the function `edit()`.

```
vector <- scan(n = 4)
vector2 <- scan()
str <- readline()
vector
```

```
## [1] 4 5 6 7
print(vector2)

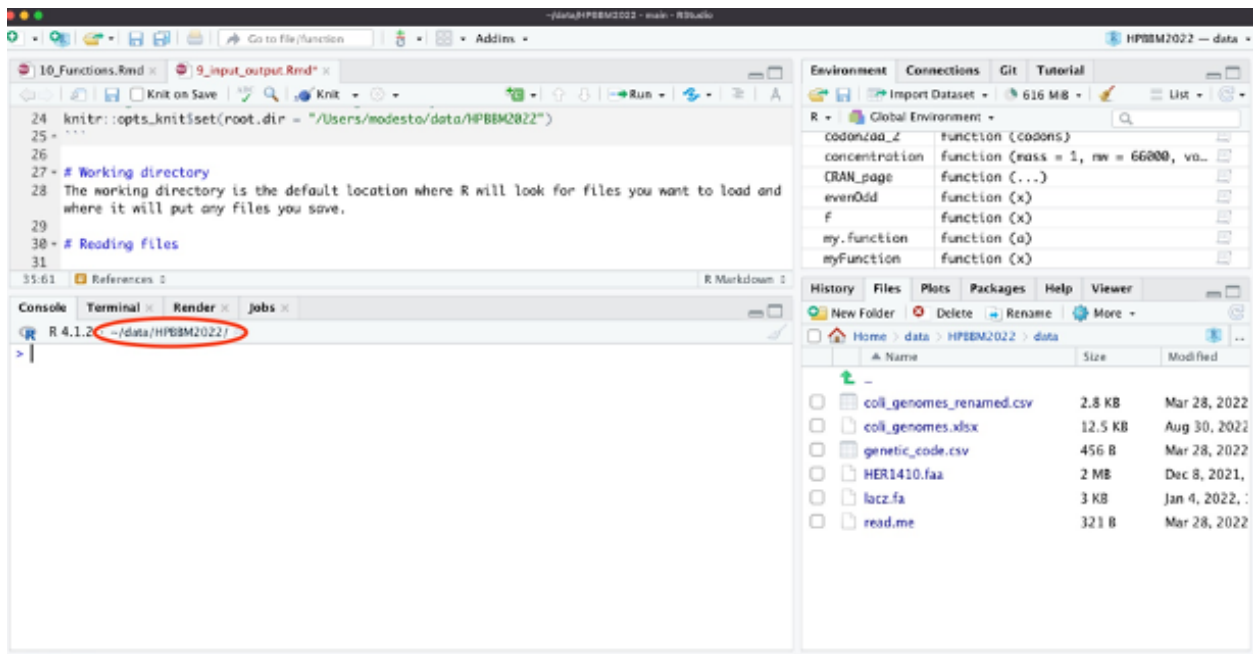
## [1] 2 4 66
print(str)

## [1] "hola clase"
edit(str)

## [1] "hola clase 2022"
```

Working directory

More often, you can process commands from a script file (a file containing R statements) and also import data from text files, databases (MySQL) or other proprietary formats, such as Excel or GraphPad Prism. We will focus on text files by the moment, as importing files in specific formats requires dedicated external packages. By default, R will read/write in the *working directory* (*wd*), which is indicated in your Console panel.



Note that the abbreviation ‘~’ stands for your home directory (for me `/Users/modesto` or `/home/modesto` on MacOS or Linux, respectively).

The functions `getwd()` and `setwd()` allow you to check and change the *wd*. As this is a Markdown document, that `setwd()` within an R chunk only changes the working directory **for that particular chunk**. Remember that you can write `?getwd()` or `?setwd()` for help.

```
getwd()

## [1] "/Users/modesto/data/HPBBM2022"

# input of setwd() is a string -> requires quotes
setwd("/Users/modesto")
getwd()
```

```
## [1] "/Users/modesto"
```

```
# you must write an existing path
setwd("/Users/mredrejo/data/HPBBM2022")
```

```
## Error in setwd("/Users/mredrejo/data/HPBBM2022"): no es posible cambiar el directorio de trabajo
```

```
setwd("/Users/modesto/data/HPBBM2022")
getwd()
```

```
## [1] "/Users/modesto/data/HPBBM2022"
```

In RStudio, the *default working directory* can be set from the “tools” and “global options” menu. Also, you can change the *wd* for your session in the menu **Session > Set Working Directory** and change it to that of source file (for instant your R script), the project or the selected directory in the files panel.

Reading/writing data in R

The most common way to read your data in R is importing it as a table, using the function `read.table()`. Note that the resultant object will become a *Dataframe*, even when all the entries got to be numeric. A followup call towards `as.matrix()` will turn it into a matrix.

In the following example we read a file called `small_matrix.csv`, located in the folder `data`.

```
sm <- read.table("data/small_matrix.csv", sep = ",")
```

```
## Warning in read.table("data/small_matrix.csv", sep = ","): incomplete final line
## found by readTableHeader on 'data/small_matrix.csv'
```

```
sm
```

```
##   V1 V2 V3
## 1  2  7 19
## 2 22 10 80
## 3 18  3 13
## 4 25  6 16
```

```
as.matrix(sm)
```

```
##      V1 V2 V3
## [1,]  2  7 19
## [2,] 22 10 80
## [3,] 18  3 13
## [4,] 25  6 16
```

You can write any data object(s) as binary data file or as text files.

```
save(vector, vector2, file = "data/vector2.Rdata")
write(vector2, file = "data/vector2.txt")
write.table(sm, "data/sm.csv")
```

Data files in RData format can be open from the *Environment* tab or with the `load()` function

```
sm_bis <- load("data/vector2.Rdata")
sm_bis
```

```
## [1] "vector" "vector2"
```

Basic Data Management in R

Now we are going to import and explore an example dataset, containing metadata from an Illumina sequencing project of pathogenic *E. coli* strains (Flament-Simon et al. 2020, <https://doi.org/10.1038/s41598-020-69356-6>). However, for didactic purposes, the original data have been simplified and manipulated and the attached datasets do not fully correspond to the actual data.

Explore a dataframe

As you can see in the R help, the function `read.table()` has several default options as `FALSE`, like `header=FALSE`. When you have a spreadsheet export file, i.e. having a table where the fields are divided by commas in place of spaces, you can use `read.csv()` in place of `read.table()`. For Spaniards, there is also `read.csv2()`, which uses a comma for the decimal point and a semicolon for the separator. The latter functions are wrappers of `read.table()` with custom default options.

```
# Note differences between read.table(), read.csv() and
# read.csv2()
coli_genomes <- read.table(file = "data/coli_genomes.csv")
```

```
## Error in scan(file = file, what = what, sep = sep, quote = quote, dec = dec, : line 2 did not have 1
```

```
head(coli_genomes)
```

```
##      Strain      Biosample Year  Source Phylogroup Serotype Clonotype Sequence.Type
## 1 LREC237 SAMN14278613   NA   Human          D  ONT:H28  CH23-331      ST524
## 2 LREC239 SAMN14278614 2010   Human          C  O153:H19   CH4-25      ST88
## 3 LREC240 SAMN14278615 2008   Human         B1  076:H30   CH29-38     ST156
## 4 LREC241 SAMN14278616   NA   Human          A  078:H11   CH11-41     ST48
## 5 LREC242 SAMN14278617 2011 Porcine          A  ONT:HNM   CH7-54      ST746
## 6 LREC243 SAMN14278618 2007 Porcine          A   09:H37   CH7-31     ST3011
##      VF Plasmids kmer  Contigs      N50 longest.contig..bp. Assembly_length
## 1 18           3  117      223 272287                662555          5341632
## 2 14           3  117      159 323172                760527          5415613
## 3 10           2   89      114 270767                738861          4875343
## 4  5           3  117      212 112160                285056          5167401
## 5  5           9   89      320 45936                 128053          4858138
## 6  7           3   93      158 106897                369508          4638334
##      contigs1kb average_contig
## 1           74      23953.51
## 2           57      34060.46
## 3           47      42766.17
## 4          101      24374.53
## 5          212      15181.68
## 6           93      29356.54
```

```
coli_genomes <- read.table(file = "data/coli_genomes.csv", sep = ";",
                           dec = ".", header = TRUE)
head(coli_genomes)
```

```
##      Strain      Biosample Year.of.isolation  Source Phylogroup Serotype Clonotype
## 1 LREC237 SAMN14278613             NA   Human          D  ONT:H28  CH23-331
## 2 LREC239 SAMN14278614          2010   Human          C  O153:H19   CH4-25
## 3 LREC240 SAMN14278615          2008   Human         B1  076:H30   CH29-38
## 4 LREC241 SAMN14278616             NA   Human          A  078:H11   CH11-41
## 5 LREC242 SAMN14278617          2011 Porcine          A  ONT:HNM   CH7-54
## 6 LREC243 SAMN14278618          2007 Porcine          A   09:H37   CH7-31
```

```
##      Sequence.Type VF No..Plasmids kmer Contigs      N50 longest.contig..bp.
## 1          ST524 18              3 117      223 272287          662555
## 2          ST88 14              3 117      159 323172          760527
## 3          ST156 10             2 89       114 270767          738861
## 4          ST48 5              3 117      212 112160          285056
## 5          ST746 5              9 89       320 45936          128053
## 6          ST3011 7            3 93       158 106897          369508
##      total.assembled.bp contigs...1kb
## 1          5341632           74
## 2          5415613           57
## 3          4875343           47
## 4          5167401          101
## 5          4858138          212
## 6          4638334           93
```

```
coli_genomes <- read.csv(file = "data/coli_genomes.csv")
head(coli_genomes)
```

```
##      Strain.Biosample.Year.of.isolation.Source.Phylogroup.Serotype.Clonotype.Sequence.Type.VF.No..Plasm
## 1                                                                 LREC237;SAMN14278613
## 2                                                                 LREC239;SAMN14278614
## 3                                                                 LREC240;SAMN14278615;
## 4                                                                 LREC241;SAMN14278
## 5                                                                 LREC242;SAMN14278617
## 6                                                                 LREC243;SAMN14278618
```

```
coli_genomes <- read.csv(file = "data/coli_genomes.csv", sep = ";")
head(coli_genomes)
```

```
##      Strain      Biosample Year.of.isolation      Source Phylogroup Serotype Clonotype
## 1 LREC237 SAMN14278613              NA      Human          D  ONT:H28  CH23-331
## 2 LREC239 SAMN14278614             2010      Human          C  0153:H19  CH4-25
## 3 LREC240 SAMN14278615             2008      Human          B1  076:H30  CH29-38
## 4 LREC241 SAMN14278616              NA      Human          A  078:H11  CH11-41
## 5 LREC242 SAMN14278617             2011 Porcine          A  ONT:HNM  CH7-54
## 6 LREC243 SAMN14278618             2007 Porcine          A   09:H37  CH7-31
```

```
##      Sequence.Type VF No..Plasmids kmer Contigs      N50 longest.contig..bp.
## 1          ST524 18              3 117      223 272287          662555
## 2          ST88 14              3 117      159 323172          760527
## 3          ST156 10             2 89       114 270767          738861
## 4          ST48 5              3 117      212 112160          285056
## 5          ST746 5              9 89       320 45936          128053
## 6          ST3011 7            3 93       158 106897          369508
##      total.assembled.bp contigs...1kb
## 1          5341632           74
## 2          5415613           57
## 3          4875343           47
## 4          5167401          101
## 5          4858138          212
## 6          4638334           93
```

```
coli_genomes <- read.csv2(file = "data/coli_genomes.csv")
head(coli_genomes)
```

```
##      Strain      Biosample Year.of.isolation      Source Phylogroup Serotype Clonotype
## 1 LREC237 SAMN14278613              NA      Human          D  ONT:H28  CH23-331
```

```
## 2 LREC239 SAMN14278614      2010   Human      C 0153:H19   CH4-25
## 3 LREC240 SAMN14278615      2008   Human      B1 076:H30   CH29-38
## 4 LREC241 SAMN14278616      NA     Human      A 078:H11   CH11-41
## 5 LREC242 SAMN14278617      2011 Porcine    A 078:H11   CH7-54
## 6 LREC243 SAMN14278618      2007 Porcine    A 09:H37    CH7-31
##   Sequence.Type VF No..Plasmids kmer Contigs   N50 longest.contig..bp.
## 1      ST524 18          3 117    223 272287      662555
## 2      ST88 14          3 117    159 323172      760527
## 3      ST156 10         2 89     114 270767      738861
## 4      ST48 5           3 117    212 112160      285056
## 5      ST746 5           9 89     320 45936      128053
## 6     ST3011 7           3 93     158 106897      369508
##   total.assembled.bp contigs...1kb
## 1          5341632          74
## 2          5415613          57
## 3          4875343          47
## 4          5167401         101
## 5          4858138         212
## 6          4638334          93
```

```
# explore the dataframe structure
dim(coli_genomes)
```

```
## [1] 25 16
```

```
length(coli_genomes)
```

```
## [1] 16
```

```
ncol(coli_genomes)
```

```
## [1] 16
```

```
nrow(coli_genomes)
```

```
## [1] 25
```

```
# dataframe esturcture in one line
str(coli_genomes)
```

```
## 'data.frame':   25 obs. of  16 variables:
## $ Strain      : chr  "LREC237" "LREC239" "LREC240" "LREC241" ...
## $ Biosample   : chr  "SAMN14278613" "SAMN14278614" "SAMN14278615" "SAMN14278616" ...
## $ Year.of.isolation : int  NA 2010 2008 NA 2011 2007 2006 2006 2010 2013 ...
## $ Source      : chr  "Human " "Human " "Human " "Human" ...
## $ Phylogroup   : chr  "D" "C" "B1" "A" ...
## $ Serotype     : chr  "ONT:H28" "0153:H19" "076:H30" "078:H11" ...
## $ Clonotype    : chr  "CH23-331" "CH4-25" "CH29-38" "CH11-41" ...
## $ Sequence.Type : chr  "ST524" "ST88" "ST156" "ST48" ...
## $ VF          : int   18 14 10 5 5 7 4 2 10 22 ...
## $ No..Plasmids  : int   3 3 2 3 9 3 7 7 1 4 ...
## $ kmer         : int  117 117 89 117 89 93 115 115 113 113 ...
## $ Contigs      : int  223 159 114 212 320 158 277 203 131 215 ...
## $ N50          : int  272287 323172 270767 112160 45936 106897 89185 94368 326769 248158 ...
## $ longest.contig..bp.: int  662555 760527 738861 285056 128053 369508 281444 280268 451887 504233 .
## $ total.assembled.bp : int  5341632 5415613 4875343 5167401 4858138 4638334 5406295 4796593 5173794
## $ contigs...1kb     : int   74 57 47 101 212 93 155 114 56 76 ...
```

```

# type of data in each variable
typeof(coligenomes$Strain)

## [1] "character"
typeof(coligenomes[, 2])

## [1] "character"
typeof(coligenomes[, 9])

## [1] "integer"
# read some data
head(coligenomes)

##      Strain      Biosample Year.of.isolation Source Phylogroup Serotype Clonotype
## 1 LREC237 SAMN14278613          NA Human          D 0NT:H28 CH23-331
## 2 LREC239 SAMN14278614        2010 Human          C 0153:H19 CH4-25
## 3 LREC240 SAMN14278615        2008 Human          B1 076:H30 CH29-38
## 4 LREC241 SAMN14278616          NA Human          A 078:H11 CH11-41
## 5 LREC242 SAMN14278617        2011 Porcine         A 0NT:HNM CH7-54
## 6 LREC243 SAMN14278618        2007 Porcine         A 09:H37 CH7-31
##      Sequence.Type VF No..Plasmids kmer Contigs      N50 longest.contig..bp.
## 1          ST524 18          3 117      223 272287          662555
## 2          ST88 14          3 117      159 323172          760527
## 3          ST156 10          2 89       114 270767          738861
## 4          ST48 5          3 117      212 112160          285056
## 5          ST746 5          9 89       320 45936          128053
## 6          ST3011 7          3 93       158 106897          369508
##      total.assembled.bp contigs...1kb
## 1          5341632          74
## 2          5415613          57
## 3          4875343          47
## 4          5167401          101
## 5          4858138          212
## 6          4638334          93

tail(coligenomes, n = 2)

##      Strain      Biosample Year.of.isolation Source Phylogroup Serotype Clonotype
## 24 LREC261 SAMN14278636        2016 Human          A 098:H26 CH27-23
## 25 LREC262 SAMN14278637        2012 Human          B1 066:H10 CH4-32
##      Sequence.Type VF No..Plasmids kmer Contigs      N50 longest.contig..bp.
## 24          ST8233 2          4 89       114 187945          537848
## 25          ST1049 4          2 113      94 325747          822206
##      total.assembled.bp contigs...1kb
## 24          4821342          53
## 25          4839344          32

coligenomes[1, ]

##      Strain      Biosample Year.of.isolation Source Phylogroup Serotype Clonotype
## 1 LREC237 SAMN14278613          NA Human          D 0NT:H28 CH23-331
##      Sequence.Type VF No..Plasmids kmer Contigs      N50 longest.contig..bp.
## 1          ST524 18          3 117      223 272287          662555
##      total.assembled.bp contigs...1kb

```

```
## 1          5341632          74
```

```
coli_genomes[, 1]
```

```
## [1] "LREC237" "LREC239" "LREC240" "LREC241" "LREC242" "LREC243" "LREC244"
## [8] "LREC245" "LREC246" "LREC247" "LREC248" "LREC249" "LREC250" "LREC251"
## [15] "LREC252" "LREC253" "LREC254" "LREC255" "LREC256" "LREC257" "LREC258"
## [22] "LREC259" "LREC260" "LREC261" "LREC262"
```

```
coli_genomes[1:6, 2:4]
```

```
##      Biosample Year.of.isolation Source
## 1 SAMN14278613          NA      Human
## 2 SAMN14278614        2010      Human
## 3 SAMN14278615        2008      Human
## 4 SAMN14278616          NA      Human
## 5 SAMN14278617        2011 Porcine
## 6 SAMN14278618        2007 Porcine
```

Some of the columns include 'chr' data that may be actually a categorical variable, so we can code them as **factor**. Using the expression `as.factor()` you can check whether the data would correspond to a text or a categorical variable.

```
coli_genomes$Source <- as.factor(coi_genomes$Source)
coli_genomes$Phylogroup <- as.factor(coi_genomes$Phylogroup)
```

```
str(coi_genomes) #dataframe esturcture updated
```

```
## 'data.frame':   25 obs. of  16 variables:
## $ Strain      : chr  "LREC237" "LREC239" "LREC240" "LREC241" ...
## $ Biosample   : chr  "SAMN14278613" "SAMN14278614" "SAMN14278615" "SAMN14278616" ...
## $ Year.of.isolation : int  NA 2010 2008 NA 2011 2007 2006 2006 2010 2013 ...
## $ Source      : Factor w/ 4 levels "Avian ","Human",...: 3 3 3 2 4 4 4 4 3 ...
## $ Phylogroup   : Factor w/ 4 levels "A","B1","C","D": 4 3 2 1 1 1 1 1 3 4 ...
## $ Serotype     : chr  "ONT:H28" "O153:H19" "O76:H30" "O78:H11" ...
## $ Clonotype    : chr  "CH23-331" "CH4-25" "CH29-38" "CH11-41" ...
## $ Sequence.Type : chr  "ST524" "ST88" "ST156" "ST48" ...
## $ VF          : int  18 14 10 5 5 7 4 2 10 22 ...
## $ No..Plasmids  : int  3 3 2 3 9 3 7 7 1 4 ...
## $ kmer         : int  117 117 89 117 89 93 115 115 113 113 ...
## $ Contigs      : int  223 159 114 212 320 158 277 203 131 215 ...
## $ N50         : int  272287 323172 270767 112160 45936 106897 89185 94368 326769 248158 ...
## $ longest.contig..bp.: int  662555 760527 738861 285056 128053 369508 281444 280268 451887 504233 .
## $ total.assembled.bp : int  5341632 5415613 4875343 5167401 4858138 4638334 5406295 4796593 5173794
## $ contigs...1kb     : int  74 57 47 101 212 93 155 114 56 76 ...
```

How many levels are there in *Source*?? It is not uncommon to see some mistake in our data, usually made when the data were recorded, for example a space may have been inserted before a data value. By default this white space will be kept in the R environment, such that 'Human' will be recognized as a different value than 'Human'. In order to avoid this type of error, we can use the *strip.white* argument.

```
unique(coi_genomes$Source)
```

```
## [1] Human      Human      Porcine  Avian
## Levels: Avian      Human Human      Porcine
```

```
table(coi_genomes$Source)
```

```
##
```



```
##      Avian      Human      Human      Porcine
##           3           1           16           5

coli_genomes <- read.csv2(file = "data/coli_genomes.csv", strip.white = TRUE)
coli_genomes$Source <- as.factor(coli_genomes$Source)
coli_genomes$Phylogroup <- as.factor(coli_genomes$Phylogroup)

unique(coli_genomes$Source)

## [1] Human      Porcine Avian
## Levels: Avian Human Porcine
```

We can also rename some variables to use more easy names.

```
names(coli_genomes) #see all variable names

## [1] "Strain"           "Biosample"         "Year.of.isolation"
## [4] "Source"           "Phylogroup"        "Serotype"
## [7] "Clonotype"        "Sequence.Type"     "VF"
## [10] "No..Plasmids"      "kmer"              "Contigs"
## [13] "N50"              "longest.contig..bp." "total.assembled.bp"
## [16] "contigs...1kb"

# rename variables
names(coli_genomes)[3] <- "Year"
names(coli_genomes)[10] <- "Plasmids"
names(coli_genomes)[15] <- "Assembly_length"
names(coli_genomes)[16] <- "contigs1kb"
# check
names(coli_genomes)

## [1] "Strain"           "Biosample"         "Year"
## [4] "Source"           "Phylogroup"        "Serotype"
## [7] "Clonotype"        "Sequence.Type"     "VF"
## [10] "Plasmids"         "kmer"              "Contigs"
## [13] "N50"              "longest.contig..bp." "Assembly_length"
## [16] "contigs1kb"
```

Change and add variables

We are going to simplify our dataframe by dropping variables:

```
coli_genomes <- coli_genomes[-c(9:11), ]
# Note that this might be also used to remove rows
```

We know the ‘Assembly length’ and the number of ‘Contigs’, but we would like to represent the average contig length.

```
coli_genomes$aaverage_contig <- coli_genomes$Assembly_length/coli_genomes$Contigs
```

Dealing with NAs

It is very easy to calculate statistics of one variable. Imagine we want to know the average year of sample isolation.

```
mean(coli_genomes$Year.of.isolation)

## Warning in mean.default(coli_genomes$Year.of.isolation): argument is not numeric
## or logical: returning NA
```

```
## [1] NA
```

Yes, that error means that there are some NA values and mean cannot be calculated. We can check that and omit the NAs.

```
# check if there is any NA
is.na(coli_genomes)
```

```
##      Strain Biosample   Year Source Phylogroup Serotype Clonotype Sequence.Type
## 1  FALSE      FALSE TRUE  FALSE      FALSE      FALSE      FALSE      FALSE
## 2  FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
## 3  FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
## 4  FALSE      FALSE TRUE   FALSE      FALSE      FALSE      FALSE      FALSE
## 5  FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
## 6  FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
## 7  FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
## 8  FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
## 12 FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
## 13 FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
## 14 FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
## 15 FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
## 16 FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
## 17 FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
## 18 FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
## 19 FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
## 20 FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
## 21 FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
## 22 FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
## 23 FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
## 24 FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
## 25 FALSE      FALSE FALSE  FALSE      FALSE      FALSE      FALSE      FALSE
##      VF Plasmids  kmer Contigs  N50 longest.contig..bp. Assembly_length
## 1  FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 2  FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 3  FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 4  FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 5  FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 6  FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 7  FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 8  FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 12 FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 13 FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 14 FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 15 FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 16 FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 17 FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 18 FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 19 FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 20 FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 21 FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 22 FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 23 FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 24 FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
## 25 FALSE      FALSE FALSE  FALSE FALSE      FALSE      FALSE
##      contigs1kb average_contig
```

```
## 1      FALSE      FALSE
## 2      FALSE      FALSE
## 3      FALSE      FALSE
## 4      FALSE      FALSE
## 5      FALSE      FALSE
## 6      FALSE      FALSE
## 7      FALSE      FALSE
## 8      FALSE      FALSE
## 12     FALSE      FALSE
## 13     FALSE      FALSE
## 14     FALSE      FALSE
## 15     FALSE      FALSE
## 16     FALSE      FALSE
## 17     FALSE      FALSE
## 18     FALSE      FALSE
## 19     FALSE      FALSE
## 20     FALSE      FALSE
## 21     FALSE      FALSE
## 22     FALSE      FALSE
## 23     FALSE      FALSE
## 24     FALSE      FALSE
## 25     FALSE      FALSE
```

```
# na.rm=TRUE will omit the NAs for this function
mean(coli_genomes$Year.of.isolation, na.rm = TRUE)
```

```
## Warning in mean.default(coli_genomes$Year.of.isolation, na.rm = TRUE): argument
## is not numeric or logical: returning NA

## [1] NA
```

What if we want to remove observations with an NA from a dataset?

```
coli_genomes2 <- na.omit(coli_genomes)
```

Finally, we are going to save our new dataset for future examples.

```
write.csv2(coli_genomes, "data/coli_genomes_renamed.csv", row.names = FALSE)
```

References

- *An introduction to R*, <https://intro2r.com/work-d.html>
- *R programming for data science*, <https://bookdown.org/rdpeng/rprogdatascience/>
- Using RStudio projects, <https://support.rstudio.com/hc/en-us/articles/200526207>
- Importar y exportar datos en R, <https://rsanchezs.gitbooks.io/rprogramming/content/chapter3/index.html>
- *R in action*. Robert I. Kabacoff. March 2022 ISBN 9781617296055
- R para análisis científicos reproducibles. Software Carpentry Foundation. <https://swcarpentry.github.io/r-novice-gapminder-es/>

Short exercises

1. Try the input/output examples from *Techvidvan* website <https://techvidvan.com/tutorials/r-input-and-output-functions/>
2. Load the file *colis3.csv* as *colis* and explore the dataset structure.
3. Calculate the mean of numerical variables: isolation date (*Year*), antimicrobial resistance genes (*AMR*), virulence factors (*VF*), integron cassettes (*Integron*) and sequencing date (*seqs*) in those strains? Note. For the *seqs* variable you will need to use the function `as.Date()`.
4. Save the tables *coli_genomes_renamed* and *colis* in a *Rdata* file.
5. Add the values of the exercise 3 as a last row in the table. Note. For the *seqs* variable you will need to use the function `format()`.