STA 2300: Introduction to Data Science

Lecture 03: Common Commands and Structures

**Today’s Topics**

**Data Structures**

**Common, Useful Commands**

**Filtering Data**

**Data Structures**

There are different types of data in R, and those types can be organized into different structures in R. The types are

• **Character:** This is text data. For example, the variable “Class” may contain the entries “freshman,” “sophomore,” “junior,” or “senior.”

• **Numeric:** These can be integer or decimal-valued numbers. These are the same as “doubles.”

• **Integer:** These are whole number values, either positive or negative. • **Logical:** These are either “TRUE” or “FALSE” designations.

• **Complex:** Values with real and imaginary parts, denoted for example as 1+4i. • **Date/Time:** These are dates, without or without a specific time stamp associated with them.

These types can be collected and represented as follows:

• **Factor:** A whole column of character data is designated a factor where each unique outcome is called a “level.”"

• **Vector:** A sequence of values of the same data type.

• **Matrix:** A two-dimensional (rows and columns) set of values and all must be numeric. • **Array:** A multi-dimensional set of values all of the same type. A matrix is a special 2-D case.

• **Data Frame:** A very commonly used data structure in R. It is two-dimensional with rows and columns, but each column can be a different data type.

• **List:** A list has slots, and within each slot, you can have a different structure of different sizes stored.

1

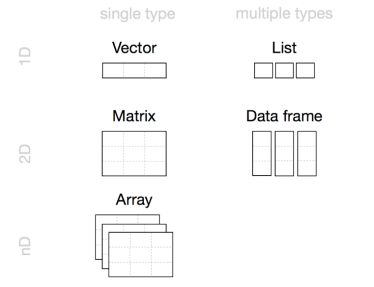


Figure from https://rstudio-education.github.io/hopr/r-objects.html

We will work with many data frames, so they deserve some special attention. You can read a dataset into R that is automatically treated as a data frame, or you can create one from scratch, as follows:

friends <- **data.frame**(id\_number = **c**(1, 2, 3, 4, 5, 6), name = **c**("Ross", "Rachel", "Monica", "Chandler", "Joey", "Phoebe"), birthday = **as.Date**(**c**("1966-11-02"1969-02-11", "1964-06-15", "1969-08-19", "1967-07-25", "1963-07-30")), num\_daily\_coffees = **c**(3, 5, 2, 3, 2, 8), stringsAsFactors = FALSE) friends

*# id\_number name birthday num\_daily\_coffees*

*# 1 1 Ross 1966-11-02 3*

*# 2 2 Rachel 1969-02-11 5*

*# 3 3 Monica 1964-06-15 2*

*# 4 4 Chandler 1969-08-19 3*

*# 5 5 Joey 1967-07-25 2*

*# 6 6 Phoebe 1963-07-30 8*

*# friends$name; friends$birthday*

You extract a particular column from a data frame by using the data frame’s name followed by the $ symbol and then the column name. You can also add a new row or column as follows:

2

*# Add a new column*

num\_breakups <- **c**(5, 8, 7, 6, 3, 8)

friends <- **cbind**(friends, num\_breakups)

*# Add a new row--requires creating a new data frame for the row*

new\_person <- **data.frame**(id\_number = 7, name = "Mike", birthday = **as.Date**("1969-04-06", format = "%Y-%m-%d"), num\_daily\_coffees = 4, num\_breakups = 0)

friends <- **rbind**(friends, new\_person)

**Example: Colorado Covid Data** The following data is downloaded from the Colorado Department of Public Health and Environment1. It contains four main variables:

• The date

• The particular utility

• SARS CoV2 copies of RNA (measured as RNA/liter of water) in wastewater • The number of new Covid-19 cases

SARS-CoV-2 is the virus that causes COVID-19, and RNA is the genetic material in each copy of the virus. SARS-CoV-2 copies per liter is one measure of how much of the virus is in the wastewater, expressed as a concentration. Studies have shown that individuals who develop COVID-19 often shed detectable SARS-CoV-2 RNA from their systems before, during, and after their infection, so higher levels of SARS-CoV-2 RNA can indicate a rise in cases in a community. Many universities used this method to monitor the wastewater from residence halls to obtain an early warning of a disease outbreak.

covid <- **read.csv**(file = "CDPHE\_COVID19\_Wastewater\_Dashboard\_Data.csv", header = T)

**class**(covid)

*# [1] "data.frame"*

**head**(covid)

*# Date Utility SARS\_CoV\_2\_copies\_L*

*# 1 08/15/2020 Metro Wastewater RWHTF - PRC NA*

*# 2 08/11/2020 Broomfield NA*

*# 3 08/15/2020 Northglenn NA*

*# 4 08/11/2020 CO Springs - JD Phillips NA*

*# 5 08/11/2020 CO Springs - Las Vegas NA*

*# 6 08/15/2020 Pueblo NA*

*# Number\_of\_New\_COVID19\_Cases\_by\_ ObjectId*

*# 1 36 1*

*# 2 0 2*

*# 3 0 3*

*# 4 6 4*

*# 5 22 5*

1https://cdphe.maps.arcgis.com/apps/opsdashboard/index.html#/d79cf93c3938470ca4bcc4823328946b 3

*# 6 5 6*

**dim**(covid)

*# [1] 3498 5*

**The Kindergarten Commands of R**

These commonly used commands are so widely used that they are often taken for granted!

• rm(list=ls()): removes everything in the global data environment, like erasing a chalkboard and starting over

• length(): returns the length of a vector or the number of elements in a matrix. See also, dim(), nrow(), and ncol.

• c(): combines values into a group, but they must all be the same data type

**c**(5, 10, **-**2.5)

*# [1] 5.0 10.0 -2.5*

• rep(): repeats the same value (or sequence of values) a certain number of times

**rep**(0, 5)

*# [1] 0 0 0 0 0*

**rep**("Hi", len = 3)

*# [1] "Hi" "Hi" "Hi"*

**rep**(20**:**25, times = 2)

*# [1] 20 21 22 23 24 25 20 21 22 23 24 25*

**rep**(20**:**25, each = 2)

*# [1] 20 20 21 21 22 22 23 23 24 24 25 25*

• seq(): creates a sequence of numbers from a lower bound to an upper bound, of a given length or seperated by a given distance

**seq**(1, 10, len = 21)

*# [1] 1.00 1.45 1.90 2.35 2.80 3.25 3.70 4.15 4.60 5.05 5.50 5.95 # [13] 6.40 6.85 7.30 7.75 8.20 8.65 9.10 9.55 10.00*

**seq**(1, 10, by = 0.5)

*# [1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0 5.5 6.0 6.5 7.0 7.5 8.0 # [16] 8.5 9.0 9.5 10.0*

• The colon : used to rapidly create a sequence of integers

1**:**7

*# [1] 1 2 3 4 5 6 7*

4

**-**3**:**5

*# [1] -3 -2 -1 0 1 2 3 4 5*

• unique() and table():

unique will return a list of the unique values in a vector or array and will discard the duplicates.

table will tabulate and count up how many times a particular unique element occurs in a vector.

*# How many utilities are represented in the Covid dataset? # unique(covid$Utility) table(covid$Utility)*

**Filtering Data**

Sometimes you want to select only those observations (rows) from a dataset that meet a certain criteria, such as only one particular utility in the Covid data. To do so, you need to know the comparison expressions used in R:

• x == y indicates x must equal y

• x != y indicates x must NOT equal y

• x >= y indicates x must be greater than or equal to y

• x <= y indicates x must be less than or equal to y

• x > y indicates x must be strictly greater than y

• x < y indicates x must be strictly less than y

These are logicals that can be used with the comparisons above:

• !a indicates not a

• a & b indicates both a AND b must be true

• a | b indicates either a OR b must be true

Finally, to apply basic filtering, you need to combine a logical statement with the square brackets. Square brackets allow you to select either the rows or columns in a data frame, indicated as dataframe[rows, columns].

*# Select row 4 and column 3*

friends[4, 5]

*# [1] 6*

*# Selects row 2*

friends[2, ]

*# id\_number name birthday num\_daily\_coffees num\_breakups # 2 2 Rachel 1969-02-11 5 8*

*# Selects column 3*

5

friends[, 3]

*# [1] "1966-11-02" "1969-02-11" "1964-06-15" "1969-08-19" "1967-07-25" # [6] "1963-07-30" "1969-04-06"*

*# Selects id\_numbers 1 to 4*

friends[1**:**4, ]

*# id\_number name birthday num\_daily\_coffees num\_breakups # 1 1 Ross 1966-11-02 3 5 # 2 2 Rachel 1969-02-11 5 8 # 3 3 Monica 1964-06-15 2 7 # 4 4 Chandler 1969-08-19 3 6*

*# Selects rows with number of daily coffees over 5*

friends[friends**$**num\_daily\_coffees **>** 5, ]

*# id\_number name birthday num\_daily\_coffees num\_breakups # 6 6 Phoebe 1963-07-30 8 8*

*# Selects rows with number of daily coffees over 3 AND number of # breakups greater than or equal to 7*

friends[(friends**$**num\_daily\_coffees **>** 4) **&** (friends**$**num\_breakups **>=** 7), ]

*# id\_number name birthday num\_daily\_coffees num\_breakups # 2 2 Rachel 1969-02-11 5 8 # 6 6 Phoebe 1963-07-30 8 8*

*# Selects rows with number of daily coffees over 3 OR number of # breakups greater than or equal to 7*

friends[(friends**$**num\_daily\_coffees **>** 4) **|** (friends**$**num\_breakups **>=** 7), ]

*# id\_number name birthday num\_daily\_coffees num\_breakups # 2 2 Rachel 1969-02-11 5 8 # 3 3 Monica 1964-06-15 2 7 # 6 6 Phoebe 1963-07-30 8 8*

**Example: Colorado Covid Data** Find the following subsets:

1. Find those rows of the dataset that pertain only to the Boulder utility.

2. Select those observations whose new Covid-19 cases are greater than 200. What proportion of the dataset is this?

3. Select the rows that do not have any NAs in the RNA column.

*# Question 1*

boulder\_covid <- covid[covid**$**Utility **==** "Boulder", ]

6

*# boulder\_covid*

*# Question 2*

high\_counts <- covid[covid**$**Number\_of\_New\_COVID19\_Cases\_by\_ **>** 200, ]

**nrow**(high\_counts)**/nrow**(covid) **\*** 100

*# [1] 5.517439*

*# Question 3*

covid\_complete <- covid[**is.na**(covid**$**SARS\_CoV\_2\_copies\_L) **==** FALSE, ]

covid\_complete <- covid[**!is.na**(covid**$**SARS\_CoV\_2\_copies\_L), ] *# Can also be split into two steps. rna\_present\_index <-*

*# is.na(covid$SARS\_CoV\_2\_copies\_L) covid\_complete <-*

*# covid[!rna\_present\_index, ]*

Are any of the columns of this dataset redundant? How could you quickly remove a column? covid <- covid[, **-**5]

7