String manipulation with stringr

stringr

apple	FALSE
banana	TRUE
pear	FALSE
persimmon	FALSE
kiwi	FALSE
mango	TRUE
orange	TRUE

Boriana Pratt
Office of Population Research (OPR)
Princeton University
January, 2019

Syntax

'stringr' is an R package with functions that handle the most common string manipulations. It is build on top of another R package – 'stringi'.

Basic syntax:

str_*fname*(*string*,…)

All 'stringr' functions are vectorized.

Let's make a vector of strings:

```
fruits <- c("apple", "banana", "pear", "persimmon", "kiwi", "mango", "orange")
```

Basic functions

str_length() – number of characters in the string - same as nchar()

str_c() – combine two strings - like paste0()

str_c(*string*, sep="", collapse=NULL)

str_c(*string*, collapse=",") - collapse a vector of strings into a single string

str_dup() – duplicate a string

str_dup(*string*, times=2)

str_dup(*string*, c(2,1,3,2,1,2))

str_sub() – extracts substring (could be used to replace a substring)

str_sub(*string*, start, end)

str_sub(*string*, start, end) <- *value*

Let's try these on our vector of strings…

Exercises – basic functions

```
fruits <- c("apple", "banana", "pear","persimmon","kiwi", "mango","orange")
str_length(fruits)
str_dup(fruits, c(2,1,3,1,2))
list_fruits <- str_c(fruits, collapse=",")

fcolor <-c("red", "yellow","green","orange","green","reddish",NA)
str_c(fruits, fcolor, sep=", ")
paste(fruits, fcolor, sep=", ")

str_sub(fruits, 1, 3)
str_sub(list_fruits, 1, 3)
str_sub(list_fruits, 21, 29)
str_sub(list_fruits, 1, 3) <- ""
str_sub(fruits, 4, 4) <- "4"
```

Find more exercises on line at:

https://cran.r-project.org/web/packages/stringr/vignettes/stringr.html

whitespace

str_trim() – removes leading and/or trailing whitespace

str_trim(*string*, side=c("both","left","right"))

str_pad() – pads string with extra whitespace either to the left, right or both sides

str_pad(*string*, width ,side=c("both","left","right"), pad=" ")

str_trunc() – truncate a string to a specified width

str_trunc(*string*, width, side=c("both","left","right"), ellipsis="…")

Exercises – whitespace functions

Pad and then trim:

```
fruits <- c("apple", "banana", "pear","persimmon","kiwi","mango","orange")
fruits3 <- str_pad(fruits, 10, "both")

str_trim(fruits3)
str_trim(fruits3, "left")

str_trunc(fruits, 5, ellipsis="..")
```

Find more exercises on line at:

https://cran.r-project.org/web/packages/stringr/vignettes/stringr.html

Functions for case conversion

str_to_upper() - makes all letters upper case

str_to_lower() - makes all letters lower case

str_to_title() - capitalizes first letter of each word

Exercises – case conversion functions

```
str_to_upper(fruits)
greetings <- c("HI", "BYE", "GOOD MORNING", "GOOD EVENING")
str_to_lower(greetings)
str_to_title(greetings)
```

Find more exercises on line at:

https://r4ds.had.co.nz/strings.html

Order functions

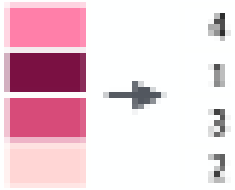
str_order() - returns the indexes of the sorted vector

str_order(*string*, decreasing=FALSE, locale= “en”, na.last=TRUE)

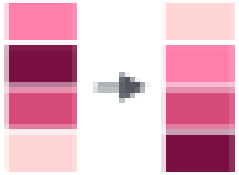
str_sort() - sorts a character vector

str_sort(*string*, decreasing=FALSE, locale= “en”, na.last=TRUE)

order:



sort:



Note: The above ‘stringr’ functions are locale-sensitive – depending on your locale you may get slightly different results.

Exercises – order and sort functions

Sort (and order) the fruits alphabetically:

```
str_order(fruits)
str_sort(fruits)
```

Example of different sorting:

```
str_sort(letters, locale="haw")
```

Find the stringr cheat sheet on line at:
https://www.rdocumentation.org/packages/stringr/versions/1.3.1

Detailed explanation on the above functions can be found here:
https://rdrr.io/cran/stringr/man/str_order.html

Functions for pattern search

str_detect() – detect a pattern (based on grepl) – returns T/F

str_count() – count the number of matches

str_locate() – finds the start and end positions of **the first match** of the pattern in the string - returns a matrix

str_locate_all() – finds the start and end positions of **all matches** of the pattern in the string - returns a list

stringr

apple	0
banana	2
pear	0
persimmon	0
kiwi	0
mango	1
orange	1

Each pattern matching function has the same first two arguments – a character vector of strings to process and a single pattern (regex) to match.

Special characters in regular expressions: **. \ | ( ) [{ ^ $ * + ?** .

Exercises - functions for pattern search

Find the fruits that start with the letter ‘p’:

```
str_detect(fruits,"^p")
```

Find the fruits that don’t start with the letter p:

```
str_detect(fruits,"^[^p]")
```

Find the fruits that don’t contain the letter p:

```
str_detect(fruits,"^[^p]([^p])*[^p]$")
```

Find the number of times ‘an’ appears in each fruit name:

```
str_count(fruits, "an")
```

Find the position of the first match (or all matches) of ‘an’ in each fruit name:

```
str_locate(fruits, "an")
str_locate_all(fruits, "an")
```

Find more exercises on line at:

http://dept.stat.lsa.umich.edu/~jerrick/courses/stat701/notes/stringmanip.html

Exercises - functions for pattern search

Let's look at the school address data and see if each address has a state (two capital letters) and a zip code (five digits).

First read in the data:

```
schools<-read.csv(file="sch_3citiesSTall.csv", header=T, as.is=T)
head(schools)

state_zip <- "[A-Z]{2} [0-9]{5}"
st_z1 <- str_detect(schools$adr,state_zip)
summary(st_z1)
```

Let's try one more pattern – with two spaces between state and zip:

```
state_zip2 <- "[A-Z]{2}  [0-9]{5}"
st_z2 <- str_detect(schools$adr,state_zip2)
summary(st_z2)
```

Let's find the location of the first occurrence in each address of the first pattern preceded by a city:

```
csz <- "[A-Z][a-z]+_ [A-Z]{2} [0-9]{5}"
st_z <- str_locate(schools$adr,csz)
```

Let's see if each school address has only one state, then only one zip code…

Functions for pattern matching

str_extract()	– extracts the text of the first match, returns a vector
str_extract_all()	– extracts the text of the all matches, returns a list
str_match()	– returns the first match in each string, as a matrix with a column for each () group
str_match_all()	– returns all matches as a list (of matrices)
str_subset()	– returns only the strings that that contain the pattern

Each pattern matching function has the same first two arguments – a character vector of strings to process and a single pattern (regex) to match.

Exercises - functions for pattern matching

Let's extract from each address the city, state and zip (two spaces between state and zip):

```
csz <- "[A-Z][a-z]+_ [A-Z]{2}  [0-9]{5}"
adr_end <- str_extract(schools$adr,csz)
head(adr_end)

adr_end2 <- str_match(schools$adr,csz)
head(adr_end2)
```

Extract city, state and zip using groups (compare to the result above):

```
csz_g <- "([A-Z][a-z]+_) ([A-Z]{2})  ([0-9]{5})"
adr_end3 <- str_match(schools$adr,csz_g)
head(adr_end3)
```

Find all of the schools in Maine:

```
schools_ME <- str_subset(dtA$adr, "ME  [0-9]{5}")
```

Find more exercises on line at:

http://dept.stat.lsa.umich.edu/~jerrick/courses/stat701/notes/stringmanip.html

Functions – replace and split

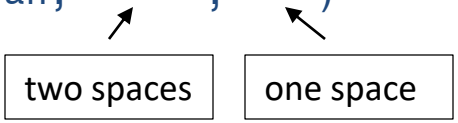
str_replace()	– replaces the first matched pattern
str_replace_all()	– replaces all instances of the matched pattern
str_split()	– splits the string based in a pattern and returns a list
str_split_fixed()	– splits the string based on a pattern and returns a matrix
str_split(…, simplify=TRUE)	– splits the string based on a pattern and returns a matrix

One more useful function: str_replace_na(*string*) – converts NA to "NA" (so you don't lose data when using other stringr functions.

Exercises - functions for replace and split

Let's clean up the addresses – remove first instance of city, state and zip (one space between city_ and state, and between state and zip):

```
csz <- "[A-Z][a-z]+_ [A-Z]{2}  [0-9]{5}"
adr_clean <- str_replace(schools$adr,  csz,"")
adr_clean <- str_replace(adr_clean, "  +", " ")
```



Extract city, state and zip into separate columns:

```
school_csz <- str_split_fixed(str_extract(adr_clean, csz)," ", n=3)
```

This should give the same result as above:

```
school_csz2 <- str_split(str_extract(adr_clean, csz)," ", simplify=TRUE)
```

More exercises

Find the addresses of the schools in NY state.

Find all schools that are only for grades 9-12 (note: use column 'grade'):

```
gr912 <- schools[str_detect(schools$grade,"9-12")==TRUE,]
```

Find all schools that are for grades K-12.

Split each line of the file "yob_notes.csv" (from the REGEX presentation) into columns:

```
names <- str_split(as.character(notes), ",", simplify=TRUE)
```