R Programming For Natural Resource Professionals

Additional Tidyverse packages

Paper discussions

- Open this link now by typing it into a browser: https://bit.ly/3GVPwuV
- Talk with your group to identify one or two thoughts, questions, and epiphanies that resonate then record them in the Google Doc.
- Read through the list as it updates.

Previewing the week

Dates and times

Standardizing date information

- ymd()
- mdy()
- dmy()

Translates date information into a standardized “YEAR-MONTH-DAY” format based on the structure indicated in the function used.

Example:

```
> mdy(“06302016”)
[1] “2016-06-30”
> str(mdy(“06302016”)) #Note data type
```

Standardizing date and time information

- `ymd_hms()`
- `mdy_hm()`
- `dmy_h()`
- etc...

Translates date and time information into a standardized “YEAR-MONTH-DAY” format based on the structure indicated in the function used.

`tz` argument useful for declaring time zones

Time zone abbreviations:
https://en.wikipedia.org/wiki/List_of_tz_database_time_zones

Extract elements of dates and times

- `year()`
- `day()`
- `minute()`
- `wday(label = TRUE)`
- `month(label = TRUE)`

Determine interval lengths

- Subtraction to determine interval
- Creates a timeDiff data type
- Use as.numeric to convert to other units
  - `secs, hours, days, mins`

- When working in a tibble, `lag()` can be useful

See lubridate cheat sheet for more functions!

Loading data from Google Drive

Read data in from Drive

```
> install.packages("googlesheets4")
> gs4_auth()
```

Authorizes R to access your Google Drive account

```
> gs4_deauth()
```

Deauthorizes R to access your Google Drive account

```
> read_sheet()
```

Reads from a Google Sheets web address

Generate conditional outputs

Glue offers interpreted string literals that are small, fast, and dependency-free.

Interpreted string literals

`-glue(string {interpreted literal})`

- Literal = fixed value
- Useful when mutating or working with single variables

```
> 50 <- age
> glue("His age is {age}")
```

Interpreted string literals

`-glue_data(string {interpreted literal})`

- Used when getting a single result from a tibble.

Working with factors

Reminder:

Factors are R’s way of representing categorical data

Inspecting factors

- `fct_count()`
  - Count entries in a factor
- `fct_unique()`
  - Display the unique values in the factor
- `fct_match()`
  - Search for a specific factor

Combine factors

- `fct_c()`
  - Append on factor onto another
- `fct_unify()`
  - Standardize the levels among various factors

Modify the order of factor levels

- `fct_relevel()`
  - Declare a modified order
- `fct_infreq()`
  - Order based on frequency
- `fct_rev()`
  - Reverse order of levels
- `fct_random()`
  - Randomize factor levels

Remove factor levels

- `fct_drop()`
  - Remove a level or all unused levels

See forcats cheat sheet for more functions!

Work with other platforms

- Reads and writes SAS, SPSS, and Strata files

SAS:

- `read_sas("path/to/file.sas7bdat")`
- `write_sas("path/to/saveLocation/file.sas7bdat")`

SPSS:

- `read_sav("path/to/file.sav")`
- `write_sav("path/to/saveLocation/file.sav")`

Strata:

- `read_dta("path/to/file.dta")`
- `write_dta("path/to/saveLocation/file.dta")`

Work with strings

Reminder:

Strings are words, phrases, and other sets of characters

Detect matching strings

- `str_detect()`
  - Detect exact string matches
- `str_starts()`
  - Detect matches that start with a given string
- `str_count()`
  - Count occurrences of a given string

- Separate multiple strings using “|” to mean “OR”

Modify string length

- `str_sub()`
  - Subset a string based on positions within the string

- `str_pad(side =, pad = )`
  - Pad strings to a given width from the left, right, or both sides of string

- `str_trim()`
  - Trim white space for left, right, or both sides of string

Mutate strings

- `str_replace()` and `str_replace_all()`
  - Replace one string with another on the first or every occurrence

- `str_to_lower()` and `str_to_upper()` and `str_to_title()`
  - Modify the case of the string

Regular expressions

- Syntax needs to be modified to account for special characters

string (type this)	matches (which matches this)	example
	a (etc.)	see("a")
\\.	.	see("\\.")
\\!	!	see("\\!")
\\?	?	see("\\?")
\\\\	\	see("\\\\")
\\(	(	see("\\(")
\\)	)	see("\\)")