Lecture 21: Strings and regular expressions

Recap: regular expressions

A *regular expression* is a pattern used to find matches in text.

Example: suppose I want to extract just the lecture number from the following file name. How would I do that?

```
"teaching/sta279-f23/slides/lecture_22.qmd"
```

Find number that comes after ____

(?<=___) \\d+

Recap: regular expressions

A *regular expression* is a pattern used to find matches in text.

Example: suppose I want to extract just the lecture number from the following file name. How would I do that?

```r
str_extract("teaching/sta279-f23/slides/lecture_22.qmd", "\\d+")
```

```
[1] "279"
```

```r
str_extract("teaching/sta279-f23/slides/lecture_22.qmd", "_\\d+")
```

```
[1] "_22"
```

```r
str_extract("teaching/sta279-f23/slides/lecture_22.qmd",
            "(?<=_)\\d+")
```

```
[1] "22"
```

Recap: regular expressions

Last time, we learned the following regular expression tools:

- `\d` matches any digit (in R, have to type `\\d` because we write the regex in a string)
- `.` matches any character (except `\n`)
- `+` means “at least once”
- `(?<=)` and `(?=)` are positive lookbehinds and lookaheads
- `|` is alternation (one pattern or another)

Recap: tools for working with strings

So far, we have learned the following:

- `str_extract` extracts the first match

```r
str_extract("teaching/sta279-f23/slides/lecture_22.qmd", "\\d+")
```

```
[1] "279"
```

- `str_exctract_all` extracts all matches

```r
str_extract_all("teaching/sta279-f23/slides/lecture_22.qmd", "\\d+")
```

```
[[1]]
[1] "279" "23"  "22"
```

Goal for today: learn more string and regex tools!

Some helpful string functions

Example: Suppose I have the following file names:

```r
file_names <- c("research/project1/code.R",
                "research/project1/data.csv",
                "teaching/sta279/lecture1.qmd",
                "teaching/sta279/example_data.csv")
```

I want to identify the files in the `research` folder. What pattern would I want to match?

"research" or "^research"

↑

"starts with"

Some helpful string functions

Example: Suppose I have the following file names:

```r
file_names <- c("research/project1/code.R",
                "research/project1/data.csv",
                "teaching/sta279/lecture1.qmd",
                "teaching/sta279/example_data.csv")
```

I want to identify the files in the `research` folder. What pattern would I want to match?

```r
str_detect(file_names, "research")
```

```
[1]  TRUE  TRUE FALSE FALSE
```

returns T or F for each entry in vector

Some helpful string functions

Example: Suppose I have the following file names:

```r
file_names <- c("research/project1/code.R",
                "research/project1/data.csv",
                "teaching/sta279/lecture1.qmd",
                "teaching/sta279/example_data.csv")
```

I want to identify the files in the `research` folder. What pattern would I want to match?

```r
str_subset(file_names, "research")
```

```
[1] "research/project1/code.R"    "research/project1/data.csv"
```

return the strings which match the pattern

Some helpful string functions

Example: Suppose I have the following file names:

```r
file_names <- c("research/project1/code.R",
                "research/project1/data.csv",
                "teaching/sta279/lecture1.qmd",
                "teaching/sta279/example_data.csv")
```

I want to identify the files in the `research` folder. What pattern would I want to match?

```r
str_view(file_names, "research")
```

```
[1] │ <research>/project1/code.R
[2] │ <research>/project1/data.csv
```

part that matched the pattern

Some helpful string functions

Example: Suppose I have the following file names:

```r
file_names <- c("research/project1/code.R",
                "research/project1/data.csv",
                "teaching/sta279/lecture1.qmd",
                "teaching/sta279/example_data.csv")
```

How would I select only the csv files?

Some helpful string functions

Example: Suppose I have the following file names:

```r
file_names <- c("research/project1/code.R",
                "research/project1/data.csv",
                "teaching/sta279/lecture1.qmd",
                "teaching/sta279/example_data.csv")
```

How would I select only the csv files?

```r
str_subset(file_names, "csv")
```

```
[1] "research/project1/data.csv"
"teaching/sta279/example_data.csv"
```

Some helpful string functions

Example: Suppose I have the following file names:

```r
file_names <- c("research/project1/code.R",
                "research/project1/data.csv",
                "research/project2/sim_output.csv",
                "teaching/sta279/lecture1.qmd",
                "teaching/sta279/example_data.csv")
```

How would I select only the csv files in the `research` directory?

research ___ csv

something

↓

.+

(any character, one or more times)

option1:

str_subset(filenames, "research") |>

str_subset("csv")

option2:

str_subset(filenames, "research.+csv")

Some helpful string functions

Example: Suppose I have the following file names:

```r
file_names <- c("research/project1/code.R",
                "research/project1/data.csv",
                "research/project2/sim_output.csv",
                "teaching/sta279/lecture1.qmd",
                "teaching/sta279/example_data.csv")
```

How would I select only the csv files in the `research` directory?

```r
str_subset(file_names, "research.+csv")
```

```
[1] "research/project1/data.csv"
"research/project2/sim_output.csv"
```

More regular expressions

```r
strings <- c("apple", "banana", "raspberry",
             "blackberry", "grrreat", "random")
```

How would I select just `raspberry` and `blackberry`?

"berry"

More regular expressions

```r
strings <- c("apple", "banana", "raspberry",
             "blackberry", "grrreat", "random")
```

How would I select just `raspberry` and `blackberry`?

```r
str_view(strings, "berry")
```

```
[3] │ rasp<berry>
[4] │ black<berry>
```

More regular expressions

```r
strings <- c("apple", "banana", "raspberry",
             "blackberry", "grrreat", "random")
```

How would I select “raspberry”, “blackberry”, “grrreat”, and “random”?

contain r

"r"

More regular expressions

```r
strings <- c("apple", "banana", "raspberry",
             "blackberry", "grrreat", "random")
```

How would I select “raspberry”, “blackberry”, “grrreat”, and “random”?

```r
str_view(strings, "r")
```

```
[3] │ <r>aspbe<r><r>y
[4] │ blackbe<r><r>y
[5] │ g<r><r><r>eat
[6] │ <r>andom
```

More regular expressions

```r
strings <- c("apple", "banana", "raspberry",
             "blackberry", "grrreat", "random")
```

How would I select just “raspberry”, “blackberry”, and “grrreat”?

rr

More regular expressions

```r
strings <- c("apple", "banana", "raspberry",
             "blackberry", "grrreat", "random")
```

How would I select just “raspberry”, “blackberry”, and “grrreat”?

at least 2 r's

```r
str_view(strings, "rr+")
```

```
[3] │ raspbe<rr>y
[4] │ blackbe<rr>y
[5] │ g<rrr>eat
```

```r
str_view(strings, "r{2,}")
```

```
[3] │ raspbe<rr>y
[4] │ blackbe<rr>y
[5] │ g<rrr>eat
```

at least 2 r's

More regular expressions

```r
strings <- c("apple", "banana", "raspberry",
             "blackberry", "grrreat", "random")
```

How would I select just “grrreat”?

```r
str_view(strings, "r{3}")
```

```
[5] │ g<rrr>eat
```

r appearing exactly 3 times

More regular expressions

```r
strings <- c("apple", "banana", "raspberry",
             "blackberry", "grrreat", "random")
```

How would I select “apple”, “raspberry”, or “blackberry”?

or “grrreat” ?

More regular expressions

```r
strings <- c("apple", "banana", "raspberry",
             "blackberry", "grrreat", "random")
```

How would I select “apple”, “raspberry”, or “blackberry”?

```r
str_view(strings, "(.)\\1{1}")
```

```
[1] │ a<pp>le
[3] │ raspbe<rr>y
[4] │ blackbe<rr>y
[5] │ g<rr>reat
```

capture group

back reference

match the capture group (any character) repeated once one letter: (.)\\1{1}

More regular expressions

```r
strings <- c("papa", "banana", "memento",
             "blackberry", "grrreat", "random")
```

How would I select “papa”, “banana”, and “memento”?

(..)\\1{1}

More regular expressions

```r
strings <- c("papa", "banana", "memento",
             "blackberry", "grrreat", "random")
```

How would I select “papa”, “banana”, and “memento”?

```r
str_view(strings, "(..)\\1{1}")
```

```
[1] │ <papa>
[2] │ b<anan>a
[3] │ <meme>nto
```

```r
str_view(strings, "(..)+")
```

```
[1] │ <papa>
[2] │ <banana>
[3] │ <mement>o
[4] │ <blackberry>
[5] │ <grrrea>t
[6] │ <random>
```

More regular expressions

```r
strings <- c("papa", "banana", "memento",
             "blackberry", "toboggan", "random")
```

How would I select “banana” and “blackberry”?

More regular expressions

```r
strings <- c("papa", "banana", "memento",
             "blackberry", "toboggan", "random")
```

How would I select “banana” and “blackberry”?

```r
str_view(strings, "^b")
```

```
[2] │ <b>anana
[4] │ <b>lackberry
```

anchor

^ means "starts with"

More regular expressions

```r
strings <- c("papa", "banana", "memento",
             "blackberry", "toboggan", "random")
```

How would I select “papa” and “banana”?

More regular expressions

```r
strings <- c("papa", "banana", "memento",
             "blackberry", "toboggan", "random")
```

How would I select “papa” and “banana”?

```r
str_view(strings, "a$")
```

```
[1] │ pap<a>
[2] │ banan<a>
```

↑ anchor

(ends with)

=> $ is a metacharacter (not just literal $)

if we want to match a literal $
we need escape characters: \$

in R: \\$

More regular expressions

```
"The mean $\\mu$ is defined by $\\mu = \\frac{1}{n} \\sum_i x_i$"
```

How would I extract `$\mu$` and `$\mu = \frac{1}{n} \sum_i x_i$`?

pattern : in between $

pattern: \\$ $\underbrace{.+}_{\text{Something}}$ \\$

More regular expressions

```
"The mean $\\mu$ is defined by $\\mu = \\frac{1}{n} \\sum_i x_i$"
```

How would I extract `$\mu$` and `$\mu = \frac{1}{n} \sum_i x_i$`?

```
str_extract("The mean $\\mu$ is defined by $\\mu = \\frac{1}{n} \\sur
            "\\$.+\\$")
```

```
[1] "$\\mu$ is defined by $\\mu = \\frac{1}{n} \\sum_i x_i$"
```

$$\$ \underbrace{\qquad\qquad\qquad}_{\underline{\text{not}}\ \$} \$$$

More regular expressions

```
"The mean $\\mu$ is defined by $\\mu = \\frac{1}{n} \\sum_i x_i$"
```

How would I extract `$\mu$` and `$\mu = \frac{1}{n} \sum_i x_i$`?

```
str_extract_all("The mean $\\mu$ is defined by $\\mu = \\frac{1}{n} \
                "\\$[^\\$]+\\$")
```

```
[[1]]
[1] "$\\mu$"                                  "$\\mu = \\frac{1}{n} \\sum_i
x_i$"
```

Start w/ $

everything except $

end w/ $

[ ] character

[^] everything except specified characters

More regular expressions

```
"The current date (today) is November 3 [2007]."
```

How would I extract “(today)” and “[2007]”?

More regular expressions

```
"The current date (today) is November 3 [2007]."
```

How would I extract “(today)” and “[2007]”?

```
str_extract_all("The current date (today) is November 3 [2007].",
                "[\\(\\[][^\\)\\]]+[\\)\\]]")
```

```
[[1]]
[1] "(today)" "[2007]"
```

What if I just want “today” and “2007”?

More regular expressions

```r
"The current date (today) is November 3 [2007]."
```

```r
str_extract_all("The current date (today) is November 3 [2007].",
                "(?<=[\\(\\[])[^\\)\\]]+(?=[\\)\\]])")
```

```
[[1]]
[1] "today" "2007"
```

More regular expressions

```
"The current date (today) is November 3 [2007]."
```

What if I only want the words?

```
str_extract_all("The current date (today) is November 3 [2007].",
                "\\w+")
```

```
[[1]]
[1] "The"       "current"   "date"      "today"     "is"        "November"
"3"
[8] "2007"
```

More regular expressions

```
"The current date (today) is November 3 [2007]."
```

What if I only want the words?

```
str_replace_all("The current date (today) is November 3 [2007].",
                "[^\\w\\s]", "")
```

```
[1] "The current date today is November 3 2007"
```

A list of some other useful tools

- `*` means “appears `0` or more times”
- `{m}` means “appears m times”
- `\b` is a word boundary (use `\\b` in R)
- `\w` is any alphanumeric character, or underscore (use `\\w` in R)
- `(  )` is a capture group
- `[  ]` is a set of characters
- `\s` denotes spaces (use `\\s` in R)
- `^` anchors at the beginning, `$` anchors at the end