Regular Expressions in R
ISAAK short lesson

Clemens Schmid

2018-10-24

Motivation

```r
last_names <- c("Maier", "Meyer", "Tietze", "Mayr", "Rinne")

## [1] "Maier"  "Meyer"  "Tietze" "Mayr"   "Rinne"
grep(pattern = "M[ae][iy]e?r", last_names)

## [1] 1 2 4
last_names[grep(pattern = "M[ae][iy]e?r", last_names)]

## [1] "Maier" "Meyer" "Mayr"
```
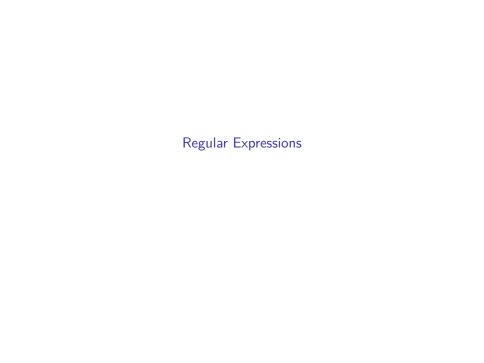
Regular Expressions

- ▶ Regular Expressions (Regex) are sign sequences that define specific search patterns
- ▶ They are pretty much universal: Many programming languages provide functions to use them and many gui/cli text editors allow to use them.
- ▶ Standards:
  - ▶ POSIX: Basic Regular Syntax (BRE)
  - ▶ **POSIX: Extended Regular Syntax (ERE)**
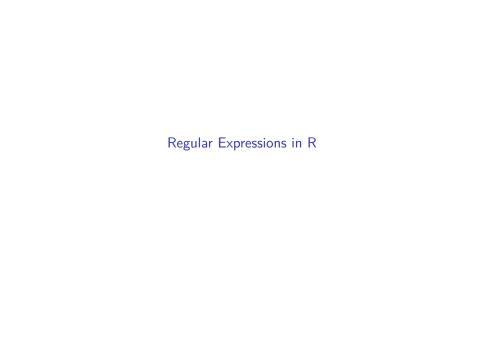 - ▶ Perl regexes

Symbols

Expression	Meaning	Example
c	Individual signs	c
.	One sign, except linebreak	.
[sign]	One of these signs	[123ufg]
[sign1 - sign2]	One sign from this sequence	[0-9]
[^sign]	One sign that is NOT one of these	[^a]
[^sign1 - sign2]	One sign NOT from this sequence	[^B-T]

Quantifiers

Expression	Meaning
?	sign before ? zero times or once
∗	sign before ∗ zero times or or any number of times
+	sign before + once or any number of times
{n}	sign before {n} n times
{n,m}	sign before {n,m} n to m times
{n,}	sign before {n,} minimum n times
{,m}	sign before {,m} maximum n times

Other Expressions

Expression	Meaning
^	Start of the line
$	End of line
\<	Start of word
\>	End of word
(...)	Definition of a subexpression
		Logical OR
\	Avoid special meaning of operators after \

Regular Expressions in R

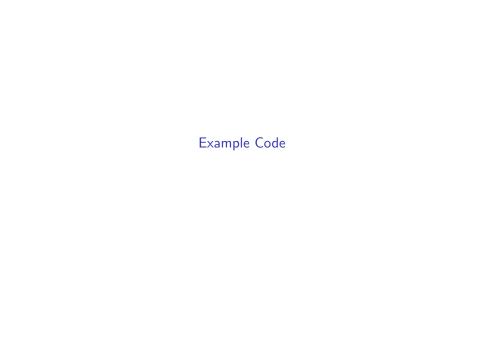## Pattern Matching and Replacement in Base R (?grep)

```r
grep(pattern, x, ignore.case = FALSE, perl = FALSE, value = FALSE,
     fixed = FALSE, useBytes = FALSE, invert = FALSE)

grepl(pattern, x, ignore.case = FALSE, perl = FALSE,
      fixed = FALSE, useBytes = FALSE)

sub(pattern, replacement, x, ignore.case = FALSE, perl = FALSE,
    fixed = FALSE, useBytes = FALSE)

gsub(pattern, replacement, x, ignore.case = FALSE, perl = FALSE,
     fixed = FALSE, useBytes = FALSE)

regexpr(pattern, text, ignore.case = FALSE, perl = FALSE,
        fixed = FALSE, useBytes = FALSE)

gregexpr(pattern, text, ignore.case = FALSE, perl = FALSE,
         fixed = FALSE, useBytes = FALSE)

regexec(pattern, text, ignore.case = FALSE, perl = FALSE,
        fixed = FALSE, useBytes = FALSE)
```

Pattern Matching and Replacement in Base R (?grep)

- **grep** & **grepl**: Search a character vector and return the indices of the matching elements. **grep** returns a vector of the indices. **grepl** returns a logical vector (TRUE/FALSE).

- **sub** & **gsub**: Replace. Search a character vector for regular expression matches and replace that match with another string. **sub** replaces the first occurrence of a pattern, **gsub** replaces all occurrences.

- **regexpr** & **gregexpr** & **regexec**: Search a character vector for matches and return the indices of the string where the match begins and the length of the match. **gregexpr** returns result as a list object. **regexec** additionally returns the locations of any parenthesized sub-expressions.

Pattern Matching and Replacement in Base R (?grep)

```r
grep(pattern, x, ignore.case = FALSE, perl = FALSE, value = FALSE,
     fixed = FALSE, useBytes = FALSE, invert = FALSE)
```

- **pattern**: Pattern to be matched. Can be regular expression or normal string (if **fixed** == TRUE).
- **x**, **text**: Character vector where matches are sought.
- **ignore.case**: Should the pattern matching be case sensitive?
- **perl**: Should Perl-compatible regexps be used?
- **value**: Should indices or matching elements be returned?
- **fixed**: Is the pattern a string to be matched as is?
- **useBytes**: Should encoding problems be ignored?
- **invert**: Should the result contain everything that does NOT match?

Example Code

Get Data

```r
library(magrittr)

radon <- c14bazAAR::get_RADON()[,c("labnr", "shortref")] %>%
  tibble::as.tibble()
```

```
## # A tibble: 20,325 x 2
##    labnr    shortref
##    <chr>    <chr>
##  1 AAR-1847 Koch 1998, 307; Aud 1995, 319
##  2 KN-2506  Breunig 1987, 142, 166
##  3 K-1983   Davidsen 1978, 55
##  4 Ua-104   <NA>
##  5 <NA>     <NA>
##  6 OxA-3197 Lanting et al. 1999/2000
##  7 KN-1988  Breunig 1987, 18
##  8 UtC-5711 Raemaekers2011, 488
##  9 ALG-33   Breunig 1987, 79
## 10 BM-1104  Breunig 1987, 96
## # ... with 20,315 more rows
```

Search for "Breunig 1987"

```r
radon %>% dplyr::filter(
    shortref == "Breunig 1987"
  )

## # A tibble: 3 x 2
##    labnr   shortref
##    <chr>   <chr>
## 1 KN-1885 Breunig 1987
## 2 OWU-59  Breunig 1987
## 3 KN-1910 Breunig 1987
```

Search for "Breunig 1987"

```r
radon %>% dplyr::filter(
    grepl("Breunig 1987", .$shortref)
  )
```

```
## # A tibble: 1,297 x 2
##    labnr      shortref
##    <chr>      <chr>
##  1 KN-2506    Breunig 1987, 142, 166
##  2 KN-1988    Breunig 1987, 18
##  3 ALG-33     Breunig 1987, 79
##  4 BM-1104    Breunig 1987, 96
##  5 KN-2985    Breunig 1987, 128
##  6 M-2166     Breunig 1987, 161
##  7 UCLA-1750B Breunig 1987, 102
##  8 Bln-1447   Breunig 1987, 172
##  9 Gro-265    Breunig 1987, 126
## 10 KN-3066    Breunig 1987, 183
## # ... with 1,287 more rows
```

Search for "Breunig 1987" from page 74–88

```r
radon %>% dplyr::filter(grepl(paste0(
    "Breunig 1987,?\\s+", # + 0 or 1 "," + 1 or more " "
    "(7[4-9]([^0-9]|$)|", # + 7 + 4-9 + no number or EOL
    "8[0-8]([^0-9]|$))"   # or 8 + 0-8 + no number or EOL
  ), .$shortref)
)
```

```
## # A tibble: 11 x 2
##    labnr shortref
##    <chr> <chr>
##  1 ALG-33 Breunig 1987, 79
##  2 M-2011 Breunig 1987, 84, 124f.
##  3 M-2165 Breunig 1987, 84, 124f.
##  4 MOC-70 Breunig 1987, 84, 123
##  5 M-2314 Breunig 1987, 84, 124f.
##  6 MOC-69 Breunig 1987, 84, 123
##  7 MOC-91 Breunig 1987, 84, 123
##  8 M-1896 Breunig 1987, 84, 123f.
##  9 M-1897 Breunig 1987, 84, 123f.
## 10 M-1986 Breunig 1987, 84, 124f.
## 11 M-2320 Breunig 1987, 84, 124f.
```

Search for "Breunig 1987" from page 74 without upper limit

```r
radon %>% dplyr::filter(grepl(paste0(
    "Breunig 1987,?\\s+",
    "([1-9]\\d{2,}|",        # + 1-9 + at least 2 more numbers
    "7[4-9]([^0-9]|$)|",     # or 7 + 4-9 + no number or EOL
    "[7-9]\\d([^0-9]|$))"    # or 7-9 + 0-9 + no number or EOL
  ), .$shortref)
)
```

```
## # A tibble: 1,279 x 2
##    labnr      shortref
##    <chr>      <chr>
##  1 KN-2506    Breunig 1987, 142, 166
##  2 ALG-33     Breunig 1987, 79
##  3 BM-1104    Breunig 1987, 96
##  4 KN-2985    Breunig 1987, 128
##  5 M-2166     Breunig 1987, 161
##  6 UCLA-1750B Breunig 1987, 102
##  7 Bln-1447   Breunig 1987, 172
##  8 Gro-265    Breunig 1987, 126
##  9 KN-3066    Breunig 1987, 183
## 10 KN-1596    Breunig 1987, 102
## # ... with 1,269 more rows
```

Resources

Resources

- ▶ Regex test and build environments:
  - ▶ https://regexr.com
  - ▶ http://buildregex.com
  - ▶ https://regex101.com
  - ▶ **https://txt2re.com**
- ▶ Fan pages:
  - ▶ https://www.rexegg.com
  - ▶ https://www.regular-expressions.info
- ▶ Short Introduction to Regex in R:
  - ▶ https://bookdown.org/rdpeng/rprogdatascience/regular-expressions.html