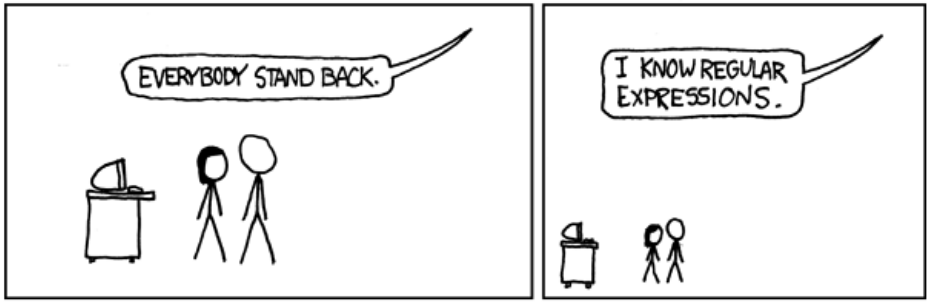
Regular expressions in R

Rita Giordano

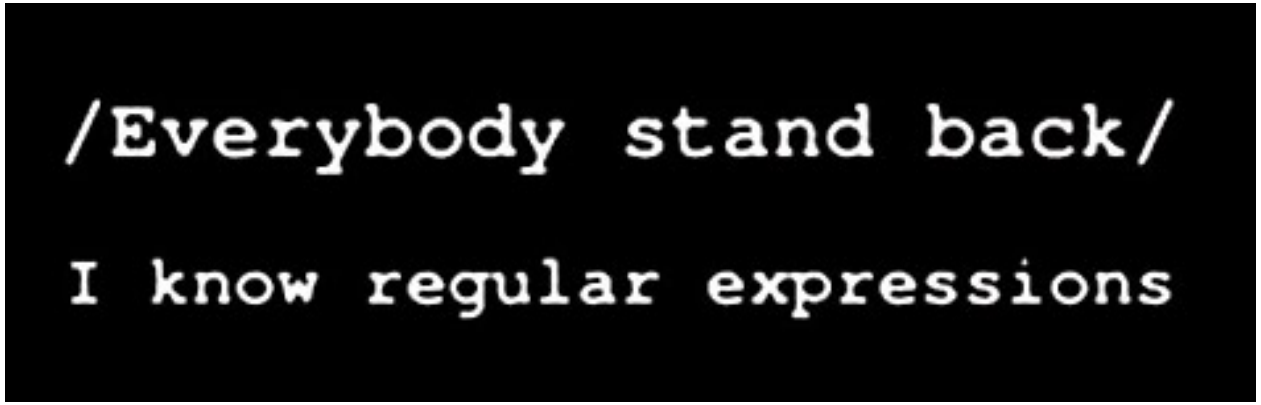
SatRday Amsterdam 01/09/2018



Regular expressions

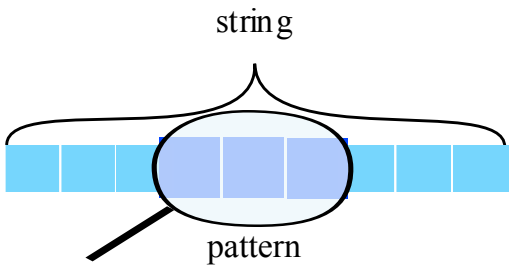
What is a regular expression?

A regular expression is a sequence of characters that define a search pattern. A pattern is a character string containing a regular expression.



What is a regular expression?

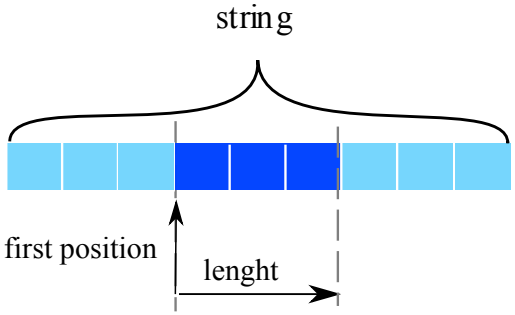
Detect pattern



grep(pattern, string)

grepl(pattern, string)

Locate pattern

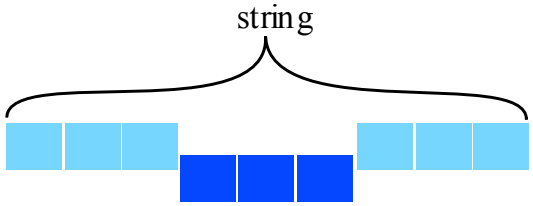


regexpr(pattern, string)

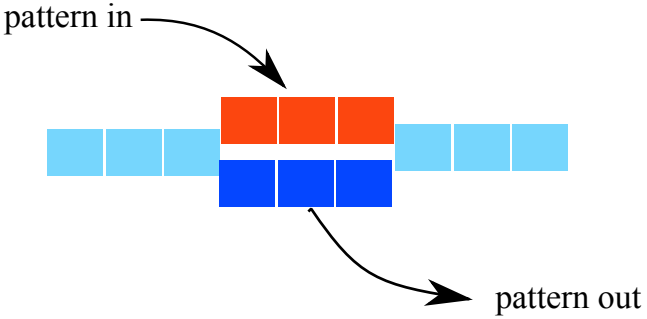
gregexpr(pattern, string)

What is a regular expression?

Extract patterns



Replace Pattern



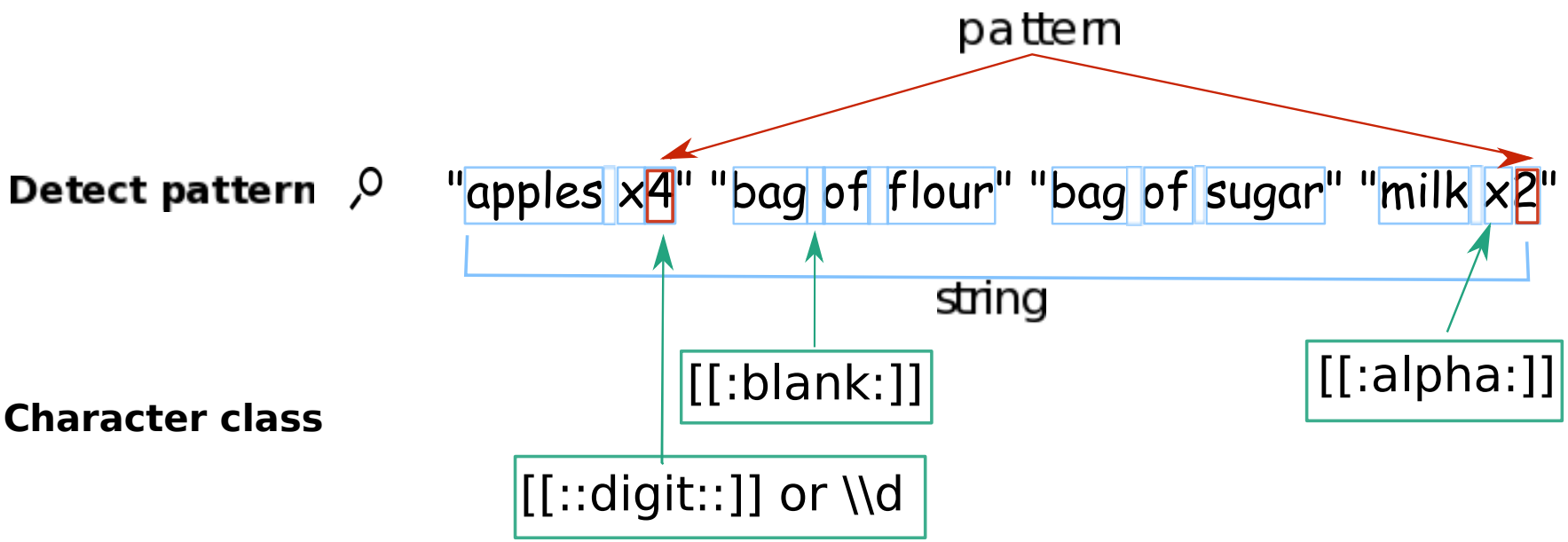
regmatches(string, regexpr(pattern, string))

regmatches(string, gregexpr(pattern, string))

sub(pattern, replacement, string)

gsub(pattern, replacement, string)

Character Class



Character Classes

```
[[:digit:]] or \\d        Digits; [0-9]
\\D                       Non-digits; [^0-9]
[[:lower:]]               Lower-case letters; [a-z]
[[:upper:]]               Upper-case letters; [A-Z]
[[:alpha:]]               Alphabetic characters; [A-z]
[[:alnum:]]               Alphanumeric characters [A-z0-9]
\\w                       Word characters; [A-z0-9_]
\\W                       Non-word characters
[[:xdigit:]] or \\x       Hexadec. digits; [0-9A-Fa-f]
[[:blank:]]               Space and tab
[[:space:]] or \\s        Space, tab, vertical tab, newline,
                          form feed, carriage return
\\S                       Not space; [^[:space:]]
[[:punct:]]               Punctuation characters;
                          !"#$%&???()*+,-./:;<=>?@[]^_`{|}~
```

https://www.rstudio.com/wp-content/uploads/2016/09/RegExCheatsheet.pdf

Example: Detect and locate pattern.

```
shopping_list <- c("apples x4", "bag of flour",
                   "bag of sugar", "milk x2")
grep("\\d", shopping_list, value = TRUE)
```

```
## [1] "apples x4" "milk x2"
```

```
grepl("\\d", shopping_list)
```

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

```
regexpr("\\d", shopping_list)
```

```
## [1]  9 -1 -1  7
## attr(,"match.length")
## [1]  1 -1 -1  1
## attr(,"useBytes")
## [1] TRUE
```

Example: Extract and remove pattern.

```
shopping_list <- c("apples x4", "bag of flour",
                   "bag of sugar", "milk x2")

# Extract first match.
regmatches(shopping_list, regexpr("\\d", shopping_list))
```

```
## [1] "4" "2"
```

```
# Replace all matche.
gsub("[[:blank:]]+", "", shopping_list)
```

```
## [1] "applesx4"   "bagofflour" "bagofsugar" "milkx2"
```

String manipulation with R: real case from cRy

Find specific pattern inside a text

Here an example of text with words and numbers: The goal is to extract the numbers and put in a data frame.

```
Text<-" Table with the numerical values
        Cycle 1, CPU 3, up/down 17.7 / 5.7,   CF 23.3
        Cycle 2, CPU11, up/down 18.6 / 7.4,   CF 26.0
        Cycle 3, CPU 4, up/down 55.3 / 34.2,  CF 89.6
        Cycle 4, CPU 7, up/down 33.8 / 17.9,  CF 51.7
        Cycle 5, CPU10, up/down 55.3 / 34.2,  CF 89.6
        Cycle 6, CPU 8, up/down 55.3 / 34.2,  CF 89.6"
```

String: Cycle 1, CPU 3, up/down 17.7 / 5.7, CF 23.3

pattern: Cycle

```r
# Read the lines of the text
data <- readLines("Text_example.dat")
# use grep to search for match the pattern
# inside the string
cycle_pat <- grep("Cycle", data, value = TRUE)
head(cycle_pat)
```

```
## [1] " Cycle 1, CPU 3, up/down 17.7 / 5.7,   CF 23.3"
## [2] " Cycle 2, CPU11, up/down 18.6 / 7.4,   CF 26.0"
## [3] " Cycle 3, CPU 4, up/down 55.3 / 34.2,  CF 89.6"
## [4] " Cycle 4, CPU 7, up/down 33.8 / 17.9,  CF 51.7"
## [5] " Cycle 5, CPU10, up/down 55.3 / 34.2,  CF 89.6"
## [6] " Cycle 6, CPU 8, up/down 55.3 / 34.2,  CF 89.6"
```

Remove elements: "gsub"

```r
# Remove: /, blanks, the words CPU and up/down
# using gsub.
tmp1 <- gsub(",|CPU|up/down|/[[:blank:]]+", "",
             cycle_pat)
cat(paste0("\t", tmp1, "\n"))
```

```
##   Cycle 1  3  17.7 5.7   CF 23.3
##       Cycle 2 11  18.6 7.4   CF 26.0
##       Cycle 3  4  55.3 34.2  CF 89.6
##       Cycle 4  7  33.8 17.9  CF 51.7
##       Cycle 5 10  55.3 34.2  CF 89.6
##       Cycle 6  8  55.3 34.2  CF 89.6
```

Split elements: "strsplit""

```r
# Split all the emlements of the string
test <- strsplit(tmp1, split = '[[:blank:]]+')
cat(paste0("\t", test, "\n"))
```

```
##  c("", "Cycle", "1", "3", "17.7", "5.7", "CF", "23.3")
##      c("", "Cycle", "2", "11", "18.6", "7.4", "CF", "26.0")
##      c("", "Cycle", "3", "4", "55.3", "34.2", "CF", "89.6")
##      c("", "Cycle", "4", "7", "33.8", "17.9", "CF", "51.7")
##      c("", "Cycle", "5", "10", "55.3", "34.2", "CF", "89.6")
##      c("", "Cycle", "6", "8", "55.3", "34.2", "CF", "89.6")
```

Create a data frame

```r
# Create a data frame using the splitted strings
tt <- as.data.frame(test, stringsAsFactors = FALSE)
#head(tt)
tt_df <- as.data.frame(t(tt), row.names = "",
                       stringsAsFactors = FALSE)
head(tt_df)
```

```
##   V1    V2 V3 V4   V5   V6 V7   V8
## 1    Cycle  1  3 17.7  5.7 CF 23.3
## 2    Cycle  2 11 18.6  7.4 CF 26.0
## 3    Cycle  3  4 55.3 34.2 CF 89.6
## 4    Cycle  4  7 33.8 17.9 CF 51.7
## 5    Cycle  5 10 55.3 34.2 CF 89.6
## 6    Cycle  6  8 55.3 34.2 CF 89.6
```

Conclusion

- Learn RegEx
- Do not copy, paste, remove . . . Use RegEx

Thank you for your attention

Dank u wel

mail: rgiordano@gmx.com

twitter: @rgiordano79

github: rgior

blog: rasrita.wordpress.com