String manipulation in R

Rita Giordano
rita@rladies.org
R-Ladies Strasbourg meetup

22 November, 2017

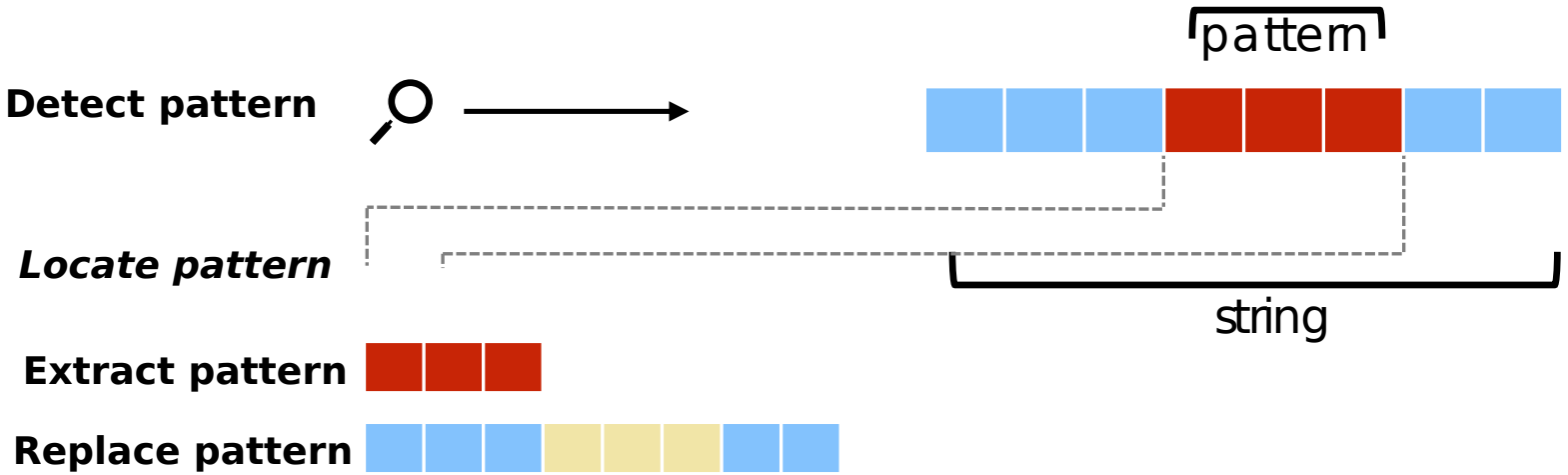
1. Regular expressions
2. String manipulation with R base: real case
3. Extract tables from a text

Regular expressions

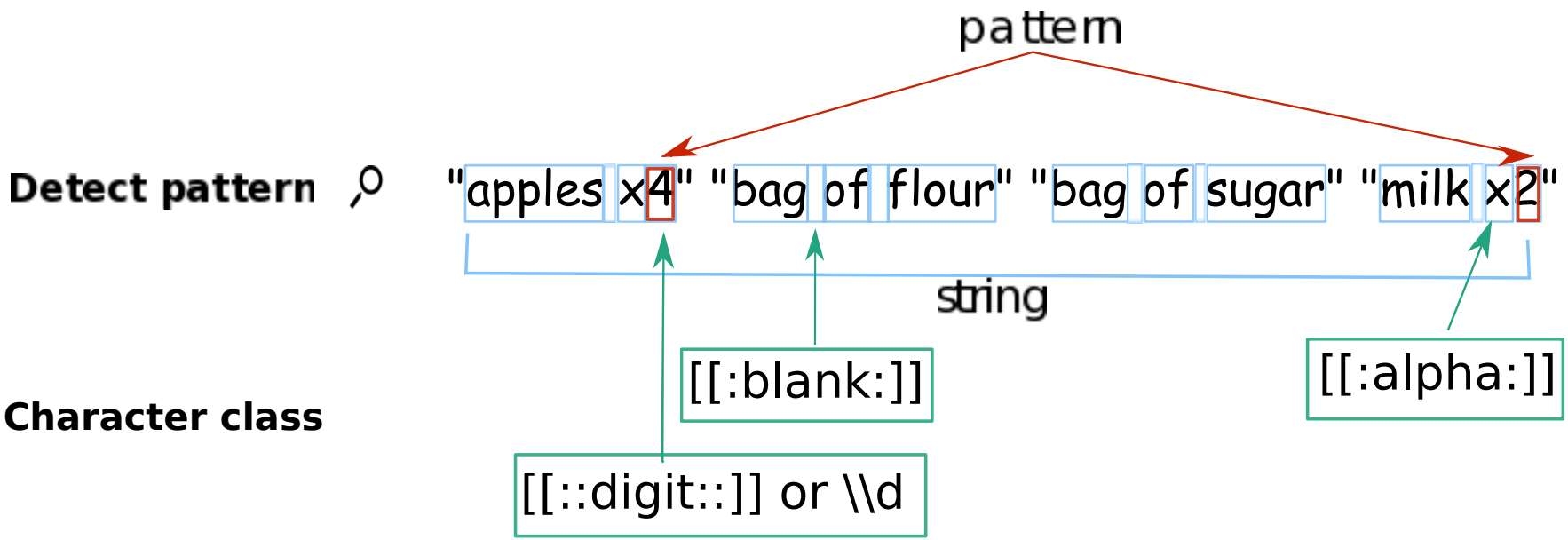
What is a regular expression?

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

A pattern is a character string containing a regular expression.



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

Functions for pattern matching

```
# ========= Detect pattern inside string ====================
grep(pattern, string)  # position of the pattern
grep(pattern, string, value = TRUE)  # print the pattern
grepl(pattern, string) # Return a Logical value
stringr::str_detect(string, pattern)
# ============== Locate Pattern ============================
regexpr(pattern, string)
stringr::str_locate(string, patter)
# ========== Extract pattern ===============================
regmatches(string, regexpr(pattern, string))
stringr::str_extract(string, pattern)
```

Example with shopping list

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

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

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

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

```
stringr::str_extract(shopping_list, "\\d")
```

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

String manipulation with R base: real case

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"

```
# 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""

```
# 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")
```

```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
```

```r
# Create a new data frame with only two columns
extract_data <-cbind(tt_df$V5,tt_df$V6)
colnames(extract_data) <- c("Up","Down")
extract_data <- as.data.frame(extract_data,
                              stringsAsFactors = FALSE)
head(extract_data)
```

```
##     Up Down
## 1 17.7  5.7
## 2 18.6  7.4
## 3 55.3 34.2
## 4 33.8 17.9
## 5 55.3 34.2
## 6 55.3 34.2
```

Extract tables from a text

File with table

```
Tables <- " Estimated mean FOM = 0.331   Pseudo-free CC = 39.43 %

 Anomalous density (in sigma units) at input heavy atom sites

  Site      x         y         z      occ*Z   density
    1    0.2955    0.6663    0.8139  34.0000   78.72
    2    0.0440    0.9597    0.9108  30.0118   72.60
    3    0.1311    0.8311    0.7595  22.7358   59.34
    4    0.2372    0.8598    0.6269  22.4944   51.30
    5   -0.1208    0.9947    1.0742  22.4638   59.70
    6   -0.1279    0.8679    0.7597  22.1476   47.47
    7   -0.2093    0.9582    1.1511  20.4102   48.09
    8   -0.0166    1.1318    1.1131  20.3048   45.05
    9   -0.1708    1.0235    1.1987  19.1454   43.51
   10    0.0860    0.9148    0.6801  18.8938   46.18
   11   -0.1698    1.0701    1.1612  18.7476   45.23
   12    0.0444    0.9156    0.7880  18.1356   42.89
   13   -0.3796    0.9911    1.1105  17.4556   41.37
   14    0.0825    0.9601    0.7197  17.2176   42.09
   15   -0.3082    1.0366    1.0438  12.3998   31.67
   16   -0.0498    0.7929    0.7227  11.9408   30.10
   17    0.0955    1.0634    1.0448   9.7376   26.47
   18   -0.0156    1.1856    1.1636   7.5922   19.92
Site     x       y       z   h(sig) near old   near new
   1  0.2957  0.6664  0.8135   78.9   1/0.04   37/1.98 43/2.01 51/2.02 23/2.06
   2  0.0436  0.9595  0.9108   72.7   2/0.03   26/2.00 25/2.03 48/2.12 20/2.91
"
```

Identify position of the table: grepl

grepl returns a logical vector, match or not for each element of x.

```
data <- readLines("Text_Table.dat")
# Identify the line number associated with the word "Site "
npos <- which(grepl("Site ", data))
# Since this word appears two time in text, npos contains
# the two line numbers corresponding to it.
npos
```

```
## [1] 373 393
```

```r
# Number of rows between the first character "Site"
# and the second.
nrow_1 <- (npos[2] - npos[1] - 2)
# Use read.table to read all the data starting from the line
# numbers selected.
Table1 <- read.table("Text_Table.dat", skip = npos[1] - 1,
                     nrows = nrow_1, header = TRUE,
                     blank.lines.skip = TRUE)
head(Table1)
```

```
##   Site       x      y      z   occ.Z density
## 1    1  0.2955 0.6663 0.8139 34.0000   78.72
## 2    2  0.0440 0.9597 0.9108 30.0118   72.60
## 3    3  0.1311 0.8311 0.7595 22.7358   59.34
## 4    4  0.2372 0.8598 0.6269 22.4944   51.30
## 5    5 -0.1208 0.9947 1.0742 22.4638   59.70
## 6    6 -0.1279 0.8679 0.7597 22.1476   47.47
```

Thank you for your attention

Merci pour votre attention

Any questions?