Text as Data

Session 3: Basics in R, Part 2

Mirko Wegemann

Universität Münster
Institut für Politikwissenschaft

29 April 2026

Your take-home tutorial

1. R and the RStudio environment

Your take-home tutorial

1. R and the RStudio environment
2. Creating vectors; assigning to objects

Your take-home tutorial

1. R and the RStudio environment
2. Creating vectors; assigning to objects
3. Data manipulation in R

This Week

- Regular Expressions
- Loops and functions

Preparation

Download the following files (again) from my website:

1. Introduction (Exercise)
2. Tutorial
3. Data on members of the German Bundestag

Regular Expressions I

A fundamental operation when working with text is Regular Expressions (see Chapter 15 in Wickham et al. (2023)). They are often abbreviated as *regex*.

- Text as a collection of different tokens
- Tokens can be classified (e.g., we know that "6" is a number, "W" is a letter, and " " is a whitespace)
- This allows us to find, extract, replace, or delete important information without knowing the exact content of the text

Regular Expressions II

No one can memorize all Regular Expressions, but an overview can be found, among other places, here.
Some of the most important regex are:

- \\w: matches all letters
- \\d: matches all numbers
- \\s: matches all whitespaces
- if you write the letters in uppercase, everything that does not match a letter/number or whitespace is matched

Regular Expressions III

stringr has also defined some additional useful regex, including

::lower:: : recognizes lowercase letters

::upper:: : recognizes uppercase letters

::punct:: recognizes punctuation

::alnum:: recognizes numbers and letters

Regular Expressions IV

"Character classes"

- we can pack a bag of patterns that target certain tokens
- this is done using square brackets []
- Example:
  1. $[a-z]$ searches for letters between a-z (the entire alphabet)
  2. $[\hat{}a-z]$ searches for everything that is not a lowercase letter between a-z

Regular Expressions V

The logical OR

- often we are not looking for one pattern, but for multiple patterns
- an "or" symbol is denoted using |
- Example: "\\d|\\w" looks for digits and word characters

Regular Expressions VI

Escaping

- some symbols are reserved for a function in R
- for example, a "." means we match all characters
- to match a period ("."), we need to escape the symbol; for this, we need two \\
- other symbols we need to escape: ., $, |, *, +, ?, ,, (,)

Regular Expressions VII

Quantifier:
Sometimes we want to match a patterna given time:

- ? recognizes a pattern that occurs 0 or 1 time
- $+$ recognizes a pattern that occurs at least 1 time
- $*$ recognizes a pattern that occurs or not
- n recognizes a pattern that occurs exactly n times
- $n,$ recognizes a pattern that occurs n times or more

Regular Expressions VIII

In R, there are different packages that allow us to use Regular Expressions. We will use *stringr* from the *tidyverse*.
It comes with different functions, some of the most important ones:

- *str_detect(string, pattern)* recognizes a certain pattern in the text and returns a boolean vector (TRUE/FALSE)
- *str_extract(string, pattern)* recognizes a pattern and extracts it
- *str_remove(string, pattern)* recognizes a pattern and deletes it from the text
- *str_replace(string, pattern, replacement)* recognizes a pattern and replaces it with another one

Regular Expressions IX

- *str_count(string, pattern)* recognizes a pattern and counts the number of occurrences in a text
- *str_split(string, pattern)* splits a variable into parts based on a certain pattern

A Simple Introduction to RegEx

Using the *str_view()* function, we can better understand how RegEx works. This function is very helpful, especially when we want to test our patterns.

```
1  > words <- c("Deutscher Bundestag", "17.
       Wahlperiode", "Tagesorientierungspunkt")
2  > str_view(words, "[a-z0-9.]+")
3   [1]  D<eutscher> B<undestag>
4   [2]  <17.> W<ahlperiode>
5   [3]  T<agesorientierungspunkt>
```

Examples of Regular Expressions

A common example is extracting years from a text. In the following code, all sequences of exactly four digits are extracted. Note the two backslashes, which are needed to escape the backslash (to indicate that it is a syntax element)

```
1 digits <- str_extract_all(syllabus, "\\d{4}")
```

Counting

We can also count the frequency of a pattern. Here, we count how often the word "ECTS" is used.

```
1 str_count(syllabus, "ECTS")
```

Removing

We can also remove a pattern. A standard example is line breaks, which are usually marked with a "\n".

```
1 syllabus2 <- str_remove_all(syllabus, "\n")
```

Replacing

When we remove a pattern, we may also remove a natural separation between the word or sequence of numbers before and after. To avoid this, we often replace it with a whitespace.

```
1 syllabus2 <- str_replace_all(syllabus, "\n", " ")
```

Tutorial – 2. Step

Now back to the application. Open "tutorial2_empty.RMD". Try to solve the exercises.

Loops I

The idea of loops is that we want to perform operations on different objects without having to write them down each time. Examples are...

- opening different web pages (important for scraping)
- importing multiple files
- editing multiple variables according to the same pattern

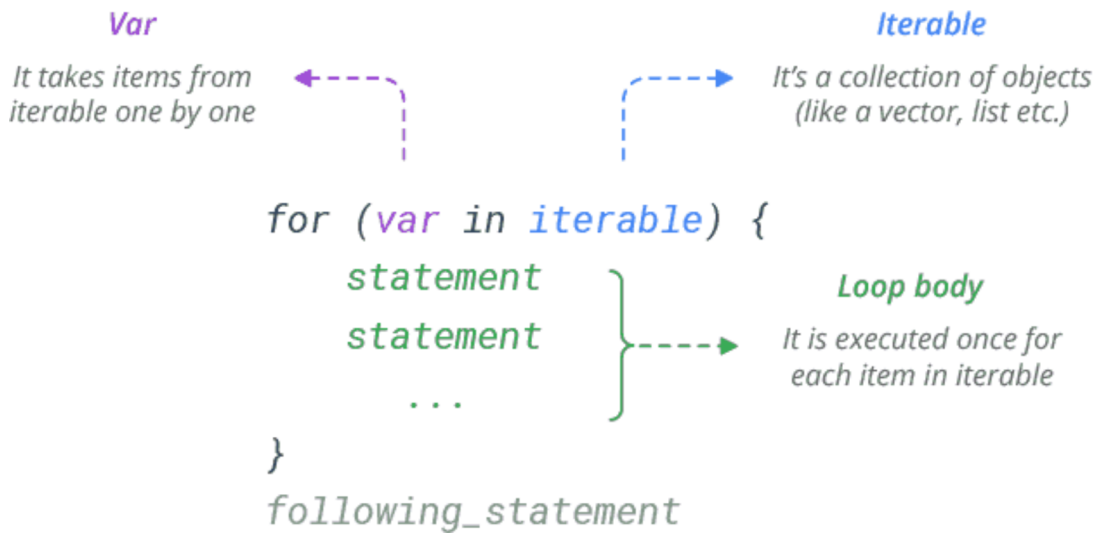
Loops II

There are different types of loops in R

- *for*-Loops go through each element of a defined list or sequence and perform an operation
- *while*-Loops go through a list until a certain condition is met

We will primarily use *for*-Loops.

The Structure of a Simple for-Loop



Tutorial on for-Loops
Graphic and Instructions for for-Loops

An Example

In the following example, we see a simple loop that goes through a sequence of numbers from 0 to 10 and displays each element of the sequence.

```
1 for(i in 0:10){
2   print(i)
3 }
```

A Brief Outlook I

Next week:

- ...we will deal with a clarification of quantitative text analysis as a form of content analysis
- Please read the following literature:
  1. Krippendorff, K. (2018). *Content Analysis: An Introduction to its Methodology* (Fourth Edition). SAGE (Chapter 2)
  2. Grimmer, J., & Stewart, B. M. (2013).Text as Data: The Promise and Pitfalls of Automatic Content Analysis Methods for Political Texts. *Political Analysis*, *21*(3), 267–297. https://doi.org/10.1093/pan/mps028

Optional: Benoit, K. (2020). Text as data: An overview.. In L. Curini & R. Franzese (Eds.), *The SAGE Handbook of Research Methods in Political Science and International Relations*. SAGE Publications Ltd. https://doi.org/10.4135/9781526486387

Literature I

Benoit, K. (2020). Text as data: An overview.. In L. Curini & R. Franzese (Eds.), *The SAGE Handbook of Research Methods in Political Science and International Relations*. SAGE Publications Ltd. https://doi.org/10.4135/9781526486387

Grimmer, J., & Stewart, B. M. (2013).Text as Data: The Promise and Pitfalls of Automatic Content Analysis Methods for Political Texts. *Political Analysis*, *21*(3), 267–297. https://doi.org/10.1093/pan/mps028

Krippendorff, K. (2018). *Content Analysis: An Introduction to its Methodology* (Fourth Edition). SAGE.

Wickham, H., Çetinkaya-Rundel, M., & Grolemund, G. (2023). *R for Data Science*. O'Reilly Media, Inc.