A Primer to Web Scraping with R

Simon Munzert, University of Konstanz
r-datacollection.com
@RDataCollection
#rwebscraping

May 12, 2015
WZB | CO:STA Session

Materials available at
https://github.com/simonmunzert/rscrapingWZB2015

Outline

Why Web Scraping?

Why Web Scraping?

Web scraping

A.k.a. screen scraping, crawling, web harvesting; computer-aided collection of predominantly unstructured data (e.g., from HTML code)

The World Wide Web is full of various kinds of new data, e.g.:

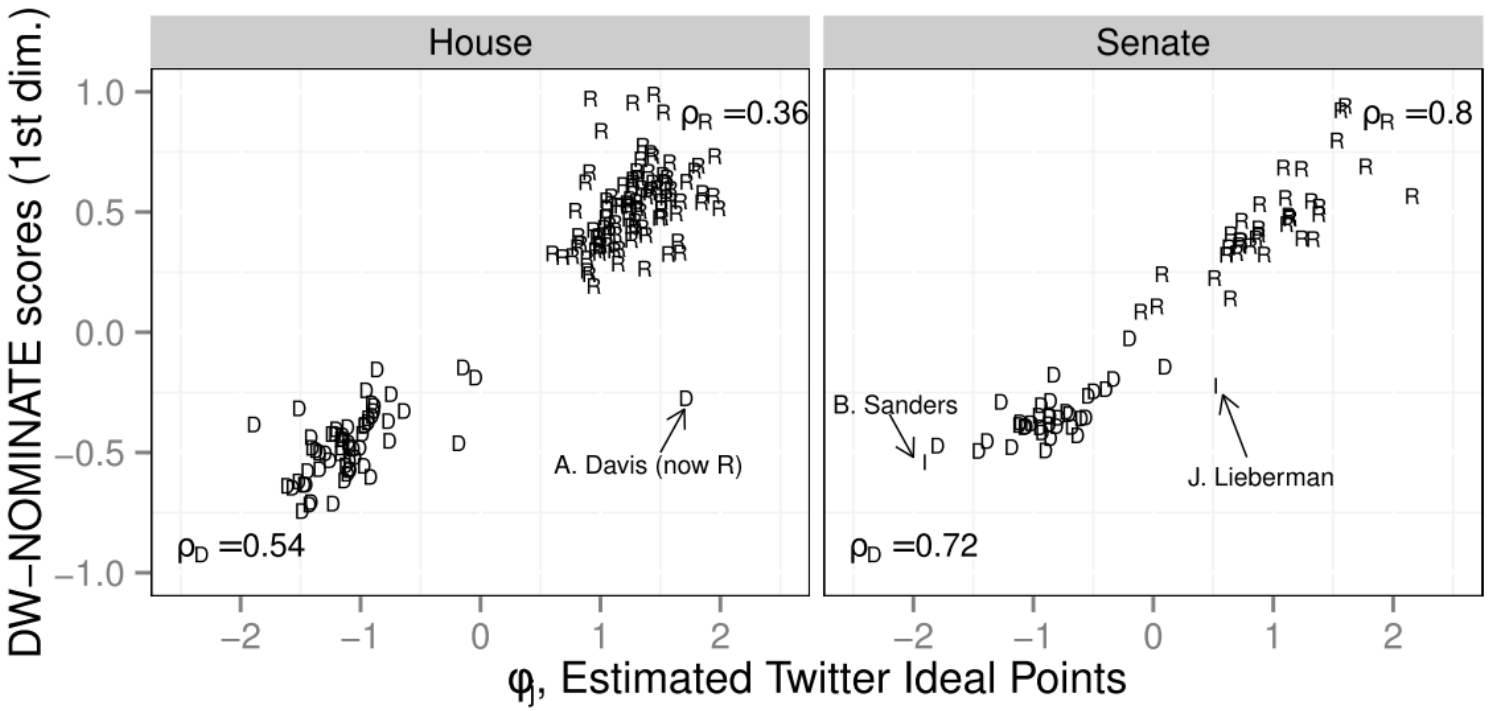
- open government data
- search engine data
- services that track social behavior

Practical arguments

- financial resources are sparse
- . . . and so is our time
- reproducibility

Barberá 2015: Ideal Point Estimation Using Twitter Data

Figure 1: Ideal Point Estimates for Members of U.S. Congress



Mellon 2014: Internet Search Data and Issue Salience

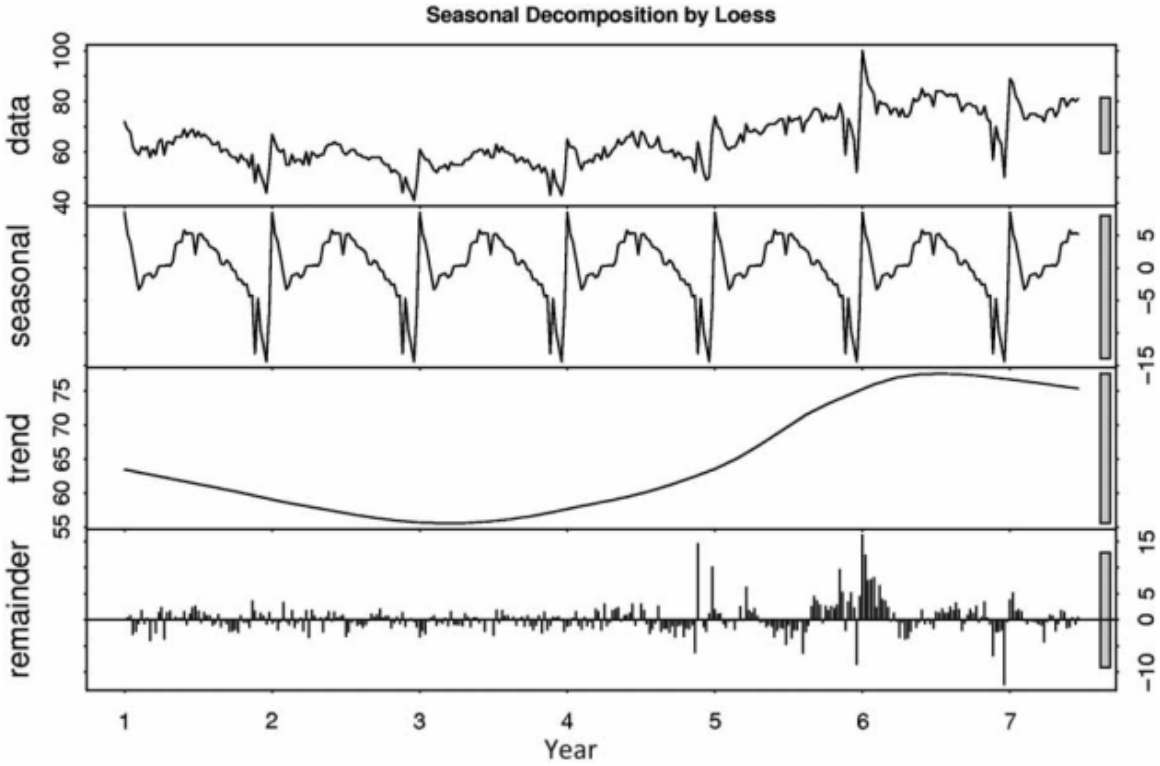


Figure 2. Seasonal trend decomposition by LOESS of searches for "jobs" on Google showing the original data and its seasonal, trend and remainder components. The axes are all expressed on the same arbitrary scale.

Measuring issue salience using Wikipedia page view data

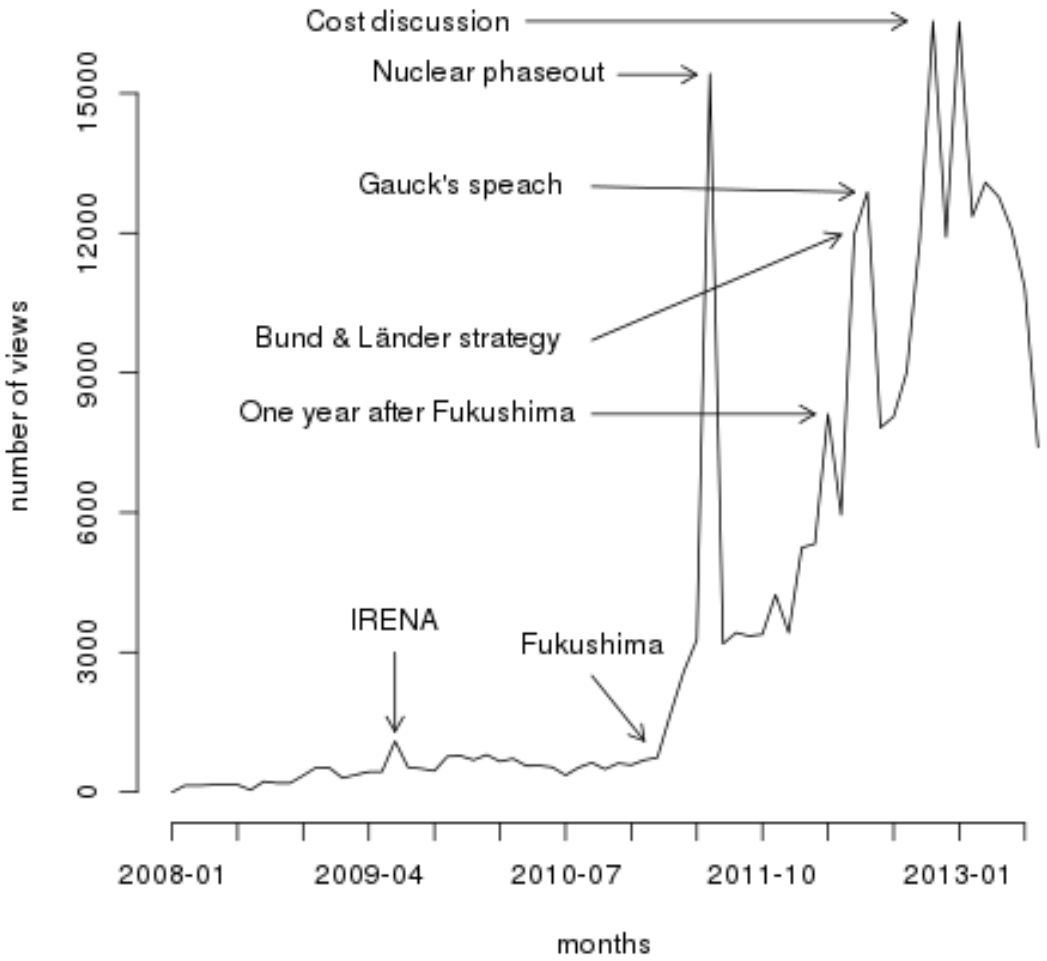
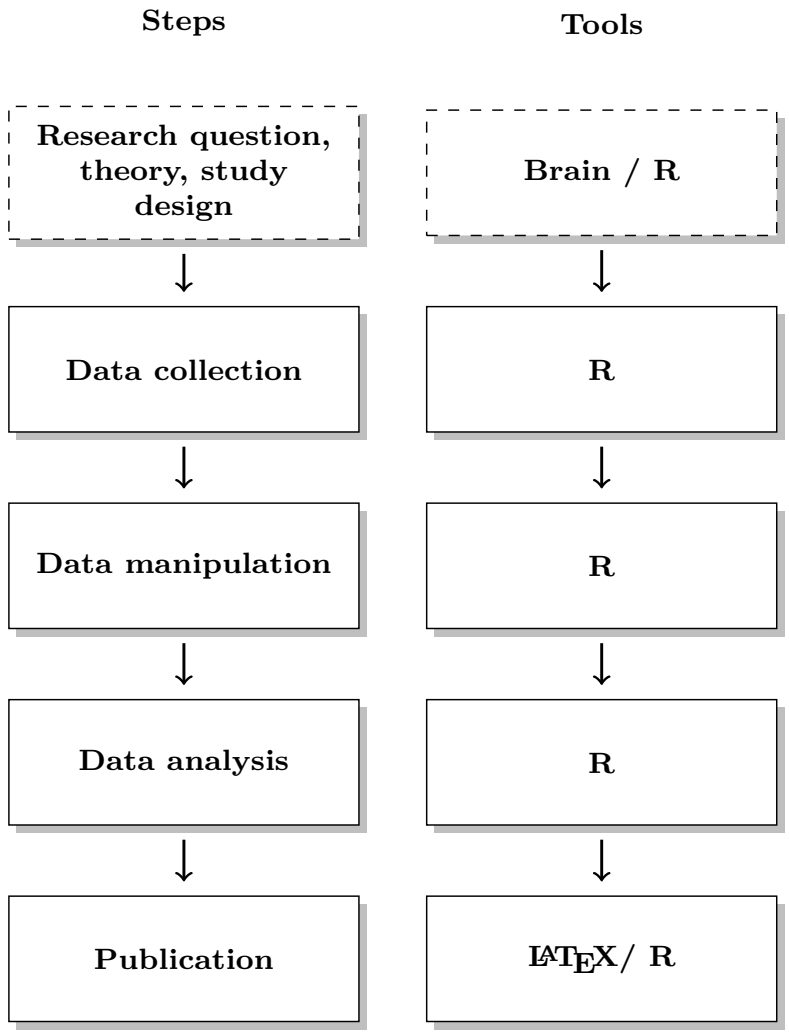


Figure 1: Wikipedia article views for "Energiewende" from January 2008 - July 2013

Why R?

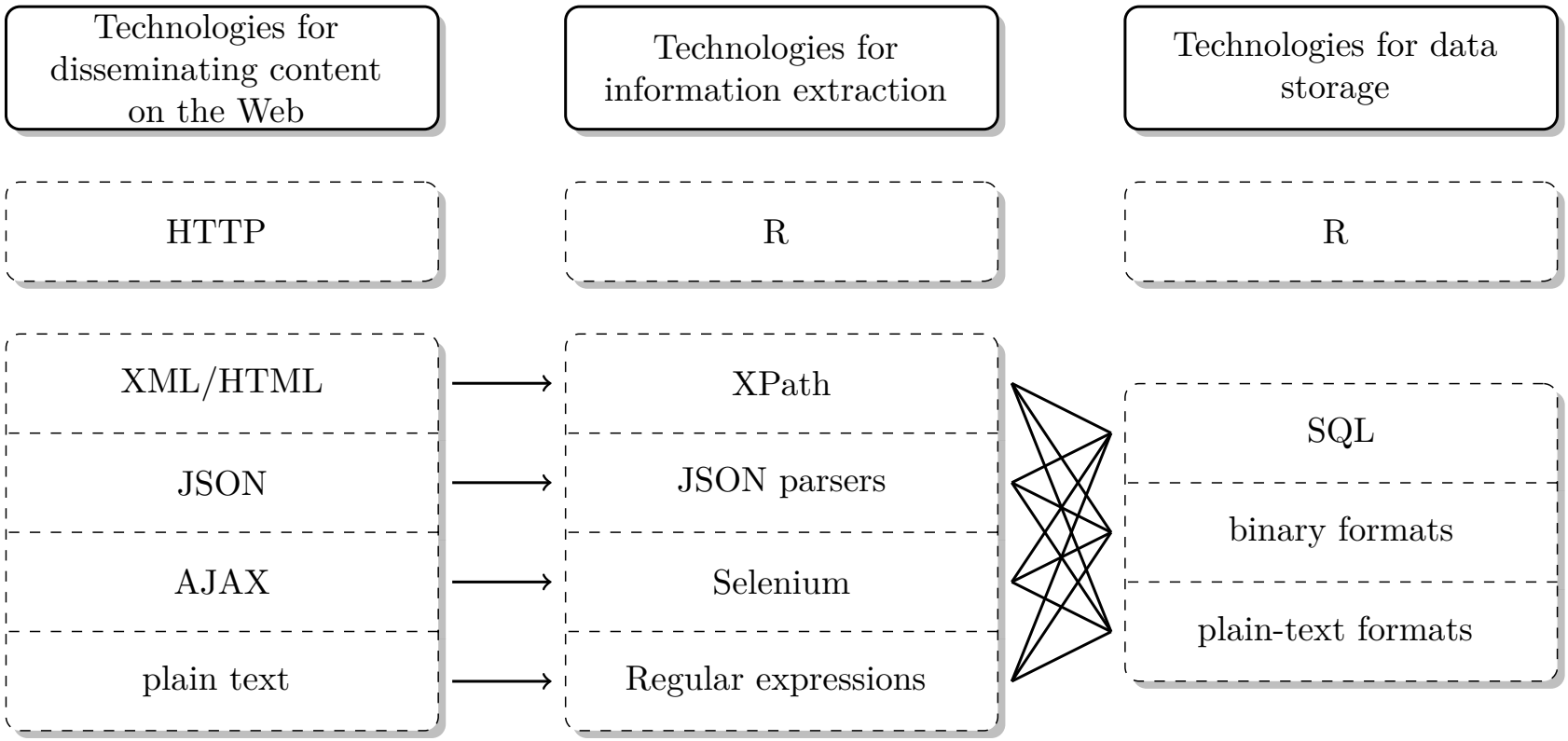
- free
- open source
- large community
- powerful tools for statistical analysis
- powerful tools for visualization
- flexible in processing all kinds of data/languages
- useful in every step of the workflow



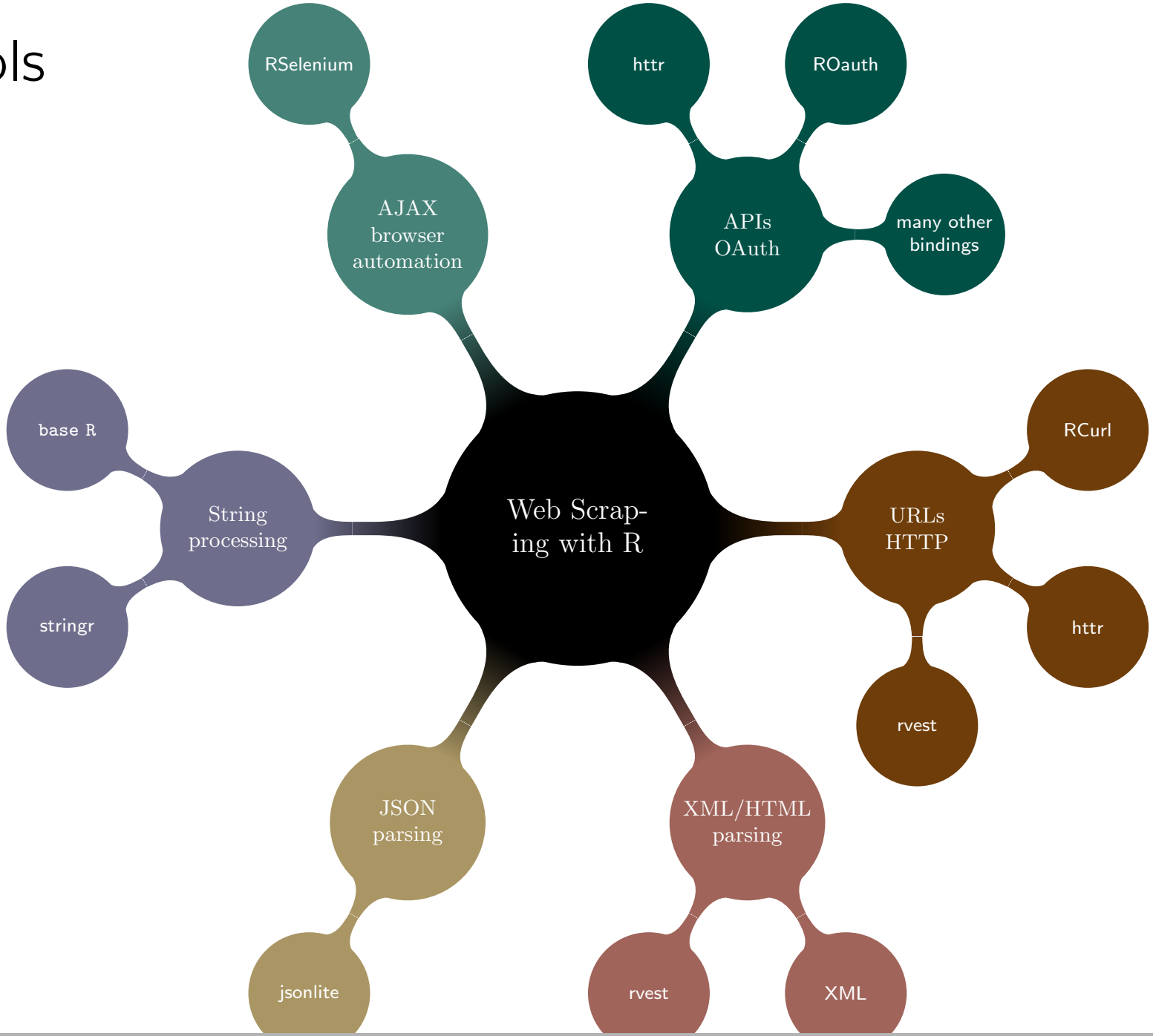
The philosophy behind web data collection with R

- no point-and-click procedure
- automation of download, parsing, and data extraction procedures
- classical screen scraping
- tapping of web services and APIs
- post-processing of text data
- reproducibility

Technologies of the World Wide Web



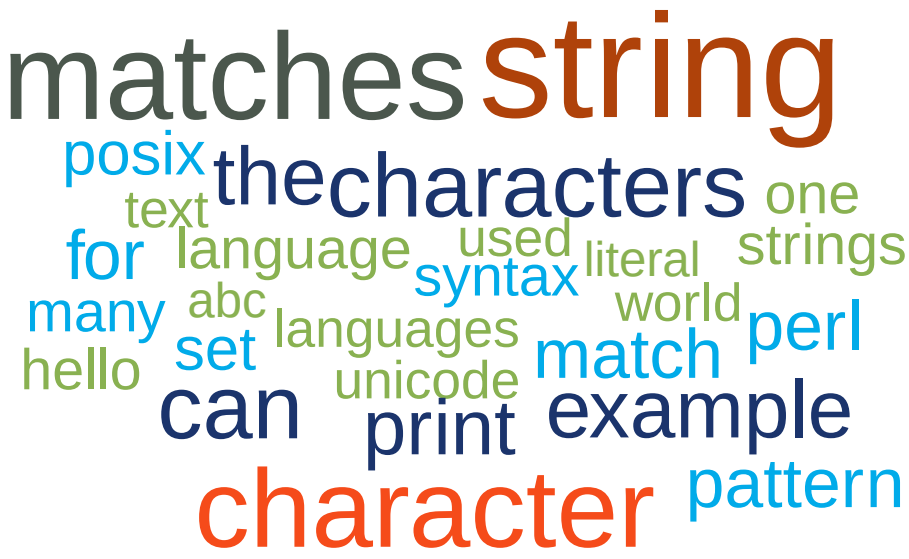
R tools


Web Scraping with R
AJAX browser automation
RSelenium
APIs OAuth
httr
ROauth
many other bindings
String processing
base R
stringr
URLs HTTP
RCurl
httr
rvest
JSON parsing
jsonlite
XML/HTML parsing
rvest
XML

Technical Setup

1. make sure that the newest version of R (currently 3.2.0; available here) is installed on your computer
2. install the newest stable version of *RStudio* (available here)
3. install the following packages:
   ```
   pkgs <- c('RCurl', 'XML', 'stringr', 'jsonlite',
   'httr', 'rvest', 'devtools', 'RSelenium', 'plyr',
   'dpylr', 'wikipediatrend', 'twitteR', 'streamR')
   ```
4. install the *Chrome* (from here) and *Firefox* browsers (from here)
5. install *Java* (from here)

Regular Expressions



What are regular expressions?

Definition

- a.k.a. *Regex* or *RegExp*
- origins in formal language theory
- sequences of characters that describe patterns in text
- implemented in many programming languages, including R

Why are regular expressions useful for web scraping?

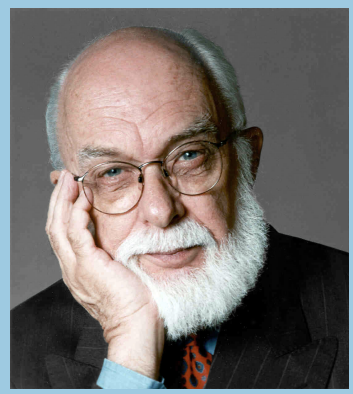
- information on the Web can often be described by patterns (email addresses, numbers, cells in HTML tables, ...)
- if the data of interest follow specific patterns, we can match and extract them—regardless of page layout and HTML overhead
- whenever the information of interest is (stored in) text, regular expressions are usful for extraction and tidying purposes

Example: mapping locations of AJPS reviewers

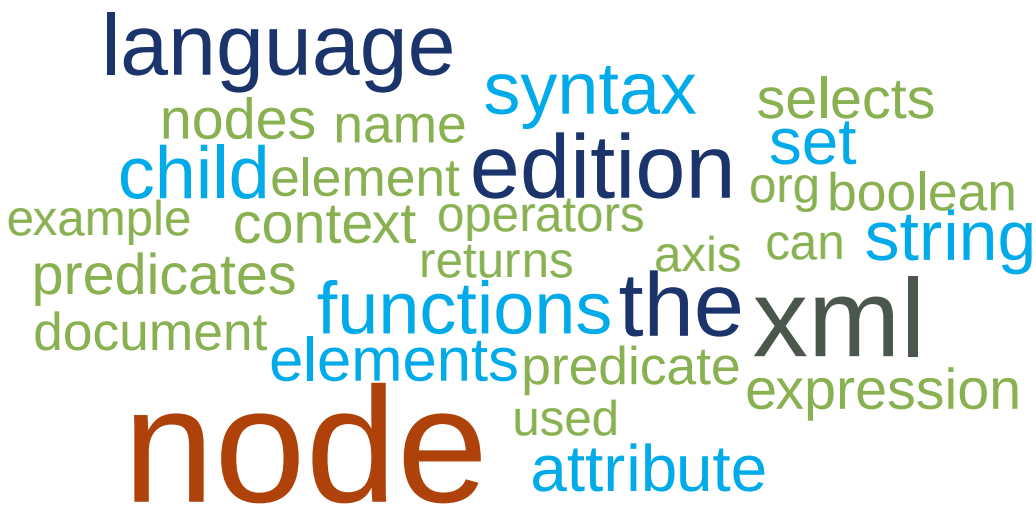
Goal: geolocate AJPS reviewers

Tasks:

- download PDF files from http://ajps.org/list-of-reviewers/
- import them into R (as plain text)
- extract information via regular expressions
- geocoding



XPath


language
syntax
selects
nodes
name
set
child
element
edition
org
boolean
example
context
operators
can
string
returns
axis
predicates
functions
the
xml
document
elements
predicate
expression
node
used
attribute

What's XPath?

Definition

- XML Path language, a W3C standard
- Query language for XML-based documents (i.e., for HTML as well)
- access node sets and extract content

Why XPath for web scraping?

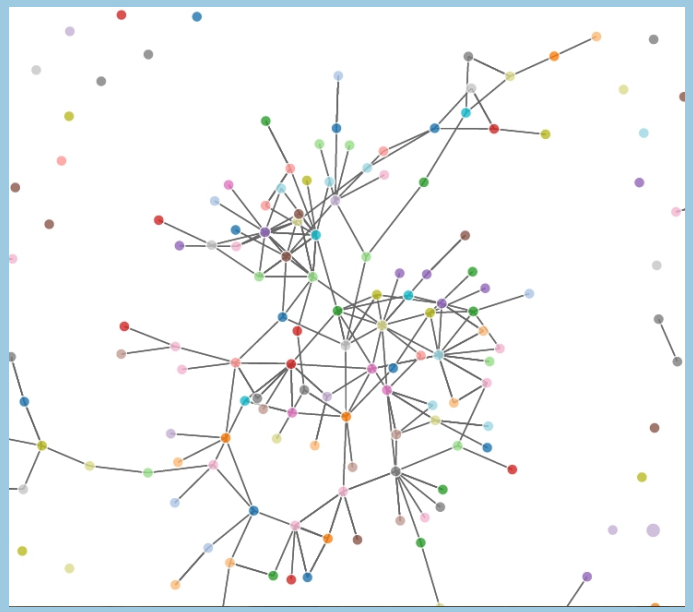
- Source code of webpages structures both layout and content
- not only content, but context matters
- enables us to extract content based on its location in the document, but (usually) regardless of its shape

Example: a Wikipedia-based network of political scientists

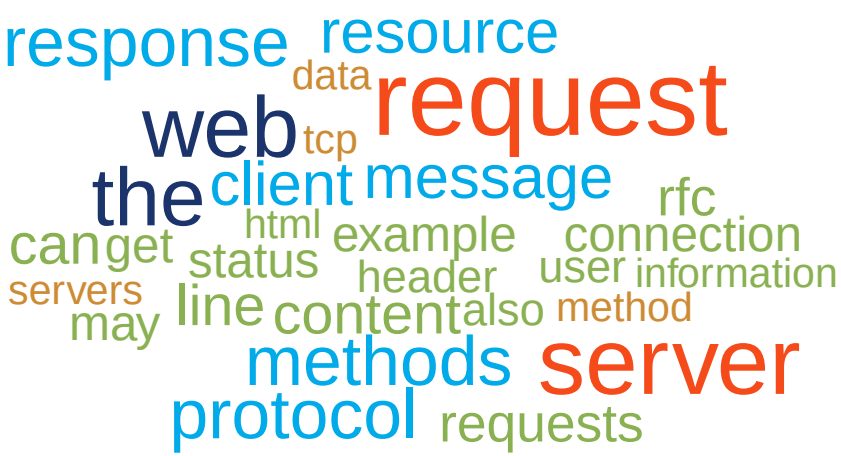
Goal: build a 'collaboration' network of political scientists

Tasks:

- gather list of political scientists
- fetch Wikipedia entries
- identify links
- construct connectivity matrix
- visualize network



APIs


response
resource
data
request
web
tcp
the
client
message
rfc
html
example
connection
can
get
status
header
user
information
servers
line
content
also
method
may
methods
server
protocol
requests

What are APIs?

Definition

- **A**pplication **P**rogramming **I**nterface
- many web services provide APIs to access their data and services (Twitter, Google, Facebook, Wikipedia, . . .)
- common data formats: XML, JSON

APIs in the context of web scraping

- instant access to clean data
- frees us from building manual scrapers
- forces us to understand the API architecture

Data gathering with APIs

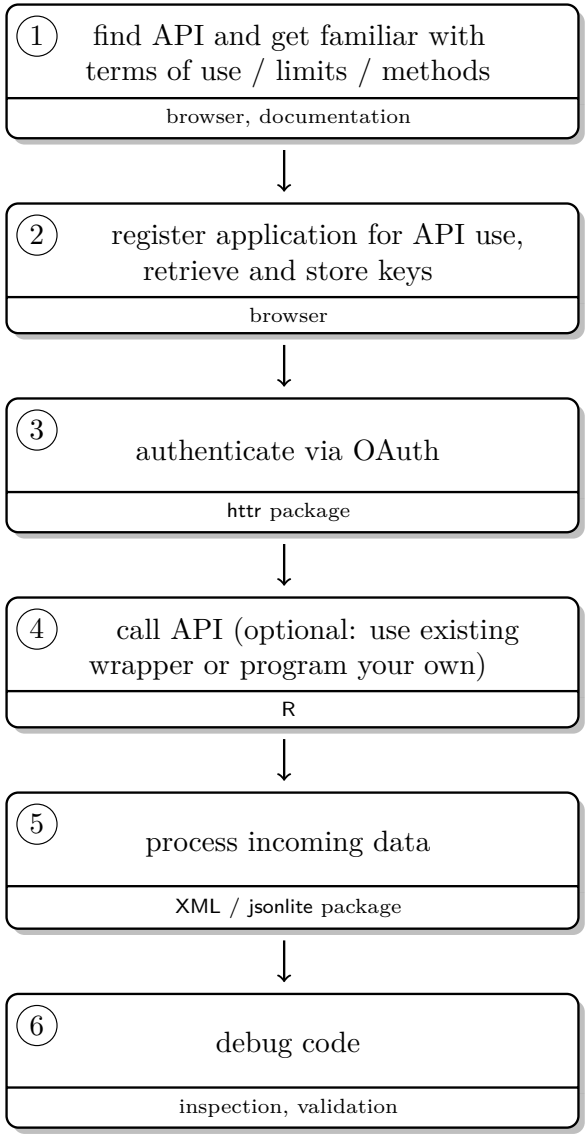
Advantages

- pure data collection without 'layout waste'
- standardized data access
- de facto automatic agreement of data owner
- robustness of calls

Disadvantages

- requires knowledge of API architecture
- dependent upon API suppliers
- not always for free

Data gathering with APIs



Finding APIs on the Web

List of APIs:
http://www.programmableweb.com/apis

rOpenSci: Collection of R-API interfaces:
http://ropensci.org/

CRAN Task View of Web Technologies:
http://cran.r-project.org/web/views/WebTechnologies.html

Social media mining with R

Why social media mining?

- network data
- communication data
- preference data

Existing R bindings

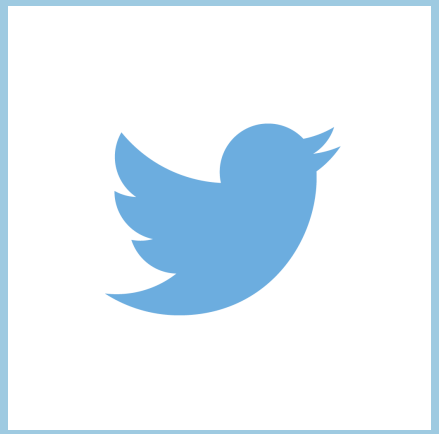
- twitteR
- streamR
- Rfacebook
- Rlinkedin
- SocialMediaMineR
- tumblR
- ...

Example: exploring Twitter's services

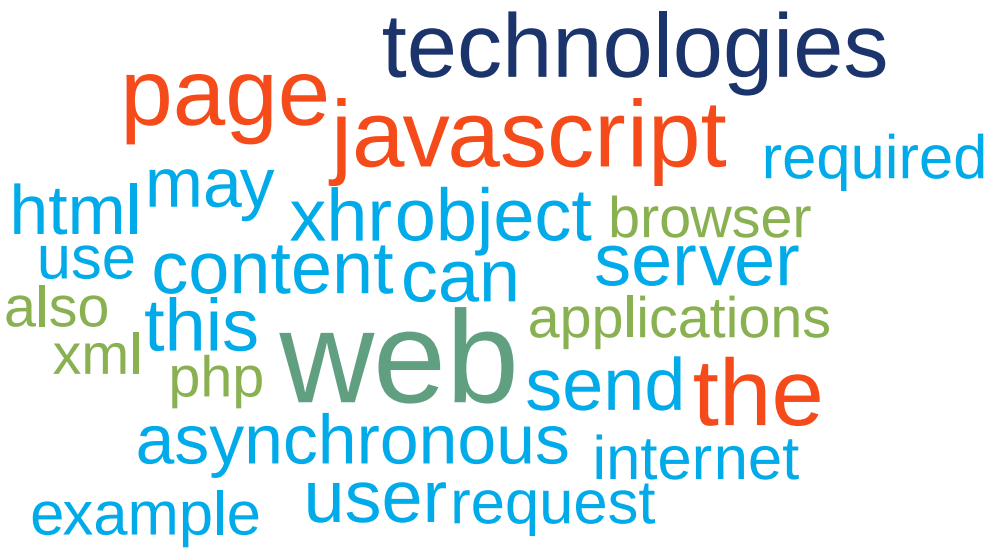
Goal: tap Twitter's REST and Streaming APIs

Tasks:

- register app
- manage authorization process
- get to know the twitteR and the streamR packages



AJAX and Selenium



What's AJAX?

- HTML/HTTP are used for static display of content
- in order to display dynamic content, they lack
  1. mechanisms to detect user behavior in the browser (and not only on the server)
  2. a scripting engine that reacts on this behavior
  3. a mechanism for asynchronous queries
- **A**synchronous **J**avaScript **a**nd **X**ML' is a set of technologies that serve these purposes
- massively used in modern webpage design and architecture
- makes classical screen scraping more difficult

Example: https://twitter.com/regsprecher

Selenium

The problem reconsidered

- dynamic data requests are not stored in the static HTML page
- therefore, we cannot access them with classical methods and packages (httr, XML, `download.file()`, etc.)

The solution

- initiate and control a web browser session with R
- let the browser do the JavaScript interpretation work and the manipulations in the live DOM tree
- access information from the web browser session

Selenium

What's Selenium?

- http://www.seleniumhq.org
- free software environment for automated web application testing
- several modules for different tasks; most important for our purposes: Selenium WebDriver
- Selenium WebDiver starts a server instance (as proxy) and passes commands (posed in R in our case) to the browser
- automated browsing via scripts

Selenium and R

Software requirements

- Java, https://www.java.com/de/download/
- Selenium server, http://selenium-release.storage.googleapis.com/2.45/selenium-server-standalone-2.45.0.jar or via RSelenium and `checkForServer()`
- Firefox browser, https://www.mozilla.org/en-US/firefox/new/
- RSelenium package

Example: tapping the IEA Global Renewable Energy database

Goal: fetch policy data from IEA database

Tasks:

- get Selenium running
- inspect HTML form on http://www.iea.org/policiesandmeasures/renewableenergy/
- access page with RSelenium
- download data output
- import data into R
- tidy data

Good Practice

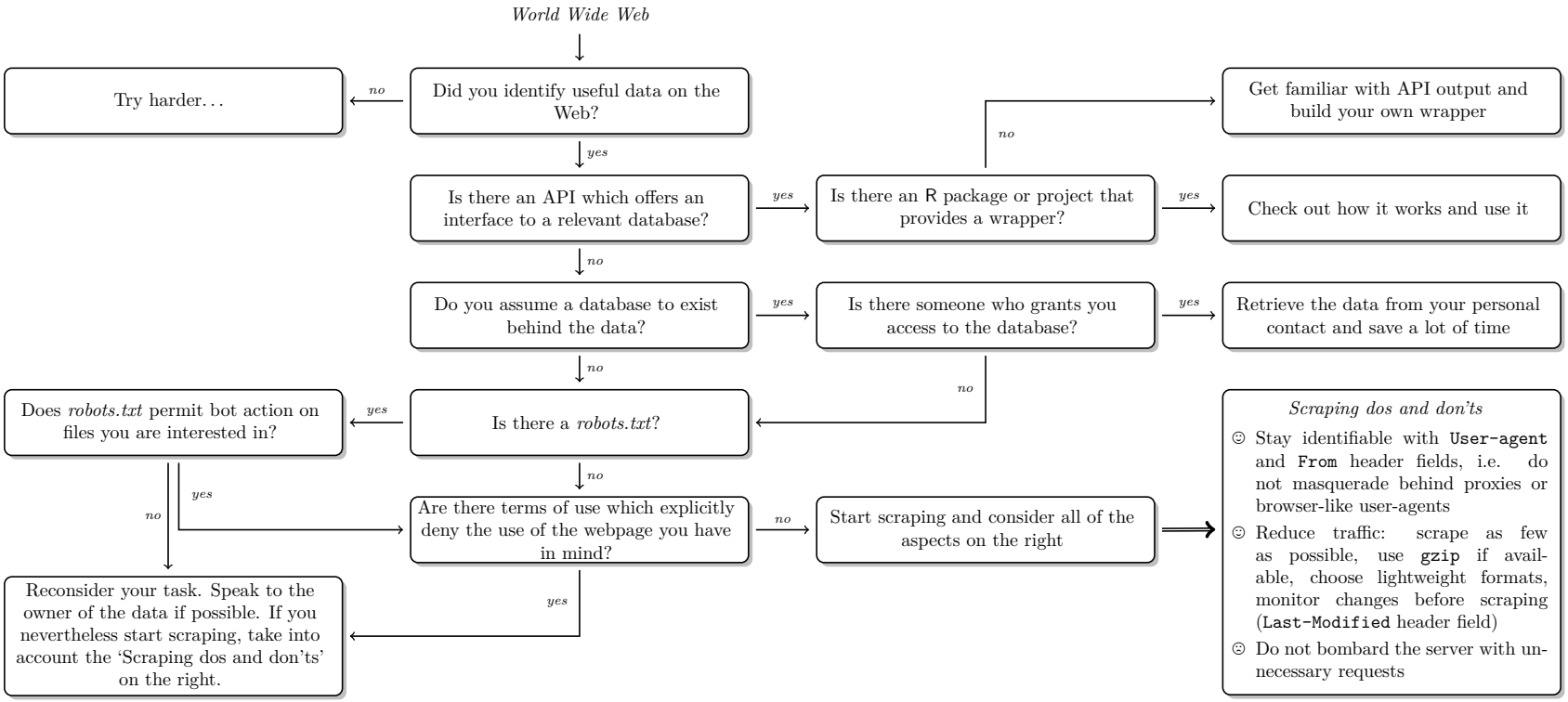
Is web scraping legal?

- no unambiguous yes or no in any country according to current jurisdiction
- so far, court cases (especially in the US) often (but not always) dealt with commercial interest and often (but not always) huge masses of data
  - eBay vs. Bidder's Edge
  - AP vs. Meltwater
  - Facebook vs. Pete Warden
  - United States vs. Aaron Swartz

A (not very useful) recommentation for your work

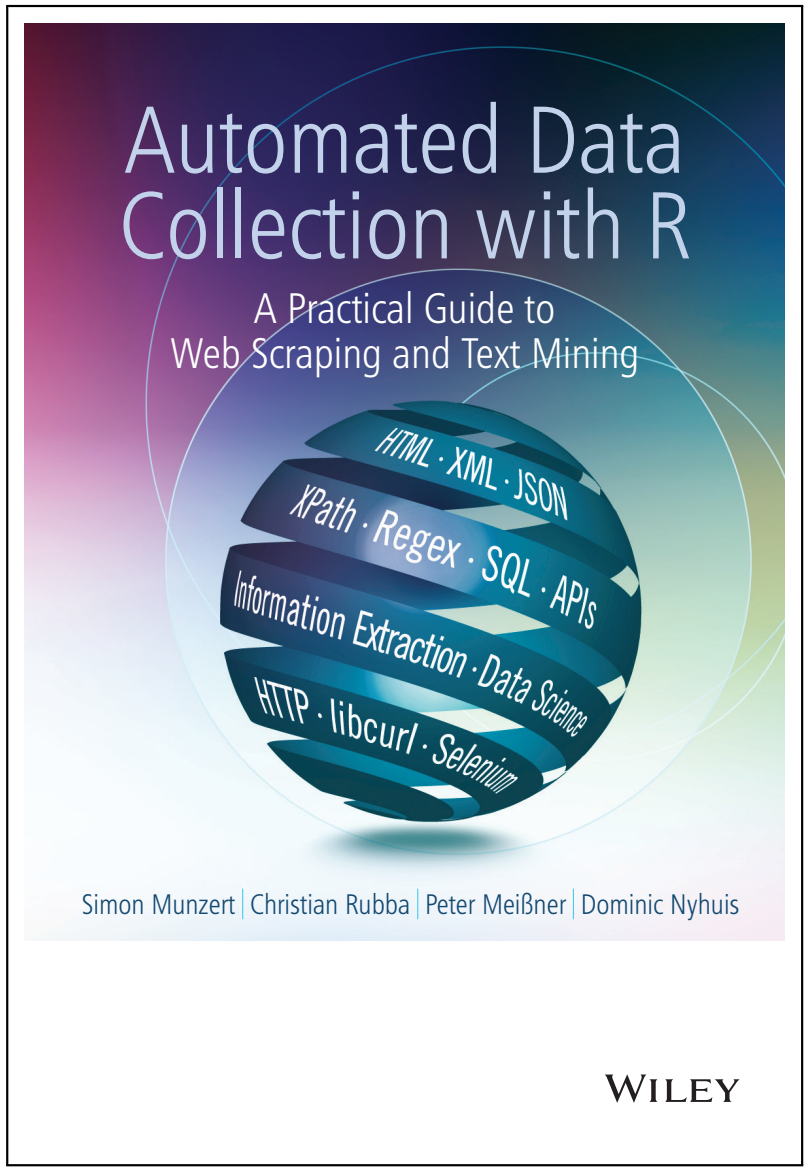
1. you take all the responsibility for your web scraping work
2. take all copyrights of a country's jurisdiction into account
3. if you publish data, do not commit copyright fraud
4. if in doubt, ask the author/creator/provider of data for permission—if your interest is entirely scientific, chances aren't bad that you get data
5. consult current jurisdiction, e.g. on http://blawgsearch.justia.com or from a laywer specialized on internet law

Scraping etiquette



Finally: a bit of self-promotion

- primers on web technologies, many scraping scenarios, and lots of code
- written between 2012 and 2014 → not entirely up-to-date anymore, but constant updates on GitHub
- homepage with materials: www.r-datacollection.com
- if you find any errors in the book, please tell us!



Thank you for your attention!

Simon Munzert
simon.munzert@uni.kn
@simonsaysnothin
@RDataCollection