ECPR Methods Summer School: Automated Collection of Web and Social Data

Pablo Barberá

School of International Relations
University of Southern California
pablobarbera.com

Networked Democracy Lab
www.netdem.org

Course website:

github.com/pablobarbera/ECPR-SC103

APIs

Social event

Save the date:

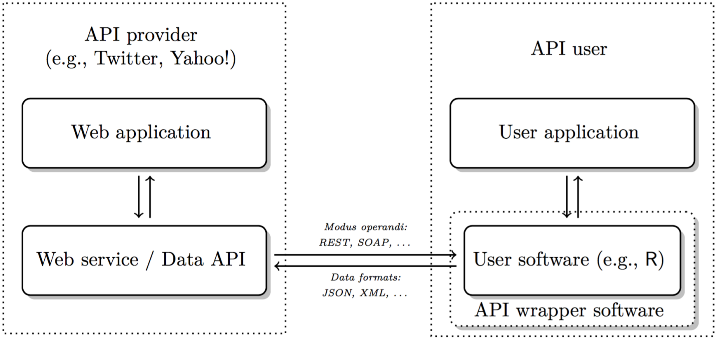
Wednesday Aug. 2nd, 6pm

Location TBA

APIs

API = Application Programming Interface; a set of structured http requests that return data in a lightweight format.

HTTP = Hypertext Transfer Protocol; how browsers and e-mail clients communicate with servers.



Source: Munzert et al, 2014, Figure 9.8

APIs

Types of APIs:

1. RESTful APIs: queries for static information at current moment (e.g. user profiles, posts, etc.)
2. Streaming APIs: changes in users' data in real time (e.g. new tweets, weather alerts...)

APIs generally have extensive documentation:

- Written for developers, so must be understandable for humans
- What to look for: endpoints and parameters.

Most APIs are rate-limited:

- Restrictions on number of API calls by user/IP address and period of time.
- Commercial APIs may impose a monthly fee

Connecting with an API

Constructing a REST API call:

- Baseline URL endpoint: `https://maps.googleapis.com/maps/api/geocode/json`
- Parameters: `?address=budapest`
- Authentication token (optional): `&key=XXXXX`

From R, use `httr` package to make `GET` request:

```
library(httr)
r <- GET(
"https://maps.googleapis.com/maps/api/geocode/json",
query=list(address="budapest"))
```

If request was successful, returned code will be `200`, where `4xx` indicates client errors and `5xx` indicates server errors.
If you need to attach data, use `POST` request.

JSON

Response is often in JSON format (Javascript Object Notation).

- Type: `content(r, "text")`
- Data stored in key-value pairs. Why? Lightweight, more flexible than traditional table format.
- Curly brackets embrace objets; square brackets enclose arrays (vectors)
- Use `fromJSON` function from `jsonlite` package to read JSON data into R
- But many packages have their own specific functions to read data in JSON format; `content(r, "parsed")`

Authentication

- Many APIs require an access key or token
- An alternative, open standard is called OAuth
- Connections without sharing username or password, only temporary tokens that can be refreshed
- `httr` package in R implements most cases (examples)

R packages

Before starting a new project, worth checking if there's already an R package for that API. Where to look?

- CRAN Web Technologies Task View (but only packages released in CRAN)
- GitHub (including unreleased packages and most recent versions of packages)
- rOpenSci Consortium

Also see this great list of APIs in case you need inspiration.

Why APIs?

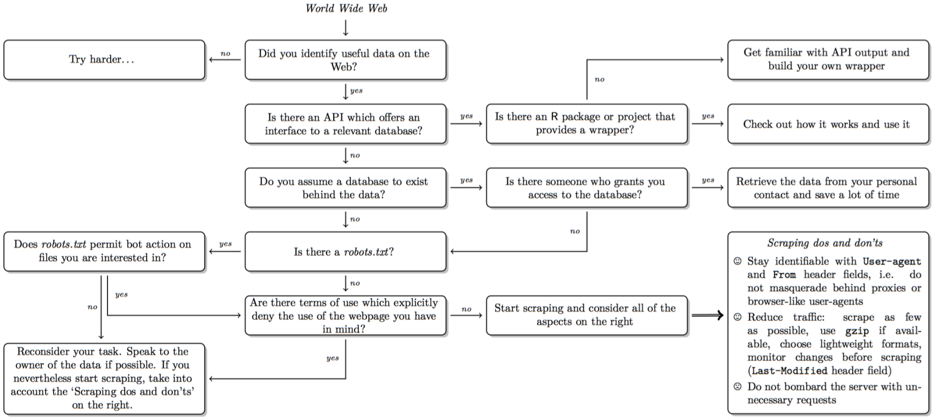
Advantages:

- ‘Pure’ data collection: avoid malformed HTML, no legal issues, clear data structures, more trust in data collection...
- Standardized data access procedures: transparency, replicability
- Robustness: benefits from ‘wisdom of the crowds’

Disadvantages

- They’re not too common (yet!)
- Dependency on API providers
- Lack of natural connection to R

Decisions, decisions...



World Wide Web

Did you identify useful data on the Web?

Try harder...

Is there an API which offers an interface to a relevant database?

Is there an R package or project that provides a wrapper?

Get familiar with API output and build your own wrapper

Check out how it works and use it

Do you assume a database to exist behind the data?

Is there someone who grants you access to the database?

Retrieve the data from your personal contact and save a lot of time

Is there a *robots.txt*?

Does *robots.txt* permit bot action on files you are interested in?

Are there terms of use which explicitly deny the use of the webpage you have in mind?

Start scraping and consider all of the aspects on the right

Reconsider your task. Speak to the owner of the data if possible. If you nevertheless start scraping, take into account the ‘Scraping dos and don’ts’ on the right.

Scraping dos and don’ts

- ☺ Stay identifiable with `User-agent` and `From` header fields, i.e. do not masquerade behind proxies or browser-like user-agents
- ☺ Reduce traffic: scrape as few as possible, use `gzip` if available, choose lightweight formats, monitor changes before scraping (`Last-Modified` header field)
- ☹ Do not bombard the server with unnecessary requests

CANT SCRAPE WEBSITE
WEBSITE USES AJAX

Advanced scraping

Selenium:

- General idea: browser control to scrape dynamically rendered web pages
- Originally developed for web testing purposes
- R will launch a browser session and all communication will be routed through that browser session.
- `phantomJS`: headless browser (will not display website)
- Capabilities: complete forms, write text, click on buttons or area of website, navigate to new URL...

Scraping newspaper websites

RSS feeds

- Really Simple Syndication, originally developed as a way to regularly check for new content on sites
- Includes list of entries (with some more information) and when they were updated
- Written in XML format (eXtensible Markup Language)
- Example: The Guardian RSS feed