


ENVIRONMENTAL DATA ANALYTICS: M10 – DATA SCRAPING

ENV 872

Nicholas School of the Environment - Duke University

Agenda

- Questions on M9 (Spatial Analysis)
- Projects!
- Next week's section: Python for R users
- M10 Data scraping…

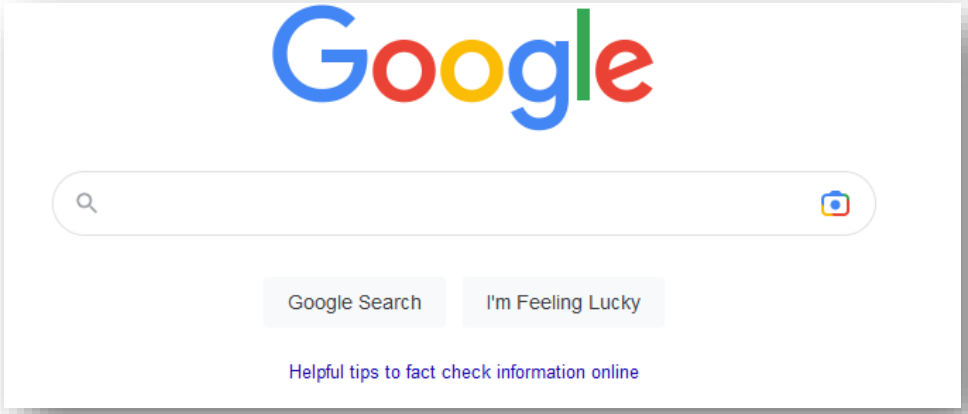
Scraping data…

- **rvest** package:
  - read_html() → Reads a web page into a parsable object
  - html_nodes() → Extracts elements with provided *tags*
  - html_text() → Gets the text associated with an element

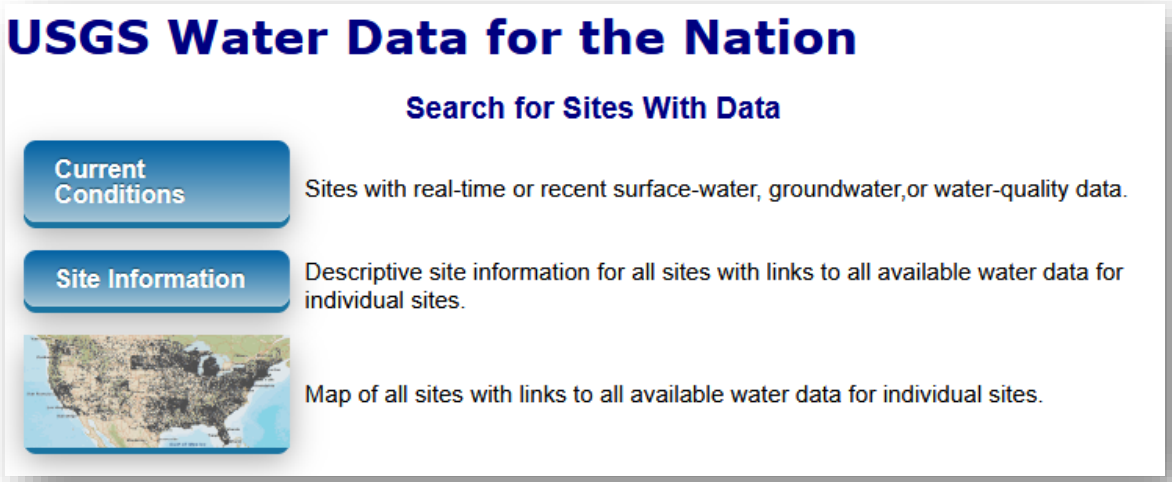
- Scraping is easier if you *understand how the web works*…
  - Structure of HTML
  - Nature of HTTP requests

- Sometimes scraping needs to be *automated*…

Understanding how the web works



https://www.google.com/search?q=Duke+University



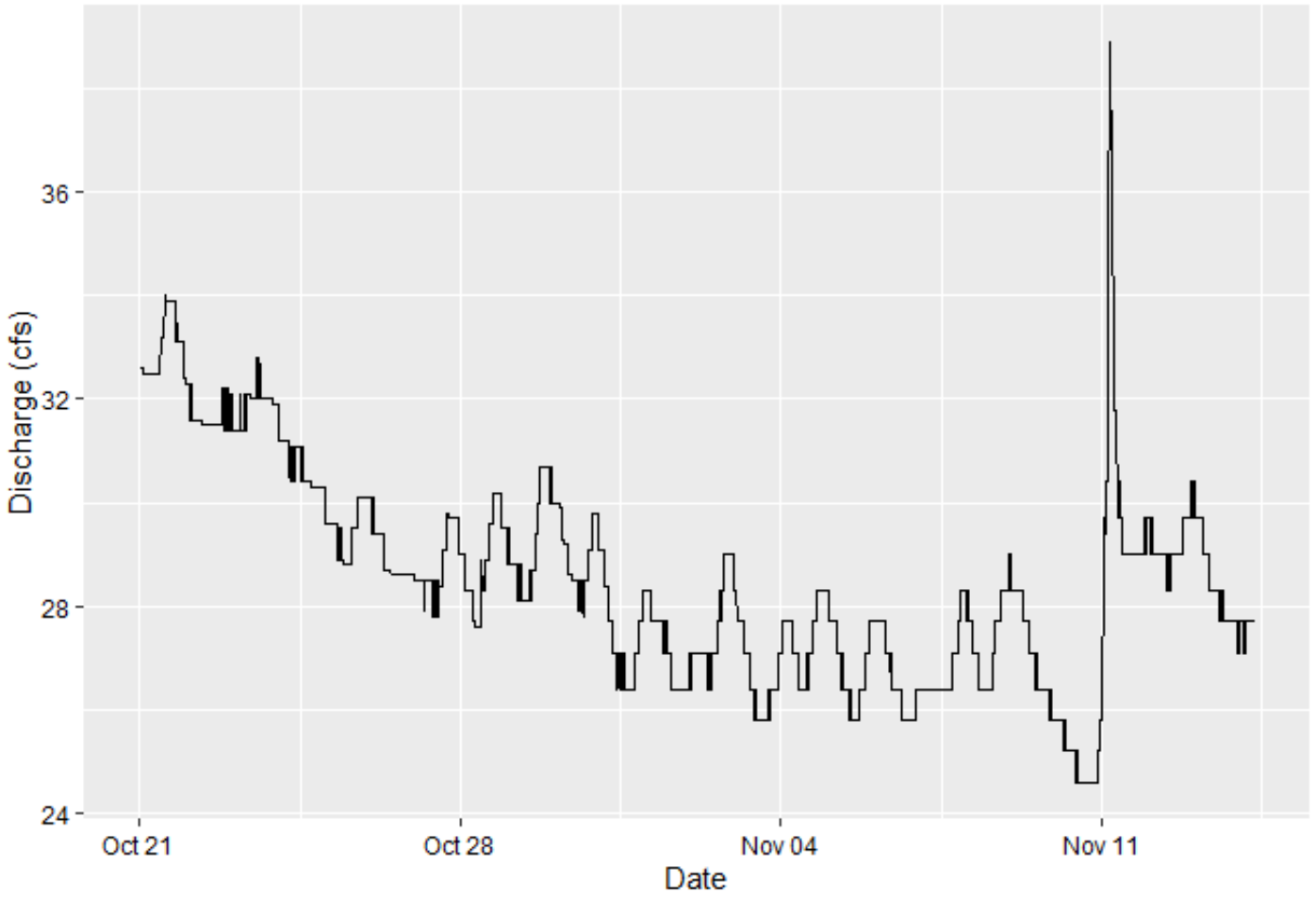
https://waterdata.usgs.gov/nwis/uv?search_site_no=02087500&period=7&format=rdb

Scraping data

1. Find tags with Selector Gadget
2. Scrape data into R:
   - read_html() → html_nodes() → html_text()
3. Construct a dataframe from scraped data…
4. Analyze data…

Exercise: Pull gage data from NWIS

Last 24 days from Gage # 02085070



Automating the scraping process

1. Analyze the URL and identify tags

2. Create a function to scrape data
   - Make variables for building URL, tags

3. Call function via `lapply` or *Purrr's* `map` function

Alternatives…

- APIs & Packages
  - Census (via “tidycensus” package)
  - USGS (via “dataRetrieval” package)

Solutions

Scrape & Plot NWIS Data

```
```{r EXERCISE - pull.and.plot.other.discharge.data}
#Set the URL
theURL <-
  'https://waterdata.usgs.gov/nwis/uv?search_site_no=02085070&period=24&format=rdb'

#Get the data, which starts on line 30
gage_data <- read.table(
  theURL,
  skip = 29,
  header=TRUE,
  sep='\t',
  stringsAsFactors = T)

#Update the column headers
colnames(gage_data) = c(
  "agency_cd", "site_no", "datetime", "tz_cd",
  "discharge_cfs", "89192_00060_cd", "gage_ht_ft", "89193_00065_cd")

#Tidy the data
gage_data <- gage_data %>%
  select(datetime, discharge_cfs, gage_ht_ft) %>%
  mutate(datetime = ymd_hm(datetime))

#Plot
ggplot(gage_data,aes(x=datetime,y=discharge_cfs)) +
  geom_line() +
  labs(x = "Date",y = "Discharge (cfs)")
```
```

Scrape Data from EIA site

```
```{r EXERCISE - scrape.data.manually eval=FALSE}
#1 Link to the web site using read_html
the_website <- read_html('https://www.eia.gov/electricity/state/')

#2&3 Locate elements and read their text attributes into variables
the_states <- the_website %>% html_nodes('td:nth-child(1)') %>% html_text()
the_price <- the_website %>% html_nodes('td:nth-child(2)') %>% html_text()
the_capacity <- the_website %>% html_nodes('td:nth-child(3)') %>% html_text()
net_generation <- the_website %>% html_nodes('td:nth-child(4)') %>% html_text()
total_retail <- the_website %>% html_nodes('td:nth-child(5)') %>% html_text()

#3 Construct a dataframe from the values
energy_data <- data.frame(
  "State" = the_states,
  "Price" = as.numeric(the_price),
  "Capacity" = as.numeric(gsub(',','',the_capacity)),
  "Net Generation" = as.numeric(gsub(',','',the_capacity)),
  "Total Retail" = as.numeric(gsub(',','',total_retail))
) %>%
  filter(State != 'U.S. Total')
```
```

Scrape as a function

```
```{r create.scrape.function}
scrape.it <- function(the_year){
  #Get the proper url
  the_url <- ifelse(
    the_year=='2024',
    'https://www.eia.gov/electricity/state/',
    paste0('https://www.eia.gov/electricity/state/archive/',the_year,'/')
    )

  #Fetch the website
  the_website <- read_html(the_url)

  #Scrape the data
  the_states <- the_website %>% html_nodes('td:nth-child(1)') %>% html_text()
  the_price <- the_website %>% html_nodes('td:nth-child(2)') %>% html_text()
  the_capacity <- the_website %>% html_nodes('td:nth-child(3)') %>% html_text()
  net_generation <- the_website %>% html_nodes('td:nth-child(4)') %>% html_text()
  total_retail <- the_website %>% html_nodes('td:nth-child(5)') %>% html_text()

  #Convert to dataframe
  energy_data <- data.frame(
    "State" = the_states,
    "Price" = as.numeric(the_price),
    "Capacity" = as.numeric(gsub(',','',the_capacity)),
    "Net Generation" = as.numeric(gsub(',','',the_capacity)),
    "Total Retail" = as.numeric(gsub(',','',total_retail))
  ) %>%
    filter(State != 'U.S. Total') %>%
    mutate(Year = the_year)

  #Return the dataframe
  return(energy_data)
}
```
```

Applying scrape function across years

```r
# Use the above function to scrape data for 2017
energy_2017 <- scrape.it(2017)

# Map the function to scrape data from 2017 to 2024
energy_17_24 <- seq(2017,2024) %>% map(scrape.it) %>% bind_rows()

# Example plot
energy_17_24 %>%
  filter(State %in% c('North Carolina','South Carolina', 'Virginia')) %>%
  ggplot(aes(x=Year, y=Price, color=State)) +
  #geom_line() +
  geom_smooth(method='loess',se=FALSE) +
  scale_x_date(date_breaks = '1 year', date_labels = '%Y')
```