

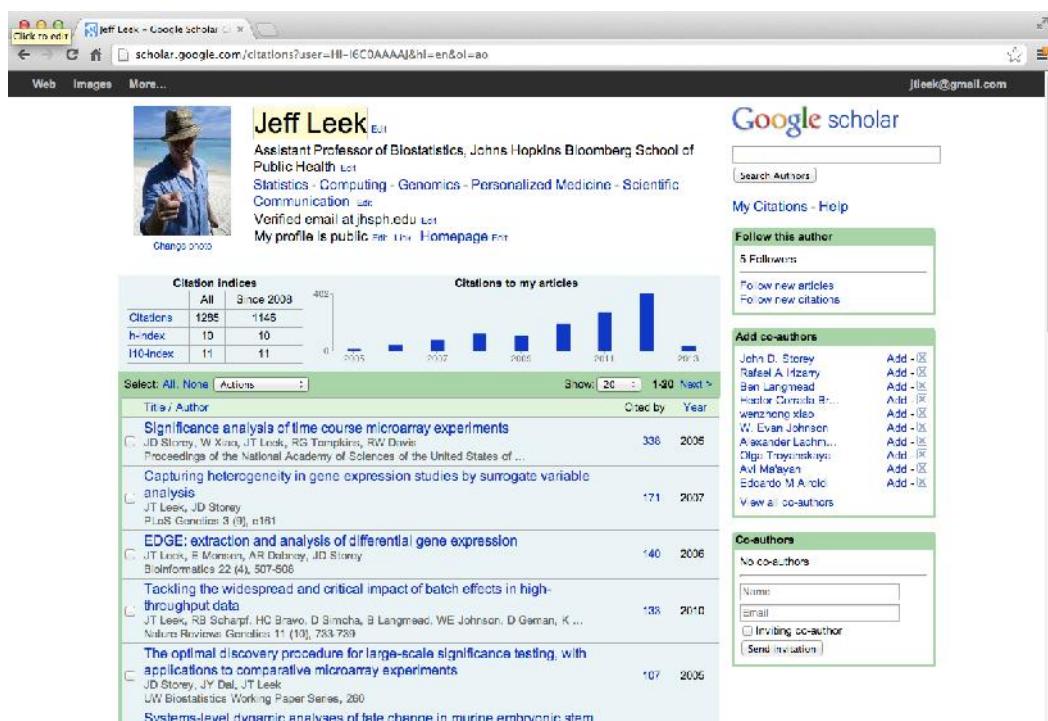
Webscraping

Webscraping: Programatically extracting data from the HTML code of websites.

- It can be a great way to get data [How Netflix reverse engineered Hollywood](#)
- Many websites have information you may want to programatically read
- In some cases this is against the terms of service for the website
- Attempting to read too many pages too quickly can get your IP address blocked

http://en.wikipedia.org/wiki/Web_scraping

Example: Google scholar



The screenshot shows the Google Scholar profile for Jeff Leek. At the top, there's a photo of Jeff Leek wearing a hat and a blue shirt, with a caption below it: "Jeff Leek Assistant Professor of Biostatistics, Johns Hopkins Bloomberg School of Public Health". Below the photo are his citation statistics: "Citation Indices All Since 2003 402", "Citations 1285 1145", "h-index 10 10", and "H-Index 11 11". To the right, there's a bar chart titled "Citations to my articles" showing citation counts by year from 2005 to 2012. A green sidebar on the left lists his publications, such as "Significance analysis of time course microarray experiments" (336 citations, 2005) and "EDGE: extraction and analysis of differential gene expression" (140 citations, 2006). On the right, there are sections for "Follow this author" (5 followers), "Add co-authors" (listing several names like John D. Storey, Rafael A. Irizarry, etc.), and "Co-authors" (listing none). The URL in the address bar is <http://scholar.google.com/citations?user=HI-I6C0AAAAJ&hl=en>.

<http://scholar.google.com/citations?user=HI-I6C0AAAAJ&hl=en>

Getting data off webpages - readLines()

```
con = url("http://scholar.google.com/citations?user=HI-I6C0AAAAJ&hl=en")
htmlCode = readLines(con)
close(con)
htmlCode
```

```
[1] "<!DOCTYPE html><html><head><title>Jeff Leek - Google Scholar Citations</title><meta name=\"rob
```

Parsing with XML

```
library(XML)
url <- "http://scholar.google.com/citations?user=HI-I6C0AAAAJ&hl=en"
html <- htmlTreeParse(url, useInternalNodes=T)

xpathSApply(html, "//title", xmlValue)
```

```
[1] "Jeff Leek - Google Scholar Citations"
```

```
xpathSApply(html, "//td[@id='col-citedby']", xmlValue)
```

```
[1] "Cited by" "397"      "259"       "237"       "172"       "138"       "125"       "122"
[9] "109"        "101"       "34"        "26"        "26"        "24"        "19"        "13"
[17] "12"         "10"        "10"        "7"         "6"
```

GET from the httr package

```
library(httr); html2 = GET(url)
content2 = content(html2,as="text")
parsedHtml = htmlParse(content2,asText=TRUE)
xpathSApply(parsedHtml, "//title", xmlValue)
```

```
[1] "Jeff Leek - Google Scholar Citations"
```

Accessing websites with passwords

```
pg1 = GET( "http://httpbin.org/basic-auth/user/passwd" )  
pg1
```

```
Response [http://httpbin.org/basic-auth/user/passwd]  
Status: 401  
Content-type:
```

<http://cran.r-project.org/web/packages/httr/httr.pdf>

Accessing websites with passwords

```
pg2 = GET("http://httpbin.org/basic-auth/user/passwd",
  authenticate("user", "passwd"))
pg2
```

```
Response [http://httpbin.org/basic-auth/user/passwd]
Status: 200
Content-type: application/json
{
  "authenticated": true,
  "user": "user"
}
```

```
names(pg2)
```

```
[1] "url"          "handle"        "status_code"   "headers"       "cookies"      "content"
[7] "times"        "config"
```

<http://cran.r-project.org/web/packages/httr/httr.pdf>

Using handles

```
google = handle("http://google.com")
pg1 = GET(handle=google, path="/")
pg2 = GET(handle=google, path="search")
```

<http://cran.r-project.org/web/packages/httr/httr.pdf>

Notes and further resources

- R Bloggers has a number of examples of web scraping <http://www.r-bloggers.com/?s=Web+Scraping>
- The httr help file has useful examples <http://cran.r-project.org/web/packages/httr/httr.pdf>
- See later lectures on APIs