



Getting Data (Part 1)

Jeffrey Leek, Assistant Professor of Biostatistics
Johns Hopkins Bloomberg School of Public Health

Get/set your working directory

Roger's lectures [windows](#), [mac](#) Andrew Jaffe's [lecture notes](#)

```
> getwd()  
[1] "/Users/jtleek/Dropbox/Jeff/teaching/2013/coursera/week2/004gettingData1"  
> setwd("/Users/jtleek/Dropbox/Jeff/teaching/2013/coursera/week2/004gettingData1/data")  
> getwd()  
[1] "/Users/jtleek/Dropbox/Jeff/teaching/2013/coursera/week2/004gettingData1/data"
```

Important difference with Windows:

```
setwd("C:\\Users\\Andrew\\Downloads")
```

Get/set your working directory (relative paths)

```
getwd()
```

```
[1] "/Users/jtleek/Dropbox/Jeff/teaching/2013/coursera/week2/004gettingData1"
```

```
setwd("./data")  
getwd()
```

```
[1] "/Users/jtleek/Dropbox/Jeff/teaching/2013/coursera/week2/004gettingData1/data"
```

```
setwd("../")  
getwd()
```

```
[1] "/Users/jtleek/Dropbox/Jeff/teaching/2013/coursera/week2/004gettingData1"
```

Get/set your working directory (absolute paths)

```
> getwd()  
[1] "/Users/jtleek/Dropbox/Jeff/teaching/2013/coursera/week2/004gettingData1"  
> setwd("/Users/jtleek/Dropbox/Jeff/teaching/2013/coursera/week2/004gettingData1/data")  
> getwd()  
[1] "/Users/jtleek/Dropbox/Jeff/teaching/2013/coursera/week2/004gettingData1/data"
```

Types of files data may come from

- Tab-delimited text
- Comma-separated text
- Excel file
- JSON File
- HTML/XML file
- Database

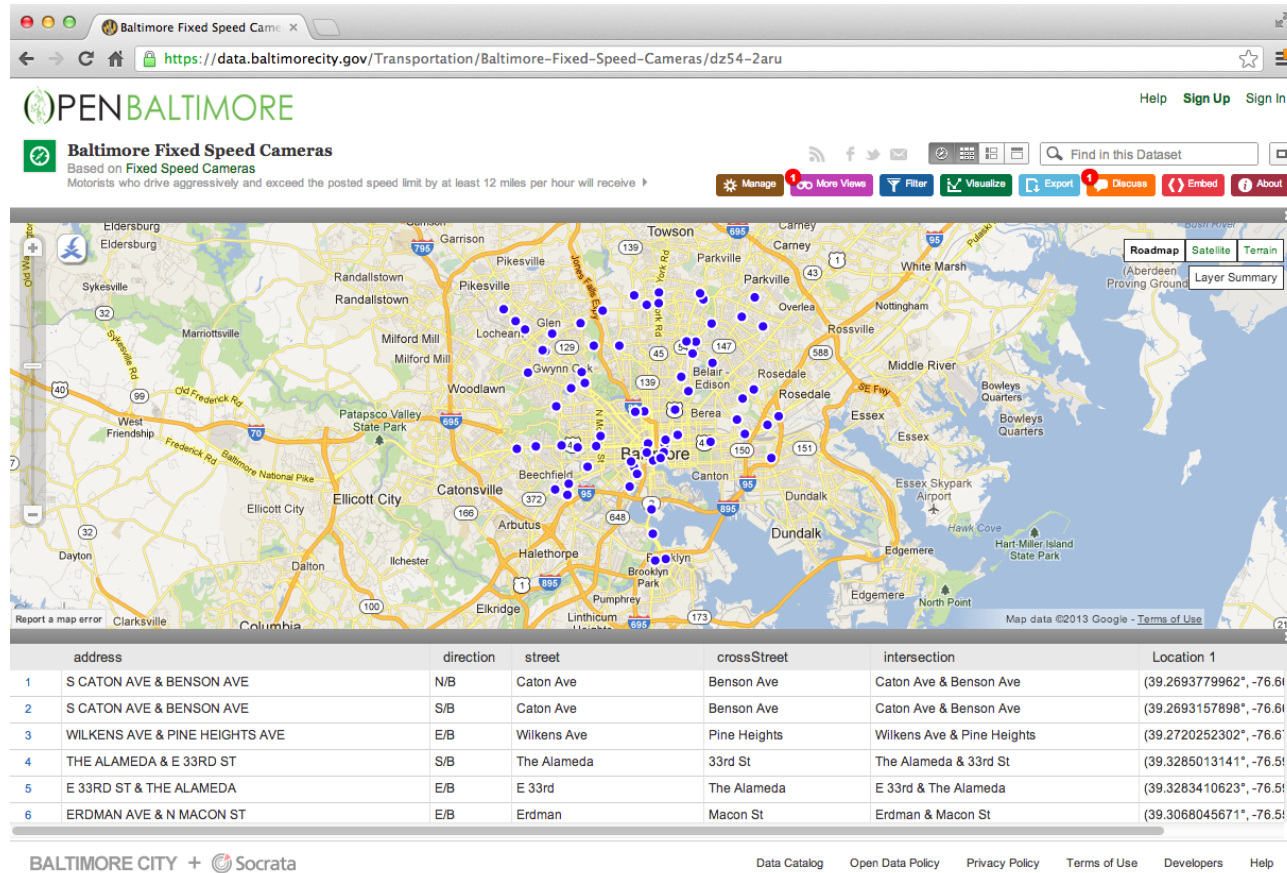
Where you can get data

- From a colleague
- From the web
- From an application programming interface
- By scraping a web page

Getting data from the internet - `download.file()`

- Downloads a file from the internet
- Even if you could do this by hand, helps with reproducibility
- Important parameters are *url*, *destfile*, *method*
- Useful for downloading tab-delimited, csv, etc.

Example - Baltimore camera data



<https://data.baltimorecity.gov/Transportation/Baltimore-Fixed-Speed-Cameras/dz54-2aru>

Example - Baltimore camera data,csv

Baltimore Fixed Speed Cameras
Based on Fixed Speed Cameras
Motorists who drive aggressively and exceed the posted speed limit by at least 12 miles per hour will receive

Find in this Dataset

Manage More Views Filter Visualize Export Discuss Embed About

Export

API

Download

Download a copy of this dataset in a static format

Download As

- Open Link in New Tab
- Open Link in New Window
- Open Link in Incognito Window
- Save Link As...
- Copy Link Address**
- Copy
- Search Google for 'CSV'
- Inspect Element
- Look Up in Dictionary
- Speech
- Search With Google
- Add to iTunes as a Spoken Track

	address	direction	street	crossStreet	intersection
1	S CATON AVE & BENSON AVE	N/B	Caton Ave	Benson Ave	Caton Ave & Be
2	S CATON AVE & BENSON AVE	S/B	Caton Ave	Benson Ave	Caton Ave & Be
3	WILKENS AVE & PINE HEIGHTS AVE	E/B	Wilkins Ave	Pine Heights	Wilkins Ave & f
4	THE ALAMEDA & E 33RD ST	S/B	The Alameda	33rd St	The Alameda &
5	E 33RD ST & THE ALAMEDA	E/B	E 33rd	The Alameda	E 33rd & The Al
6	ERDMAN AVE & N MACON ST	E/B	Erdman	Macon St	Erdman & Maco

BALTIMORE CITY + Socrata

Data Catalog Open Data Policy Privacy Policy Terms of Use Developers Help

<https://data.baltimorecity.gov/Transportation/Baltimore-Fixed-Speed-Cameras/dz54-2aru>

Download a file from the web

```
fileUrl <- "https://data.baltimorecity.gov/api/views/dz54-2aru/rows.csv?accessType=DOWNLOAD"  
download.file(fileUrl, destfile = "./data/cameras.csv", method = "curl")  
list.files("./data")
```

```
[1] "camera.json"      "camera.xlsx"      "cameras.csv"  
[4] "cameras.rda"      "camerasModified.csv"
```

```
dateDownloaded <- date()  
dateDownloaded
```

```
[1] "Wed Aug 28 13:27:31 2013"
```

Some notes about `download.file()`

- If the url starts with *http* you can use `download.file()`
- If the url starts with *https* on Windows you may be ok
- If the url starts with *https* on Mac you may need to set *method="curl"*
- If the file is big, this might take a while
- Be sure to record when you downloaded.

Loading data you have saved - `read.table()`

- This is the main function for reading data into R
- Flexible and robust but requires more parameters
- Reads the data into RAM - big data can cause problems
- Important parameters *file*, *header*, *sep*, *row.names*, *nrows*
- Related: *read.csv()*, *read.csv2()*

Example: Baltimore camera data

```
getwd()
```

```
[1] "/Users/jtleek/Dropbox/Jeff/teaching/2013/coursera/week2/004gettingData1"
```

```
cameraData <- read.table("./data/cameras.csv")
```

```
Error: line 1 did not have 13 elements
```

```
head(cameraData)
```

```
Error: error in evaluating the argument 'x' in selecting a method for  
function 'head': Error: object 'cameraData' not found
```

Example: Baltimore camera data

```
getwd()
```

```
[1] "/Users/jtleek/Dropbox/Jeff/teaching/2013/coursera/week2/004gettingData1"
```

```
cameraData <- read.table("./data/cameras.csv", sep="," , header=TRUE)
```

Example: Baltimore camera data

```
head(cameraData)
```

	address	direction	street	crossStreet
1	S CATON AVE & BENSON AVE	N/B	Caton Ave	Benson Ave
2	S CATON AVE & BENSON AVE	S/B	Caton Ave	Benson Ave
3	WILKENS AVE & PINE HEIGHTS AVE	E/B	Wilkins Ave	Pine Heights
4	THE ALAMEDA & E 33RD ST	S/B	The Alameda	33rd St
5	E 33RD ST & THE ALAMEDA	E/B	E 33rd	The Alameda
6				
1	Caton Ave & Benson Ave (39.2693779962, -76.6688185297)			
2	Caton Ave & Benson Ave (39.2693157898, -76.6689698176)			
3	Wilkins Ave & Pine Heights (39.2720252302, -76.676960806)			
4	The Alameda & 33rd St (39.3285013141, -76.5953545714)			
5	E 33rd & The Alameda (39.3283410623, -76.5953594625)			
6	Erdman & Macon St (39.3068045671, -76.5593167803)			

Example: Baltimore camera data

`read.csv` sets `sep=","` and `header=TRUE`

```
cameraData <- read.csv("./data/cameras.csv")
head(cameraData)
```

	address	direction	street	crossStreet
1	S CATON AVE & BENSON AVE	N/B	Caton Ave	Benson Ave
2	S CATON AVE & BENSON AVE	S/B	Caton Ave	Benson Ave
3	WILKENS AVE & PINE HEIGHTS AVE	E/B	Wilkins Ave	Pine Heights
4	THE ALAMEDA & E 33RD ST	S/B	The Alameda	33rd St
5	E 33RD ST & THE ALAMEDA	E/B	E 33rd	The Alameda
6				
1	Caton Ave & Benson Ave (39.2693779962, -76.6688185297)			
2	Caton Ave & Benson Ave (39.2693157898, -76.6689698176)			
3	Wilkins Ave & Pine Heights (39.2720252302, -76.676960806)			
4	The Alameda & 33rd St (39.3285013141, -76.5953545714)			
5	E 33rd & The Alameda (39.3283410623, -76.5953594625)			
6	Erdman & Macon St (39.3068045671, -76.5593167803)			

read.xlsx(), read.xlsx2() {xlsx package}

- Reads .xlsx files, but slow
- Important parameters *file, sheetIndex, sheetIndex, rowIndex, colIndex, header*
- read.xlsx2() relies more on low level Java functions so may be a bit faster

read.xlsx() - Baltimore camera data

You may need to run `install.packages("xlsx")` if the xlsx package is not already installed

```
library(xlsx)
fileUrl <- "https://data.baltimorecity.gov/api/views/dz54-2aru/rows.xlsx?accessType=DOWNLOAD"
download.file(fileUrl, destfile = "./data/camera.xlsx", method = "curl")
cameraData <- read.xlsx2("./data/camera.xlsx", sheetIndex = 1)
```

read.xlsx() - Baltimore camera data

```
head(cameraData)
```

	address	direction	street	crossStreet
1	S CATON AVE & BENSON AVE	N/B	Caton Ave	Benson Ave
2	S CATON AVE & BENSON AVE	S/B	Caton Ave	Benson Ave
3	WILKENS AVE & PINE HEIGHTS AVE	E/B	Wilkins Ave	Pine Heights
4	THE ALAMEDA & E 33RD ST	S/B	The Alameda	33rd St
5	E 33RD ST & THE ALAMEDA	E/B	E 33rd	The Alameda
6				
1	Caton Ave & Benson Ave (39.2693779962, -76.6688185297)			
2	Caton Ave & Benson Ave (39.2693157898, -76.6689698176)			
3	Wilkins Ave & Pine Heights (39.2720252302, -76.676960806)			
4	The Alameda & 33rd St (39.3285013141, -76.5953545714)			
5	E 33rd & The Alameda (39.3283410623, -76.5953594625)			
6	Erdman & Macon St (39.3068045671, -76.5593167803)			

Picking a file - less reproducible, but useful

```
cameraData <- read.csv(file.choose())
```

