Introduction to Data Management and Processing

Kylie Ariel Bemis

9/7/2017

Course information

Schedule: Tuesdays/Fridays 3:25 pm - 5:05 pm

Location: International Village 022

Dates: Sep 06, 2017 - Dec 16, 2017

Website: https://kuwisdelu.github.io/ds5110-fall17.html

Administration: https://piazza.com/northeastern/fall2017/ds5110

Textbooks

- *R for Data Science* by Wickham and Grolemund,
- *Advanced R* by Wickham
- Both are available for free online
- Deadtree copies can be obtained via Amazon
- Supplementary (free, online) readings may be posted later in the course

Preview of course topics

- Data visualization
- Data transformation
- Importing and tidying data
- Modeling
- OOP in R
- Data representation
- Data science deliverables
- Reproducibility
- Performance, profiling, and debugging
- Big data
- Interactivity
- Communication

Office hours

Instructor: Kylie Bemis | k.bemis@northeastern.edu

- *Office Hours:* Tuesdays 2:00 pm - 3:00 pm in WVH 310G

TA: Sara Taheri | mohammadtaheri.s@husky.neu.edu

- *Office Hours:* Thursdays 5:00 pm - 6:00 pm in WVH 310

TA: Jesse (Jianchao) Yang | yang.jianc@husky.neu.edu

- *Office Hours:* Saturdays 10:00 am - 11:00 am in Snell Library (2nd floor)

Piazza

- All questions, homework postings, and course announcements are handled via Piazza.
- Enroll today at:
  - https://piazza.com/northeastern/fall2017/ds5110
- **Please do not email the instructor or TAs with questions – use Piazza instead**

Academic integrity

- Be familiar with the university's academic integrity policy on cheating and plagiarism.
- All submitted solutions on homeworks and exams must be your own work.
- Discussion of topics presented in class and general strategies for solving homework problems is allowed and encouraged.
- Sharing of fully-worked or near-complete solutions is not tolerated however, and will be considered cheating.
- If your work exhibits a very high degree of similarity to another student's, it is considered plagiarism.
- **Students found to be cheating or plagiarising work will receive an F and be reported to the university.**
  - You are here to learn, not manufacture good grades.
  - If you need help, ask the instructor or a TA.
  - Remember you can ask questions anonymously and privately message instructors on Piazza

Homework

- There will be 3 homework assignments to be completed individually.
- They are due before class on the date indicated on the class website.
- Homeworks are to be submitted via email (see instructions on each homework for details)
- Each homework must be submitted as an R Markdown source file and PDF output (this will be discussed more soon)
- Homeworks will be posted on Piazza ~2 weeks before they are due (sometimes earlier)
- Requests for re-grades must be made in writing no later than 1 week after receiving the grade
  - The new grade may be lower than the original

Exams

- There will be 2 exams
- 1 midterm and 1 final
- Both will be completed in-class
- Both will be pen(cil)-and-paper

Final Project

- There will be 1 final project to be completed in groups of 2-3
- Project groups will be due at the midterm
- Use a dataset of your choice
- Presentations in-class (last week) and written report
- Proposal due $\sim$1 month before final presentation/report

Data Visualization "Mini-Poster"

- A data visualization "mini-poster" will also be assigned, due before HW2, to be graded as part of HW2.
- Details will be posted with HW2
- Use a dataset of your choice (start looking!)
- Will be compiled together for whole class to view
- This is your "pitch"/"resume" for others in the class, to help you find people with whom you would like to collaborate for the final project
- *You do not need to use the same dataset for the final project*

Grading

Grades are based out of 1000 points, distributed as follows:

- Homework: 3 x 100 pts
- Midterm Exam: 200 pts
- Final Exam: 200 pts
- Final Project: 300 pts
  - Project proposal: 50 pts
  - In-class presentation: 100 pts
  - Final written report: 150 pts

Final grades will be given on the following scale:

- A : 900 - 1000 pts
- B : 800 - 899 pts
- C : 700 - 799 pts
- F : $<=$ 699 pts

Half-letter-grades ('+' and '-') may be given as well.

Tools for this course

- Required:
  - Install R from CRAN
  - https://cloud.r-project.org
- Recommended:
  - Install RStudio
  - https://www.rstudio.com/download

Install packages

- Tidyverse

```
install.packages("tidyverse")
library(tidyverse)
```

- Additional packages

```
install.packages(c("nycflights13", "gapminder", "Lahman"))
```

- R Markdown (installed automatically with RStudio)

```
install.packages("rmarkdown")
```

Introduction to Data Science

Data Science pipeline

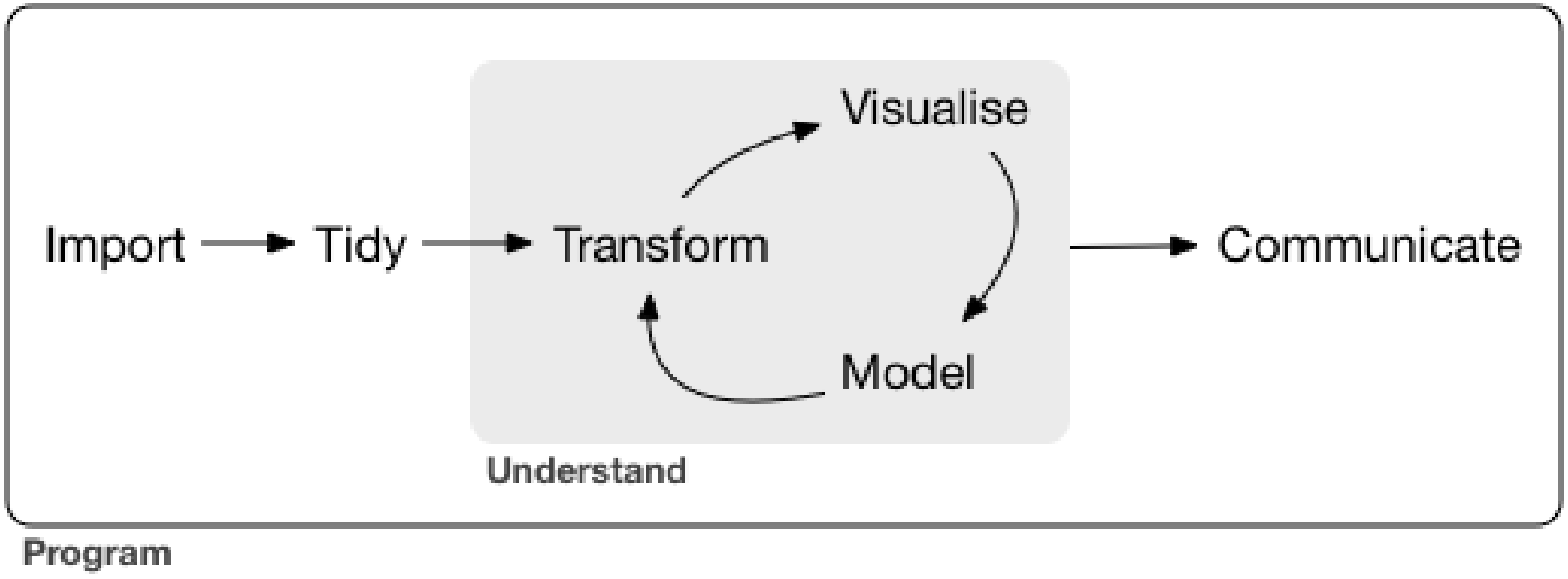


Figure 1: Wickham and Grolemund, *R for Data Science*

Reproducibility

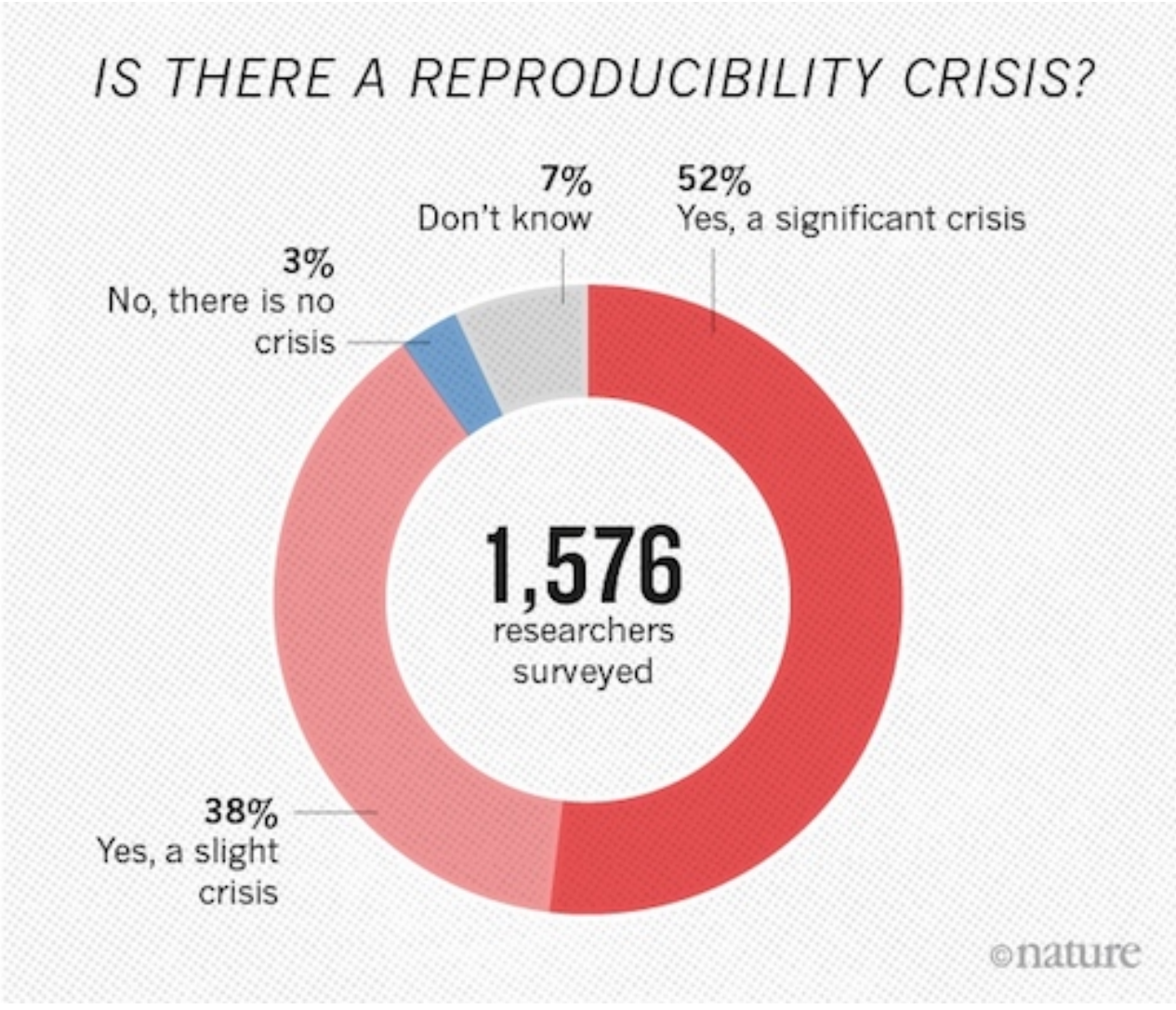


Figure 2: Baker, "1,500 scientists lift the lid on reproducibility," *Nature*

Failure to reproduce experiments

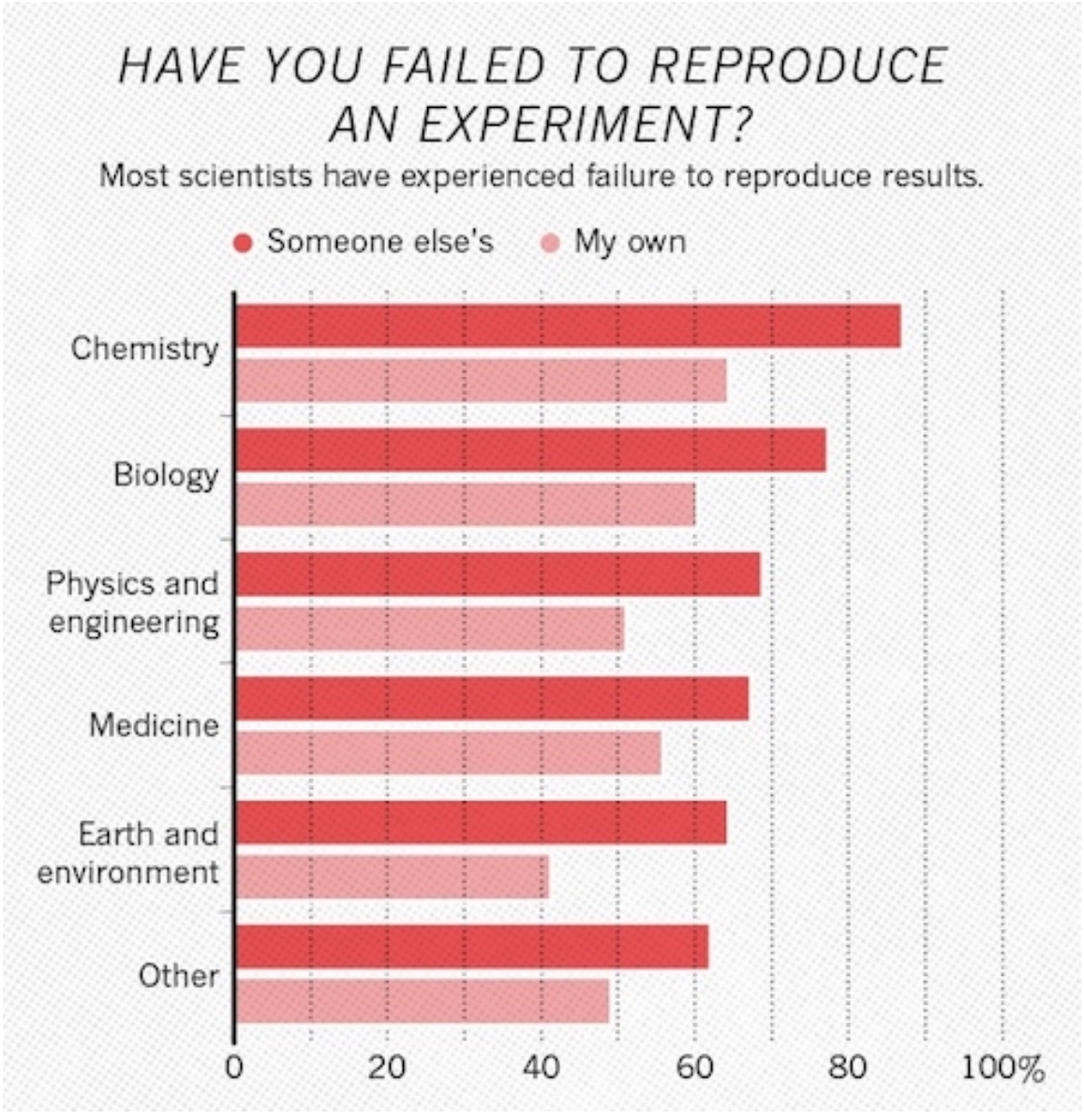


Figure 3: Baker, "1,500 scientists lift the lid on reproducibility," *Nature*

Gene name errors due to Excel

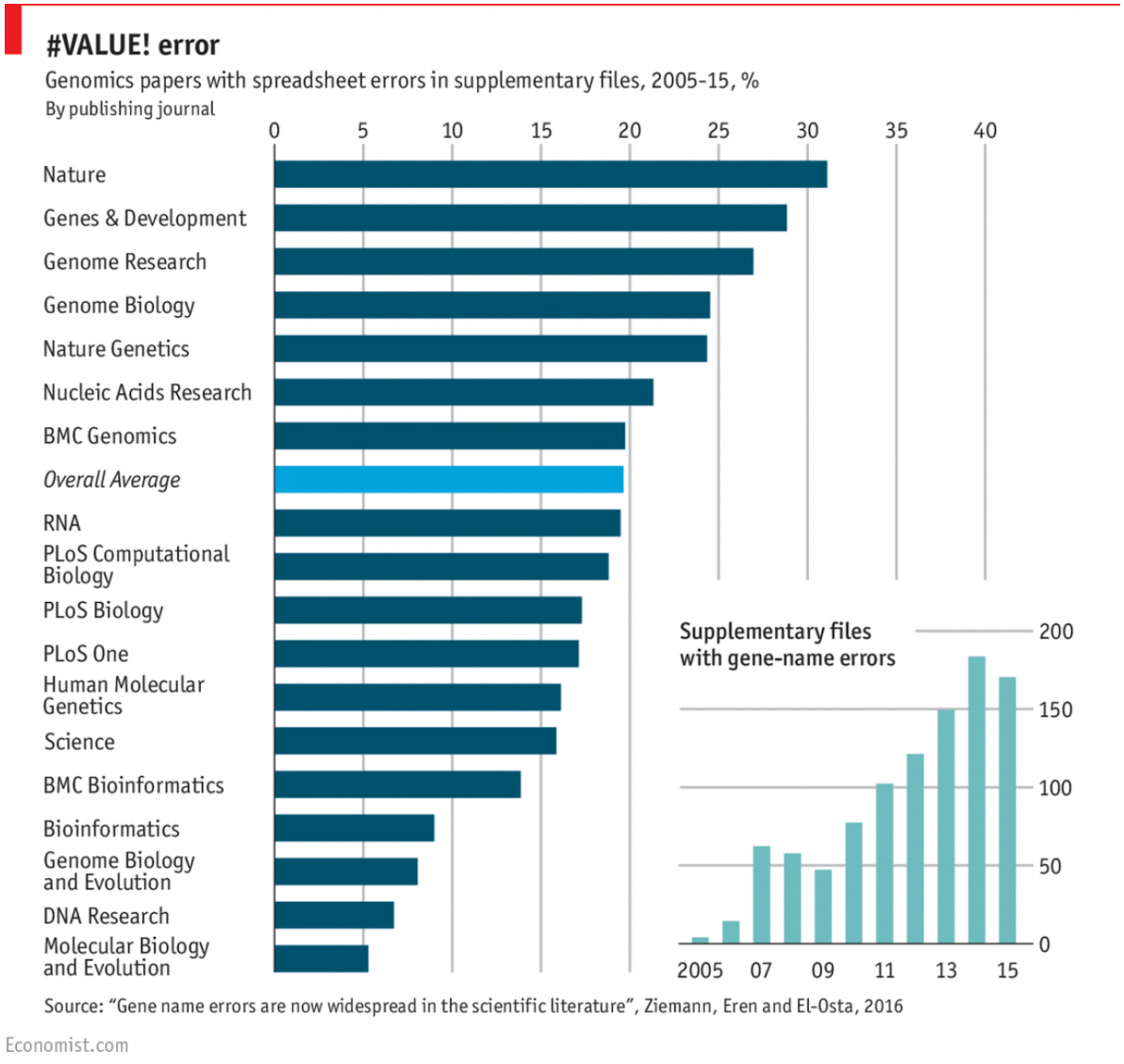


Figure 4: Zieman, Eren, and El-Osta, "Gene name errors are widespread in the scientific literature," *Genome Biology*

Why program?

- Often asked by experimentalists who wish to analyze data
- Writing code encourages reproducible research
- Code can be documented and analyses re-run
- Open-source encourages others to test and reproduce results
- Interactive analysis can be done in a programmatic environment
- Programming is an integral part of reproducibility

Why R?

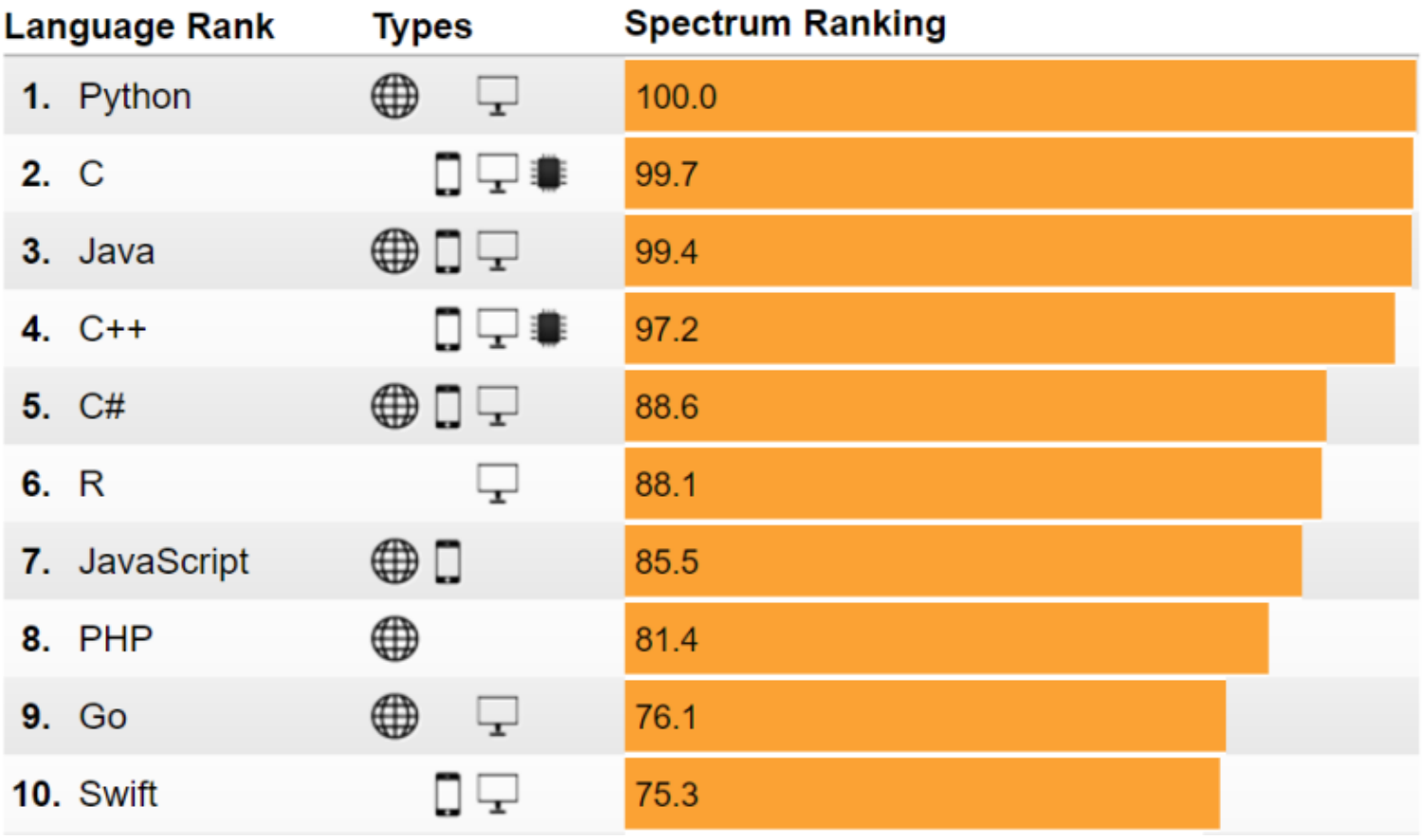


Figure 5: IEEE Spectrum 2017 Top Programming Languages

R is a language for data

- Designed specifically for data analysis
- Consistently one of the top languages for data science
- Interactive environment for visualization and statistical analysis
- Rich community of over 12,000 packages on CRAN
- Emphasizes reproducible research

R Markdown

- Authoring framework for data science
- Code for analysis embedded in report
- Documents are fully reproducible
- You will submit homeworks in R Markdown

Demo

Introduction to R

A brief history of R

- S created by John Chambers at Bell Laboratories in 1976
- "Turn ideas into software, quickly and faithfully"
- R created in 1993 as a free, open-source implementation of S
- Influenced by S, Scheme, Common Lisp, XLispStat

Basic data types (atomic)

There are six 'atomic' data types:

- **character**
- **double**
- **integer**
- **logical**
- raw
- complex

Most of the time, R will not distinguish between integer and double, as both are considered *numeric* vectors by R.

```
"a"
```

```
## [1] "a"
```

```
1
```

```
## [1] 1
```

```
1L
```

```
## [1] 1
```

```
TRUE
```

```
## [1] TRUE
```

- Everything is a vector in R
- "Scalars" are actually length-1 vectors
- Vectors of atomic types are created using **c()**
- Sequences of integers can be shortcut via **start:end**
- Assignment is done via <-

```r
c("a", "b", "c", "d")
```

```
## [1] "a" "b" "c" "d"
```

```r
c(1, 2.2, 3.33, 4.444)
```

```
## [1] 1.000 2.200 3.330 4.444
```

```r
c(1L, 2L, 3L, 4L)
```

```
## [1] 1 2 3 4
```

```r
c(TRUE, FALSE, TRUE, NA)
```

```
## [1]  TRUE FALSE  TRUE    NA
```

```r
x <- c(1, 2, 3, 4)
x
```

```
## [1] 1 2 3 4
```

```r
y <- 1:4
y
```

```
## [1] 1 2 3 4
```

Basic data types (non-atomic)

Atomic vector types are used to build more complex data types:

- matrix
- array
- list
- data.frame

This class will primarily discuss and make use of data.frames, which are the most common way of storing data in R.

For most types of data, systematic and tidy use of data.frames makes data analysis quick and easy.

```r
matrix(c(1, 2, 3, 4), nrow=2, ncol=2)
```

```
##      [,1] [,2]
## [1,]    1    3
## [2,]    2    4
```

```r
array(1:12, dim=c(2,3,2))
```

```
## , , 1
##
##      [,1] [,2] [,3]
## [1,]    1    3    5
## [2,]    2    4    6
##
## , , 2
##
##      [,1] [,2] [,3]
## [1,]    7    9   11
## [2,]    8   10   12
```

```r
list(1L, "a", 3.14)
```

```
## [[1]]
## [1] 1
##
## [[2]]
## [1] "a"
##
## [[3]]
## [1] 3.14
```

```r
data.frame(x=c(1, 2, 3), y=c("a", "b", "c"))
```

```
##   x y
## 1 1 a
## 2 2 b
## 3 3 c
```

The table below visualizes the relationship between these basic data types:

R data types	Same types	Different types
1d	vector	list
2d	matrix	data.frame
nd	array	

Each of these data types can be subsetted in multiple ways:

R subscripts	Simplifying	Preserving
vectors	x[[1]]	x[1:6]
matrices/data.frames	x[,1]	x[,1,drop=FALSE]
lists	x[[1]], x$name	x[1]

Note that indexing in R begins with 1, *not* 0.

Also note that data.frames are actually a special type of list, and can also be subset in the same ways that a list can.

```r
z <- list(1L, "a", 3.14)
z[1:2]
```

```
## [[1]]
## [1] 1
##
## [[2]]
## [1] "a"
```

```r
z[1]
```

```
## [[1]]
## [1] 1
```

```r
z[[1]]
```

```
## [1] 1
```

```r
df <- data.frame(x=c(1, 2, 3), y=c("a", "b", "c"))
df$x
```

```
## [1] 1 2 3
```

```r
df[,"y"]
```

```
## [1] a b c
## Levels: a b c
```

```r
df[1:2,]
```

```
##   x y
## 1 1 a
## 2 2 b
```

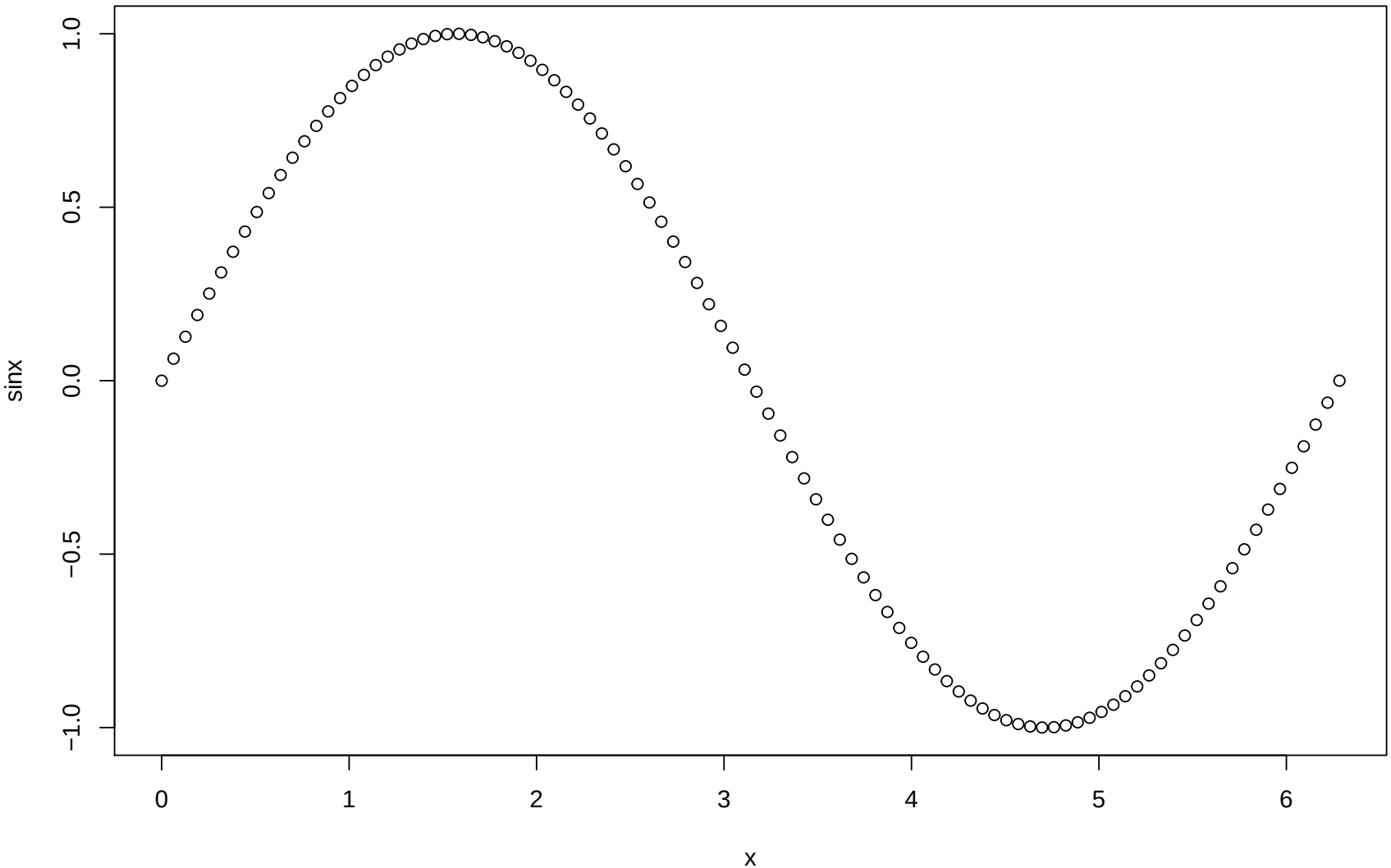
Documentation

- Access help page for any R function using **help(name)**
- Syntactic sugar is available as **?name**
- You may need to quote special names with backticks

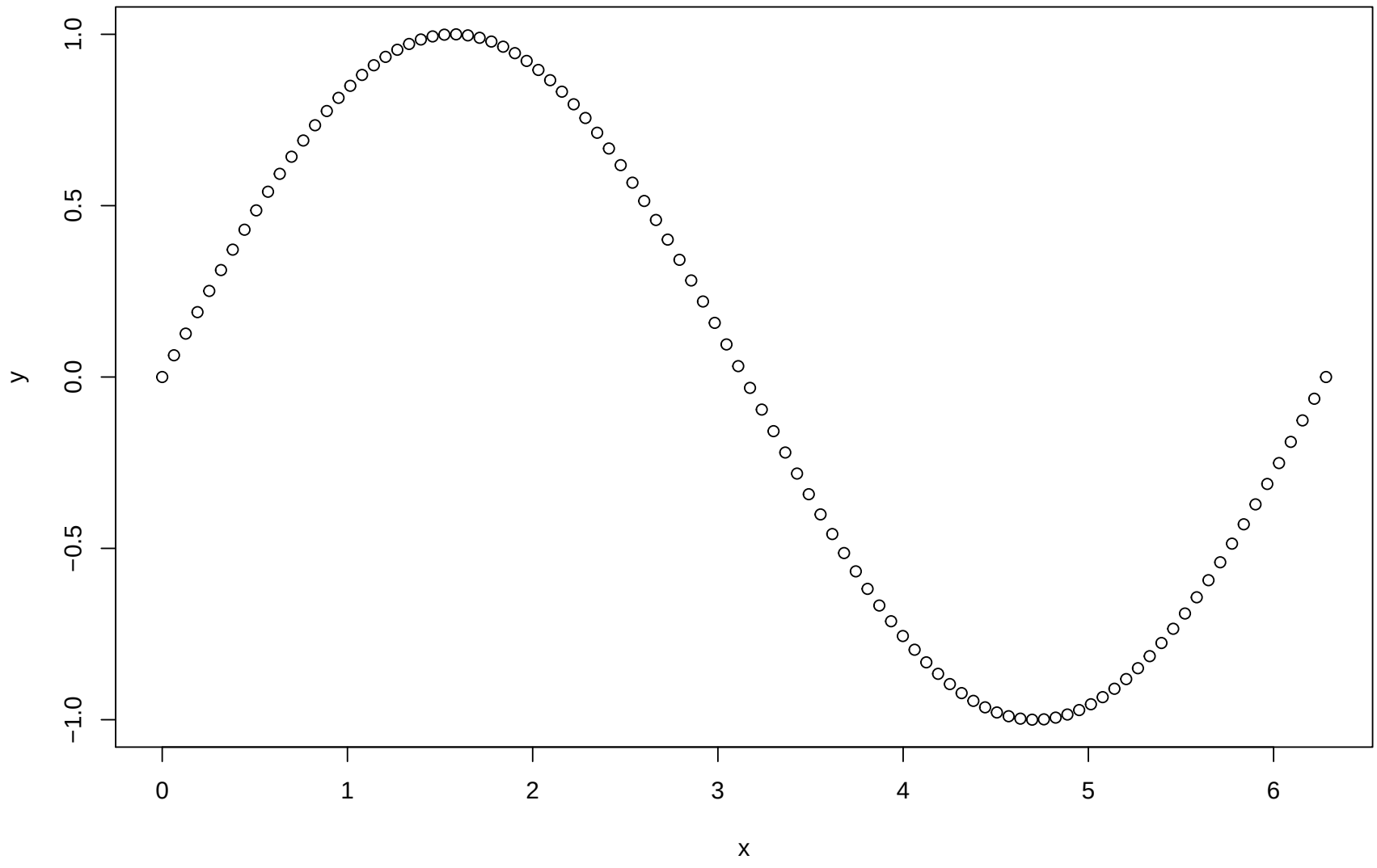
Plotting

- Base R plotting uses “brush on canvas” philosophy
- Plot elements are plotted individually, one on top of another
- Allows “formula” interface
- Good enough for basic plots
- Simple and flexible
- We will learn a more powerful alternative (ggplot2) next week

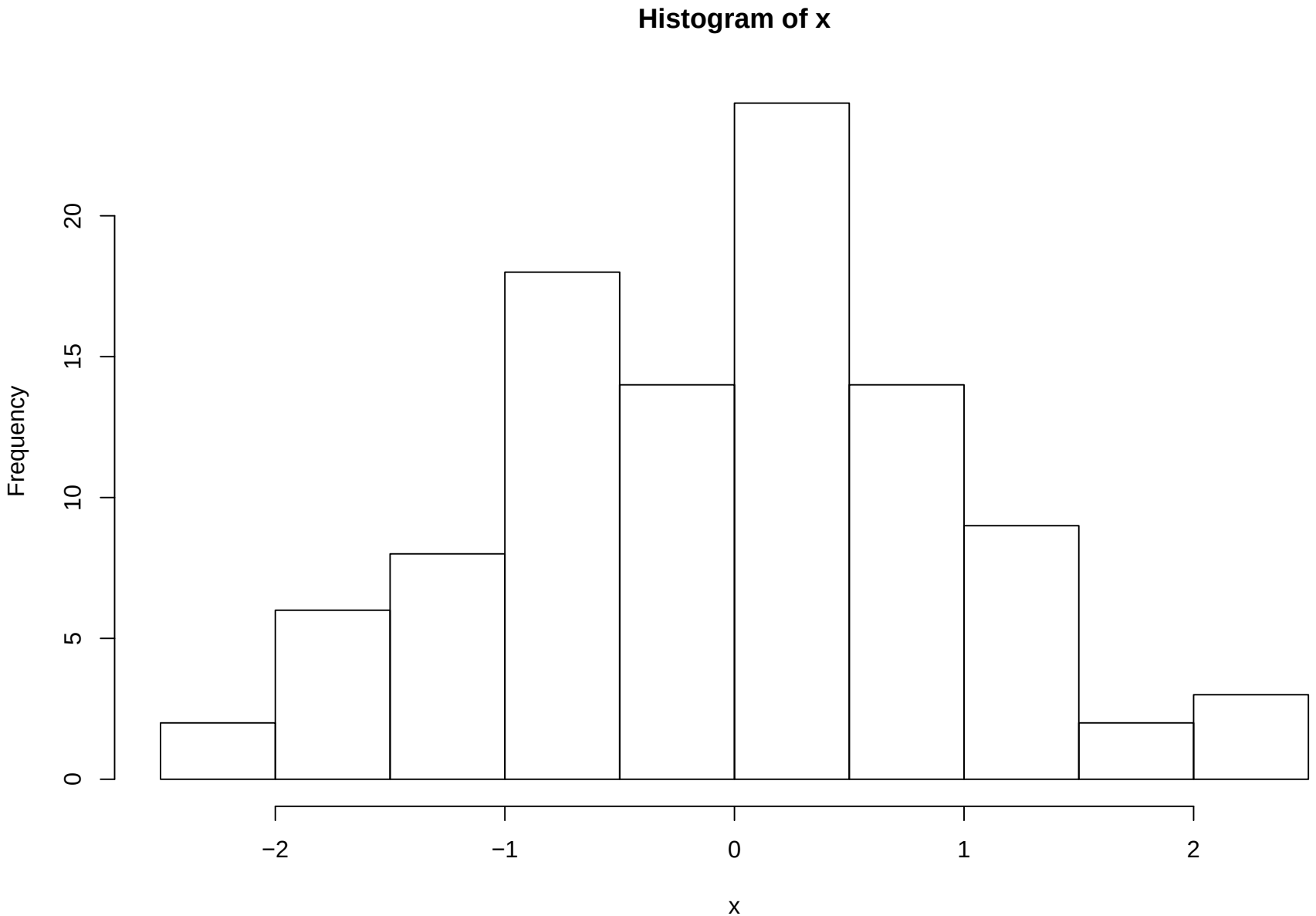
```r
x <- seq(from=0, to=2*pi, length.out=100)
sinx <- sin(x)
plot(x, sinx)
```



```r
df <- data.frame(x=x, y=sinx)
plot(y ~ x, data=df)
```



```r
x <- rnorm(100)
hist(x)
```



Next week we will discuss more advanced and systematic data visualization with a layered "grammar of graphics" using ggplot2