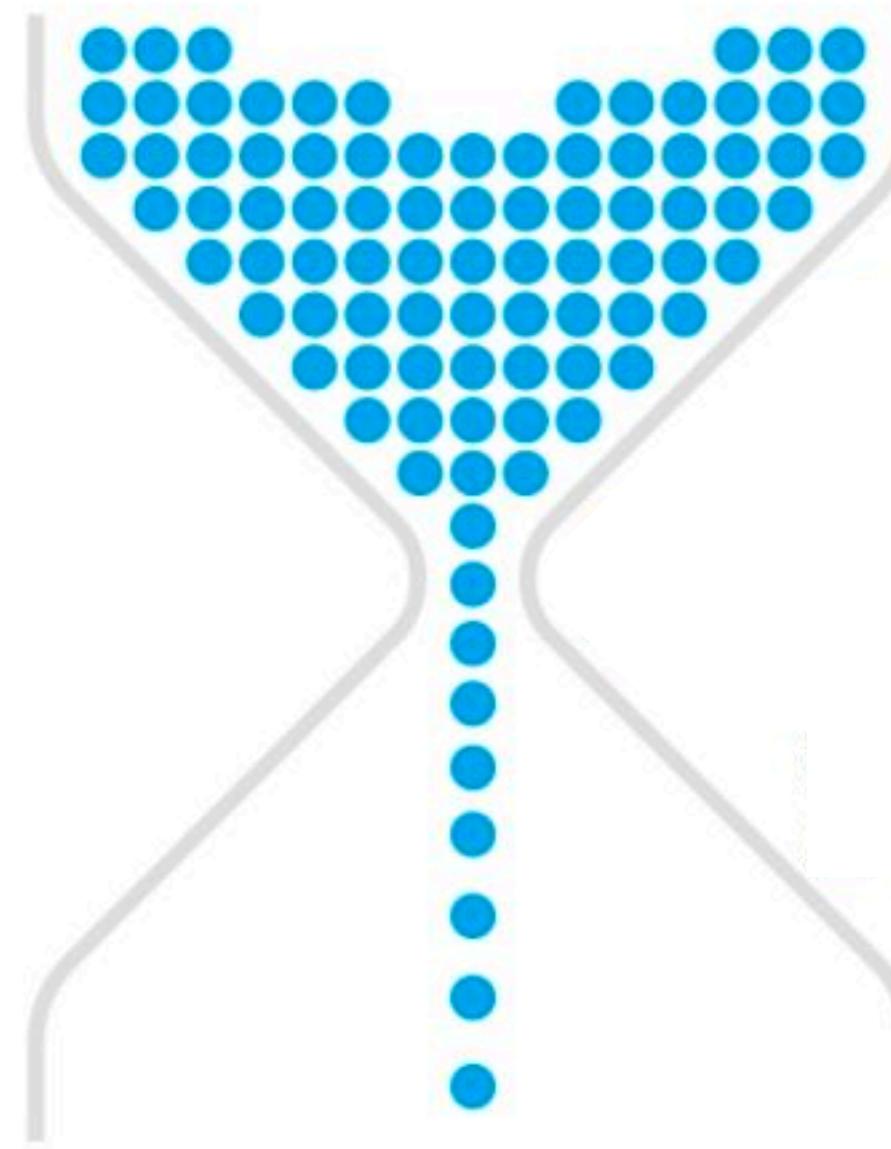




Basic R Programming

Mustafa Anil Kocak **March 4th 2024**

Why learn statistical programming?



Data is the new commodity

- We have better access to data than most of the world
- Not using it efficiently is leaving money on the table!

Analysis/analysts are typical bottlenecks

- Analysts are working one experiment at a time
- Understanding the experiment, meta-data and questions takes time

Most of the analyses are ...

- straightforward once you got the experimental thinking
- can be performed with a limited vocabulary!

Most analyses are doable with limited vocabulary

- **Visualization (ggplot)**
- **Vectors, lists and data tables (tibbles)**
- **Libraries and how to get help**
- **Loading/Saving datasets:** `data.table::fread`, `readRDS()`, `write_csv()`
- **Taking a peek:** `head()`, `tail()`, `summary()`, `names()`, `View()`, `nrow()`, `table()`, `dim()`,
`summarize()`, `describe()`
- **Data manipulation:** `select()`, `distinct()`, `filter()`, `group_by()`, `ungroup()`, `summarize()`,
`summarize_all()`, `arrange()`, `count()`
- **Joins:** `left/right/inner/full/anti/semi_join()`
- **Extra:** `top_n()`, `is.finite()`, `is.na()`, `%in%`, `paste0()`, `max()`, `min()`, `mean()`, `median()`, `sd()`,
`mad()`, `quantile()`, `is.finite()`, `toupper()`, `word()`, `n()`, `%>%`, `write_csv()`, `bind_rows()`, `c()`,
`psych::pairs.plots()`, `pheatmap()`, `reorder()`, `taiga (?)`

What is our objective?

How to draw an owl

1.



2.



1. Draw some circles

2. Draw the rest of the ~~fucking~~ owl

What is our objective?

How to draw an owl

1.



2.



1. Draw some circles

2. Draw the rest of the ~~fucking~~ owl

Knowledge

Skill

Attitude

What is our objective?

How to draw an owl

1.



2.



1. Draw some circles

2. Draw the rest of the ~~fucking~~ owl

Knowledge

- What R is good for; tutorials, resources, lectures

Skill

Attitude

What is our objective?

How to draw an owl

1.



2.



1. Draw some circles

2. Draw the rest of the fucking owl

Knowledge

- What R is good for; tutorials, resources, lectures

Skill

- Using what you learned in your projects next week

Attitude

What is our objective?

How to draw an owl

1.



2.



1. Draw some circles

2. Draw the rest of the fucking owl

Knowledge

- What R is good for; tutorials, resources, lectures

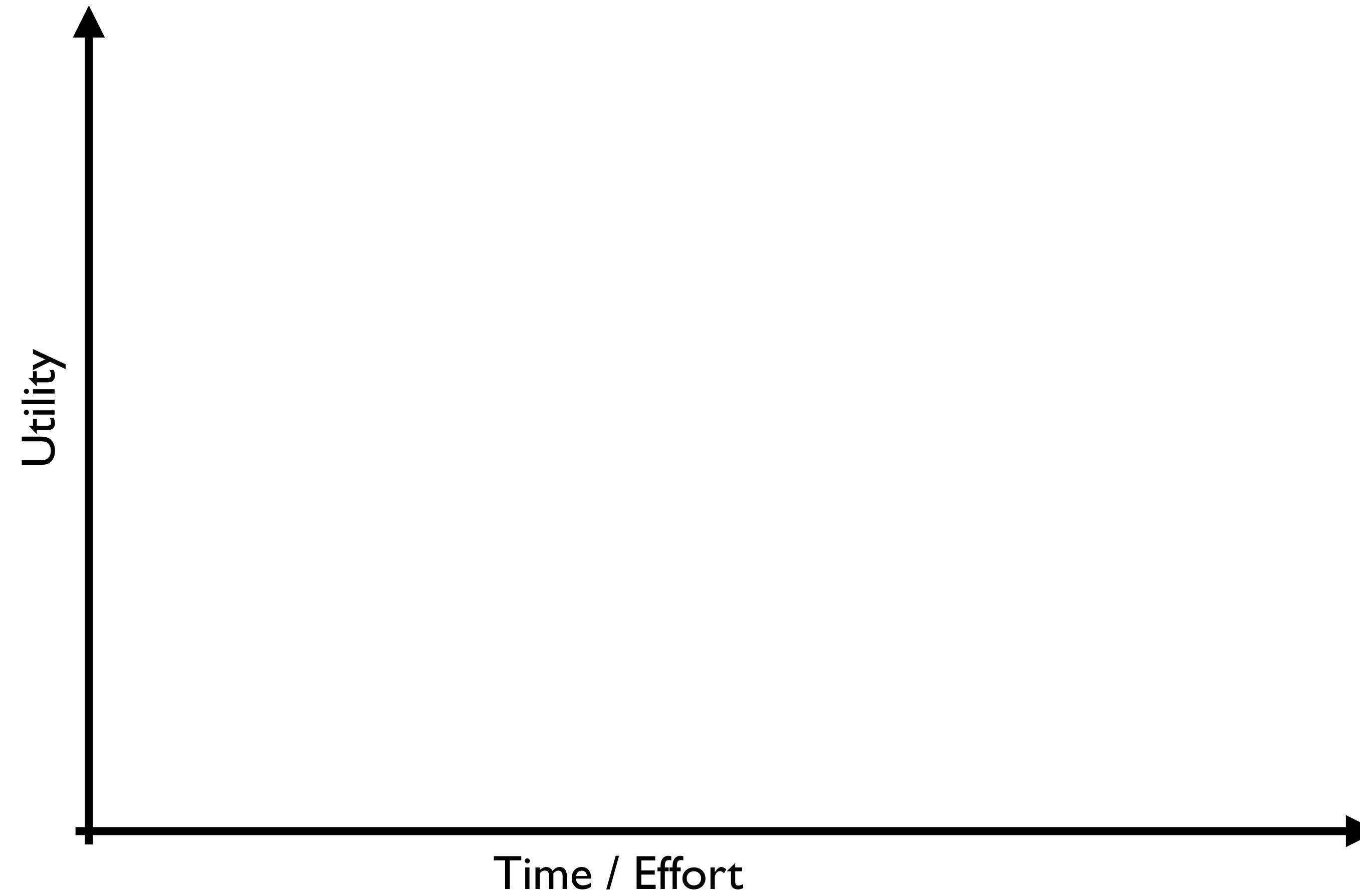
Skill

- Using what you learned in your projects next week

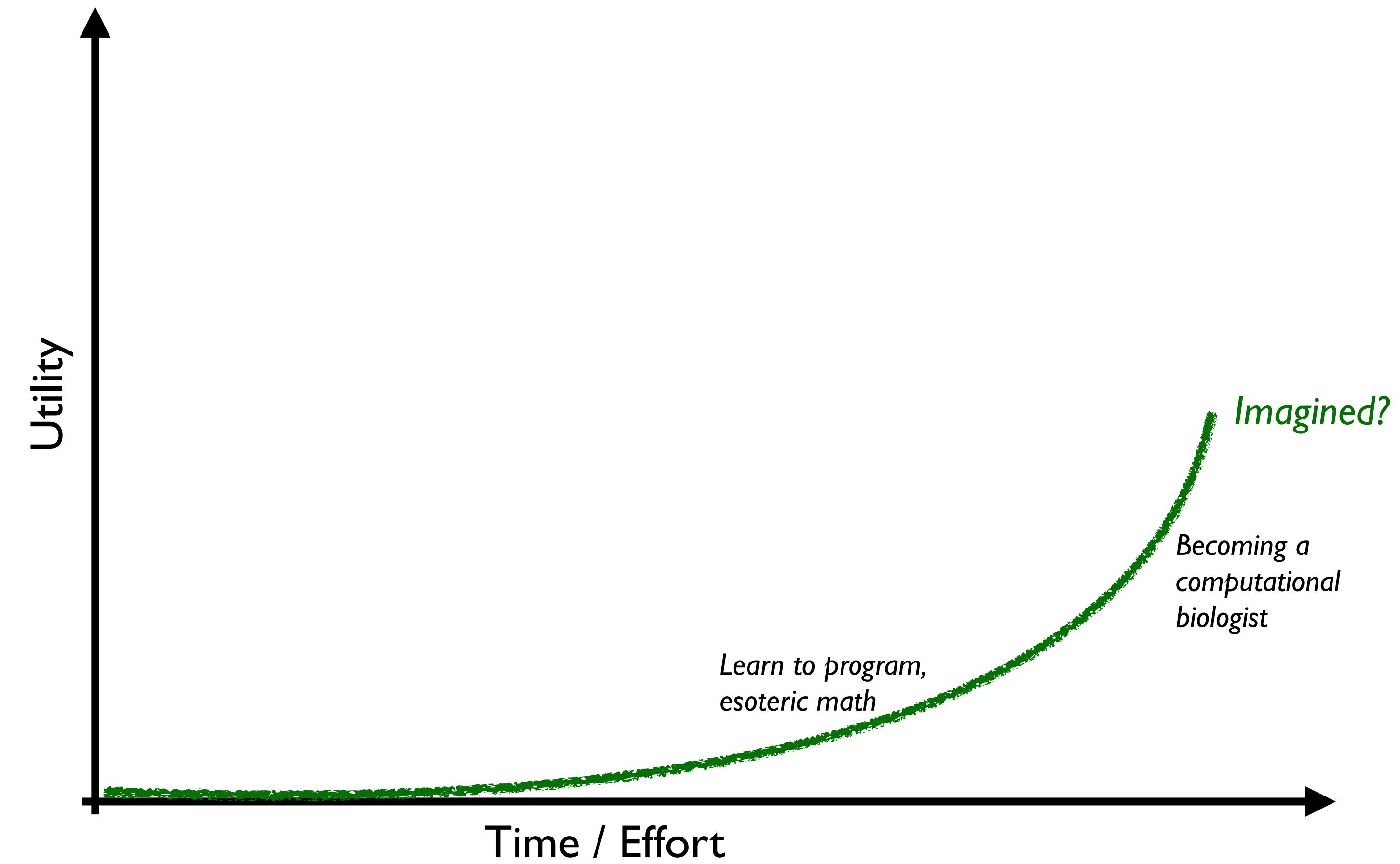
Attitude

- Keep using what you learned in the bootcamp
- Having joy through playing with data!

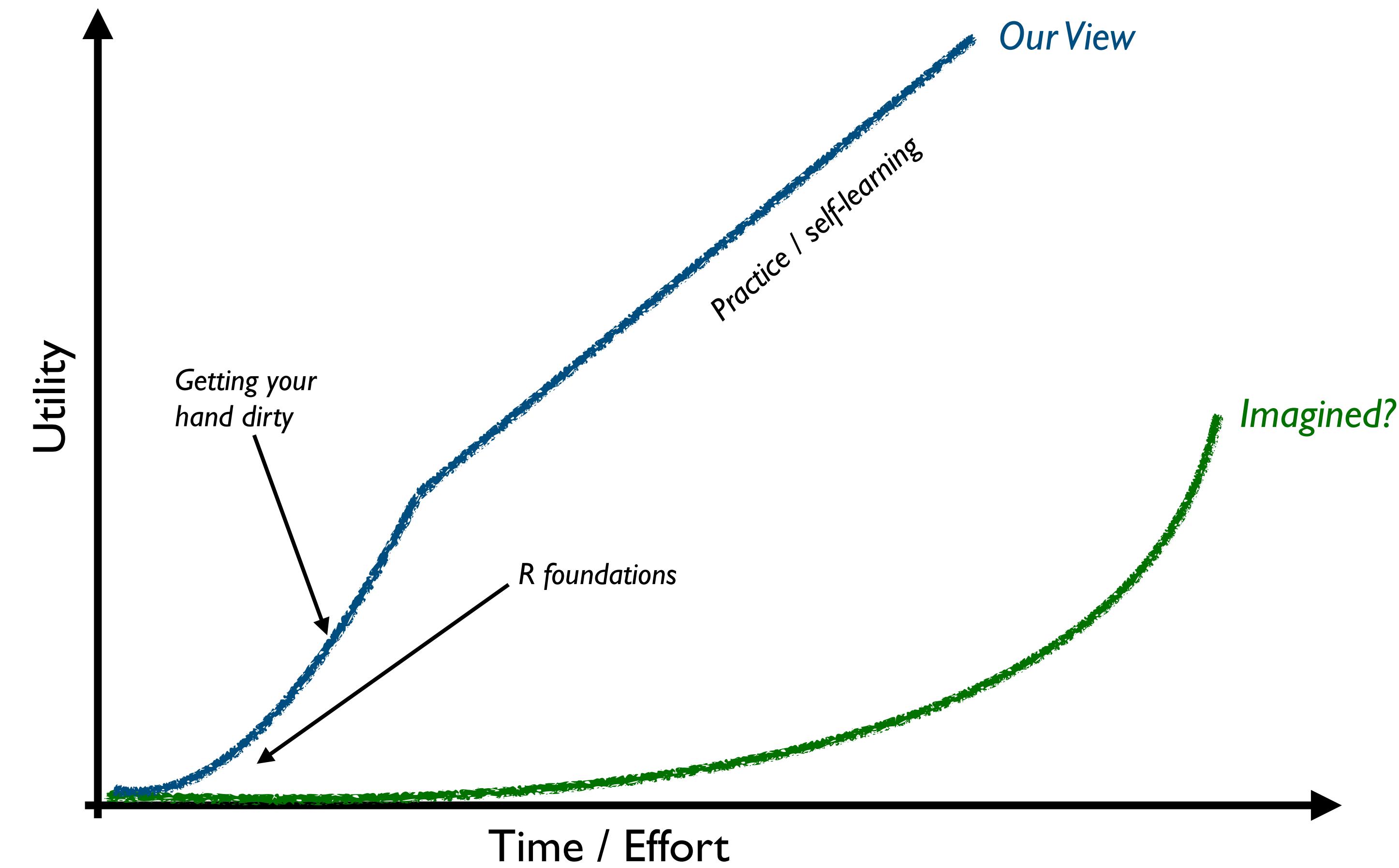
Our View of the Learning Curve



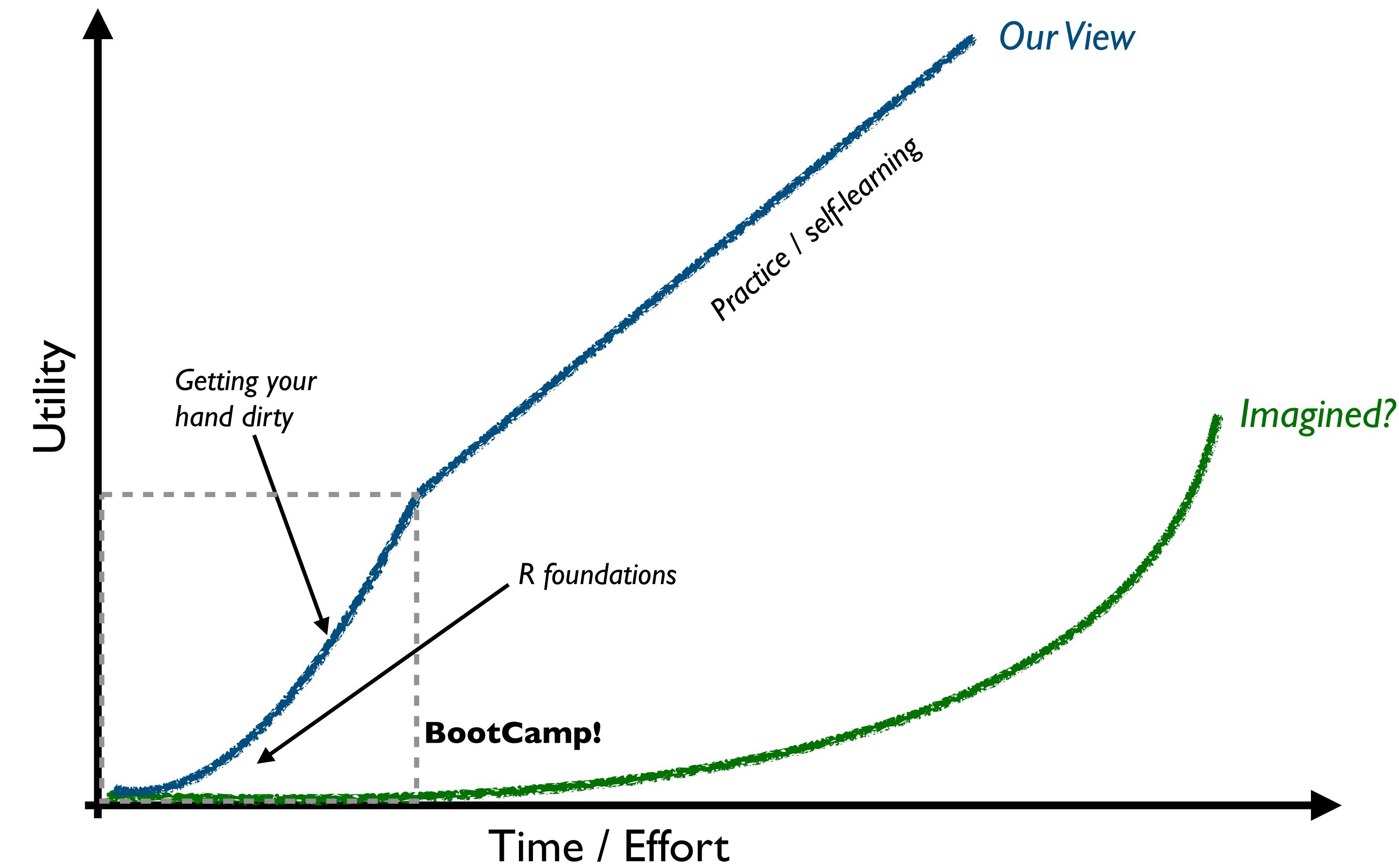
Our View of the Learning Curve



Our View of the Learning Curve

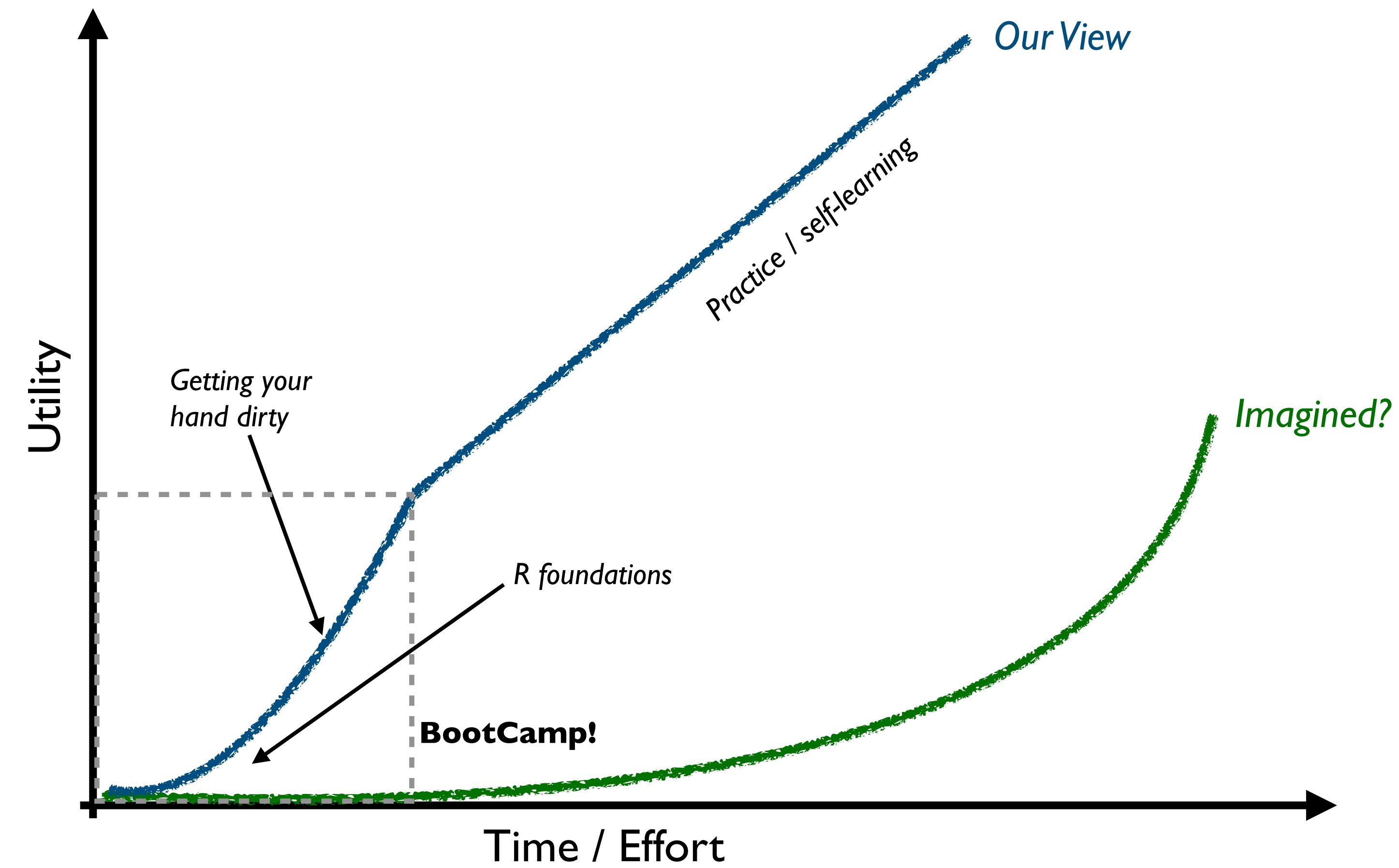


Our View of the Learning Curve



Our View of the Learning Curve

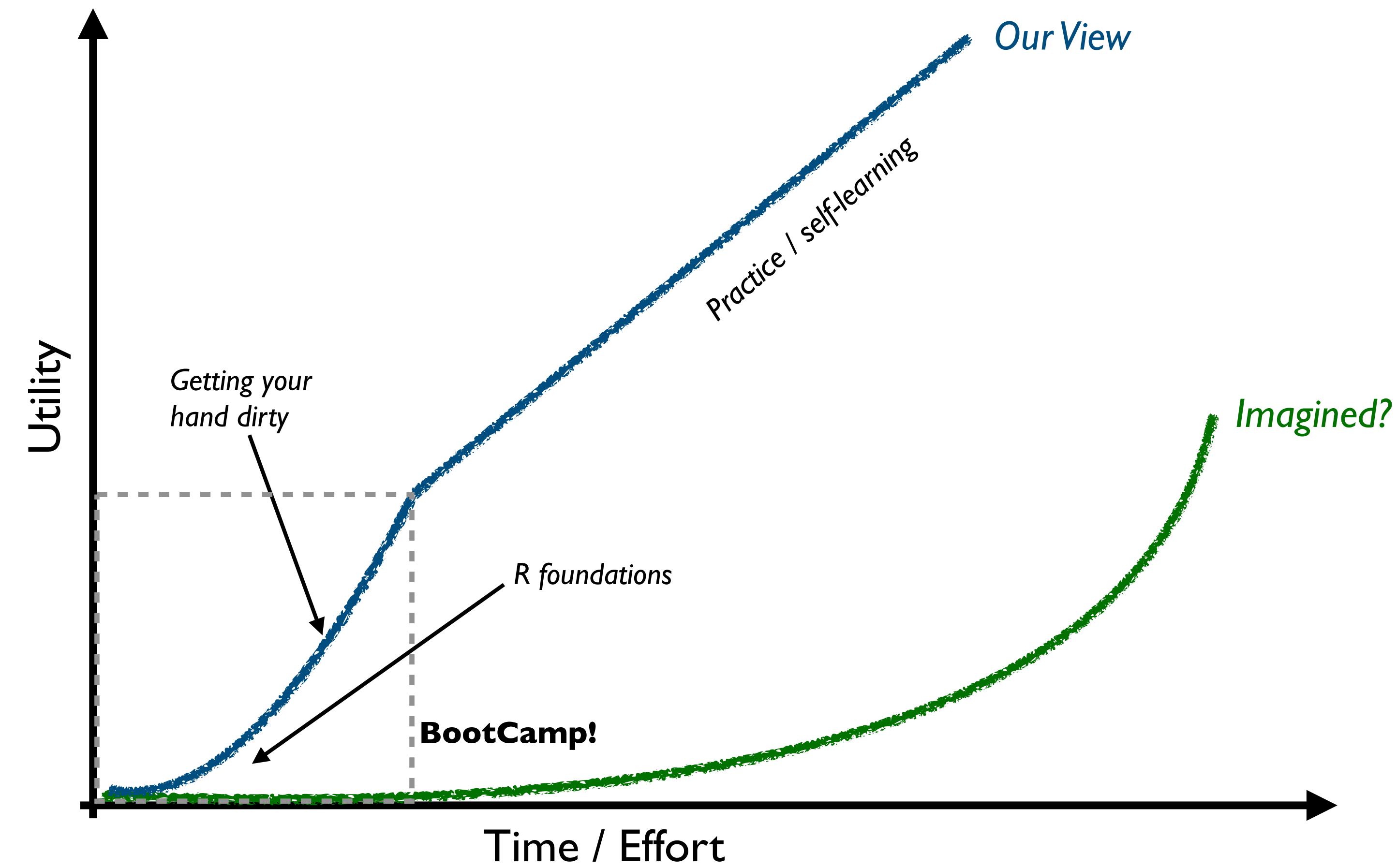
I need your help!



Our View of the Learning Curve

I need your help!

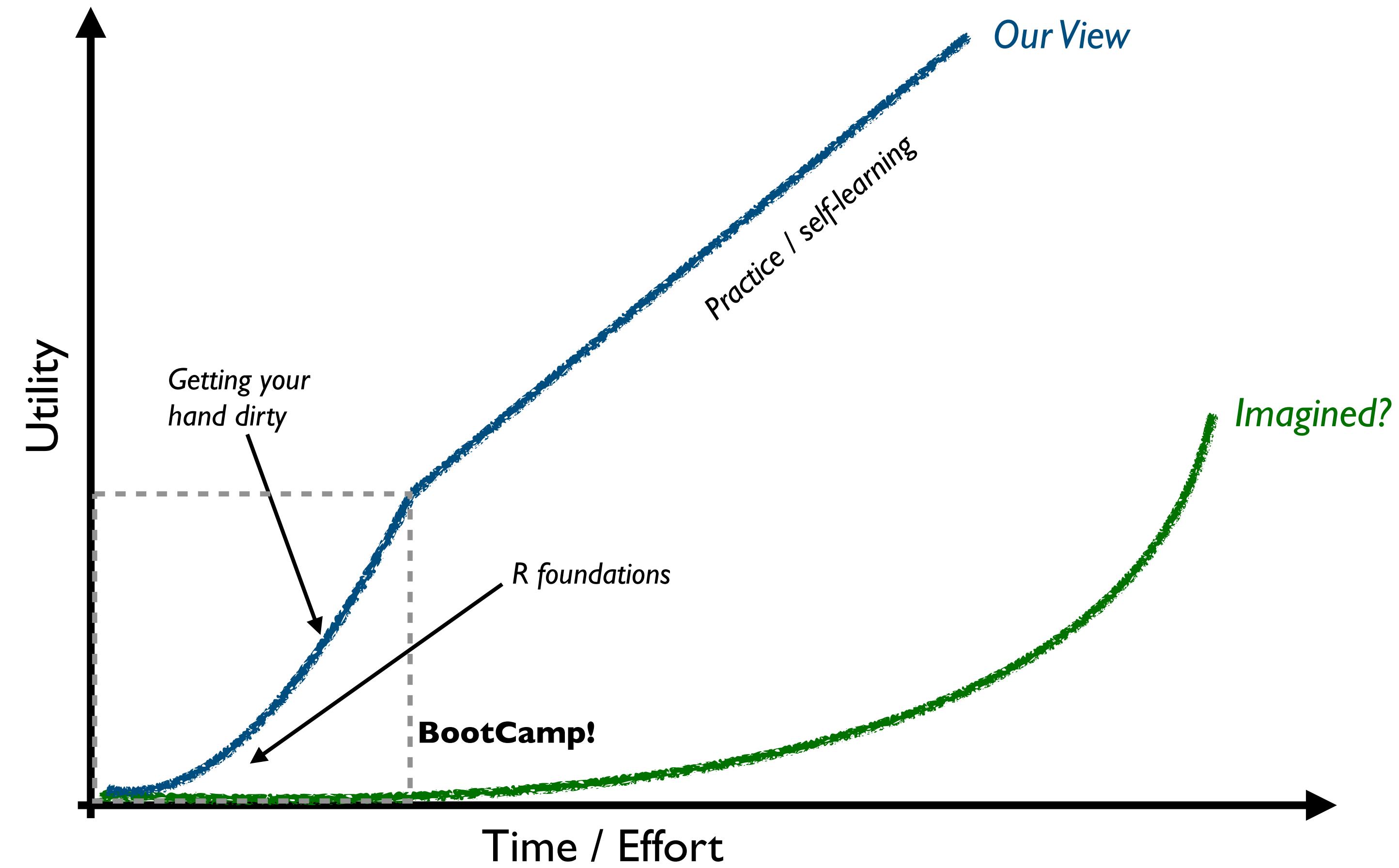
- This is my first second bootcamp



Our View of the Learning Curve

I need your help!

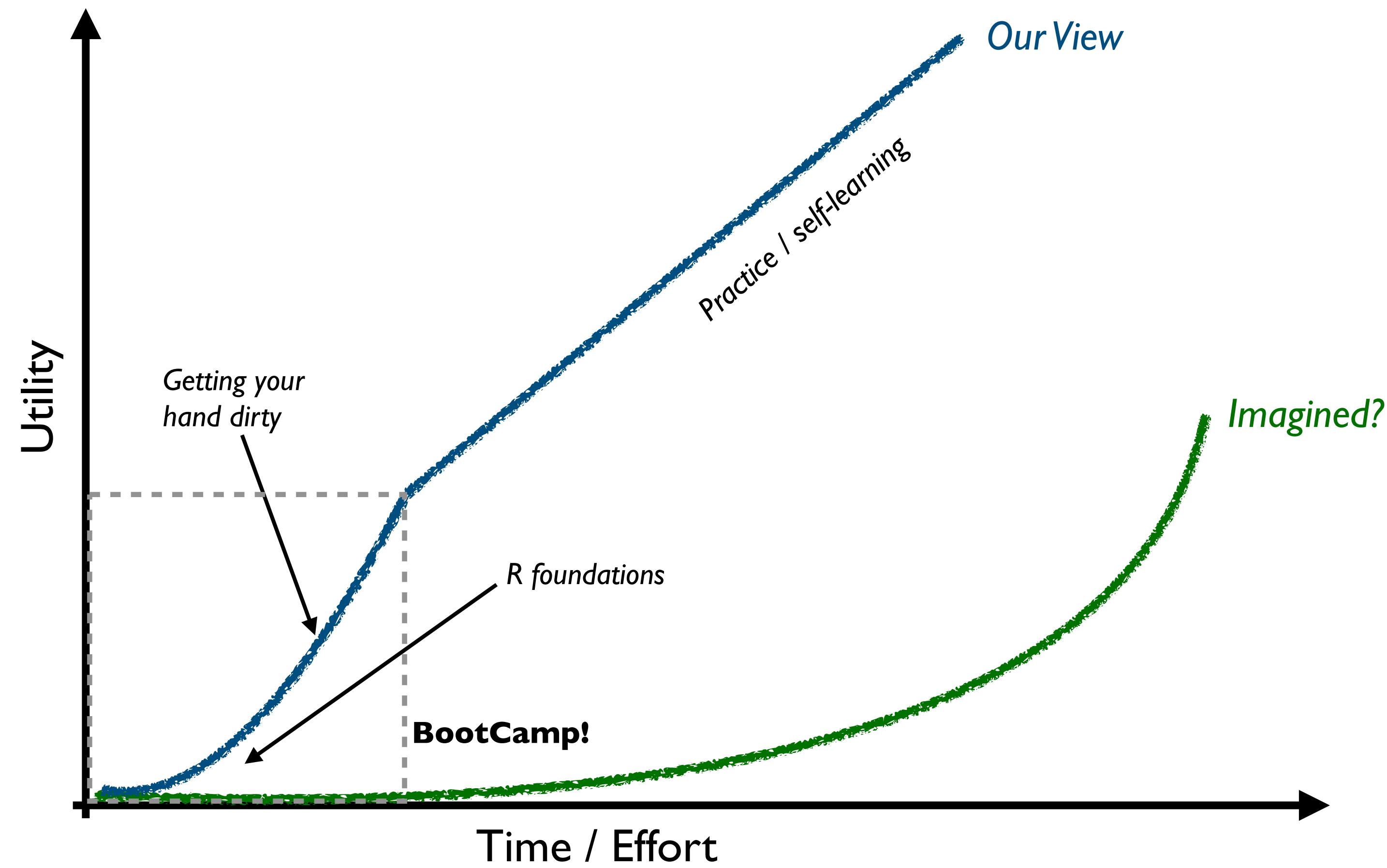
- This is my first second bootcamp
- And I am nervous :)



Our View of the Learning Curve

I need your help!

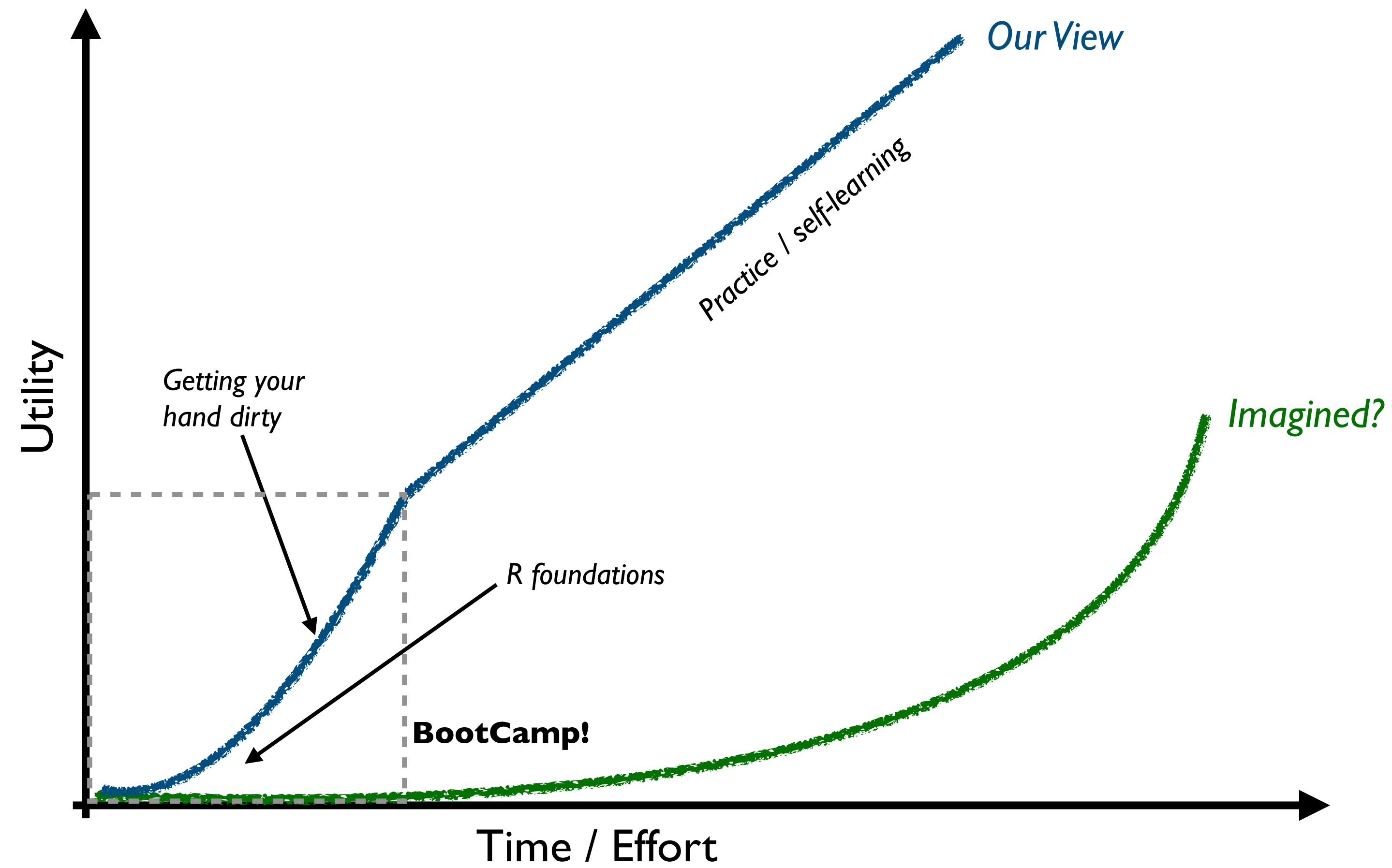
- This is my first second bootcamp
- And I am nervous :)
- We will figure it out together



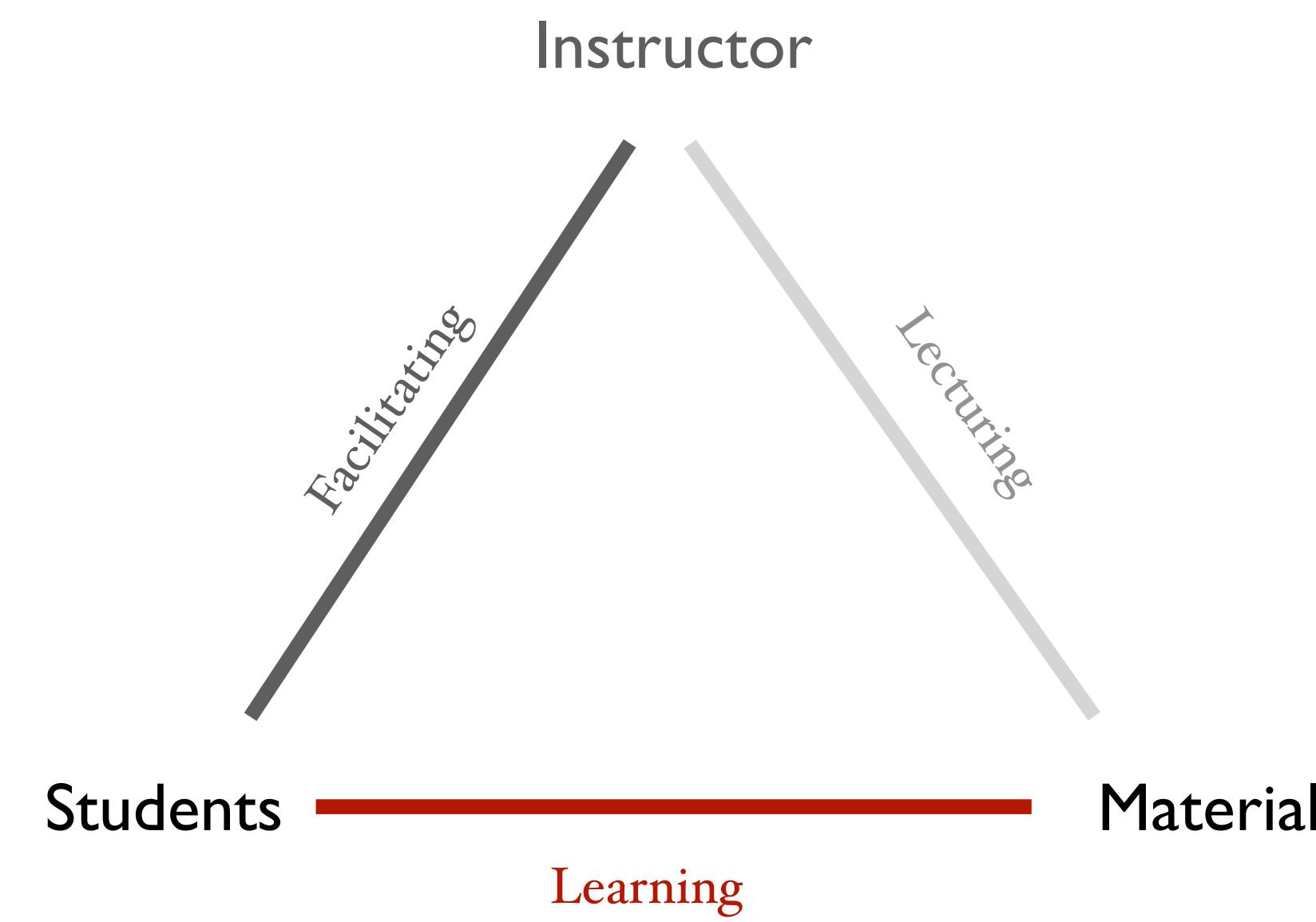
Our View of the Learning Curve

I need your help!

- This is my first second bootcamp
- And I am nervous :)
- We will figure it out together
- Please ask questions and give feedback



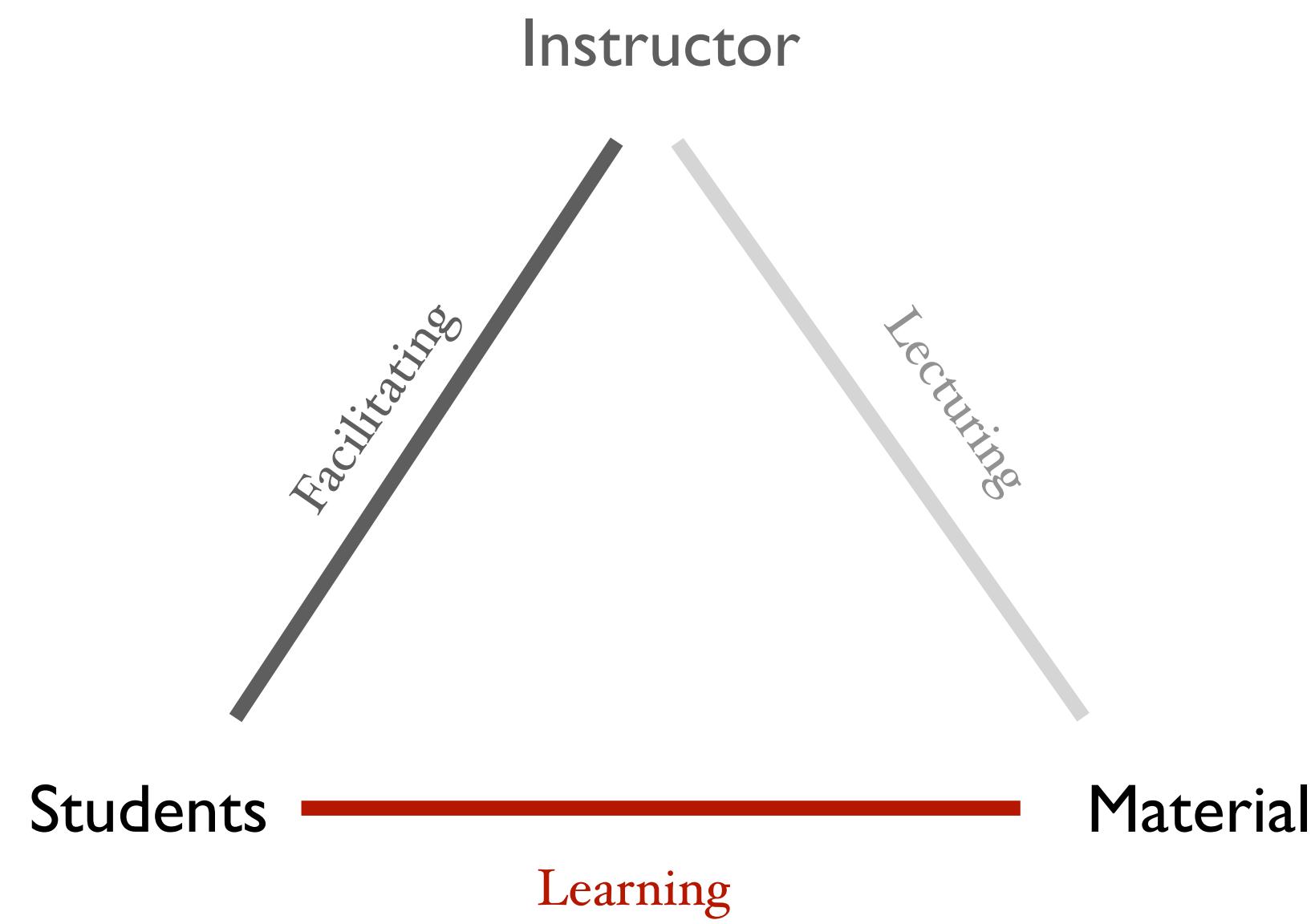
Approach



Approach

Emphasis on the practical skills

- Basics, visualization, wrangling, ...



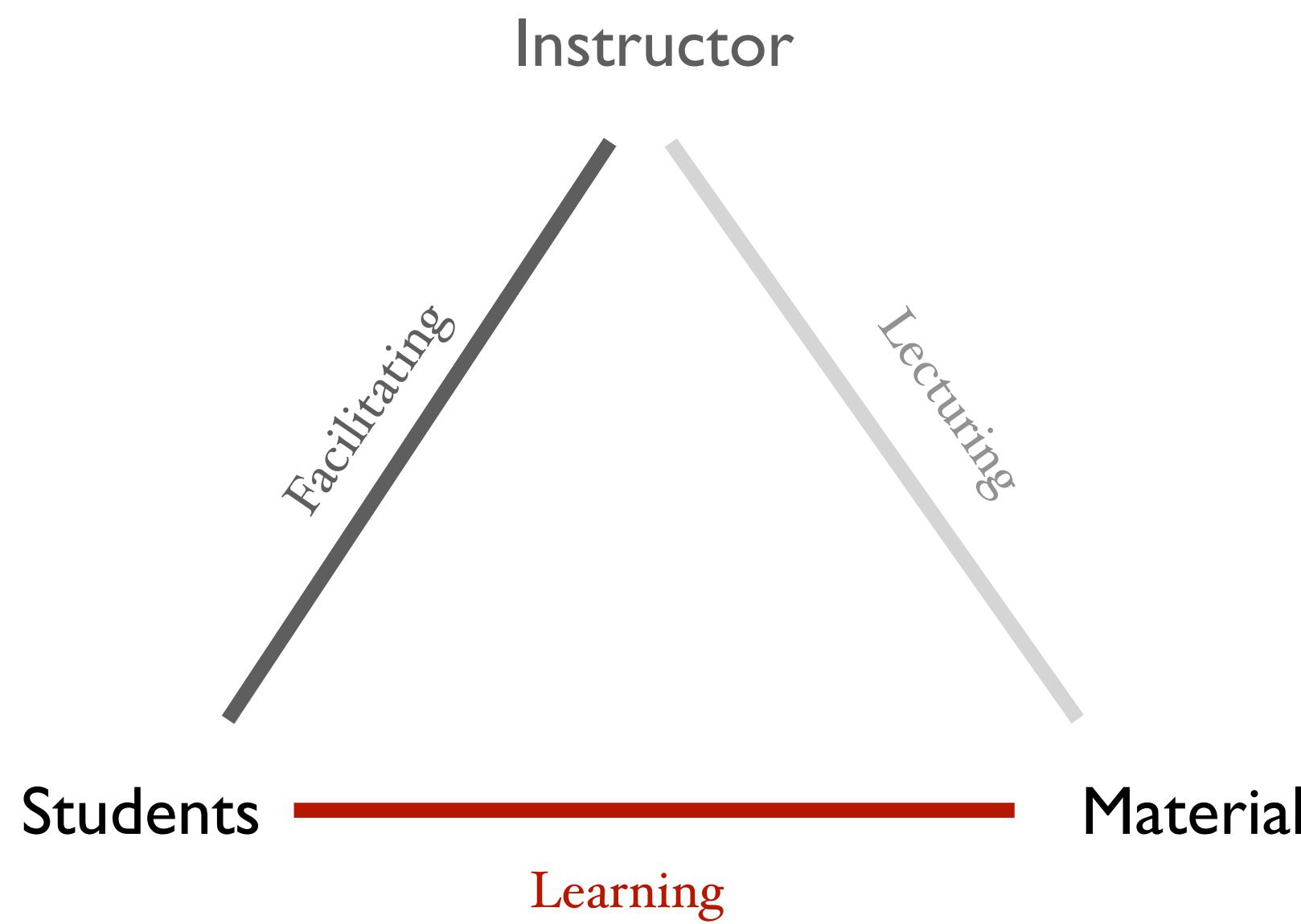
Approach

Emphasis on the practical skills

- Basics, visualization, wrangling, ...

Pointing out resources but
mostly giving the ‘need-to-know’

- Think about what else would you
like to do with R and data?



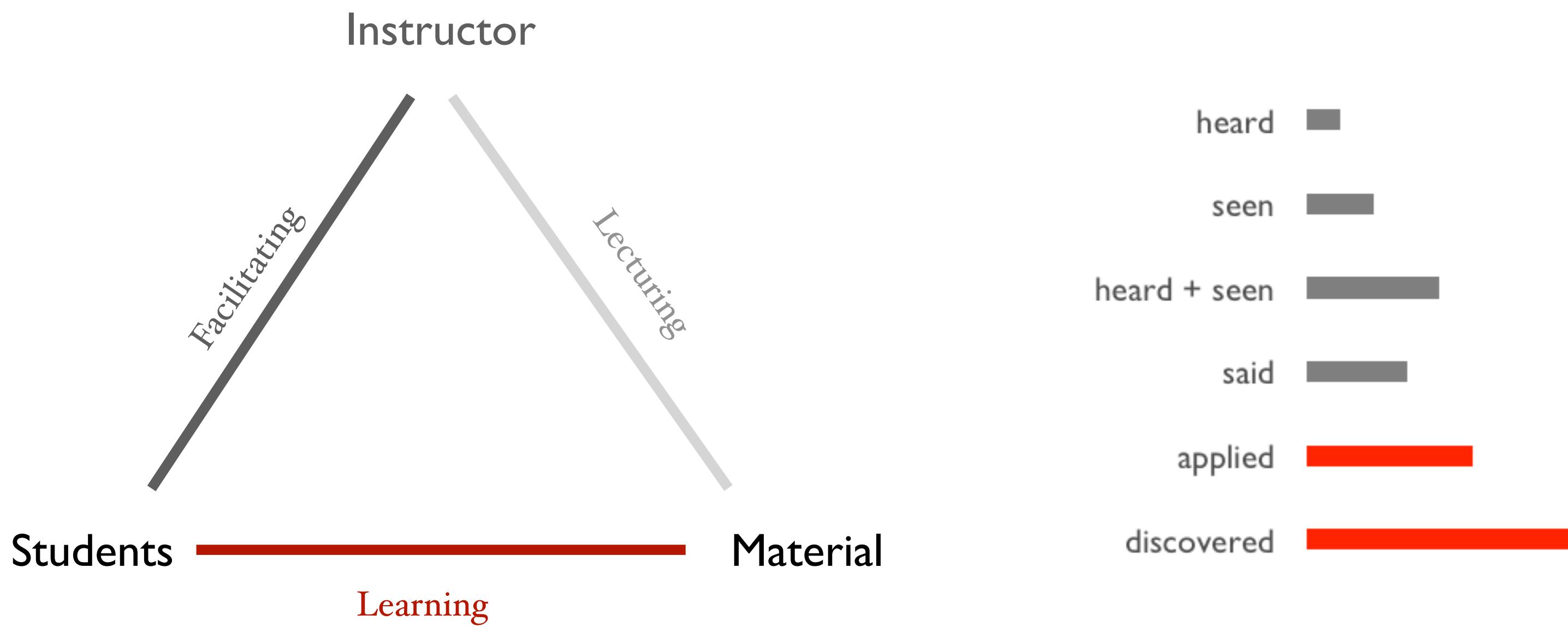
Approach

Emphasis on the practical skills

- Basics, visualization, wrangling, ...

Pointing out resources but
mostly giving the ‘need-to-know’

- Think about what else would you
like to do with R and data?



Approach

Emphasis on the practical skills

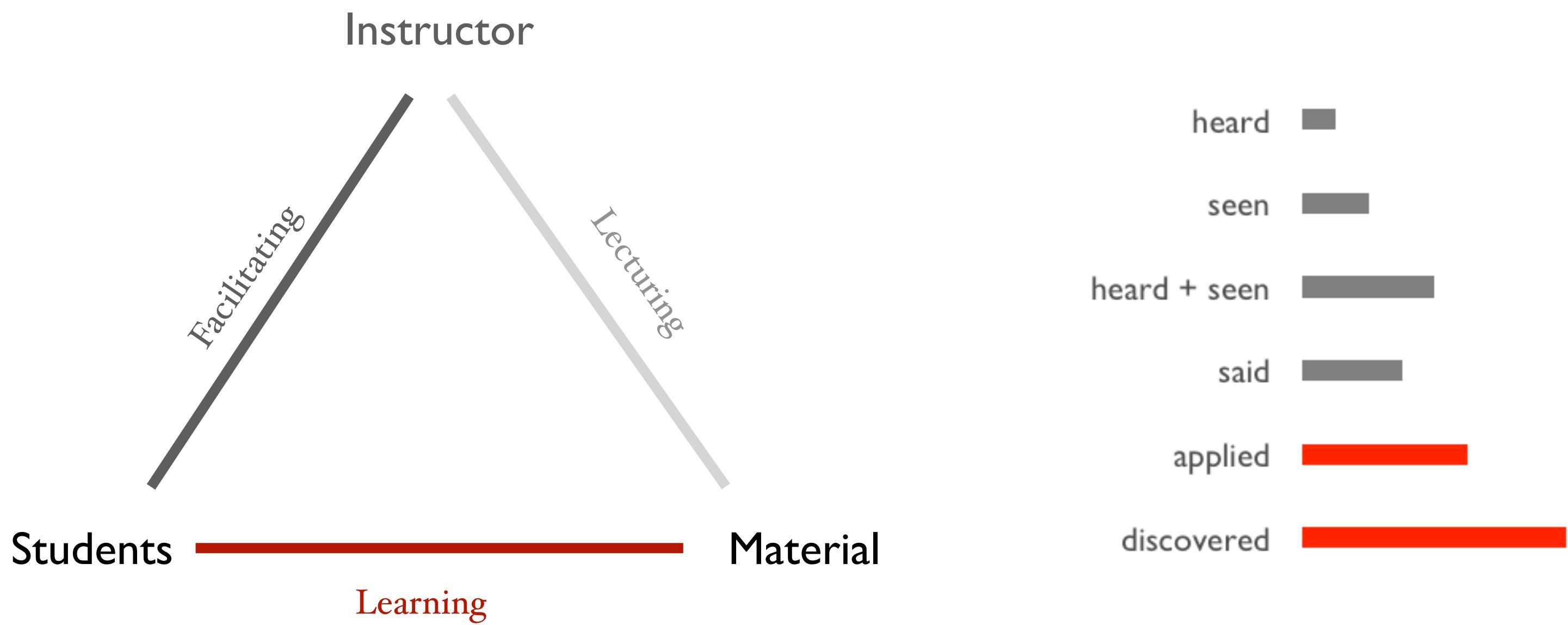
- Basics, visualization, wrangling, ...

Pointing out resources but mostly giving the 'need-to-know'

- Think about what else would you like to do with R and data?

Slides are reminders, we will write code during the sessions.

- Learn from your experience and peers!



Course Content

First Encounter with R Programming!

- R studio, scripts, markdowns
- Functions, packages, help pages
- R Objects, notation & modifying values
- Data structures, programming

Introduction to Tidyverse

- Visualization
- Wrangling
- Exploratory Data Analysis

Miscellaneous

- Practice and case studies
- Advanced data visualization
- Machine learning examples

Course Content

First Encounter with R Programming!

- R studio, scripts, markdowns**
- Functions, packages, help pages**
- R Objects, notation & modifying values**
- Data structures, programming**

Introduction to Tidyverse

- Visualization
- Wrangling
- Exploratory Data Analysis

Miscellaneous

- Practice and case studies
- Advanced data visualization
- Machine learning examples

A hands-on introduction to basic R

The Very Basics
R Objects & Notation
Basic Programming

First Encounter with R Programming!

Freely available online:

- <https://rstudio-education.github.io/hopr/>

Little pedestrian, but great introduction.

We will skip chapters 8 and 10.

- Recommended for self-study
- Great basis, but we will significantly refine in the following sessions.



Garrett Grolemund

The Very Basics

User Interface

Objects

Functions

Writing your Own Functions

Packages

Scripts / Mark-downs

R Studio Interface

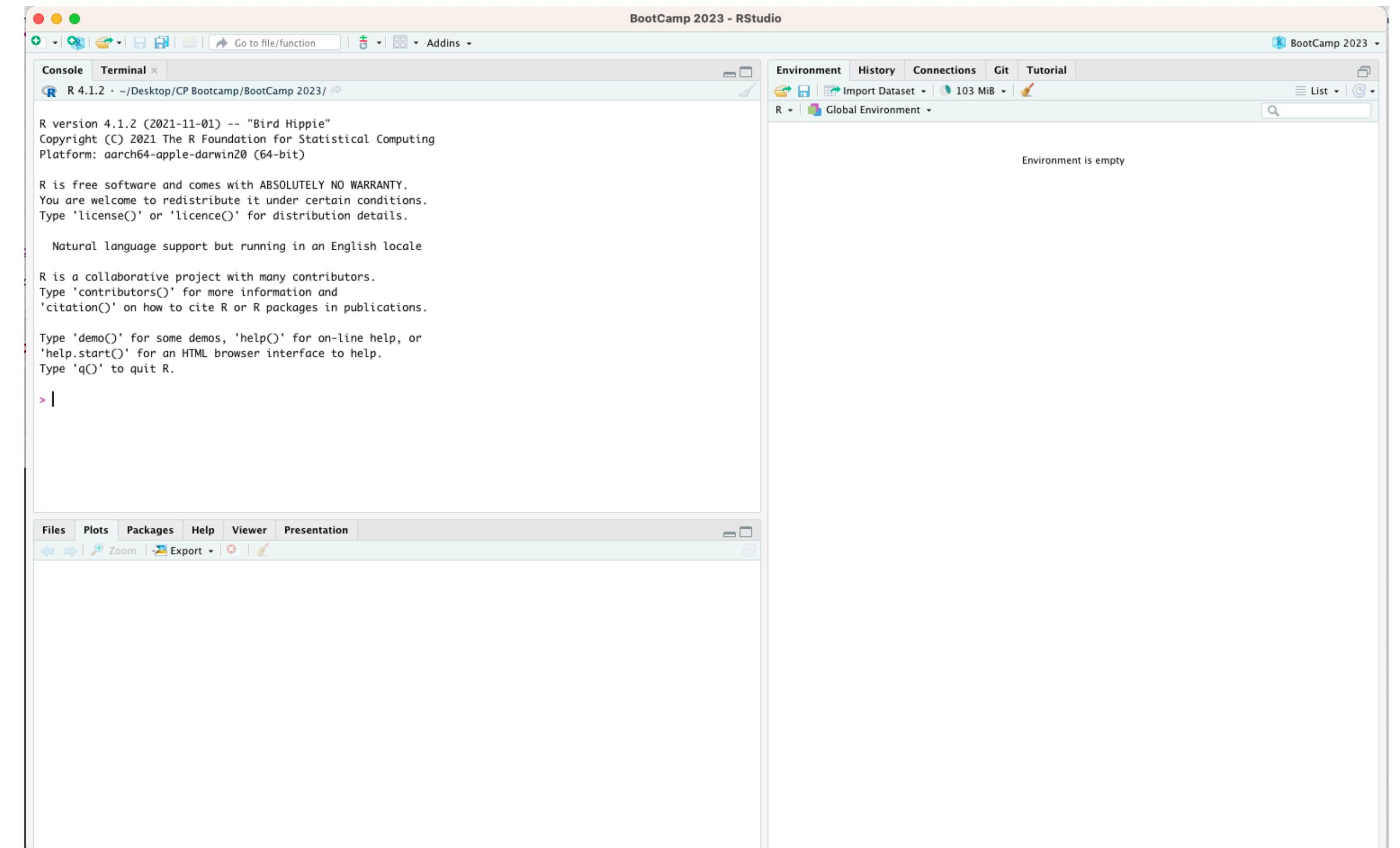
□ Interface

□ Projects

□ R as a calculator

□ Scripts and mark-downs

□ Arithmetic operators



R Objects

Variables

Vectors

Naming convention

Vector recycling

$$\begin{array}{rcl} \text{die} & * & \text{die} \\ \hline 1 & * & 1 \\ 2 & * & 2 \\ 3 & * & 3 \\ 4 & * & 4 \\ 5 & * & 5 \\ 6 & * & 6 \end{array} = \begin{array}{r} 1 \\ 4 \\ 9 \\ 16 \\ 25 \\ 36 \end{array}$$

$$\begin{array}{rcl} \text{die} & * & \text{c}(1,2) \\ \hline 1 & * & 1 \\ 2 & * & 2 \\ 3 & * & 1 \\ 4 & * & 2 \\ 5 & * & 1 \\ 6 & * & 2 \end{array} = \begin{array}{r} 1 \\ 4 \\ 4 \\ 6 \\ 6 \\ 8 \end{array}$$

Functions

□ Calling a function

□ Arguments

□ Help pages

□ Writing a function

□ Extract function

1. **The name.** A user can run the function by typing the name followed by parentheses, e.g., roll2().

3. **The arguments.** A user can supply values for these variables, which appear in the body of the function.

4. **The default values.** Optional values that R can use for the arguments if a user does not supply a value.

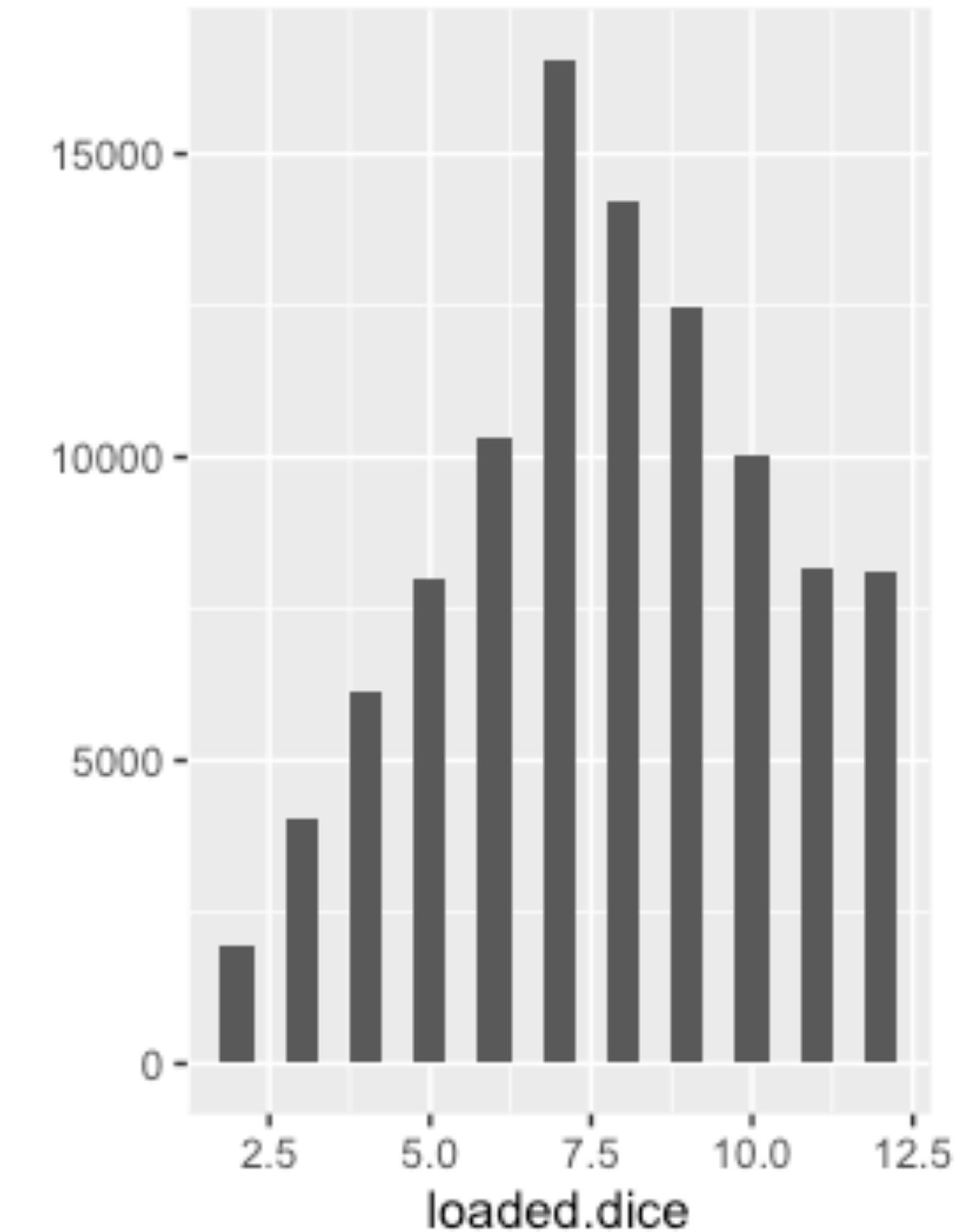
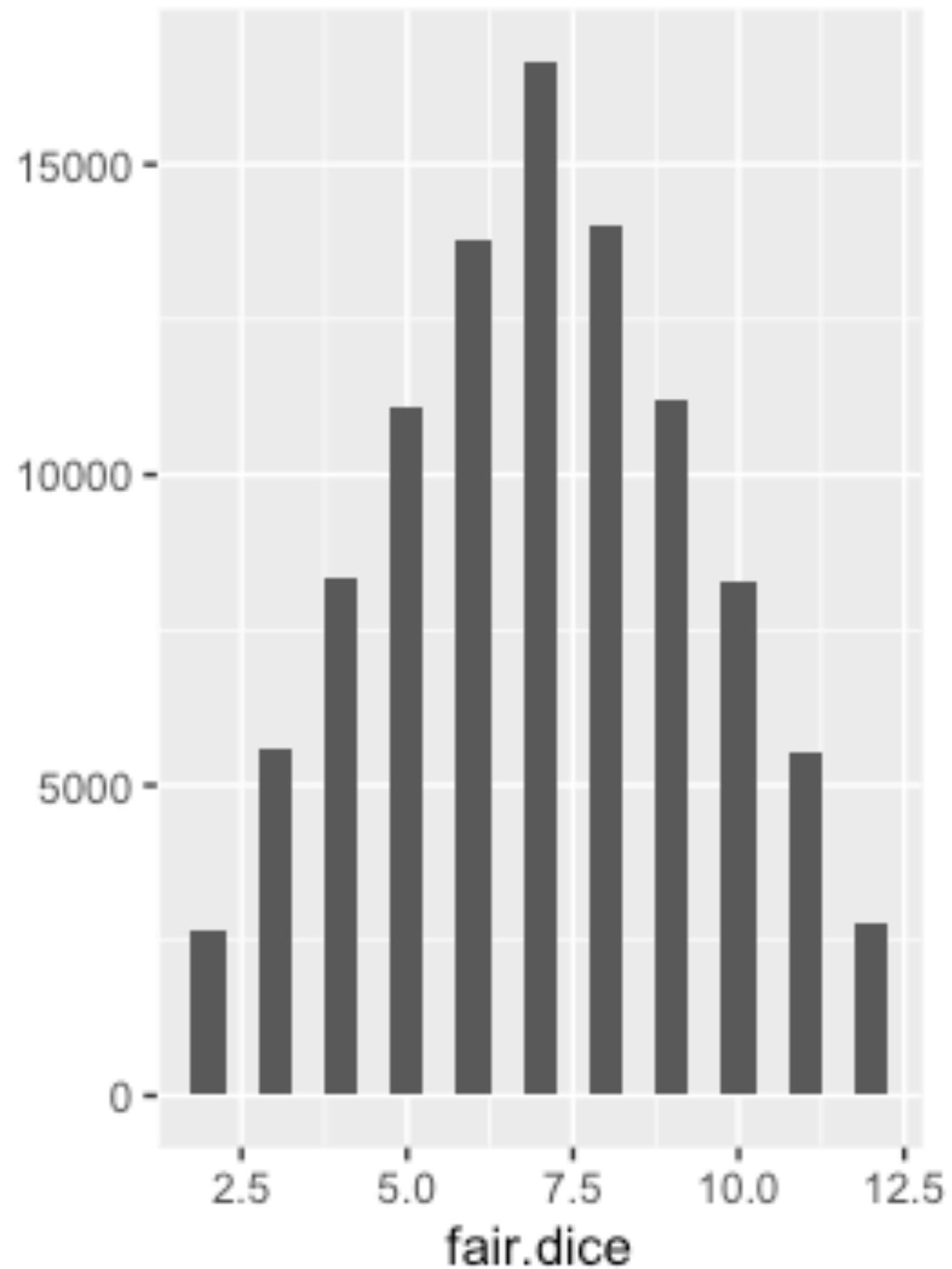
2. **The body.** R will run this code whenever a user calls the function.

```
roll2 <- function(bones = 1:6) {  
  dice <- sample(bones, size = 2,  
    replace = TRUE)  
  sum(dice)  
}
```

5. **The last line of code.** The function will return the result of the last line.

Packages

- Installing packages
- Loading packages
- Functions from a package
- Help pages re-visited
- Misc:
 - c / replicate / qplot / ctrl + c / pipe / plot_grid



R Objects & Notation

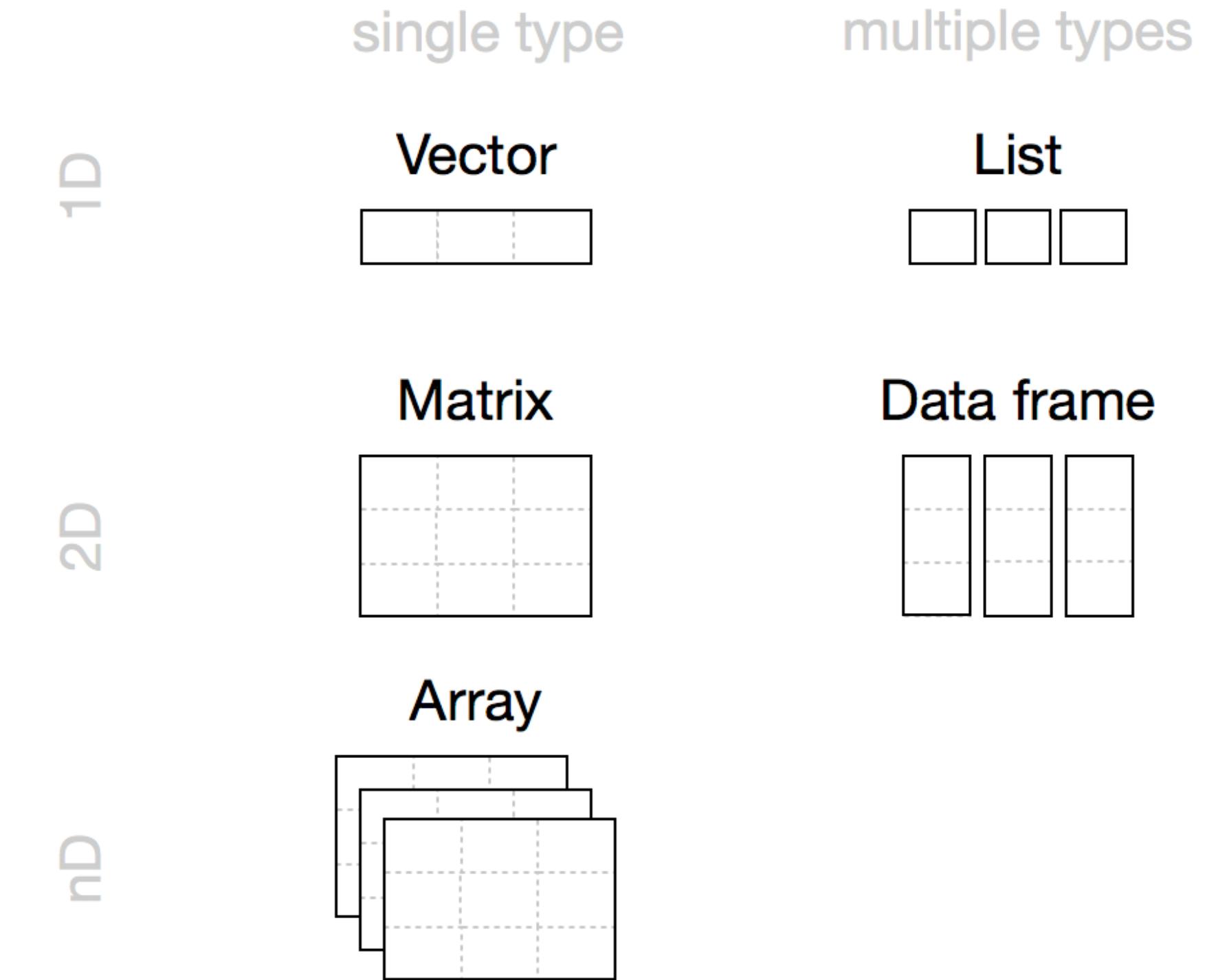
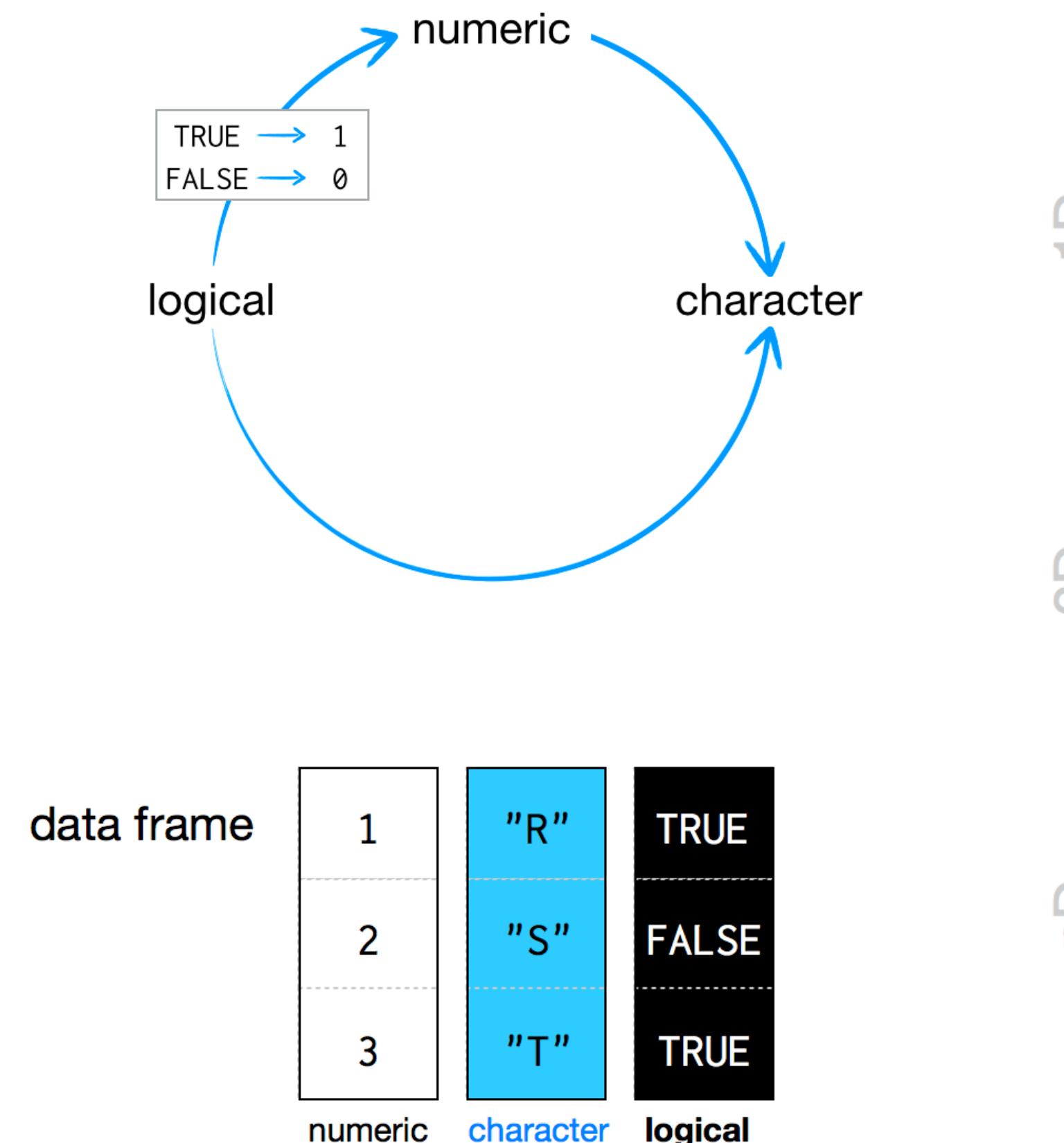
R Objects

Slicing

Modifying Values

R Objects

- Atomic vectors
- Basic types: **doubles**, integers, **characters**, **logicals**, complex, raw
- Coercion
- Matrices & arrays
- Attributes & class
- Lists & data frames
- Loading & saving data



Slicing and Modifying Values

Slicing datasets: []

`6 1 3 6 10 5`

John	1940	guitar
Paul	1941	bass
George	1943	guitar
Ringo	1940	drums

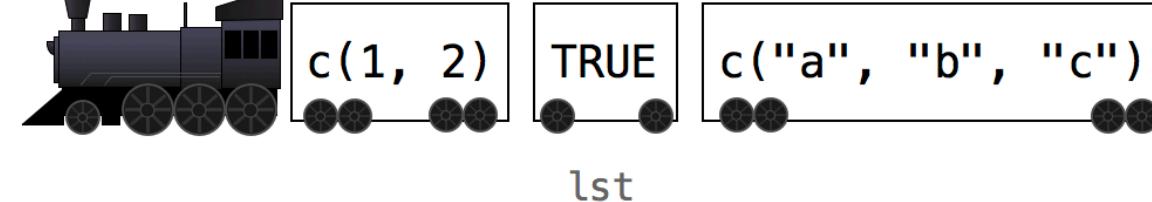
\$ and [[]] to slice lists

`vec[5]`

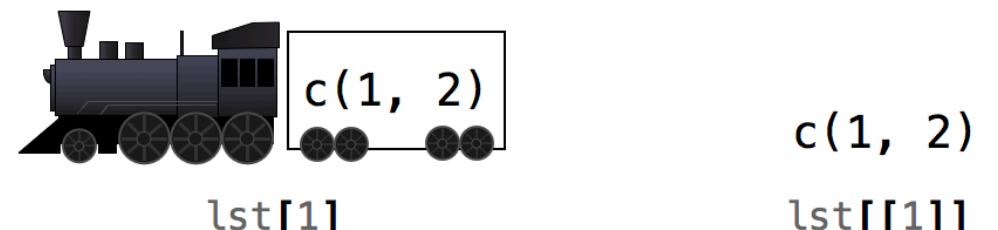
`df[2, c(2,3)]`

Modifying values

Changing in-place



Logical subsetting



Missing values

`x > 2 & x < 9`

`TRUE & TRUE`
↓
`TRUE`

`x > 2 & < 9`

`TRUE & Error!`
↓
`Error!`

Operator	Syntax	Tests
<code>></code>	<code>a > b</code>	Is a greater than b?
<code>>=</code>	<code>a >= b</code>	Is a greater than or equal to b?
<code><</code>	<code>a < b</code>	Is a less than b?
<code><=</code>	<code>a <= b</code>	Is a less than or equal to b?
<code>==</code>	<code>a == b</code>	Is a equal to b?
<code>!=</code>	<code>a != b</code>	Is a not equal to b?
<code>%in%</code>	<code>a %in% c(a, b, c)</code>	Is a in the group c(a, b, c)?

Operator	Syntax	Tests
<code>&</code>	<code>cond1 & cond2</code>	Are both <code>cond1</code> and <code>cond2</code> true?
<code> </code>	<code>cond1 cond2</code>	Is one or more of <code>cond1</code> and <code>cond2</code> true?
<code>xor</code>	<code>xor(cond1, cond2)</code>	Is exactly one of <code>cond1</code> and <code>cond2</code> true?
<code>!</code>	<code>!cond1</code>	Is <code>cond1</code> false? (e.g., <code>!</code> flips the results of a logical test)
<code>any</code>	<code>any(cond1, cond2, cond3, ...)</code>	Are any of the conditions true?
<code>all</code>	<code>all(cond1, cond2, cond3, ...)</code>	Are all of the conditions true?

Exercises / Optional Reading

Chapter 17-20 from

<https://stat545.com/r-objects.html>

Basic Programming

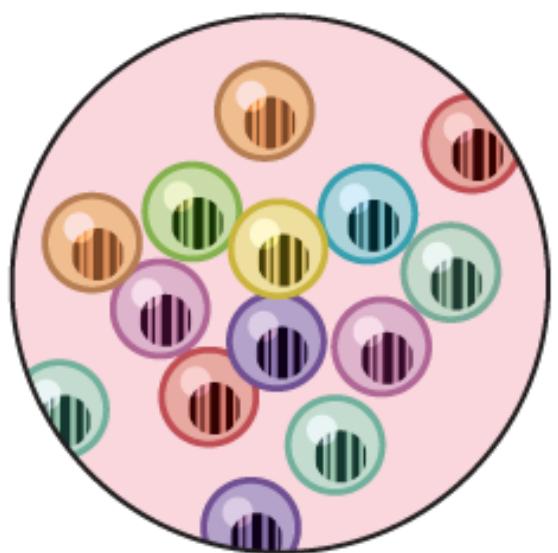
Pooled Cell Lines (Example)

Conditionals

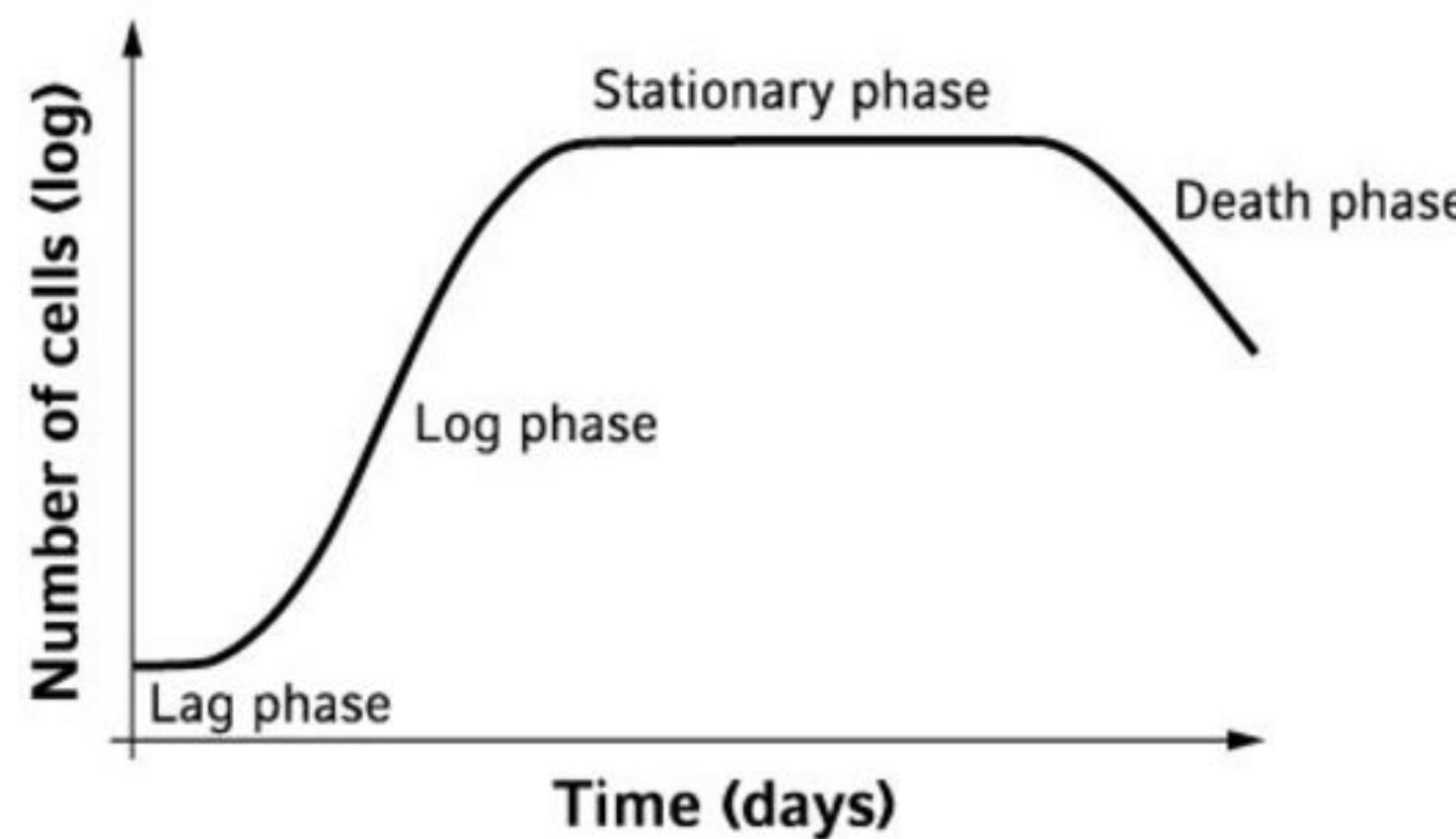
Expected Returns (Loops)

Speed

Pooled Cell Lines Example



Assay-ready
pools



- Cell Lines are characterized by**
 - Growth rates (doubling per day)
 - Initial cell count (cells)
 - Duration of lag-phase (days)
- Assay is characterized by**
 - End-point (days)
 - PCR bottleneck (cells)
 - Sequencing depth (counts)
- # of cell lines with more than 400 counts.**

Pooled Cell Lines and Conditionals

Strategy:

- Break complex tasks into simple subtasks.
- Use concrete examples.
- Describe your solutions in English, then convert them to R.

If / else statements

```
if (condition){  
  # This will run only when  
  # the condition is TRUE  
} else {  
  # This will run only when  
  # the condition is FALSE  
}
```

```
# First attempt (planning)  
simulate_experiment <- function(){  
  
  # calculate cell counts at the end-point  
  lysate_cell_counts <- simulate_growth()  
  
  # simulate the pcr bottleneck  
  pcr_counts <- simulate_pcr()  
  
  # simulate sequencing  
  sequencing_counts <- simulate_sequencing()  
  
  # visualize final population  
  hist(sequencing_counts, 100)  
  
  # count the detected cell lines  
  detected_lines <- count_detected()  
}  
}
```

Loops

- Expected values
- expand.grid
- for loops
- while loops
- repeat loops

$$E(x) = \sum_{i=1}^n (x_i \cdot P(x_i))$$

$$\begin{aligned} E(\text{die}) &= \sum_{i=1}^n (\text{die}_i \cdot P(\text{die}_i)) \\ &= 1 \cdot \frac{1}{6} + 2 \cdot \frac{1}{6} + 3 \cdot \frac{1}{6} + 4 \cdot \frac{1}{6} + 5 \cdot \frac{1}{6} + 6 \cdot \frac{1}{6} \\ &= 3.5 \end{aligned}$$

```
for (value in that) {  
  this  
}
```

```
while (condition) {  
  code  
}
```

Speed

Vectorized code

logical tests

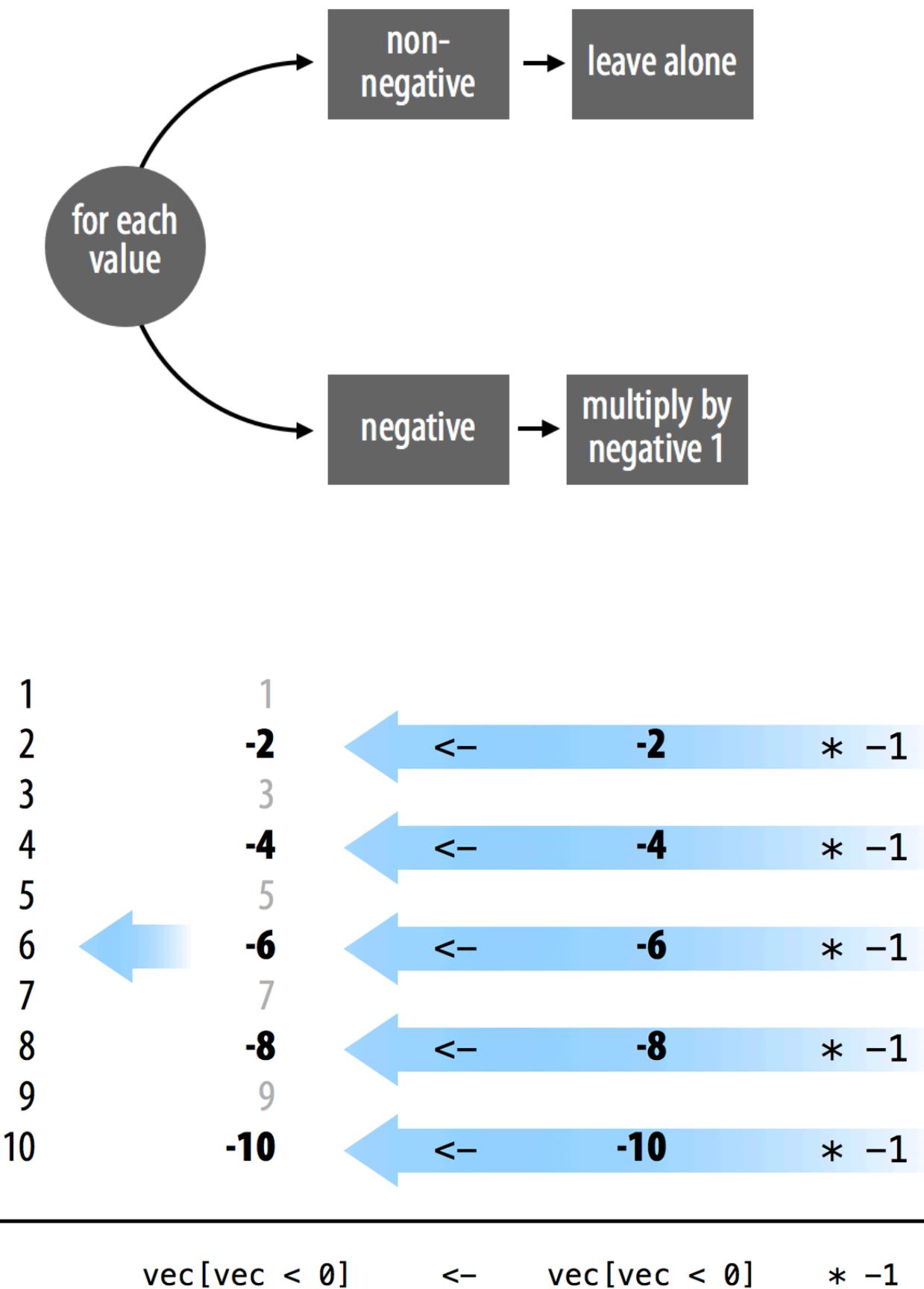
subsetting

element-wise execution

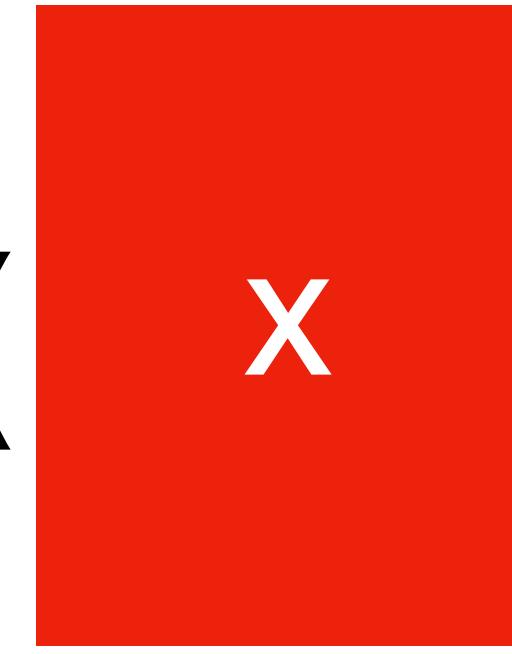
Faster for loops

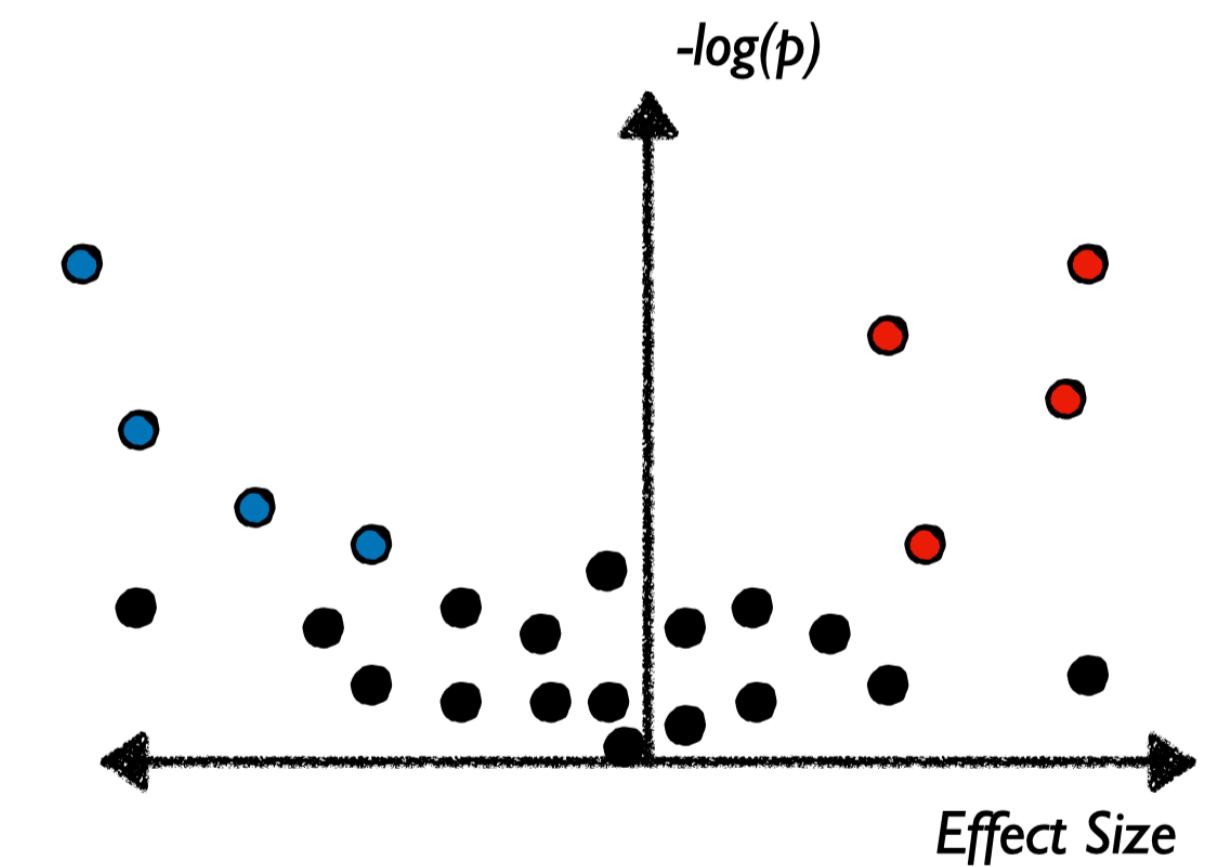
apply() and its friends

Monte Carlo simulations



Practice: Correlations and Volcano Plots

volcano( , ) 



- Are rows matching?
- What if there are missing values in X or y?
- Multiple hypothesis correction — q values (FDR)
- Bonus: Replace y with a matrix Y

$$z = r \sqrt{\frac{n-2}{1-r^2}} \sim t_{n-2}$$

Homework

1. From 2019 HW's: I & 2
2. Review all the code we wrote and Mustafa sent.

Check out chapters 7,9,11, and 12 (optional) from [HOPR](#)

3. Videos from Roger Peng

Data Types: [I](#) / [II](#) / [III](#)

Subsetting: [I](#) / [II](#) / [III](#) / [IV](#) / [V](#)

Control Structures: [I](#) / [II](#) / [III](#) / [IV](#) / [V](#)

