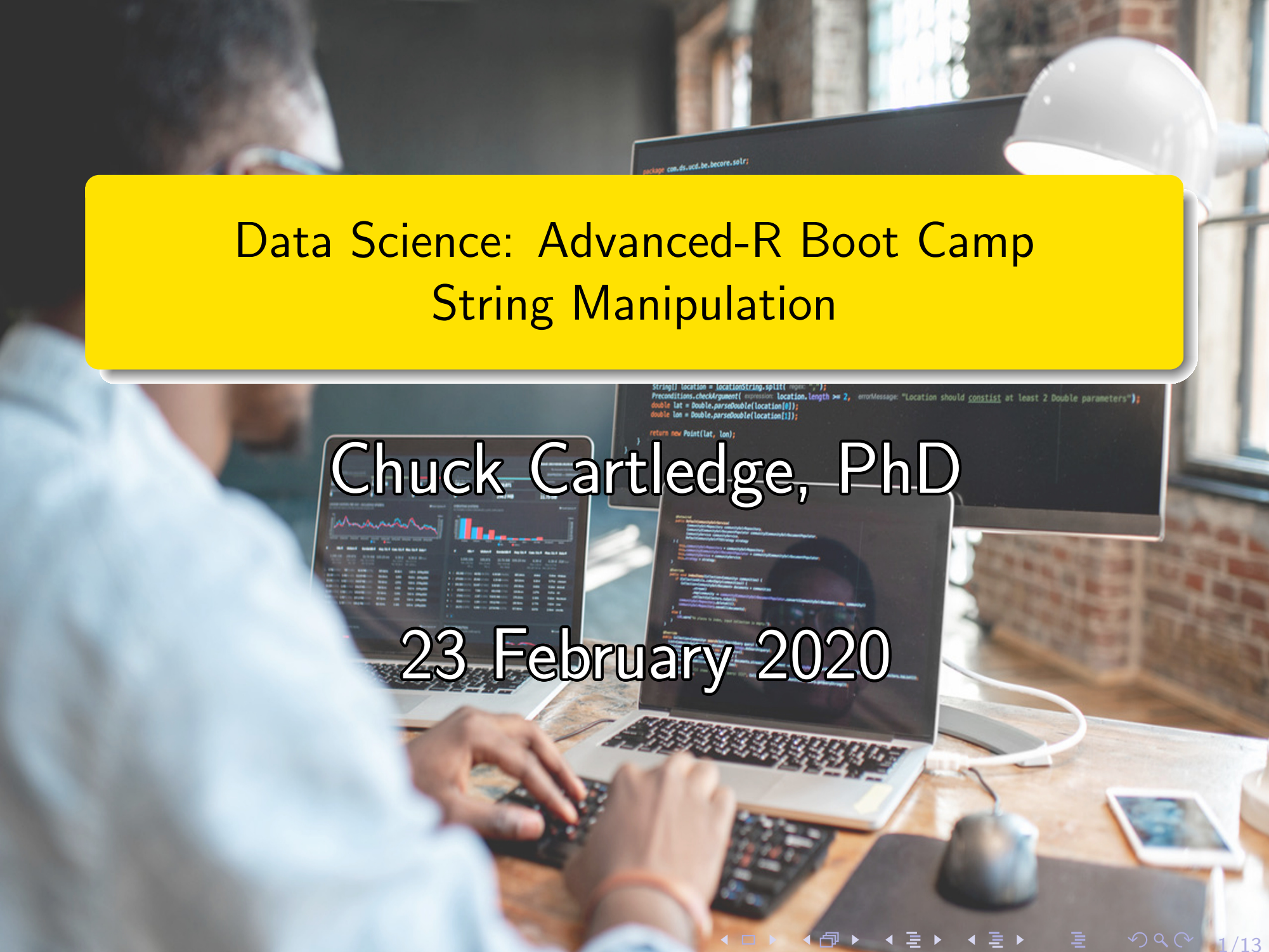


# Data Science: Advanced-R Boot Camp
## String Manipulation

Chuck Cartledge, PhD

23 February 2020

# Table of contents (1 of 1)

# What are we going to cover?

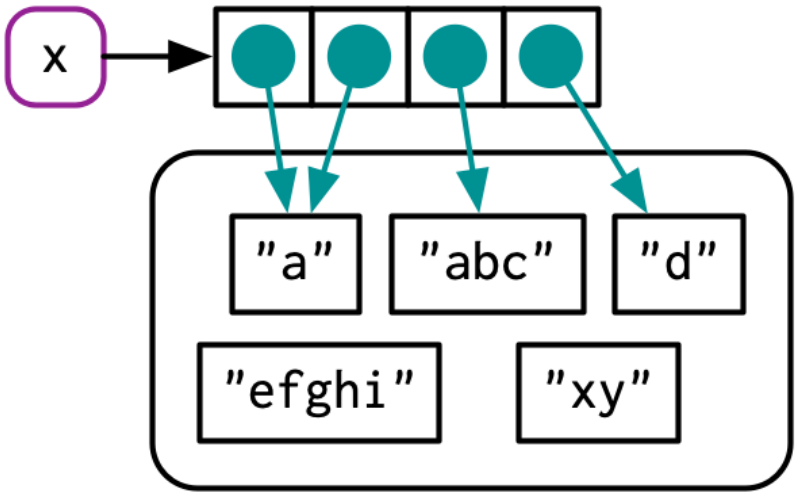
We're going to talk strings (and not string theory)

- How to create them
- How they are stored (and the "ripple" effect that has)
- How to effectively find data in strings

# All strings are part of a "global pool"

- Each collection of characters exists once.
- A collection may be referenced more than once.
- Changes in collection membership causes a new collection to be created.

Examples to follow.




The global string pool

Image from [1].

# Exploring strings

```
library(lobstr)
x <- c(1, 2, 3)
y <- x
obj_addr(x)
obj_addr(y)
y[[3]] <- 4
x
obj_addr(x)
obj_addr(y)
x <- c("a", "a", "abc", "d")
ref(x, character=TRUE)
```

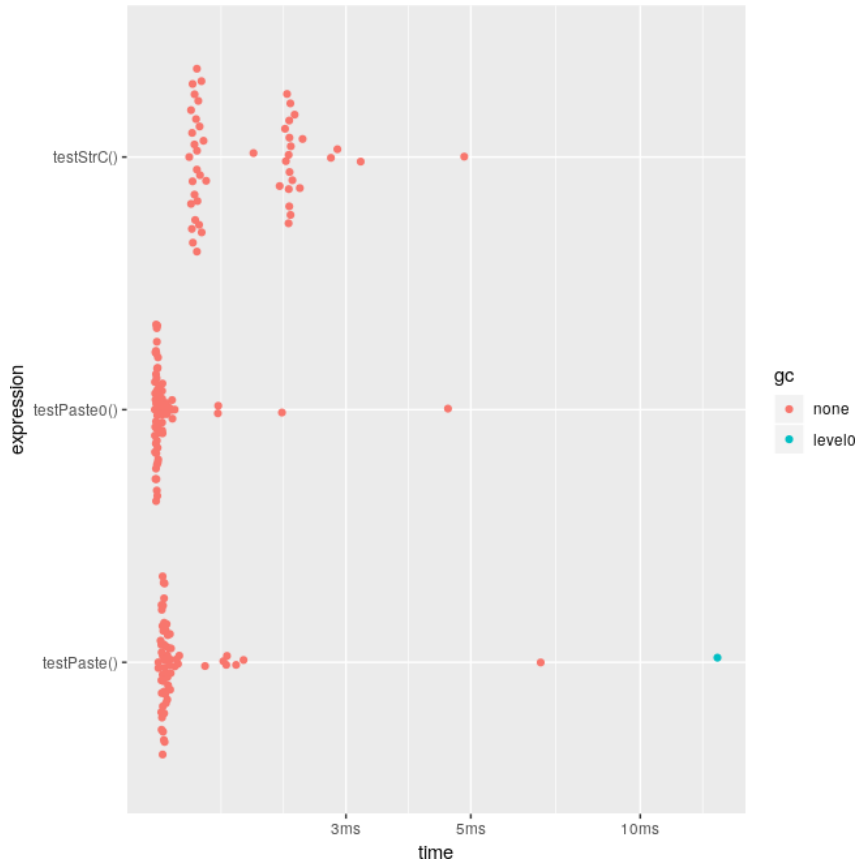
The idea of strings being unique "bubbles" into strings as factors with levels.

# paste() or paste0()

Which is faster/better?

- Based on a series of tests, `paste0()` is about 20% faster.
- Individual executions of either function in single digit micro-second range.
- Plotting routines have unforeseen limitations.
- Library `stringr` has many string handling functions.

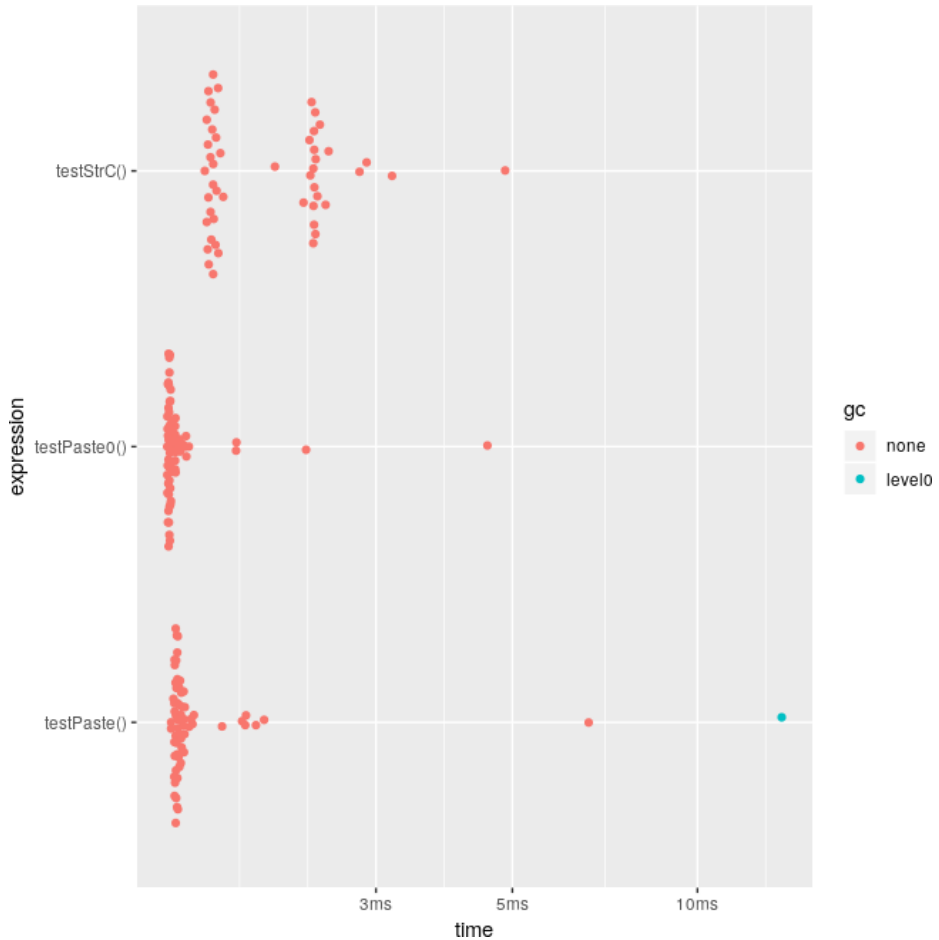
Test code is in the embedded file.




Code in embedded file.

Strings aren't stored like you might think.

# Same image.




Code in embedded file.

# Regular expressions (regexp) are the "work horse" in R.

"grep" comes from "global regexp print," part of the Unix ed program commands: g/re/p.

- regexp is a pattern matching language
- Lines that match the pattern are returned
- Lines are character sequences

```
url <-
   "http://www.gutenberg.org/
    cache/epub/1112/pg1112.txt"
lines <- readLines(url)

grep("rose", lines)
grep("rose", lines, value=TRUE)
grep("rose", lines, value=TRUE,
   ignore.case=TRUE)
grep("rose\\b", lines, value=TRUE,
   ignore.case=TRUE)
grep("rose[s]\\b", lines, value=TRUE,
   ignore.case=TRUE)
```

Books have been written about regexp. It is a powerful tool.

# Looking at Romeo and Juliet

Shakespeare is acknowledged author of "Romeo and Juliet". Many people have questioned that based on textual analysis of the play, when compared to others he may have written.
Use the `tm` library to:

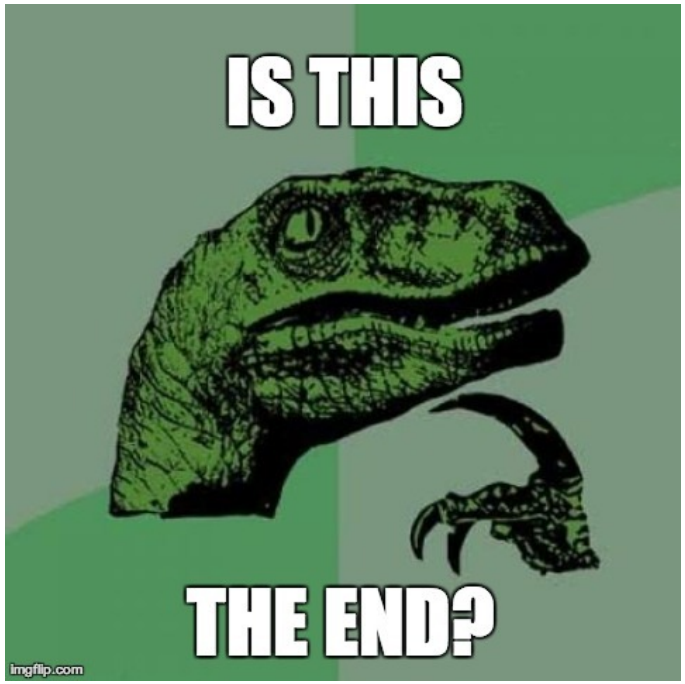
1. Convert all the lines of text into individual words
2. Convert all words to lower case
3. Create a histogram of words sorted from high to low by usage
4. Modify the histogram by removing words that are too common
5. Modify the histogram by "stemming" the words from the previous requirement

# Q & A time.

Q: How did you get into artificial intelligence?
A: Seemed logical – I didn't have any real intelligence.

# What have we covered?

- Looked at how strings are stored
- Looked at different ways to create and modify strings
- Measured the performance of different ways to create strings
- Looked at ways strings could be analyzed

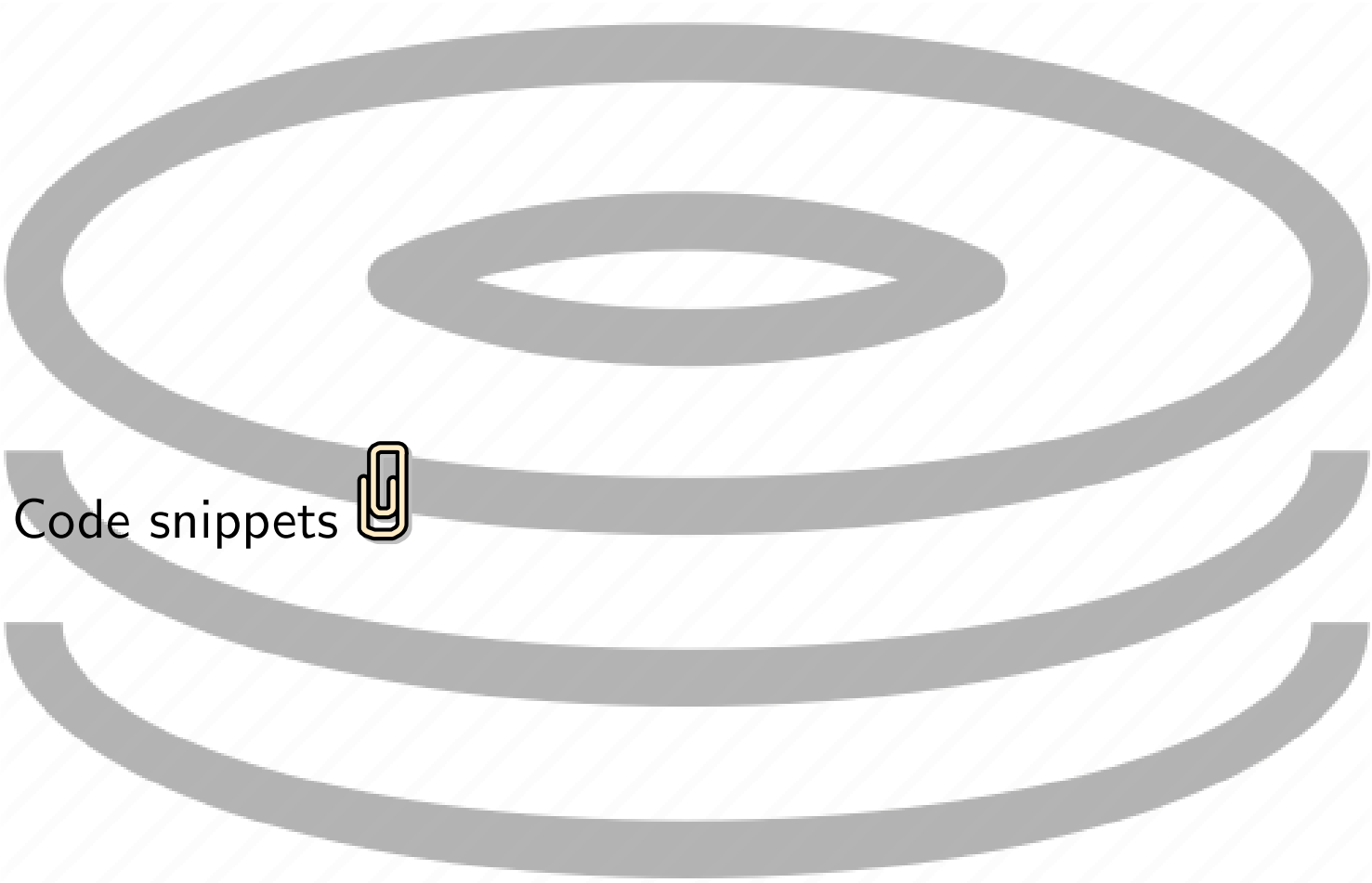



Next: Data insights

# References (1 of 1)

[1] Hadley Wickham, Advanced R, Chapman and Hall/CRC, 2019.

# Files of interest



1. Code snippets