DATA SCIENCE SEMINAR SERIES
SPRING, 2016

WEEK 01

DATA MANAGEMENT AND MANIPULATION IN R

INTO TO R AND SCRIPTING

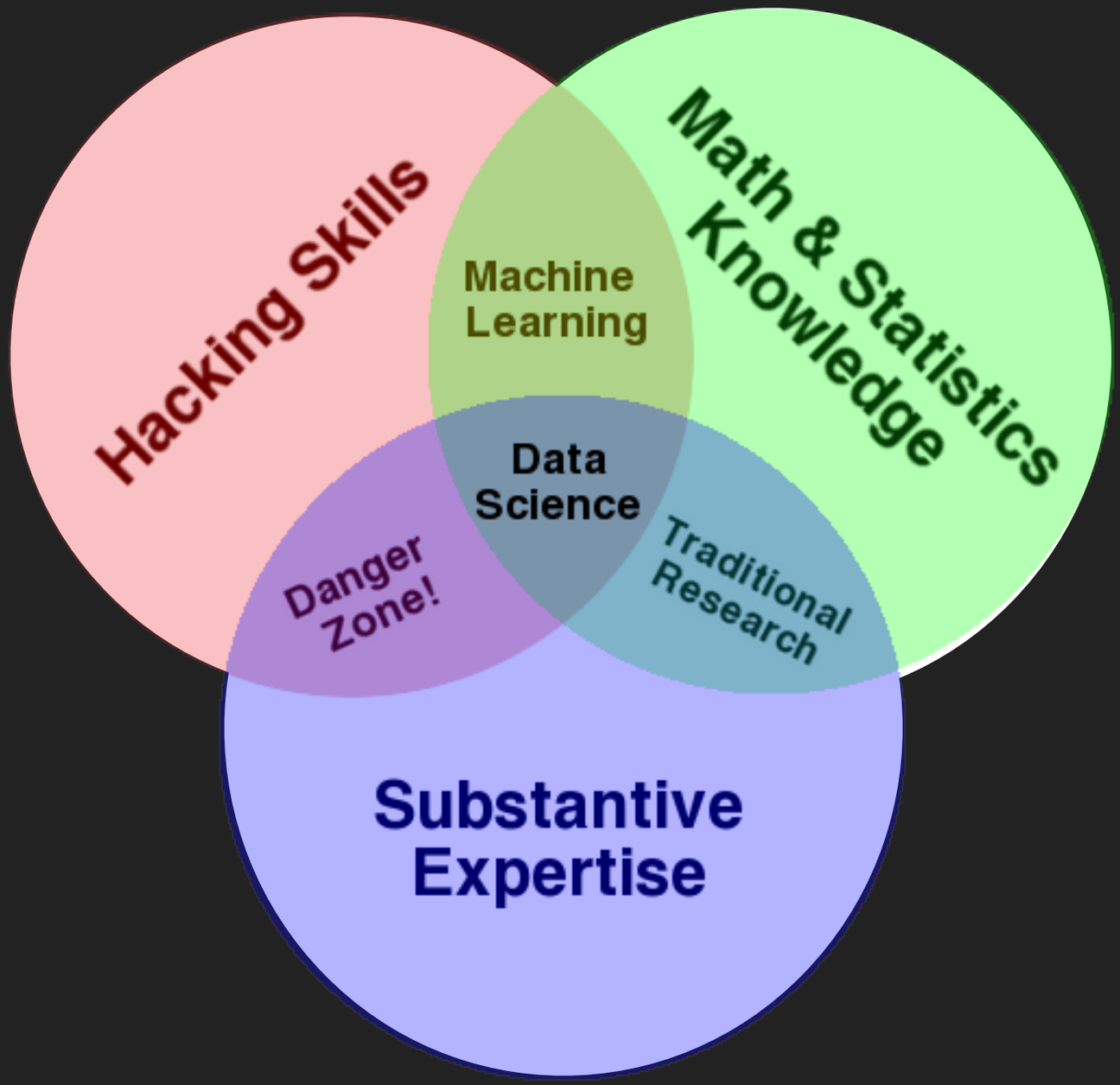
AGENDA

1. Introductions
2. Seminar Overview
3. Introduction to `R`
4. Scripting for `R`

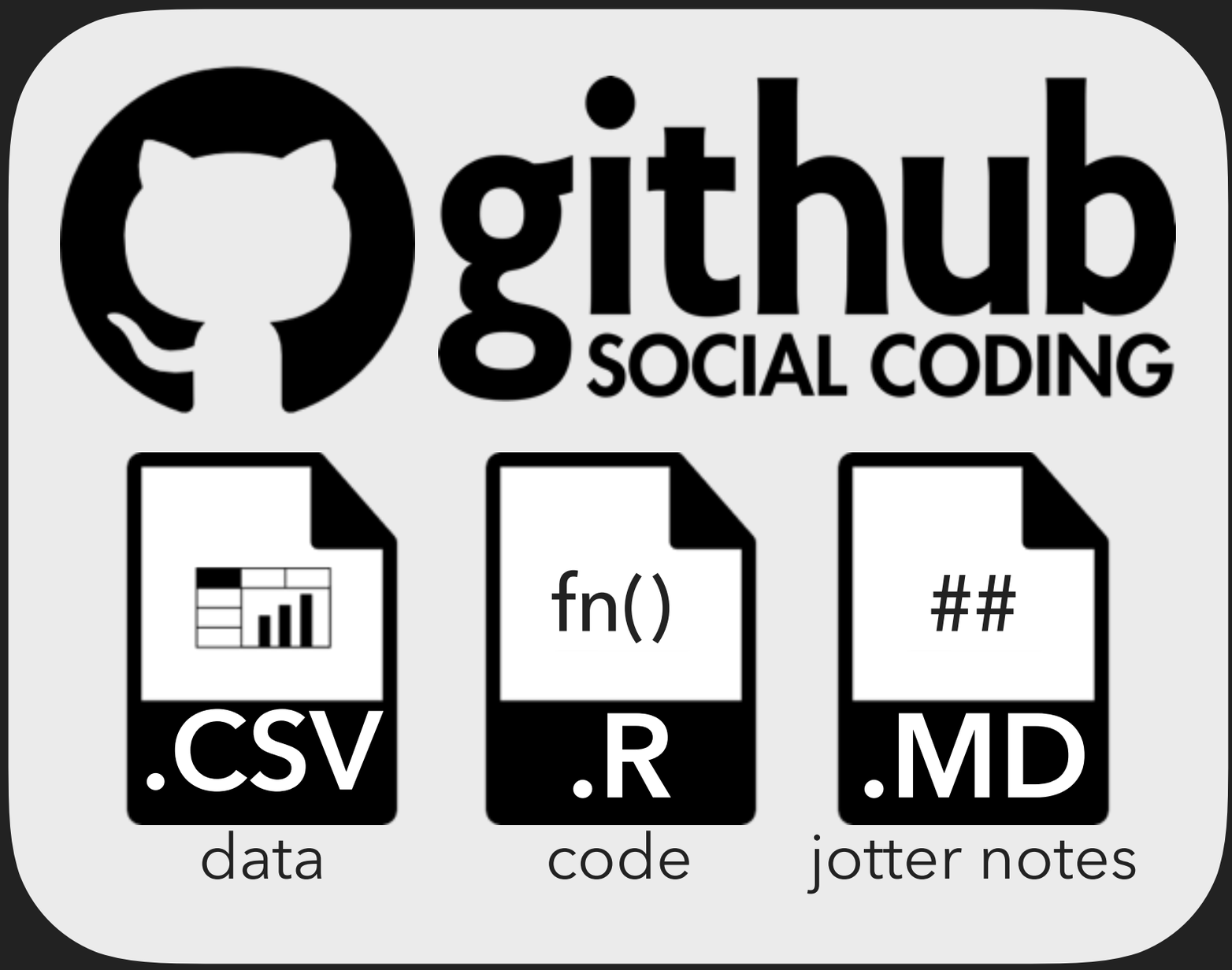
1 INTRODUCTIONS

2 SEMINAR OVERVIEW

WHAT IS DATA SCIENCE?

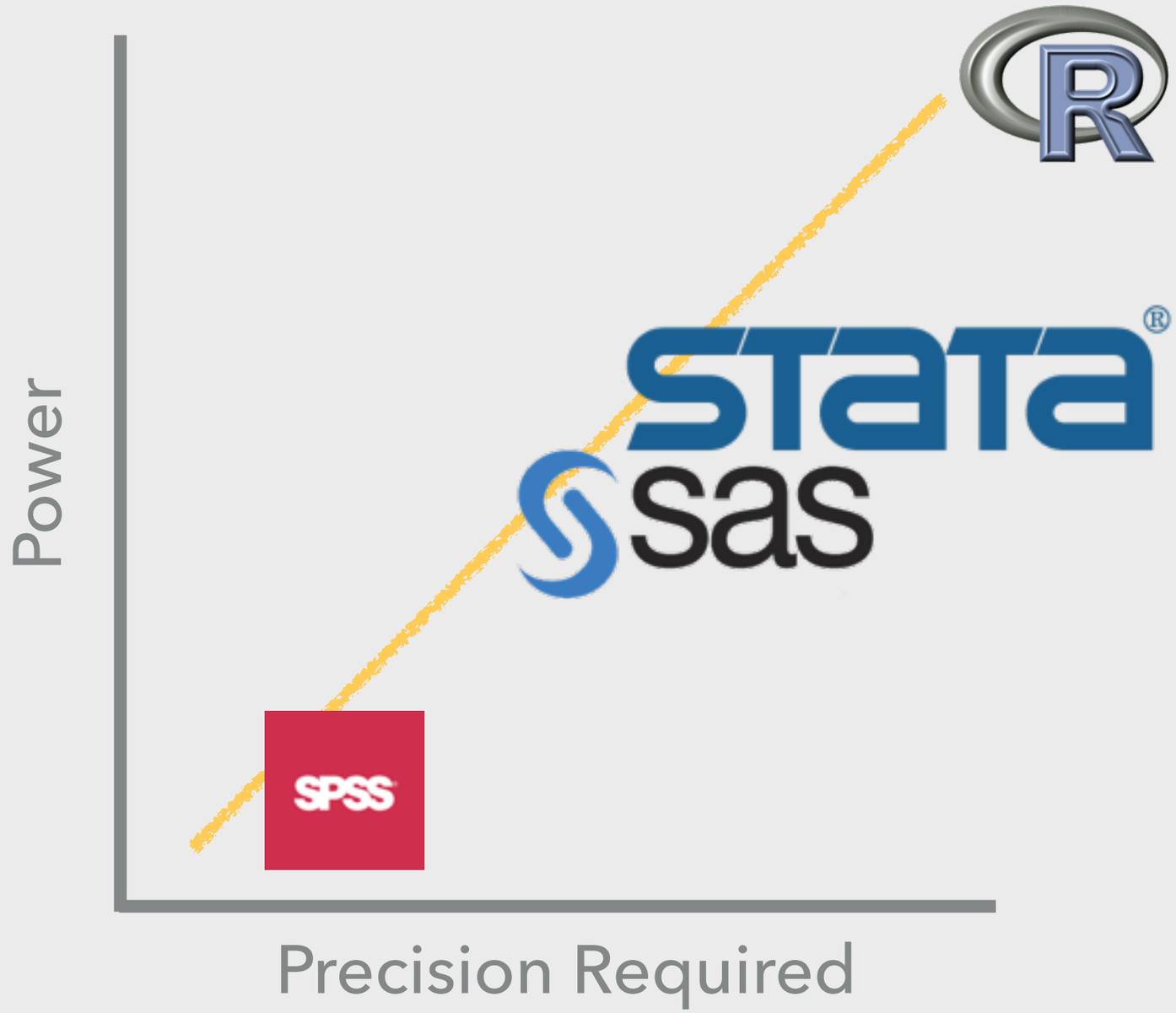


SEMINAR OVERVIEW



3 INTRO TO R

DEMANDS OF STATS PACKAGES

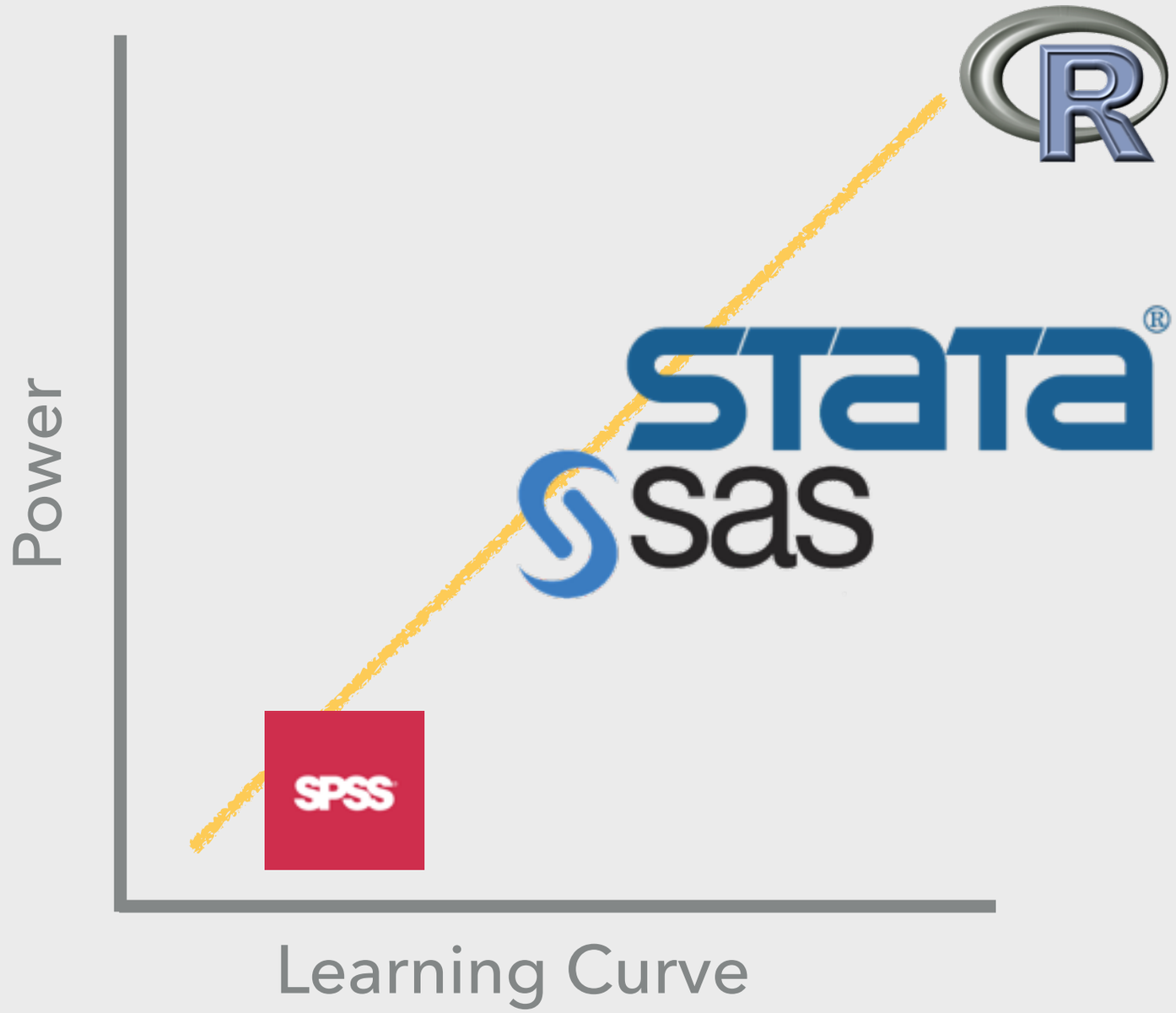


IMPLICIT CONTRACT WITH THE COMPUTER / SCRIPTING LANGUAGE: COMPUTER WILL DO TEDIOUS COMPUTATION FOR YOU. IN RETURN, YOU WILL BE COMPLETELY PRECISE IN YOUR INSTRUCTIONS. TYPOS MATTER. CASE MATTERS. GET BETTER AT TYPING.

Jenny Bryan, PhD

University of British Columbia Stats/Data Science Prof

DEMANDS OF STATS PACKAGES



BASIC STRUCTURE OF STATS PACKAGES

observations	var1	var2	var3
1	a	1	0
2	b	2	1
...	a	4	1
k	b	3	0

BASIC STRUCTURE OF R

THE VECTOR

```
> [1] 1 2 3 4
> [2] "chris" "christy" "kelly"
```

BASIC STRUCTURE OF R

ATOMIC VECTOR

```
> [1] 1 2 3 4
```

NUMERIC / DOUBLE / INTEGER*

```
> [2] “chris” “christy” “kelly”
```

CHARACTER

```
> [3] 0 1 0 1
```

LOGICAL

```
> [4] a b b a
> Levels: a b
```

FACTOR

BASIC STRUCTURE OF R

LISTS

```
> $ : int [1:4] 1 2 3 4
> $ : chr “chris” “christy” “kelly”
> $ : logi [1:4] 0 1 0 1
> $ : Factor w/ 2 levels "a","b": 1 2 2 1
```

BASIC STRUCTURE OF R

DATA FRAME

```
> $ : int [1:4] 1 2 3 4
> $ : chr “chris” “christy” “kelly” “mark”
> $ : logi [1:4] 0 1 0 1
> $ : Factor w/ 2 levels "a","b": 1 2 2 1
```

4 SCRIPTING FOR R

LET US CHANGE OUR TRADITIONAL ATTITUDE TO THE CONSTRUCTION OF PROGRAMS: INSTEAD OF IMAGINING THAT OUR MAIN TASK IS TO INSTRUCT A COMPUTER WHAT TO DO, LET US CONCENTRATE RATHER ON EXPLAINING TO HUMANS WHAT WE WANT THE COMPUTER TO DO.

Donald E. Knuth

Stanford University Computer Scientist

NAMING CONVENTIONS COUNT

- Be consistent with how you name objects
- Names should be intuitive
- A vector representing gender should ideally be named ‘gender’, not z4
- Names should be formatted consistently:

```
some_people_use_snake_case

other.people.use.periods

iUseCamelCase
```

READABILITY COUNTS

```
# This is a comment. Comments are highlighted green. There are no

# multi-line comments. If you want to continue writing a comment

# you need a new number sign. Use comments extensively!
```

```
# These are examples of dividers:

# ==============================================================

# ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

# Dividers make your code easier to read. Use them!
```

DOCUMENTATION COUNTS

```
# ============================================================

# DATA SCIENCE SEMINAR, SPRING 2016, WEEK 01

# ============================================================

# opening options

rm(list = ls()) # clear workspace

week <- 1 # define week number

# set working directory based on operating system

if (Sys.info()["sysname"] == "Darwin") {

  setwd(paste(Sys.getenv("HOME"),'/Desktop',sep=""))

} else if (Sys.info()["sysname"] == "Windows") {
```

THE ZEN OF R

Beautiful is better than ugly.

Explicit is better than implicit.

Simple is better than complex.

Complex is better than complicated.

Readability counts.

In the face of ambiguity, refuse the temptation to guess.

BORROWED FROM 'THE ZEN OF PYTHON'

The Zen of Python

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than right now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
—Tim Peters