Live Session Week 1

Data Science Workflows

Dr Zak Varty

Outline

1. Review
2. Discussion
3. Break
4. Minimal R Package

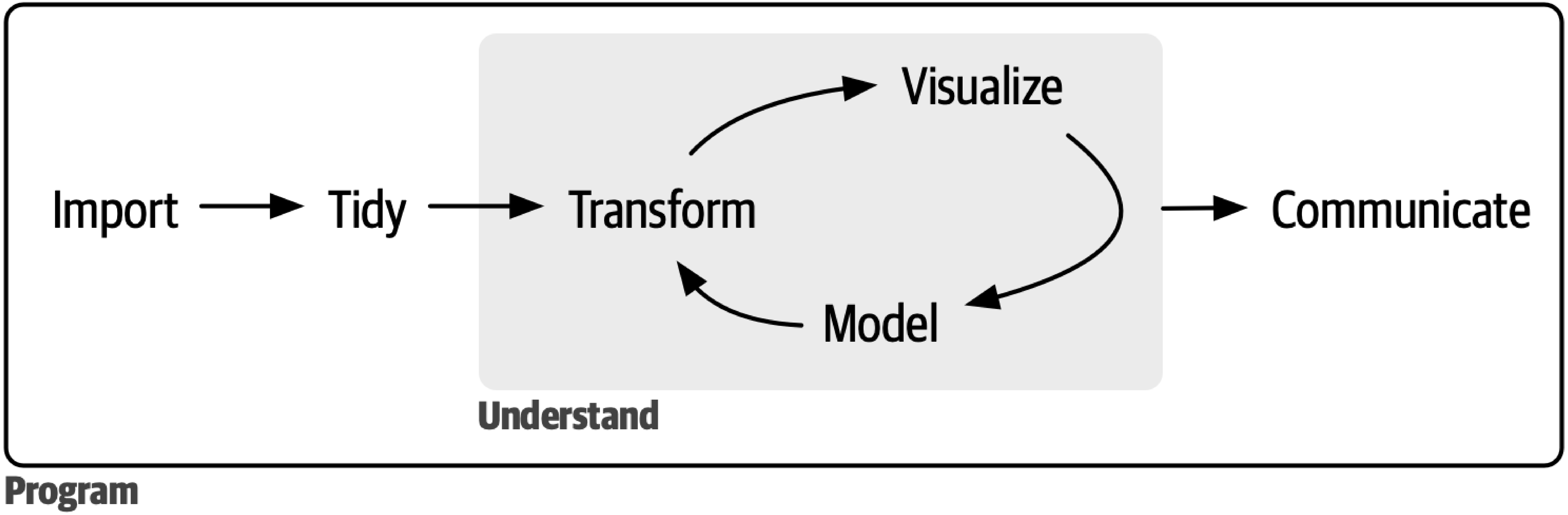
Why are you here?

(Or more specifically, what do you want to get out of this course?)

`https://www.menti.com/alauvs7o`

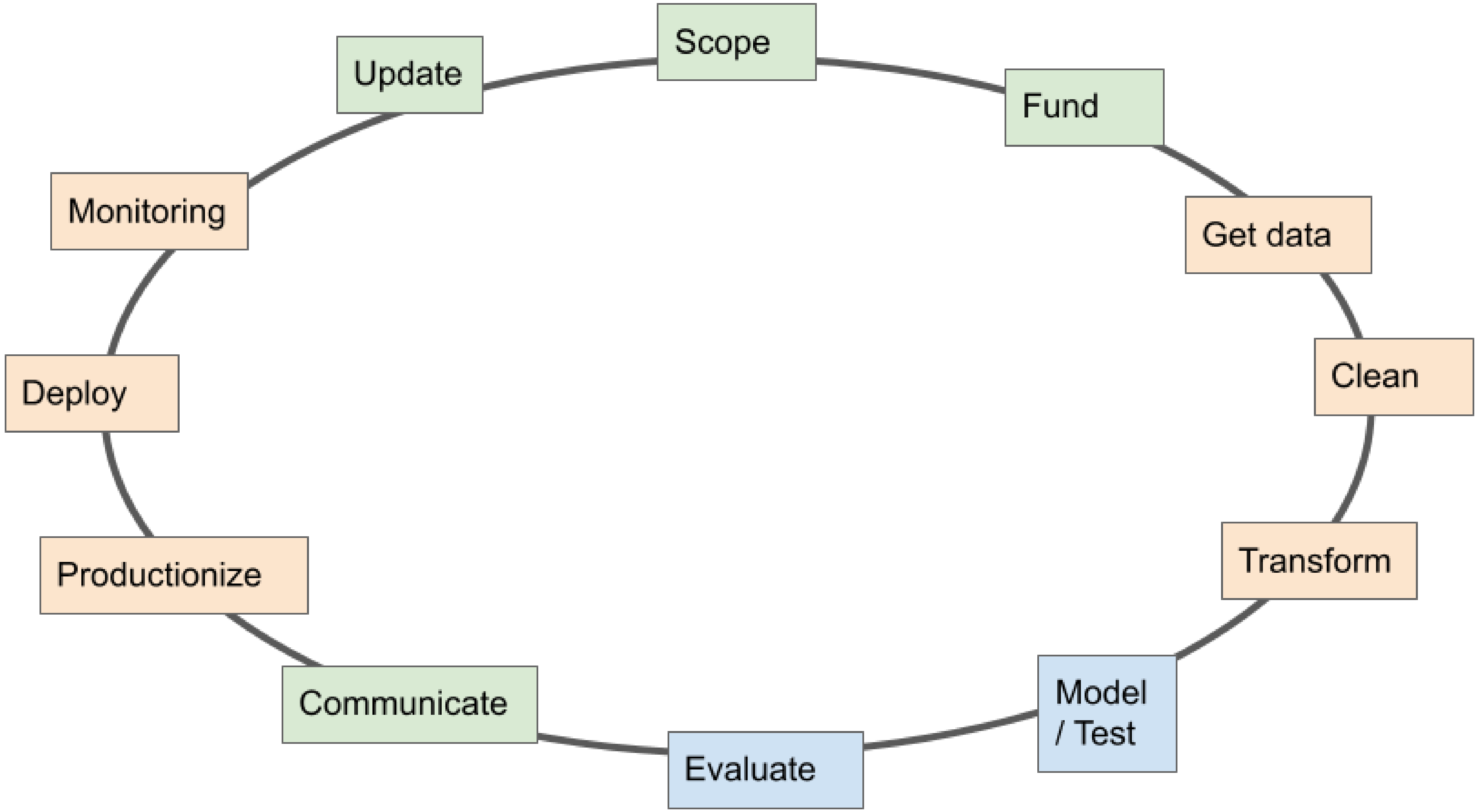
Review

Life Cycle of A Data Science Project

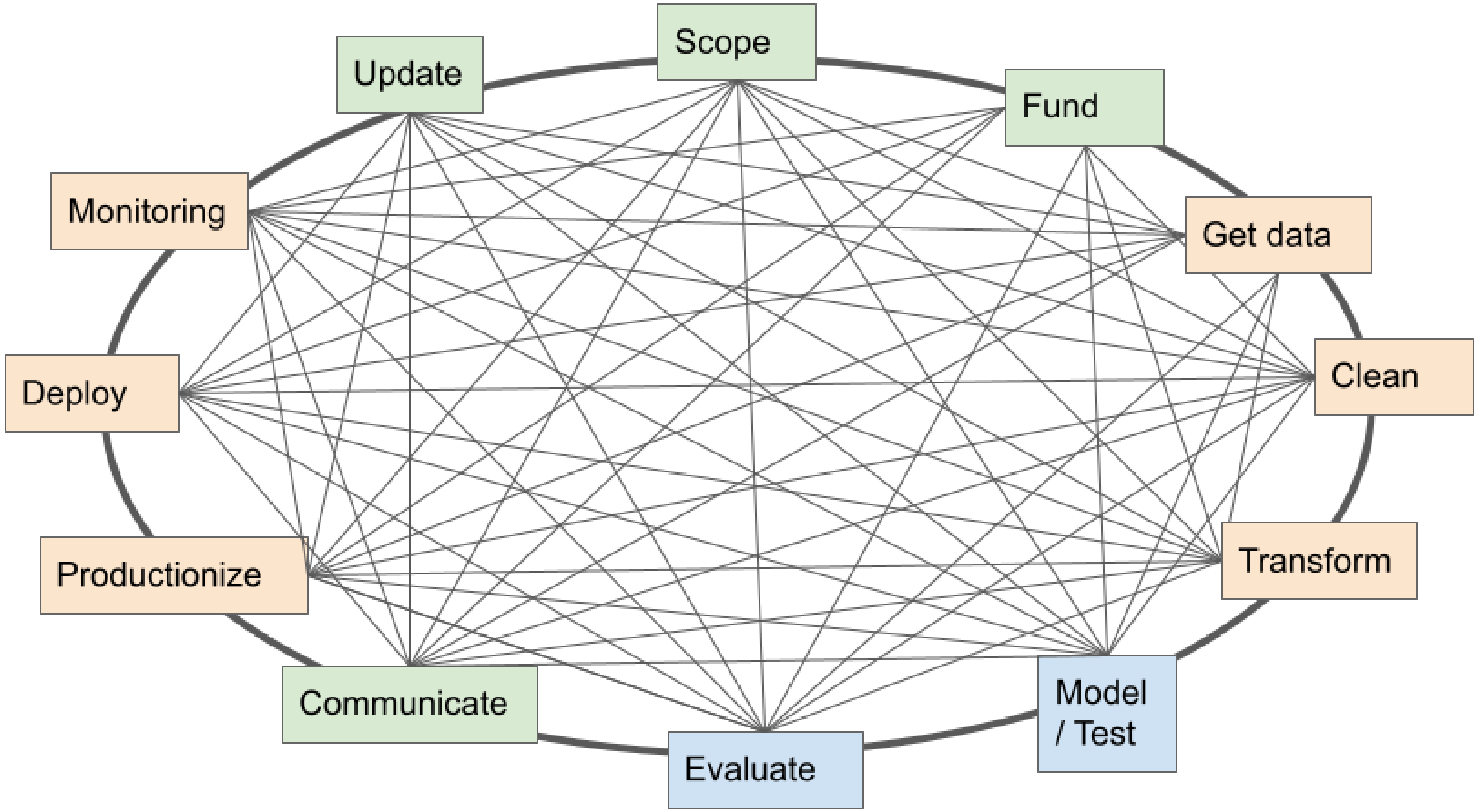


source: R4DS

Life Cycle of A Data Science Project



Life Cycle of A Data Science Project



Task Discussion

Review - Organising your work

Project management spans all aspects of the data science cycle.

- **Organising your directories:**
  - Benefits of a common directory structure
  - Example project structure
- **Organising your files:**
  - Name with humans, computers and ordering in mind.
  - Pick file types with care.
- **Organising your code:**
  - Functional vs Object Oriented Programming
  - Naming conventions and style guide (verb & nouns)

Review: Task 1 - Github Show and Tell

> **I asked you to:** Find 3 data science projects on Github and explore how they organise their work.

In groups of 2-3, each pitch one of the projects you found.

- What does the project do?
- Who made it?
- Why do you think it is interesting?
- What did you learn by looking at it?

05:00

Review: Task 2 - Make Project Templates

> **I asked you to:** Create your own project directory (or directories) for this course and its assignments.

`https://www.menti.com/alrej2iedgtr`

Find a person you haven’t spoken to today and explain your reasoning.

05:00

Review: Task 3 - Function writing

> **I asked you to:** write a function to calculate the rolling arithmetic mean of a numeric vector.

Pair up with a third person that you haven’t spoken to yet today.

Discuss your thought process and compare code.

- What did you have to consider when writing this function?
- Report back: One decision that you made differently **or** a decision that you made the same, but implemented differently.

12:00

My thought process (1/n)

```r
rolling_mean <- function(x){}
```

- Do I pick window length or does the user? (User)
  - What values should I allow? (Integers > 1.)

- Gah, need two inputs: `x` = Vector to smooth, `window_length` = # obs in rolling window. Are these good names?

```r
rolling_mean <- function(x, window_length){}
```

My thought process (2/n)

- Is the window centred, left-aligned or right aligned?
  - centred for smoothing , right for prediction
  - would I ever want left-aligned?

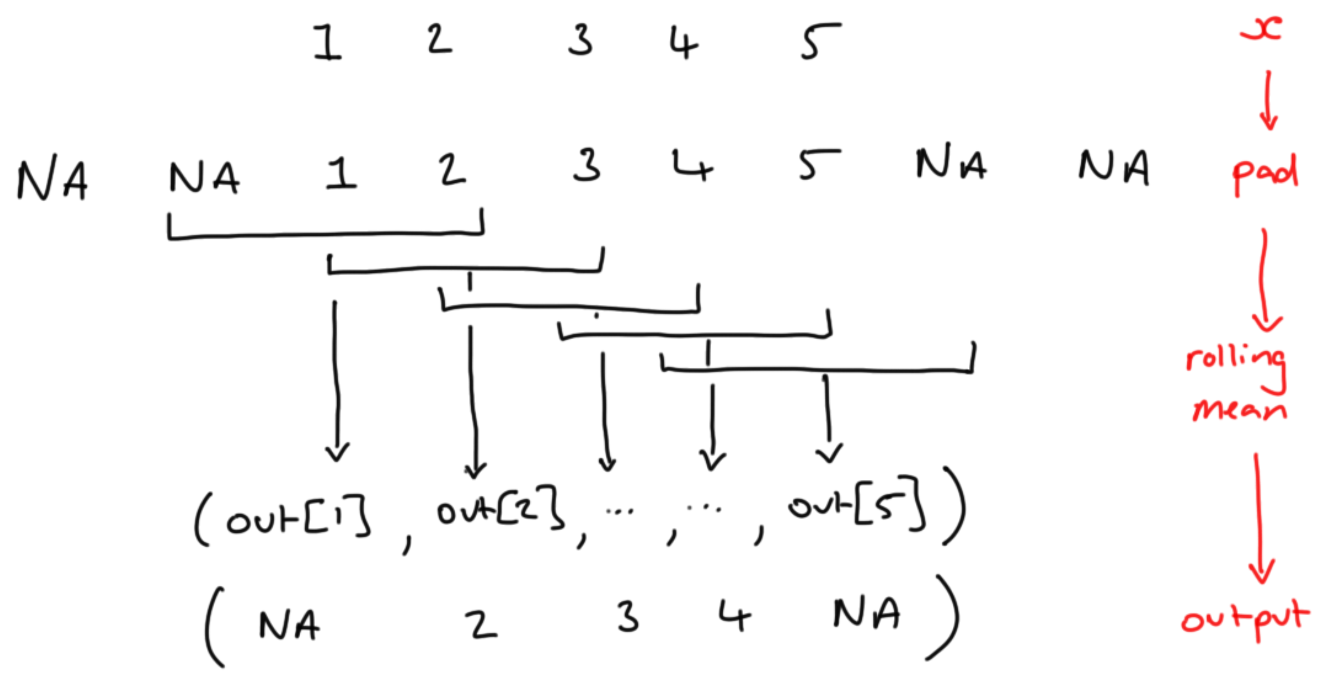
```
rolling_mean <- function(x, window_length, window_position = "centre"){}
```

- `window_position` as an argument should this be three separate functions?
  - One function should do one job well. Focus on centred.
  - Can always make a wrapper function later.

```
rolling_mean <- function(x, window_length){}
```

My thought process (3/n)

- What do I do at the edges?
  - Miss them and return a shorter vector
  - Average fewer terms
  - Pad with NAs ✅



Let the worrying commence (4/n)

- What if `window_length` is even?
  - Stupid. Left and right would have worked. Regret.
  - Do I want to return a data frame? Something else?
  - Limit to non-negative odd numbers.
  - How do I do that? `is.integer()`? Nope.

Let the worrying commence (5/n)

- How could this go wrong?
  - Could give a non-vector for `x`
  - Could give a non-numerical vector for `x`
  - Could give a vector of values for `window_width`
  - What happens if they give nothing or `NULL`?
  - What else have I missed …

The final code

```r
rolling_mean <- function(x, window_width, ...){
  # -----Input Checks ----------------------------------------------------------
  # Check that x is a vector with numerical interpretation
  stopifnot(is.logical(x) | is.integer(x) | is.double(x) | is.complex(x))
  stopifnot(length(x) > 0)

  # Check window_width is an odd, positive integer
  stopifnot(length(window_width) == 1)
  stopifnot(window_width %% 1 == 0)
  stopifnot((window_width / 2) %% 1 != 0)
  stopifnot(window_width > 0)

  # ----- Function Body --------------------------------------------------------

  # number of values left and right to include in each mean
  half_width <- floor(window_width / 2)
  x_padded <- pad_with_NAs(x, n_left = half_width, n_right = half_width)
  evaluation_locations <- seq_along(x) + half_width

  output <- rep(NA, length(x))
```

My thought process: documenting (5/n)

```r
#' Calculate the rolling mean of a vector
#'
#' @param x Vector of values that can be interpreted numerically.
#' @param window_width The number of values included in each mean calculation. Should be an odd, positive integer.
#' @param ... Additional arguments to pass to the mean() function call.
#'
#' @return A vector of rolling mean values of the same length as `x`.
#' @export
#'
#' @examples
#'
#' rolling_mean(x = 1:5, window_width = 3)
#' rolling_mean(x = 1:5, window_width = 5)
#' rolling_mean(x = 1:5, window_width = 7)
#' rolling_mean(x = c(TRUE, TRUE, TRUE, FALSE, TRUE, TRUE, TRUE), window_width = 3)
#'
rolling_mean <- function(x, window_width, ...){}
```

BREAK

05:00

Minimal R project in 1 hour (or less)