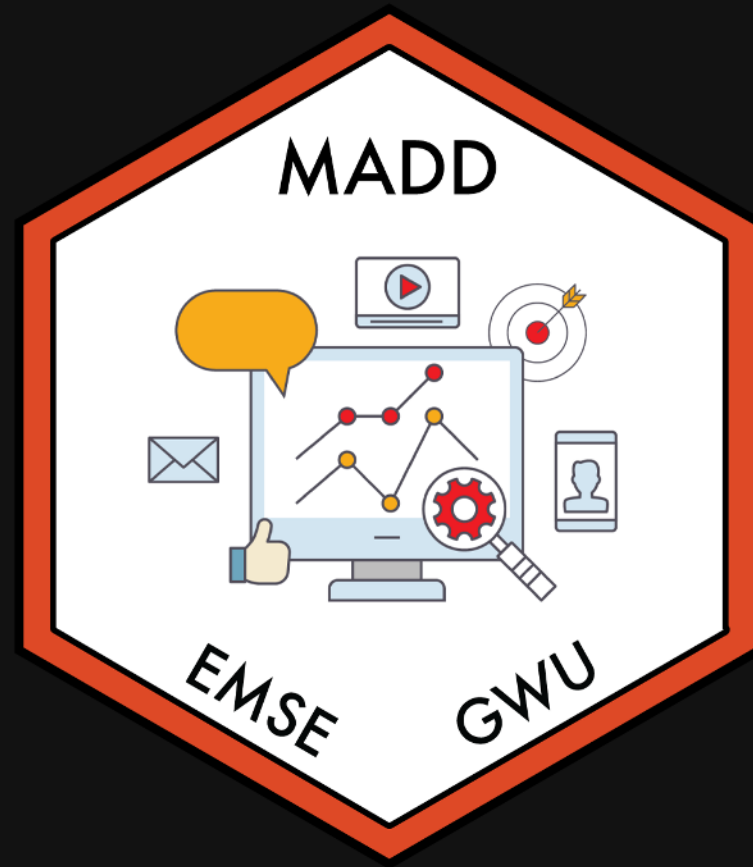




# *Optimization & ML*

🏛️ EMSE 6035: Marketing Analytics for Design Decisions

👤 John Paul Helveston

📅 October 21, 2026

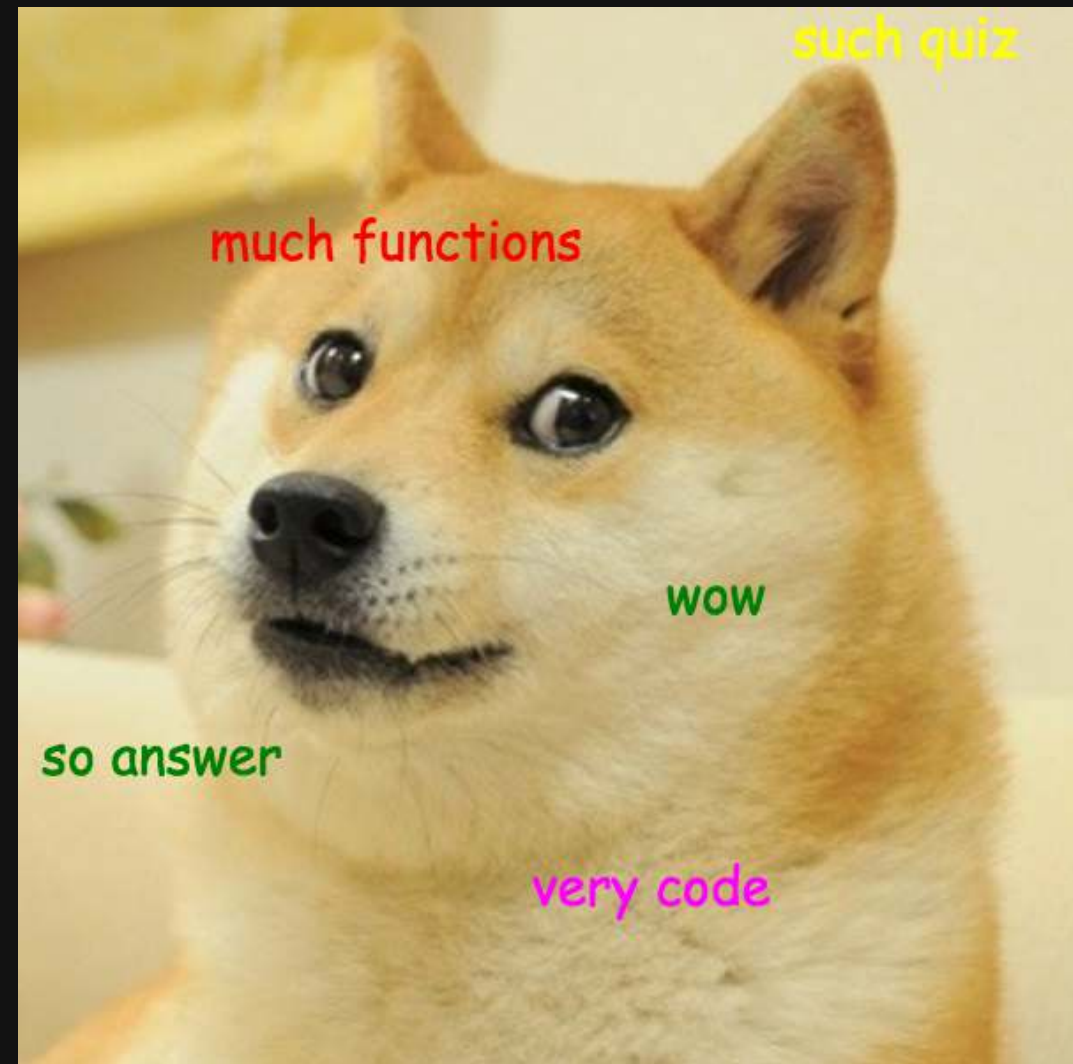
# Quiz 3

10:00

Write your name on the quiz!

## Rules:

- Work alone; no outside help of any kind is allowed.
- No calculators, no notes, no books, no computers, no phones.



# Week 9: *Optimization & MLE*

1. Maximum likelihood estimation
2. Optimization (in general)
3. Joins
4. Pilot data cleaning

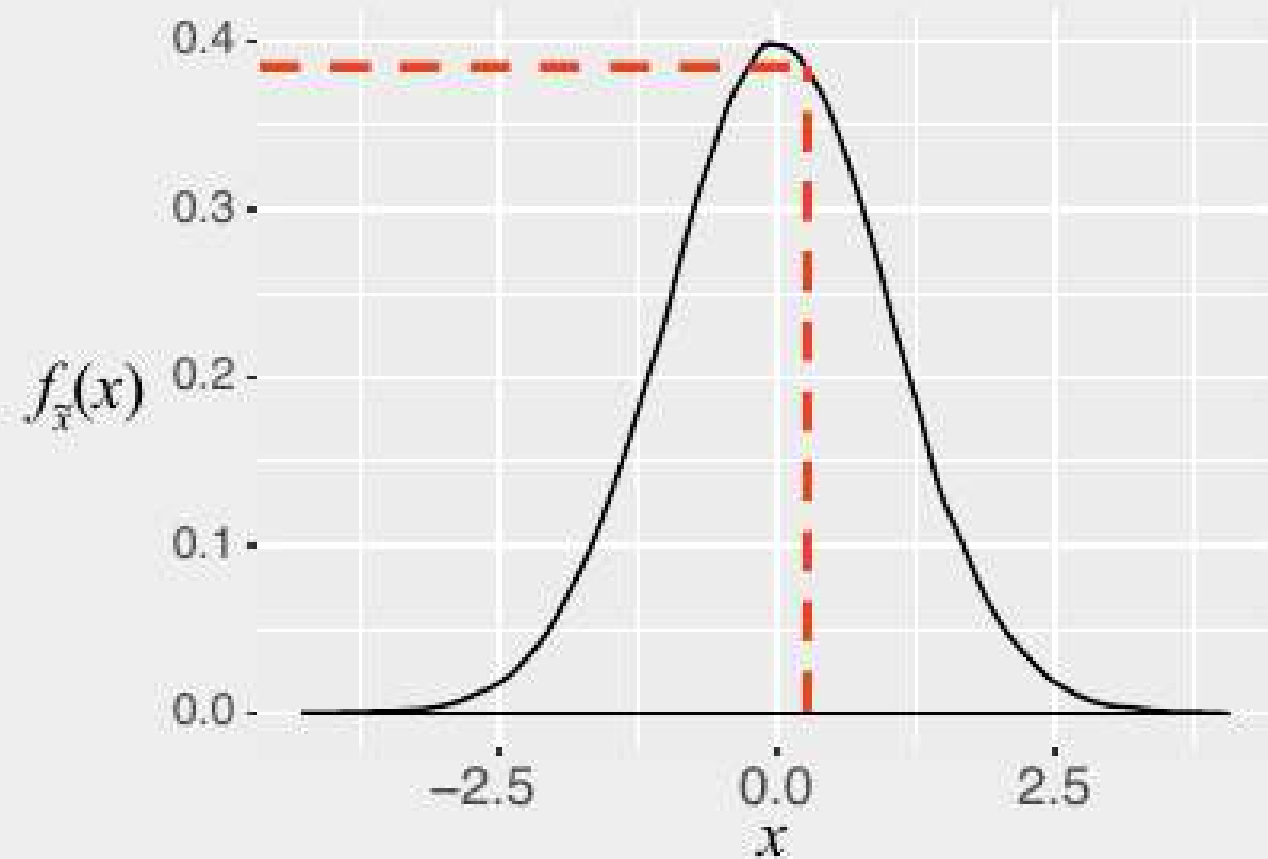
# Week 9: *Optimization & MLE*

1. Maximum likelihood estimation
2. Optimization (in general)
3. Joins
4. Pilot data cleaning

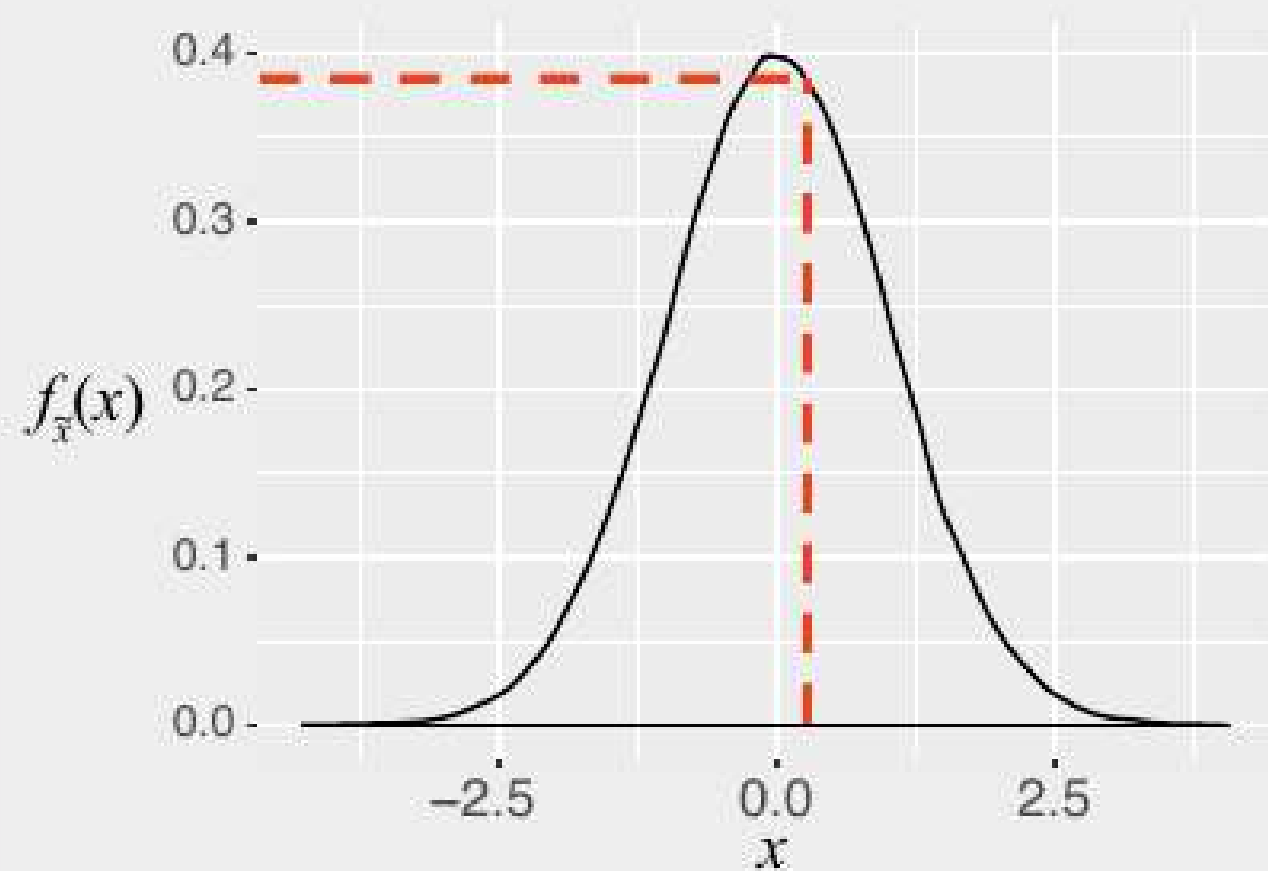
# Computing the likelihood

$x$ : an observation

$f(x)$ : probability of observing  $x$



# Computing the likelihood



$x$ : an observation

$f(x)$ : probability of observing  $x$

$\mathcal{L}(\theta|x)$ : probability that  $\theta$  are the true parameters, given that observed  $x$

**We want to estimate  $\theta$**

We actually compute the *log*-likelihood  
(converts multiplication to addition)

|      |      |      |      |      |      |      |      |      |      |
|------|------|------|------|------|------|------|------|------|------|
| 0.39 | 0.35 | 0.24 | 0.39 | 0.40 | 0.11 | 0.33 | 0.35 | 0.07 | 0.37 |
|------|------|------|------|------|------|------|------|------|------|

$$\mathcal{L}(\boldsymbol{\theta}|\mathbf{x}) = f_{\tilde{x}}(x_1) f_{\tilde{x}}(x_2) \dots f_{\tilde{x}}(x_n) = 1.63\text{e-}6$$

$$\log \mathcal{L}(\boldsymbol{\theta}|\mathbf{x}) = f_{\tilde{x}}(x_1) + f_{\tilde{x}}(x_2) + \dots + f_{\tilde{x}}(x_n) = 3$$

# Practice Question 1

**Observations** - Height of students (inches):

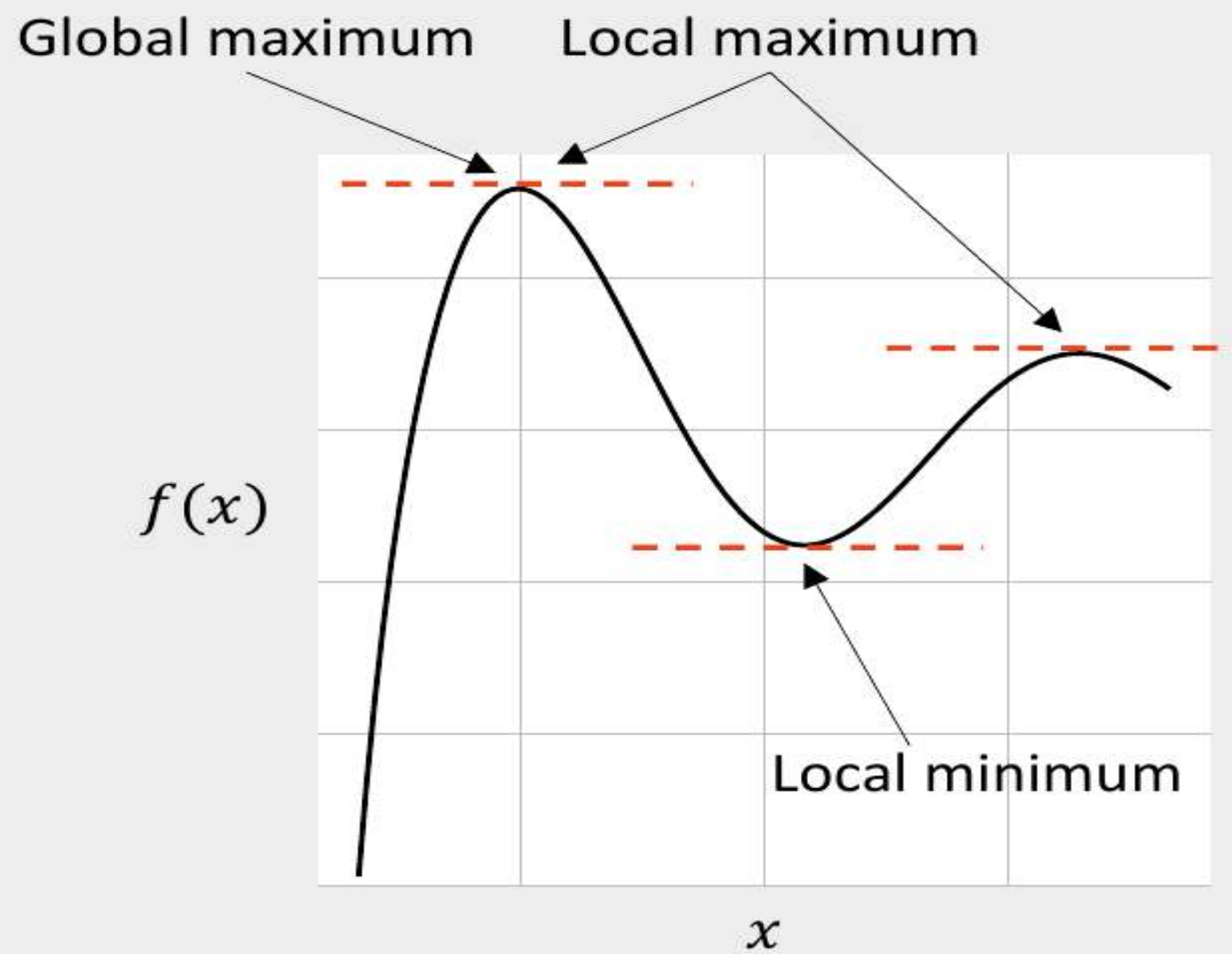
```
#> [1] 65 69 66 67 68 72 68 69 63 70
```

- Let's say we know that the height of students,  $\tilde{x}$ , in a classroom follows a normal distribution. A professor obtains the above height measurements students in her classroom. What is the log-likelihood that  $\tilde{x} \sim \mathcal{N}(68, 4)$ ? In other words, compute  $\ln \mathcal{L}(\mu = 68, \sigma = 4)$ .
- Compute the log-likelihood function using the same standard deviation ( $\sigma = 4$ ) but with the following different values for the mean,  $\mu$  : 66, 67, 68, 69, 70. How do the results compare? Which value for  $\mu$  produces the highest log-likelihood?

# Week 9: *Optimization & MLE*

1. Maximum likelihood estimation
2. Optimization (in general)
3. Joins
4. Pilot data cleaning

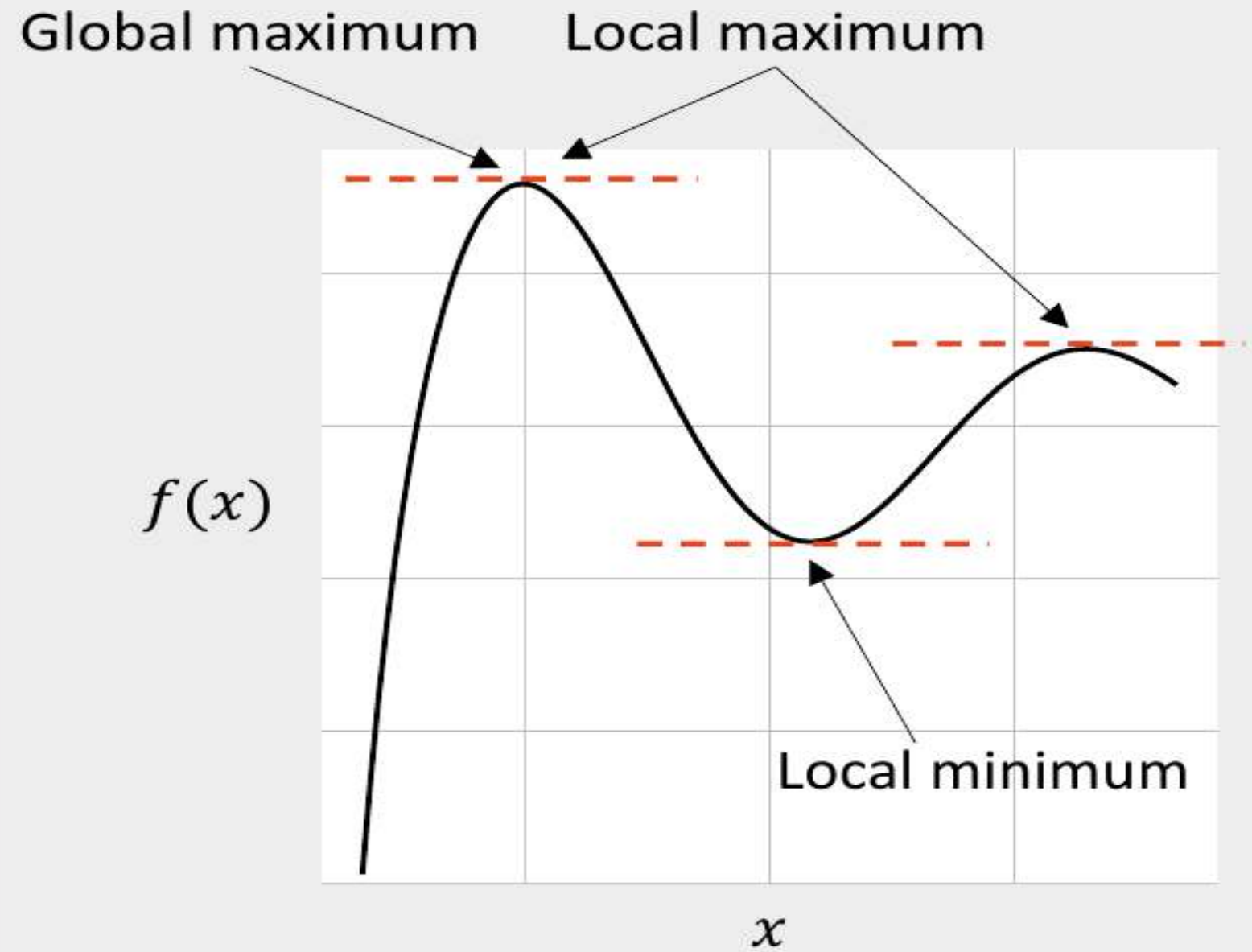
$$f(x)$$



First order necessary condition

$x^*$  is a “stationary point” when

$$\frac{df(x^*)}{dx} = 0$$



First order necessary condition

$x^*$  is a “stationary point” when

$$\frac{df(x^*)}{dx} = 0$$

Second order sufficiency condition

$x^*$  is a local *maximum* when

$$\frac{d^2f(x^*)}{dx^2} < 0$$

$x^*$  is a local *minimum* when

$$\frac{d^2f(x^*)}{dx^2} > 0$$

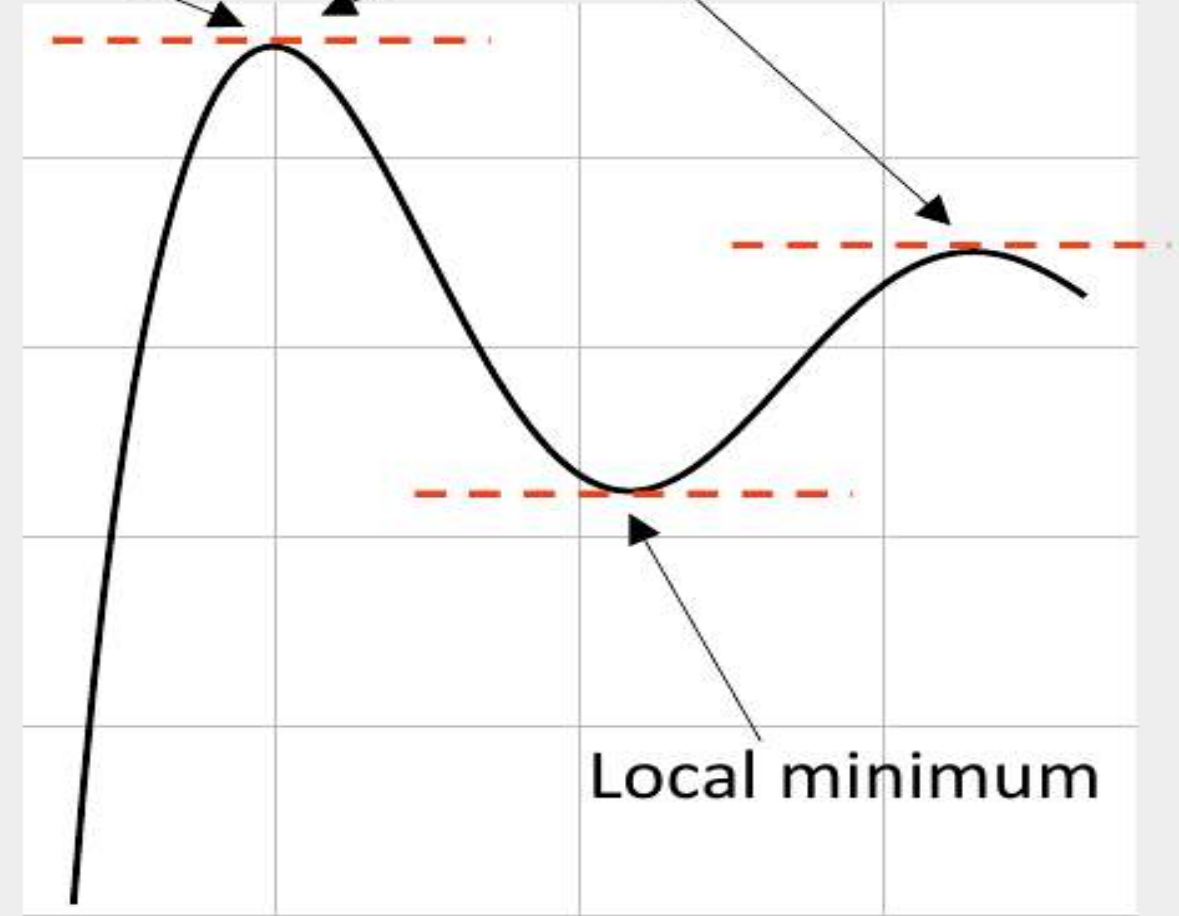
Global maximum

Local maximum

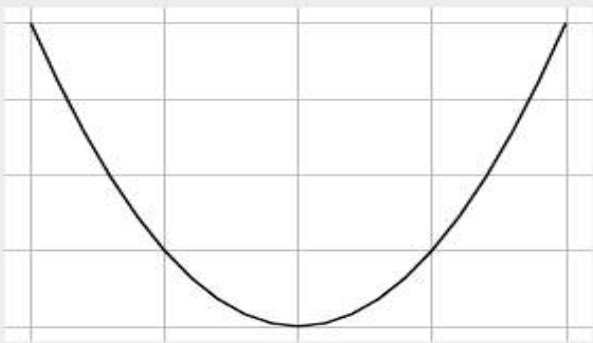
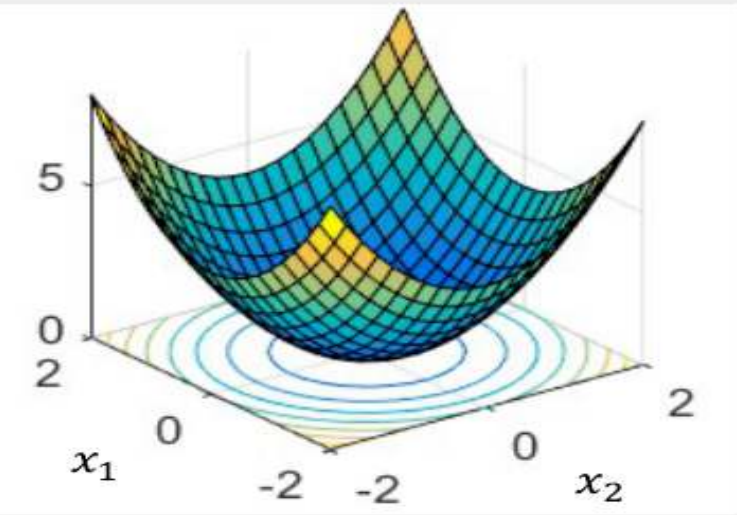
$f(x)$

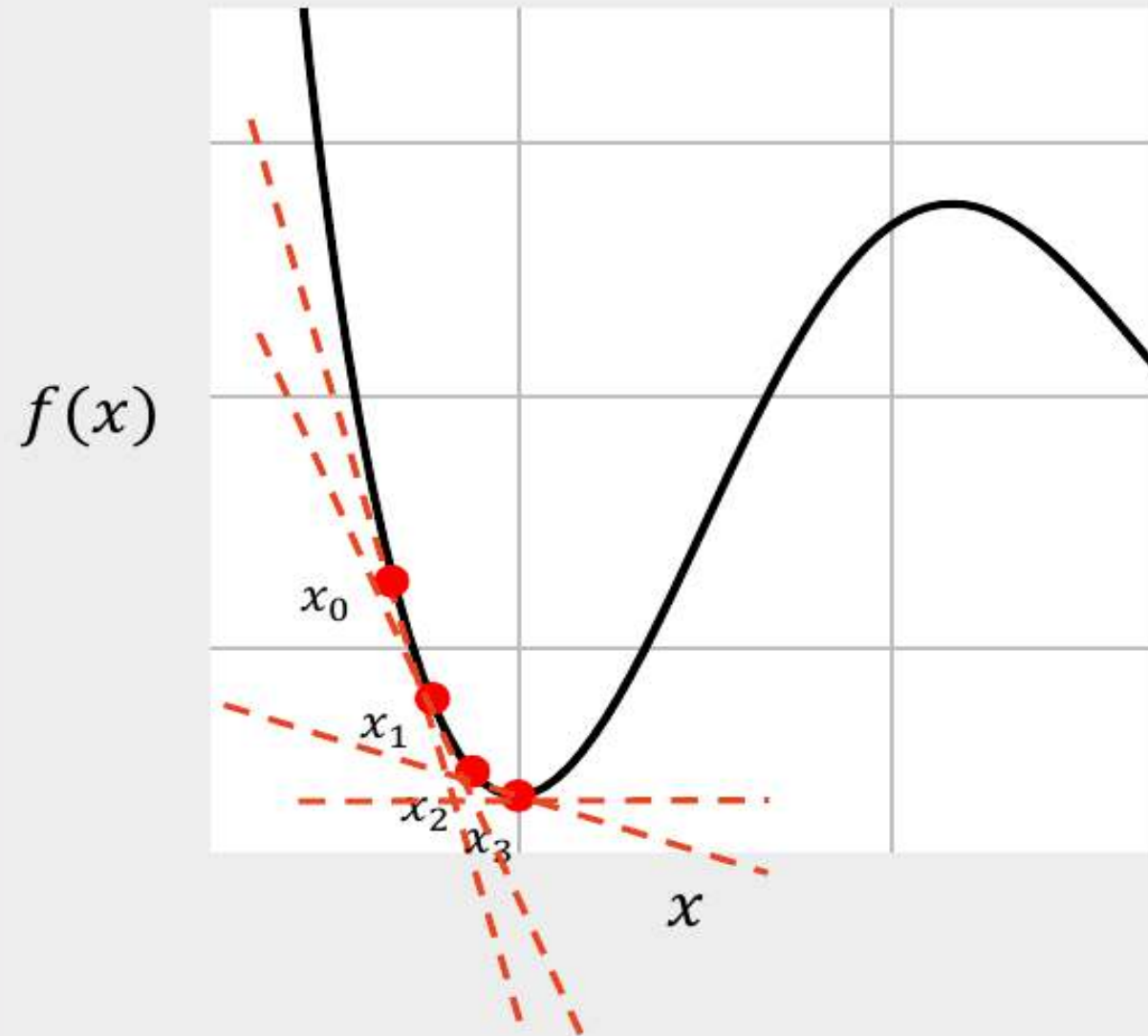
Local minimum

$x$



# Optimality conditions for local **minimum**

| Number of dimensions | First order condition                                                                                                                                                                                               | Second order condition                                                                                                                                                                                                                                                                                                                                            | Example                                                                              |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------|
| One                  | $\frac{df(x^*)}{dx} = 0$                                                                                                                                                                                            | $\frac{d^2f(x^*)}{dx^2} > 0$                                                                                                                                                                                                                                                                                                                                      |   |
| Multiple             | <p>“Gradient”<br/> <math>\nabla f(x_1, x_2, \dots, x_n)</math></p> $= \left[ \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \dots, \frac{\partial f}{\partial x_n} \right]$ $= [0, 0, \dots, 0]$ | <p>“Hessian”<br/> <math>\nabla^2 f(x_1, x_2, \dots, x_n)</math></p> $= \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \dots & \frac{\partial^2 f}{\partial x_n \partial x_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_1 \partial x_n} & \dots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$ <p>Must be “positive definite”</p> |  |



### Gradient Descent Method:

1. Choose a starting point,  $x_0$
2. At that point, compute the gradient,  $\nabla f(x_0)$
3. Compute the next point, with a step size  $\gamma$  :

$$x_{n+1} = x_n - \gamma \nabla f(x_n)$$

\*Stop when  $\nabla f(x_n) < \delta$  Very small number  
or

\*Stop when  $(x_{n+1} - x_n) < \delta$

# Practice Question 2

Consider the following function:

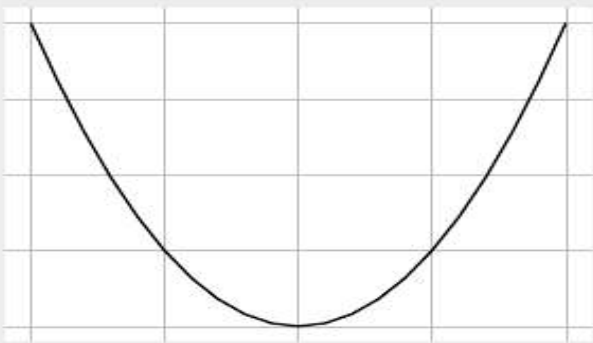
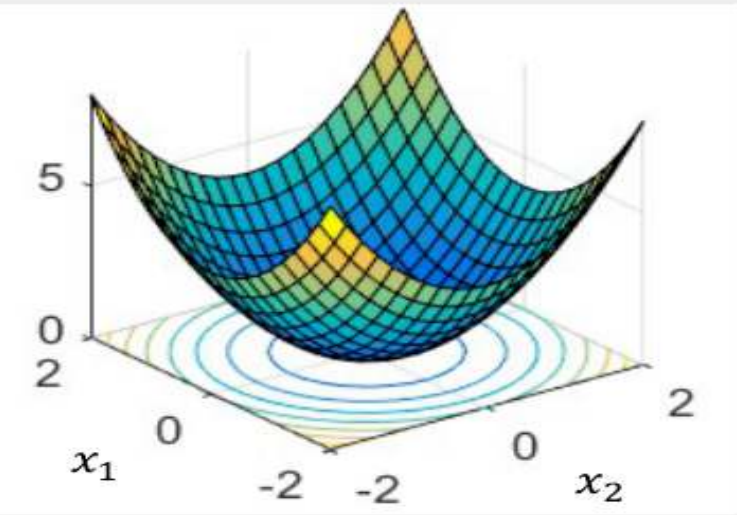
$$f(x) = x^2 - 6x$$

The gradient is:

$$\nabla f(x) = 2x - 6$$

Using the starting point  $x = 1$  and the step size  $\gamma = 0.3$ , apply the gradient descent method to compute the next **three** points in the search algorithm.

# Optimality conditions for local **minimum**

| Number of dimensions | First order condition                                                                                                                                                                                               | Second order condition                                                                                                                                                                                                                                                                                                                                            | Example                                                                              |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------|
| One                  | $\frac{df(x^*)}{dx} = 0$                                                                                                                                                                                            | $\frac{d^2f(x^*)}{dx^2} > 0$                                                                                                                                                                                                                                                                                                                                      |   |
| Multiple             | <p>“Gradient”<br/> <math>\nabla f(x_1, x_2, \dots, x_n)</math></p> $= \left[ \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \dots, \frac{\partial f}{\partial x_n} \right]$ $= [0, 0, \dots, 0]$ | <p>“Hessian”<br/> <math>\nabla^2 f(x_1, x_2, \dots, x_n)</math></p> $= \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \dots & \frac{\partial^2 f}{\partial x_n \partial x_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_1 \partial x_n} & \dots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$ <p>Must be “positive definite”</p> |  |

# Practice Question 3

Consider the following function:

$$f(\underline{x}) = x_1^2 + 4x_2^2$$

The gradient is:

$$\nabla f(\underline{x}) = \begin{bmatrix} 2x_1 \\ 8x_2 \end{bmatrix}$$

Using the starting point  $\underline{x}_0 = [1, 1]$  and the step size  $\gamma = 0.15$ , apply the gradient descent method to compute the next **three** points in the search algorithm.

Download the [logitr-cars](#) repo from GitHub

# Estimating utility models

1. Open the `logitr-cars` folder in Positron
2. Open `code/3.1-model-mnl.R`

# Maximum likelihood estimation

$$\begin{aligned}\tilde{u}_j &= v_j + \tilde{\varepsilon}_j \\ &= \beta_1 x_{j1} + \beta_2 x_{j2} + \dots + \tilde{\varepsilon}_j \\ &= \boldsymbol{\beta}' \mathbf{x}_j + \tilde{\varepsilon}_j\end{aligned}$$

Estimate  $\boldsymbol{\beta} = [\beta_1, \beta_2, \dots, \beta_n]$   
by maximizing the likelihood function

$$\begin{aligned}\text{minimize } -\log \mathcal{L} &= - \sum_{j=1}^J P_j(\boldsymbol{\beta} | \mathbf{x})^{y_j} \\ &\text{with respect to } \boldsymbol{\beta}\end{aligned}$$

$y_j = 1$  if alternative  $j$  was chosen  
 $y_j = 0$  if alternative  $j$  was not chosen

For logit model:

$$P_j = \frac{e^{v_j}}{\sum_{k=1}^J e^{v_k}} = \frac{e^{\boldsymbol{\beta}' \mathbf{x}_j}}{\sum_{k=1}^J e^{\boldsymbol{\beta}' \mathbf{x}_k}}$$

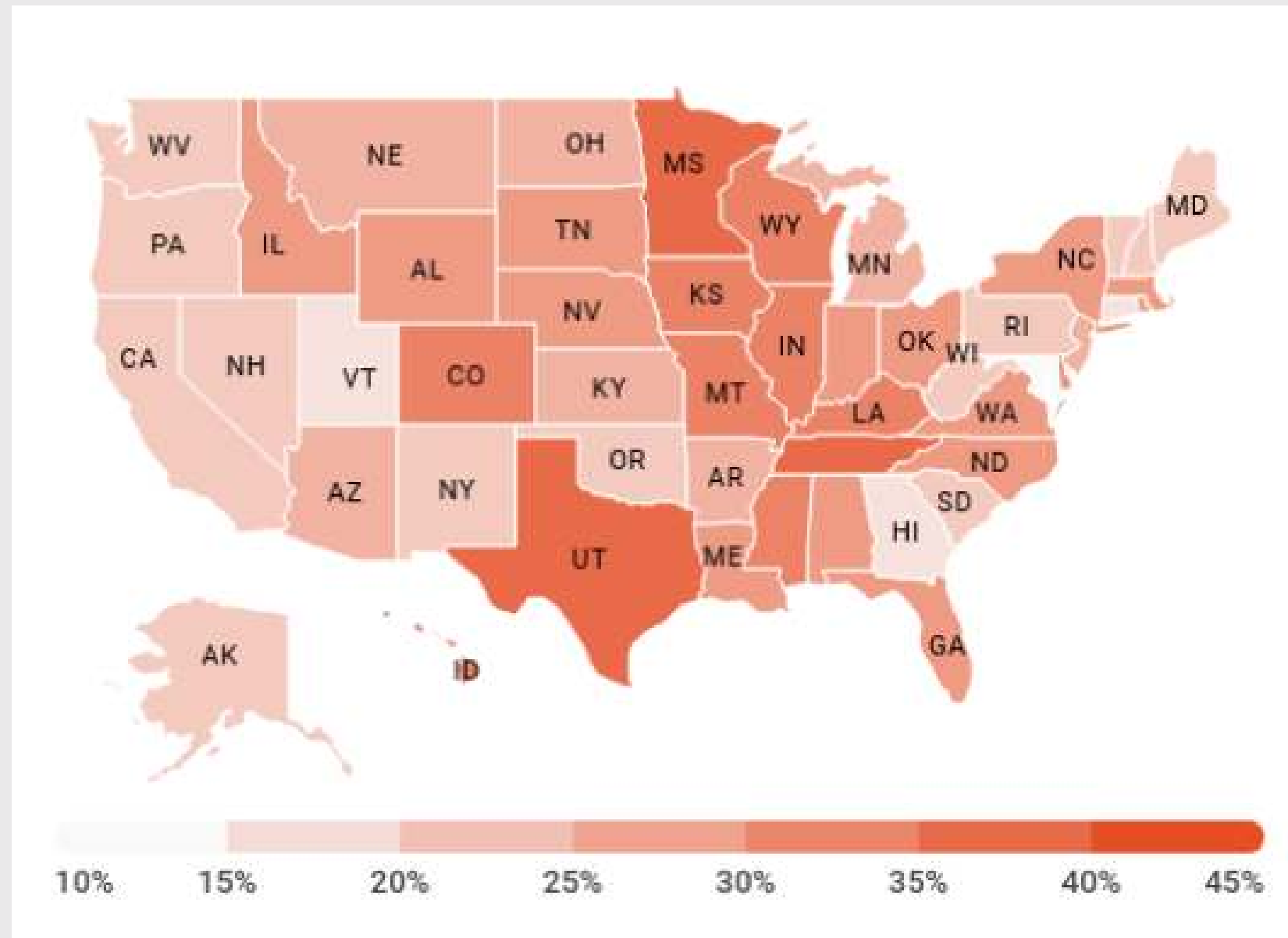
*Break*

05 : 00

# Week 9: *Optimization & MLE*

1. Maximum likelihood estimation
2. Optimization (in general)
3. Joins
4. Pilot data cleaning

# What's wrong with this map?



# Likely culprit: Merging two columns

```
1 head(names)
```

```
#>           state_name
#> 1           Alabama
#> 2           Alaska
#> 3           Arizona
#> 4           Arkansas
#> 5 Armed Forces Africa
#> 6 Armed Forces Americas
```

```
1 head(abbs)
```

```
#> state_abb
#> 1      AA
#> 2      AE
#> 3      AE
#> 4      AE
```

```
1 result <- cbind(names, abbs)
2 head(result)
```

```
#>           state_name state_abb
#> 1           Alabama      AA
#> 2           Alaska      AE
#> 3           Arizona      AE
#> 4           Arkansas      AE
#> 5 Armed Forces Africa      AE
#> 6 Armed Forces Americas      AK
```

# Joins

1. `inner_join()`
2. `left_join()` / `right_join()`
3. `full_join()`

Example: `band_members` & `band_instruments`

```
1 band_members
```

```
#> # A tibble: 3 × 2  
#>   name  band  
#>   <chr> <chr>  
#> 1 Mick  Stones  
#> 2 John  Beatles  
#> 3 Paul  Beatles
```

```
1 band_instruments
```

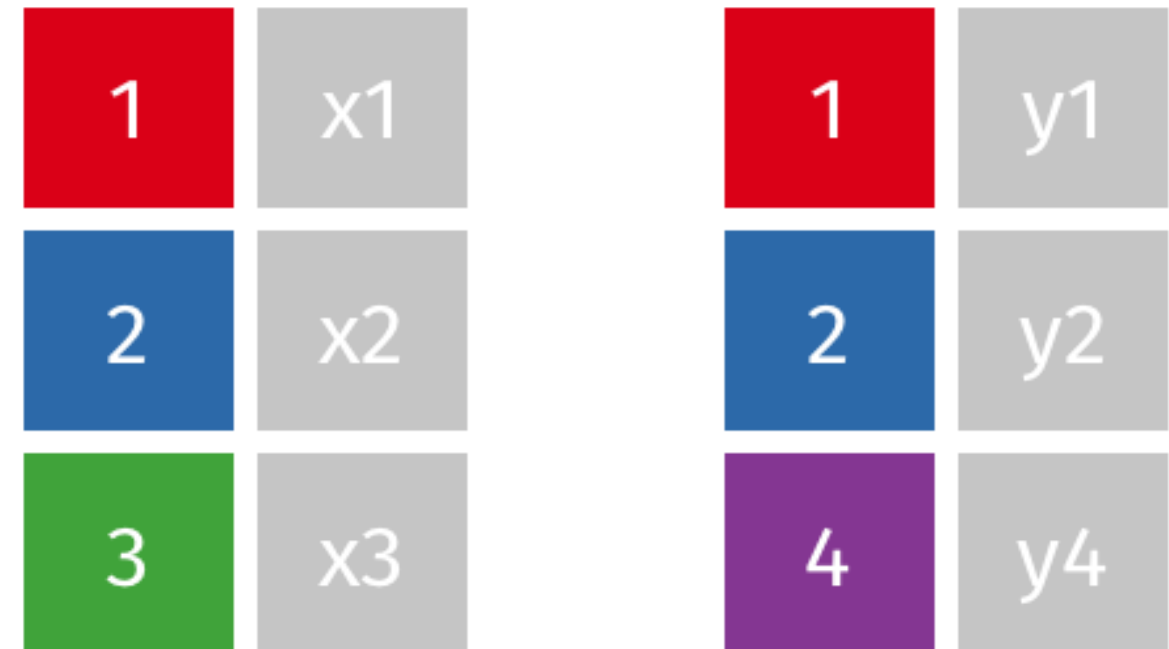
```
#> # A tibble: 3 × 2  
#>   name  plays  
#>   <chr> <chr>  
#> 1 John  guitar  
#> 2 Paul  bass  
#> 3 Keith guitar
```

# inner\_join()

```
1 band_members %>%  
2   inner_join(band_instruments)
```

```
#> # A tibble: 2 × 3  
#>   name  band  plays  
#>   <chr> <chr> <chr>  
#> 1 John  Beatles guitar  
#> 2 Paul  Beatles  bass
```

inner\_join(x, y)

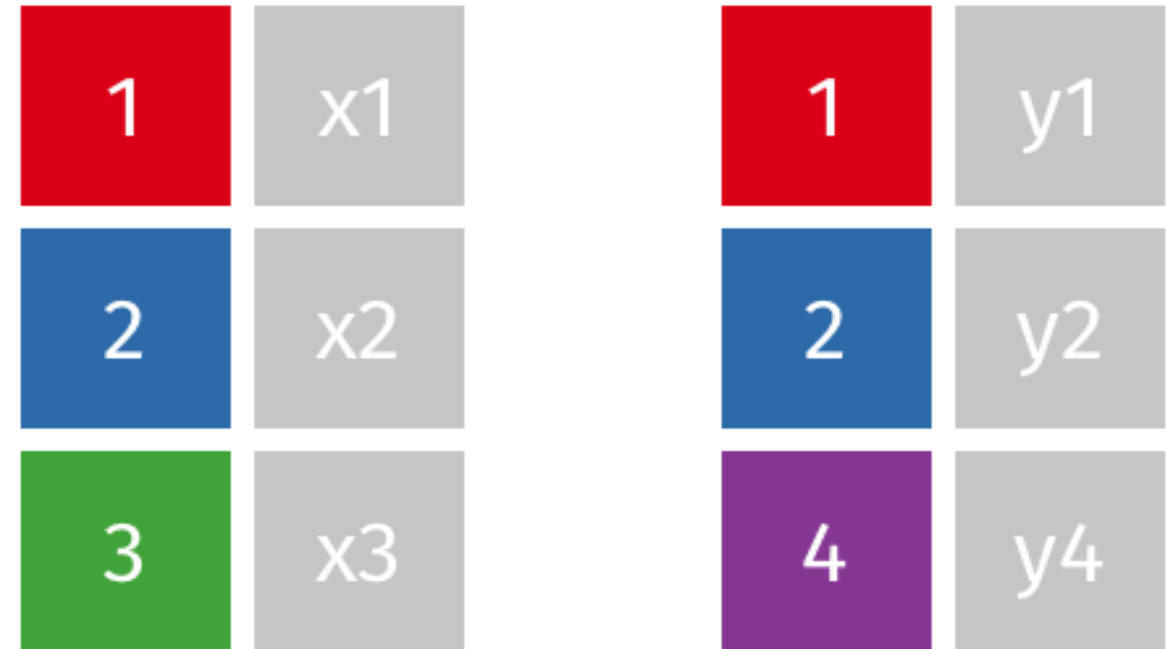


# full\_join()

```
1 band_members %>%  
2   full_join(band_instruments)
```

```
#> # A tibble: 4 × 3  
#>   name  band  plays  
#>   <chr> <chr> <chr>  
#> 1 Mick  Stones <NA>  
#> 2 John  Beatles guitar  
#> 3 Paul  Beatles bass  
#> 4 Keith <NA>   guitar
```

full\_join(x, y)

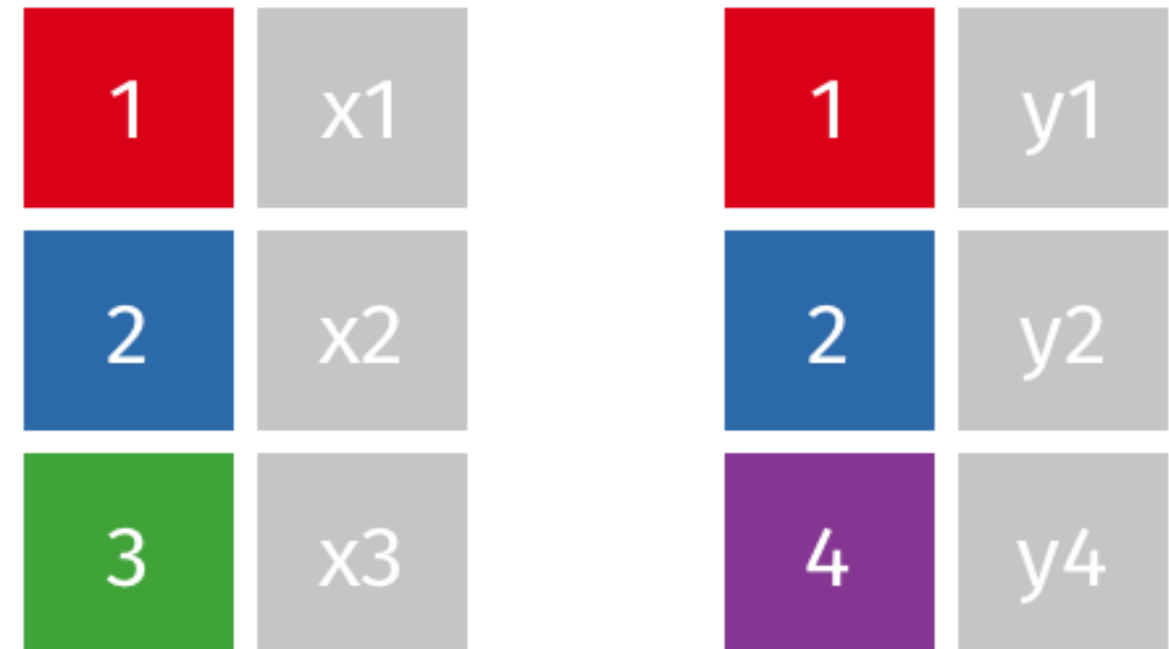


# left\_join()

```
1 band_members %>%  
2   left_join(band_instruments)
```

```
#> # A tibble: 3 × 3  
#>   name  band  plays  
#>   <chr> <chr> <chr>  
#> 1 Mick  Stones <NA>  
#> 2 John  Beatles guitar  
#> 3 Paul  Beatles bass
```

left\_join(x, y)

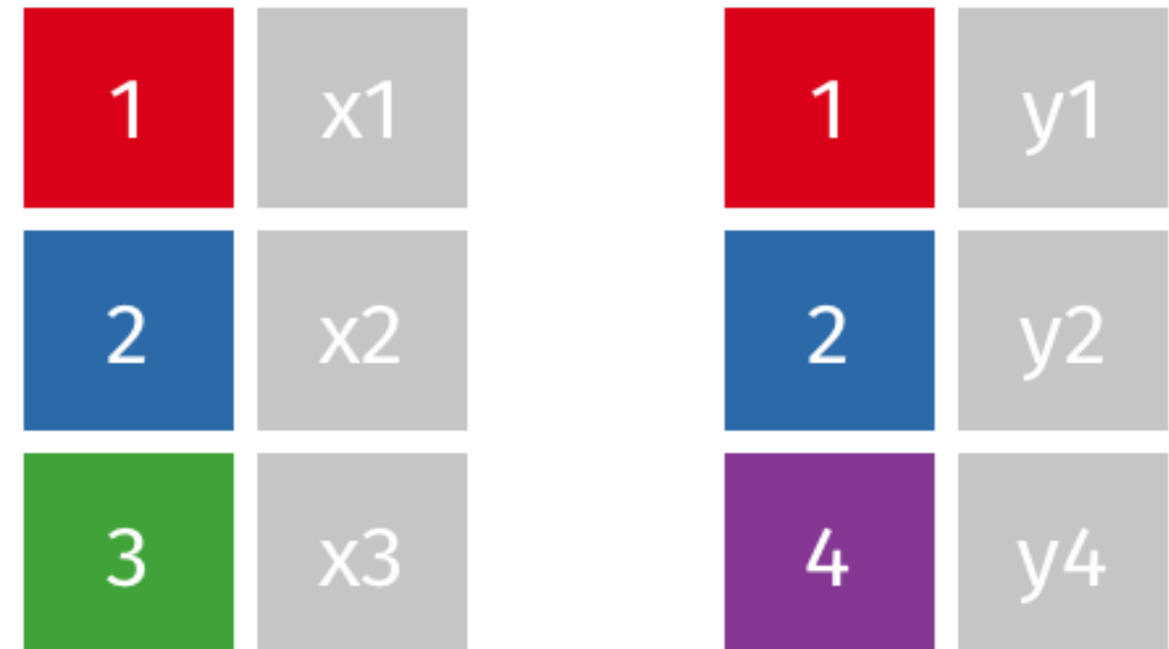


# right\_join()

```
1 band_members %>%  
2   right_join(band_instruments)
```

```
#> # A tibble: 3 × 3  
#>   name  band  plays  
#>   <chr> <chr> <chr>  
#> 1 John  Beatles guitar  
#> 2 Paul  Beatles bass  
#> 3 Keith <NA>   guitar
```

right\_join(x, y)



# Specify the joining variable name

```
1 band_members %>%  
2   left_join(band_instruments)
```

```
#> Joining with `by = join_by(name)`
```

```
#> # A tibble: 3 × 3  
#>   name band plays  
#>   <chr> <chr> <chr>  
#> 1 Mick  Stones <NA>  
#> 2 John  Beatles guitar  
#> 3 Paul  Beatles bass
```

```
1 band_members %>%  
2   left_join(  
3     band_instruments,  
4     by = 'name'  
5   )
```

```
#> # A tibble: 3 × 3  
#>   name band plays  
#>   <chr> <chr> <chr>  
#> 1 Mick  Stones <NA>  
#> 2 John  Beatles guitar  
#> 3 Paul  Beatles bass
```

# Specify the joining variable name

If the names differ, use `by = c("left_name" = "joining_name")`

```
1 band_members
```

```
#> # A tibble: 3 × 2  
#>   name  band  
#>   <chr> <chr>  
#> 1 Mick  Stones  
#> 2 John  Beatles  
#> 3 Paul  Beatles
```

```
1 band_instruments2
```

```
#> # A tibble: 3 × 2  
#>   artist plays  
#>   <chr>   <chr>  
#> 1 John   guitar  
#> 2 Paul   bass  
#> 3 Keith  guitar
```

```
1 band_members %>%  
2   left_join(  
3     band_instruments2,  
4     by = c("name" = "artist")  
5   )
```

```
#> # A tibble: 3 × 3  
#>   name  band  plays  
#>   <chr> <chr>   <chr>  
#> 1 Mick  Stones <NA>  
#> 2 John  Beatles guitar  
#> 3 Paul  Beatles bass
```

# Specify the joining variable name

Or just rename the joining variable in a pipe

```
1 band_members
```

```
#> # A tibble: 3 × 2  
#>   name  band  
#>   <chr> <chr>  
#> 1 Mick  Stones  
#> 2 John  Beatles  
#> 3 Paul  Beatles
```

```
1 band_instruments2
```

```
#> # A tibble: 3 × 2  
#>   artist plays  
#>   <chr>   <chr>  
#> 1 John   guitar  
#> 2 Paul   bass  
#> 3 Keith  guitar
```

```
1 band_members %>%  
2   rename(artist = name) %>%  
3   left_join(  
4     band_instruments2,  
5     by = "artist"  
6   )
```

```
#> # A tibble: 3 × 3  
#>   artist band    plays  
#>   <chr>   <chr>   <chr>  
#> 1 Mick   Stones  <NA>  
#> 2 John   Beatles guitar  
#> 3 Paul   Beatles bass
```

# Your turn

15:00

1. Create a new data frame called `state_data` by joining the `state_abbs` and `state_regions` data frames. The result should be a data frame with variables `state_name`, `state_abb`, and `state_region`. It should look like this:

```
1 head(state_data)
```

```
#> # A tibble: 6 × 3
#>   state_name state_abb state_region
#>   <chr>      <chr>      <chr>
#> 1 Alabama    AL          Southeast
#> 2 Alaska     AK          Pacific
#> 3 Arizona    AZ          Mountain
#> 4 Arkansas   AR          Delta States
#> 5 California CA          Pacific
#> 6 Colorado   CO          Mountain
```

2. Join the `state_data` data frame to the `wildlife_impacts` data frame, adding the variables `state_region` and `state_name`.

```
1 glimpse(wildlife_impacts)
```

```
#> Rows: 56,978  
#> Columns: 23  
#> $ state_abb      <chr> "FL", "IN", NA, NA, NA, "FL", "FL", NA, NA, "FL", NA, "TX", NA, N  
#> $ state_name     <chr> "Florida", "Indiana", NA, NA, NA, "Florida", "Florida", NA, NA, "  
#> $ state_region   <chr> "Southeast", "Corn Belt", NA, NA, NA, "Southeast", "Southeast", N  
#> $ incident_date  <dttm> 2018-12-31, 2018-12-29, 2018-12-29, 2018-12-27, 2018-12-27, 2018  
#> $ airport_id     <chr> "KMIA", "KIND", "ZZZZ", "ZZZZ", "ZZZZ", "KMIA", "KMC0", "ZZZZ", "  
#> $ airport        <chr> "MIAMI INTL", "INDIANAPOLIS INTL ARPT", "UNKNOWN", "UNKNOWN", "UN  
#> $ operator       <chr> "AMERICAN AIRLINES", "AMERICAN AIRLINES", "AMERICAN AIRLINES", "A  
#> $ atype          <chr> "B-737-800", "B-737-800", "UNKNOWN", "B-737-900", "B-737-800", "A  
#> $ type_eng       <chr> "D", "D", NA, "D", "D", "D", "D", "D", "D", "D", "D", "D", "D", "  
#> $ species_id     <chr> "UNKBL", "R", "R2004", "N5205", "J2139", "UNKB", "UNKBS", "ZT001"  
#> $ species        <chr> "Unknown bird - large", "Owls", "Short-eared owl", "Southern lapv  
#> $ damage         <chr> "M?", "N", NA, "M?", "M?", "N", "N", "N", "N", "N", "N", "N", "N"  
#> $ num_engs       <dbl> 2, 2, NA, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2  
#> $ incident_month  <dbl> 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 1  
#> $ incident_year   <dbl> 2018, 2018, 2018, 2018, 2018, 2018, 2018, 2018, 2018, 2018, 2018,  
#> $ time_of_day     <chr> "Day", "Night", NA, NA, NA, "Day", "Night", NA, NA, "Day", "Night  
#> $ time           <dbl> 1207, 2355, NA, NA, NA, 955, 948, NA, NA, 1321, 15, 1612, NA, NA,  
#> $ height         <dbl> 700, 0, NA, NA, NA, NA, 600, NA, NA, 0, NA, 0, NA, NA, 0, 500, 10  
#> $ speed          <dbl> 200, NA, NA, NA, NA, NA, 145, NA, NA, 130, NA, NA, NA, NA, NA, 16  
#> $ phase_of_flt   <chr> "Climb", "Landing Roll", NA, NA, NA, "Approach", "Approach", NA,  
#> $ sky            <chr> "Some Cloud", NA, NA, NA, NA, NA, "Some Cloud", NA, NA, "No Cloud  
#> $ precip         <chr> "None", NA, NA, NA, NA, NA, "None", NA, NA, "None", NA, "None", N  
#> $ cost_repairs_infl_adj <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA
```

# Week 9: *Optimization & MLE*

1. Maximum likelihood estimation
2. Optimization (in general)
3. Joins
4. Pilot data cleaning

Download the [conjoint\\_buttons](#) template repo

# Cleaning surveydown survey data

For the rest of class, work with your team to pull in your pilot survey responses and clean your pilot data.

1. Open the `survey` folder in Positron
2. Open `code/2_get_raw_data.R`
3. Open `code/3_data_cleaning.R`