# WEEK 41: CLASS EXERCISES

Practical exercise: MouseTrap analysis

The data file which we’ll analyze can be found [here](https://drive.google.com/drive/folders/11KwZrEoLGHW7EvJBFkRfqX3PMmfKUcGt)

The solution markdown can be found in the [GitHub](https://github.com/laurabpaulsen/PercAct2022)

The following exercises are based on the mouse-tracking experiment which you tried in the last class. The experiment asks the participant to choose the red circle as fast as possible. The alternative (incorrect) option presented is either a red square (coded as ‘similar’) or a blue square (coded as ‘dissimilar’). We’ll now preprocess and analyze the data from that experiment in our beloved R studio.

A red and black background

Description automatically generated with medium confidenceA red and blue squares

Description automatically generated

Mousetrap R package

* Install the package “mousetrap” and load the library.
* Download the data file from the above google folder and load it into R using standard read functions (e.g. read.csv) and briefly inspect it.
* Open the [MouseTrap documentation page](https://www.rdocumentation.org/packages/mousetrap/versions/3.1.5/topics/mousetrap). In your group, do the following steps by finding and applying appropriate mousetrap functions. Also, writing ?function\_name() in the console is great way of reading about the function in question. Call me for help whenever you get stuck 😊

1. Turn the raw data into a mousetrap object. Now take a moment to look at the structure that it created. What are the elements, what information does it provide?
2. Make a plot. Try to understand what the lines mean and how to “read” the graph. What seems weird about it when you think about how the experiment task looked like?

1. The demo-experiment had different trial types for which we have different predictions. Make a plot that distinguishes these two conditions, e.g. by different colors.

1. Find a function that does a mirror-symmetric mapping of all the movements from the right side to the left side so that all movements overlap. Plot again. What did this function also do? If you’ve done everything correctly, it should look like this:

A graph with lines and numbers

Description automatically generated

1. The standard plotting function shows x and y coordinates. Modify it so you plot timestamps by xpos.

What do you see? What is this line in the beginning?

1. Find a function to remove this “line” and plot again. If you now plotted x and y coordinates again, what would have changed in the plot? Think first, then try it!

--> Please call me over for a brief check-in!

1. Apply the function “mt\_time\_normalize”. Now look at your data variable (the mt object) where a new matrix appeared. What do you notice? What does the function do? After you have thought about it yourself, check the documentation.

1. Find out how to plot the normalized trajectories instead of the raw data.

1. Take a moment to play around with different numbers of steps in “mt\_time\_normalize” and see how that changes the shape of the trajectories, esp. when using very small numbers. Discuss what the decision about normalizing implies for the data analysis and interpretation. In the end, set the steps back to the default of 101.

1. Now we want to visualize our “findings”. Find a function that will plot averages of all the “similar” movements and all the “dissimilar” movements.

Think: Which trajectories do we need to use, the original or the time normalized? Why? Try plotting both to see whether you were right.

1. Apply the function “mt\_measures” and look at the outcome in your data variable.
2. Now find a function that helps you aggregate some measures of your pleasing over the two trial\_types. It’ll look e.g. like this:

A screenshot of a computer

Description automatically generated

1. Take a moment to think what these “results” could mean if this was a proper experiment and didn’t just have the experimenter herself run it once ;-) How would you interpret this? Does this match your hypothesis?  Especially look at the MAD (maximum absolute deviation from the direct path), the AUC (area under the curve) and the reaction time.

When you’ve come this far, please call me over for a brief check-in!

Challenge level:

1. Normally, you'd want to remove incorrect trials. As the mouse\_trap object does not function with tidyverse, figure out how to remove potential incorrect trials from your mousetrap object.

1. Would the function ‘mt\_align’ be useful for this data? Why or why not?

Your own mouse tracking experiment

Next class, we will make our own mouse tracking experiments by modifying the circle square experiment. Remember to keep it simple, as you only have the next class to make the experiment and run it on a few classmates. Use some time to think about what you’d like to do, maybe find the stimuli, which you’ll use etc.