Models for Perception and Action

Mouse-tracking exercises

Line, Mille, Phila, and Telma

## 

In your study group, try to do the following analysis steps with a single trial of the hand priming data in R. Upload your commented code to Blackboard (or share it with me in some other way). If you get stuck anywhere make a comment about it - just see how far you get! We’ll discuss it together during the next class.

Deadline for submission: Tuesday, April 9th.

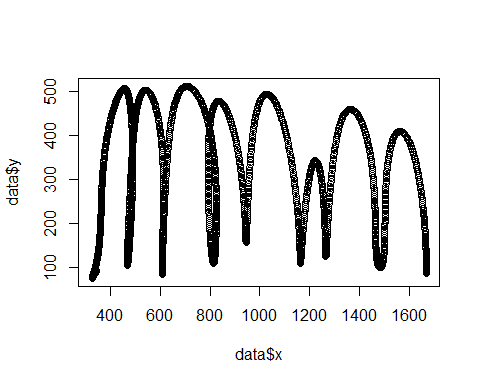
Note: Use the data 21, not any of the others I had previously uploaded. The data folder contains all 20 trials of that one participant. The data are already filtered.

1. **Remind yourselves what we did in the study and what we expected and why.**
2. **Load one trial of your choice into R. The four columns refer to: sample, condition, x and y.**

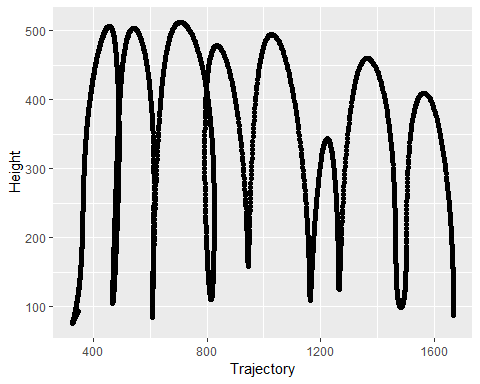
# read in the data  
data <- read.delim("C:/Users/telmi/Documents/Cognitive science/3\_computational modelling/assignments/semester4/MPA/21/21\_1.txt", sep = ",", header = F )  
  
  
# name the columns  
colnames(data) <- c("sample", "condition", "x", "y")

1. **Plot x against y.**

plot(data$x, data$y)

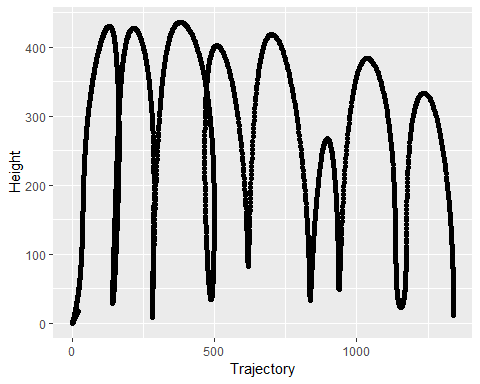


# use ggplot to make the plot nicer  
ggplot(data) +  
 geom\_point( aes(x,y)) +  
 labs(x = "Trajectory", y = "Height") # add axis titles



1. **Map the y data to a more intuitive screen output with 0/0 in the lower left corner and where upwards movements are increasingly positive (i.e. so that it looks like the movements we made on the screen).**

# center the data by substracting the minimum values  
data$x\_center <- data$x - min(data$x)  
data$y\_center <- data$y - min(data$y)  
  
# plot it  
ggplot(data) +  
 geom\_point( aes(x\_center,y\_center)) +  
 labs(x = "Trajectory", y = "Height")



1. **Calculate the distance travelled (for x and y combined).**

dist <- sum(sqrt((data$x-lag(data$x))^2+(data$y-lag(data$y))^2), na.rm = T)  
 # you could also use function diff()  
 # lag takes the previous datapoint  
dist

## [1] 6135.357

1. **Add a time vector. I forgot to measure time so I had to estimate it post-hoc (fine for this exercise, not great if it was a real experiment). We can assume that each sample takes .0025 s.**

data$time <- 0.0025\*data$sample # multiply by the sample number

1. **Calculate velocity and acceleration for y, i.e. vertical movement. Remember: Velocity = distance difference / time difference. Acceleration = velocity difference / time difference.**

# velocity  
data$velocity <- (data$y-lag(data$y) ) / (data$time - lag(data$time))  
  
# acceleration  
data$acceleration <- (data$velocity - lag(data$velocity)) / (data$time - lag(data$time))

1. **Play around with some filter to smoothen velocity and acceleration (most common: Butterworth filter).**

Comment:

We can’t really solve this task yet.   
The following code runs. However, the unreasonable results suggest there might be mistakes in the code.

#install.packages("mFilter")  
  
library(mFilter)

## Warning: package 'mFilter' was built under R version 3.5.3

#f <- bwfilter(data$velocity,freq=400,nfix=NULL,drift=FALSE)   
 # What should our parameters be?  
  
  
#matrix <- (f$fmatrix)  
  
#par(mfrow=c(2,1),mar=c(3,3,2,1),cex=.8)  
#plot(f$x,  
 # main="Butterworth filter of unemployment: Trend,  
 # drift=TRUE",col=1, ylab="")  
#lines(f$trend,col=2)  
  
#("topleft",legend=("nice try"))

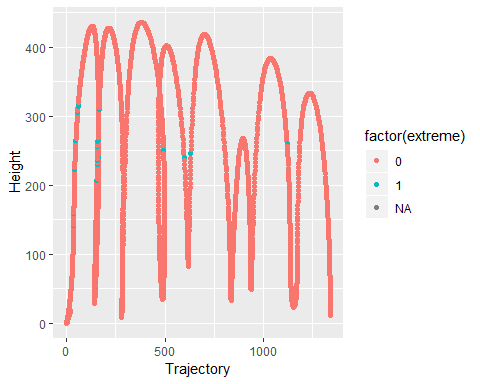
1. **Using zero crossings in the acceleration, calculate velocity extreme values (maxima and minima). Mark those extreme values on top of the x/y plot you made earlier.**

Comment:

The derivative of velocity is acceleration => the local maxima and minima of velocity are at those points where the derivative is 0 (i.e. zero-crossings in acceleration).

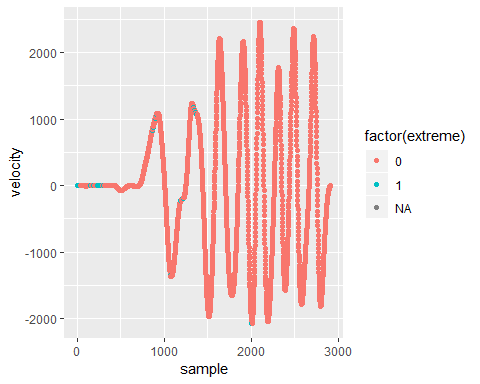
We would expect to see a blue point at each local minimum and maximum when we plot the velocity. However, this is not the case in the velocity plot.

# create a new column where all the acceleration = 1 are coded by 1, otherwise 1  
data$extreme <- ifelse(data$acceleration == 0, 1,0)  
  
  
ggplot(data) +  
 geom\_point( aes(x\_center,y\_center, color = factor(extreme))) +   
 # plot by this factor, you get the zero crossing plotted in the different color  
 labs(x = "Trajectory", y = "Height")



ggplot(data) +  
 geom\_point( aes(sample,velocity, color = factor(extreme)))

## Warning: Removed 1 rows containing missing values (geom\_point).



# Here we should see a blue point at each local minimum and maximum, but that is not the case. We don’t know why yet.

1. **Using e.g. a range of x positions, specify a segment during which the critical movement happens, i.e. the 6th movement is what we would want to compare in this study. Mark the peak in that segment in a different color.**

Comment: We eye-balled the interval of the 6th peak to color-code the critical movement.

data$color<- ifelse(data$sample>2260 & data$sample<2440,1,0)  
  
  
ggplot(data) +  
 geom\_point( aes(x\_center,y\_center, color = factor(color))) +   
 # plot by this factor, you get the zero crossing plotted in the different color  
 labs(x = "Trajectory", y = "Height") +  
 theme(legend.title = element\_blank(), legend.position='none')

