Achievement Emotions Analysis Pipeline

Pavel Chernyavskiy

2024-11-26

Preliminary Data Manipulation

Preamble

FaceReader9 intensity ratings were output to Excel files, concatenated row-wise, extra columns were removed, and data were assigned a common timescale in seconds. Some of the participants' genders did not appear correctly, so we correct the gender variable using code below. The result was saved as an R object ("D_emo_01042023.rds"), where the current analysis begins.

Code preliminaries

We load all necessary R packages and make a useful text processing function that we will use later:

```
#* load R packages *#
library(dplyr)
library(tidyverse)
library(ggplot2)
library(forcats)
library(hms)
library(lubridate)
library(stringr)
library(readxl)
library(scales)

#* make text processing function to use later *#
substrRight <- function(x, n){substr(x, nchar(x)-n+1, nchar(x))}
```

Read & filter emotion data

Here, we set working directory (where R object is located), read in the saved R object, only retain emotions encoded between 40 and 250 seconds (3.5 mins of instruction), and correct the subjects' gender variable. The sub-section ends in a check of subjects' gender; the numbers reflect the count of 1/100th of a second of encoded emotion for each subject. The rows are subject IDs and columns are the two genders:

```
#* set working directory, read in and process data *#
setwd("C:/Users/pcher/Box/EMO Strategies/Cleaned FaceReader Data")
D_emo<-readRDS("D_emo_01042023.rds")
D_emo<-D_emo %>%
       filter(Time_sec > 40) %>%
       filter(Time_sec < 250)

#* correct the gender variable *#
D_emo[which(D_emo$ID == '046'),9]<-"Male"
```

```
D_emo[which(D_emo$ID == '049'),9]<-"Female"
D_emo[which(D_emo$ID == '054'),9]<-"Male"
D_emo[which(D_emo$ID == '066'),9]<-"Male"
D_emo[which(D_emo$ID == '072'),9]<-"Female"
D_emo[which(D_emo$ID == '077'),9]<-"Male"
D_emo[which(D_emo$ID == '080'),9]<-"Male"
D_emo[which(D_emo$ID == '191'),9]<-"Male"
D_emo[which(D_emo$ID == '194'),9]<-"Female"
D_emo[which(D_emo$ID == '204'),9]<-"Female"
D_emo[which(D_emo$ID == '212'),9]<-"Female"
D_emo[which(D_emo$ID == '213'),9]<-"Female"
D_emo[which(D_emo$ID == '261'),9]<-"Male"
D_emo[which(D_emo$ID == '283'),9]<-"Male"

#* correct gender and row counts of 15 subjects *#
table(D_emo$ID,D_emo$Gender)
```

```
##
##       Female  Male
##   046      0 25181
##   049  37764     0
##   054      0 37764
##   066      0 44058
##   072  50352     0
##   077      0 44058
##   080      0 56646
##   191      0 44095
##   194  31495     0
##   204  25208     0
##   212  50377     0
##   213  31501     0
##   230  54447     0
##   261      0 38307
##   283      0 25176
```

Set a time sampling rate & convert to time-wide format

Here, we set a sampling rate for time (1 second, 1/10th of a second, or 1/100th of a second), and convert from a "long" format to a "wide" format for functional regression analysis. Less frequent (slower) sampling rates will result in smoother data. Additionally, we filter out any FaceReader file (i.e. a subject's session), where at least 30% of the emotions could not be coded.

Because we currently analyze one emotion at a time, we select an emotion to analyze in the code below. Joy/happiness ("Happy") is currently selected. Other emotions ("Sad","Suprised") are commented out, but can be commented in to perform the analysis. The amount of coded emotion data that remains depends on the sampling rate: least data for 1 second, most data for 1/100th of a second.

Below we have 70 rows (70 sessions of 15 subjects with >70% coded emotions) and 214 time points (one column per second), across ~210 seconds of instruction.

```
#* set a time sampling rate *#
t_samp<-0 #0 for 1 second, 1 for 1/10 of sec, 2 for 1/100 of sec
###
D_emo1w<-D_emo %>%
  mutate(temp=substr(`Participant Name`,1,6)) %>%
```

```
  #* remove sessions where >30% emotions are NAs **#
  filter(!(temp %in%c('212_10','077_13','072_5_','080_8_',
                      '072_4_','212_13','046_7_','077_6_',
                      '054_13','080_10','194_8_','049_13'))) %>%
  #* apply time sampling rate here *#
  mutate(Time_round = round(Time_sec,t_samp)) %>%
  #* select one emotion to analyze & other variables to carry forward*#
  select(
    #   Surprised,#Sad, <-same code can be used for other emotions
    Happy,
    Gender,ID,Session,Time_round,
    -Time_sec) %>%
  #* recast from long to wide time format *#
  pivot_wider(values_from = 1,
              names_from = 5,
              values_fn = mean) #<-average w/in time sampling rate

# retain 70 participants (rows) and 204 time points (columns)
dim(D_emo1w)
```

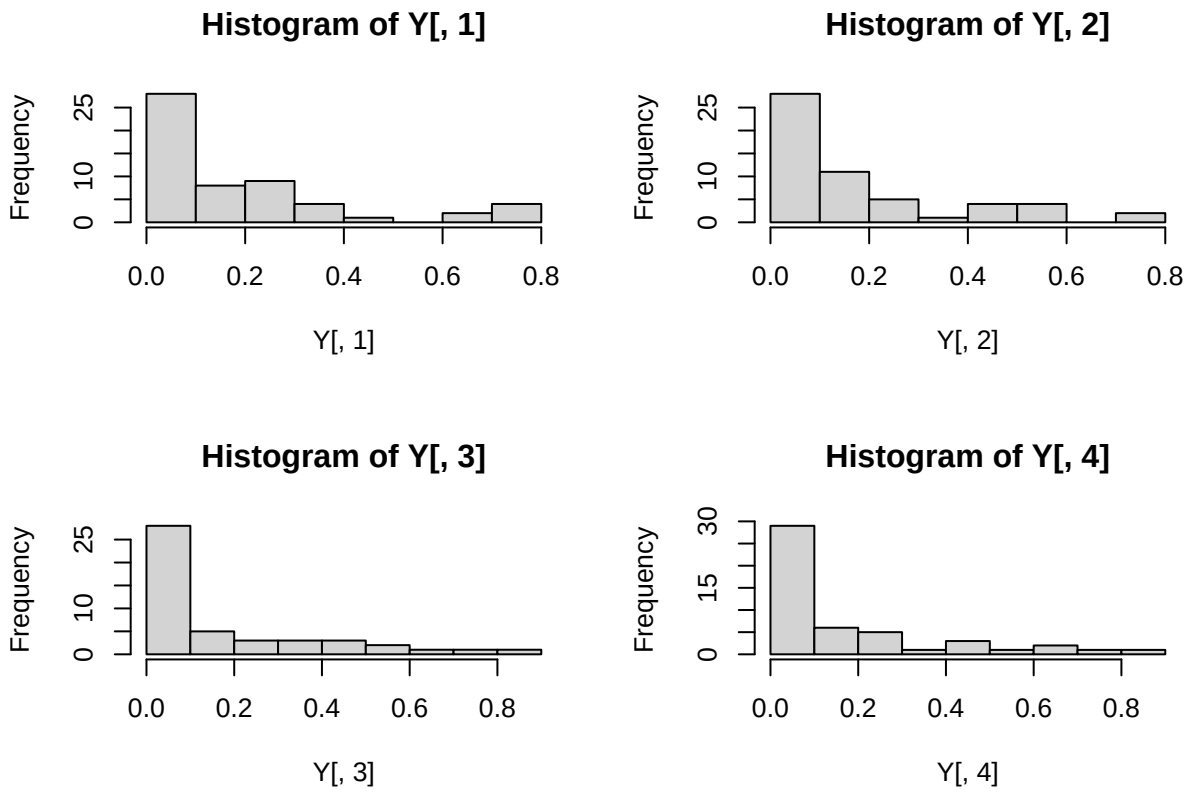
```
## [1]  70 214
```

Emotional Intensity Interpolation

Exploring emotional intensity

The goal of this sub-section is to interpolate and smooth the emotional intensity recorded during the retained sessions. Once interpolated, the intensities will serve as functional covariates in function-on-scalar regression. Here, we extract the intensities (without subject ID, gender) and visualize its distribution, for example for the first 4 sessions:

```
# extract without ID and gender and recast as a matrix #
Y<-D_emo1w %>% dplyr::select(-1:-3) %>% as.matrix()

par(mfrow=c(2,2))
hist(Y[,1])
hist(Y[,2])
hist(Y[,3])
hist(Y[,4])
```



Importantly, intensity is defined as continuous between 0 and 1, so interpolation must account for this. We perform all interpolation on the logit scale, i.e. we transform the data from the (0,1) interval into the unbounded (-infinity, infinity) interval. This transformation is performed using the qlogis() function in the code or the link="logit" option.

Non-parametric regression to interpolate & smooth

The method used to interpolate intensity, which in-turn will be used as a functional covariate, affects the functional regression output. In other words, how we choose to "fill in" the missing intensity and how smooth we make the emotions over time influences the noise in the explanatory variable and thus the realized statistical significance. Here, we use a form of non-parametric regression, called nonparametric kernel smoothing regression (Hayfield and Racine, 2008), such that fewer assumptions are required for interpolation. The smoothness of the predicted line, which affects the degree of smoothing and the accuracy of the interpolation (i.e., "bandwidth"), is selected using AIC. The smoothness is stationary, i.e. does not vary over time.

"Multistart 1 of 1" will appear in the output window to indicate when a single regression and interpolation has finished:

```
#** Nonparametric regression to interpolate **#
# read in np package
# Tristen Hayfield and Jeffrey S. Racine (2008). Nonparametric Econometrics:
# The np Package. Journal of Statistical Software 27(5). DOI 10.18637/jss.v027.i05
library(np)

# unique times & time resolution #
t<-unique(round(D_emo$Time_sec,t_samp))
```

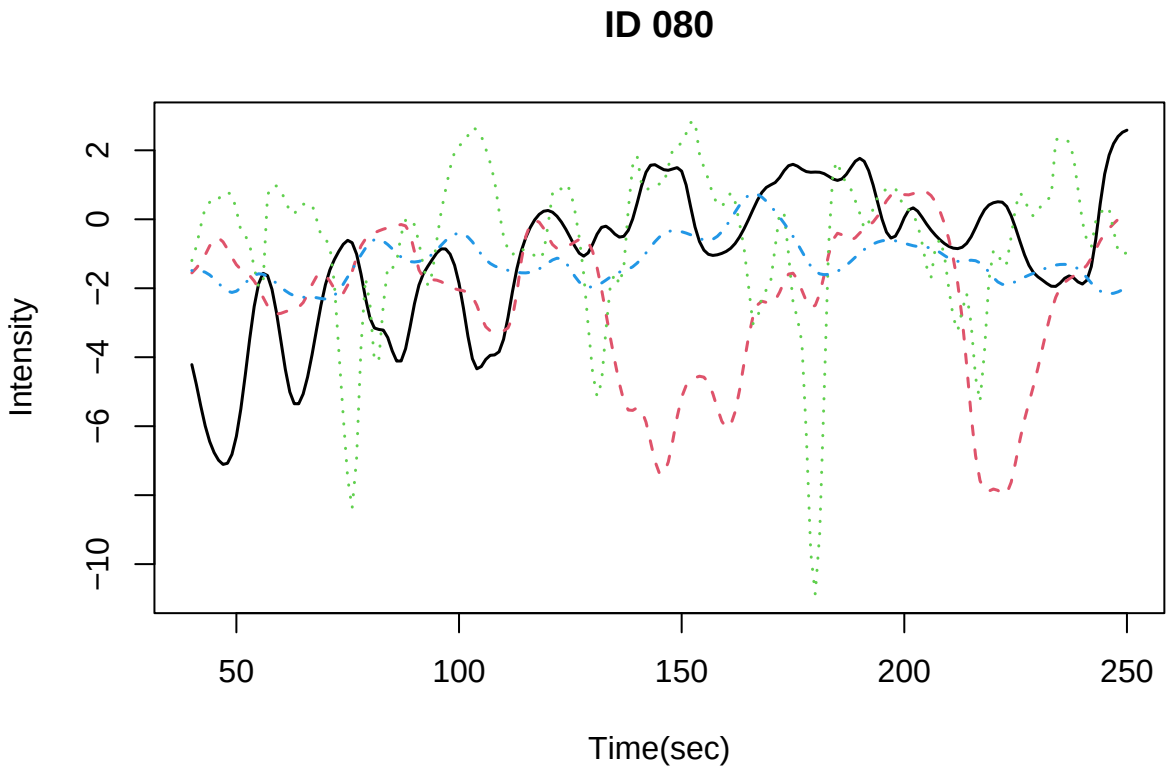
```
tres<-ncol(Y) #<-set resolution (original = num of Y columns)#

# make time to predict #
pred_t<-data.frame(t=seq(min(t),max(t),length=tres))
# make a temporary repository
temp<-sm_list<-list()

# loop over the instructional sessions #
for(i in 1:nrow(Y)){
  temp[[i]]<- npregbw(ydat=qlogis(Y[i,]), xdat=t, nmulti=1,
                      regtype="lc",
                      bwtype = "adaptive_nn",
                      bwmethod="cv.aic") %>%
              npreg(exdat=pred_t)
  sm_list[[i]]<- temp[[i]]$mean
}
```

```
## Multistart 1 of 1 |Multistart 1 of 1 |Multistart 1 of 1 |Multistart 1 of 1 /Multistart 1 of 1 |Multi
```

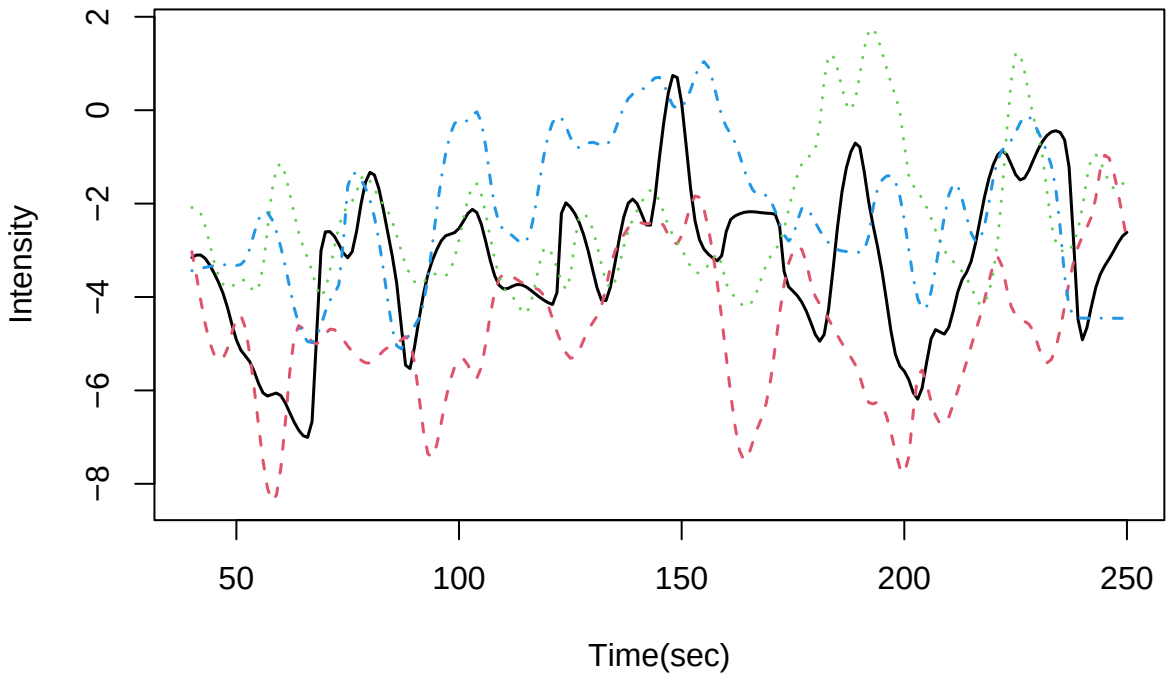
Now, we replace the original data with the smoothed data and visualize a few subjects. For every subject, we plot the functional data (ie emotional intensity of Joy, sampled every second over 30 to 230 sec) for all the sessions for that subject. Different colored lines delineate different instructional sessions. Different participants appear in different plots.

Note that the intensity is plotted on the same scale it was interpolated (logit scale), not the orignial (0 to 1) intensity, but the shapes would be very similar.

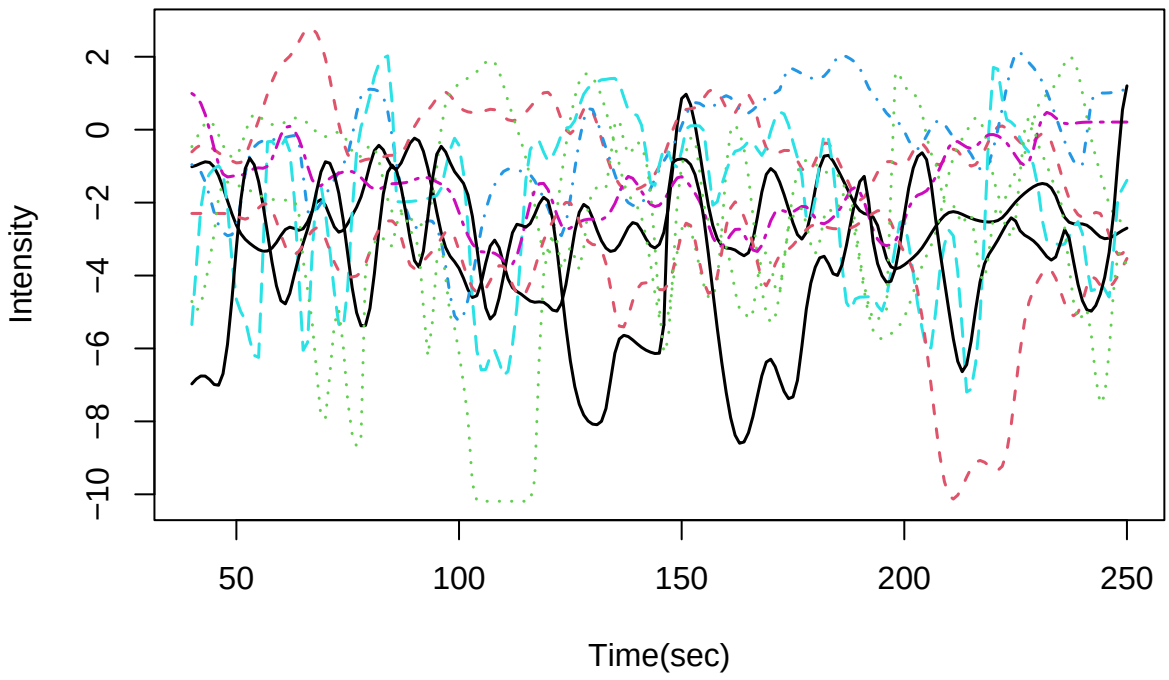
```
#* replace original data with smoothed emotion data *#
Y_sm<-matrix(unlist(sm_list),
             nrow=nrow(Y),byrow=TRUE) %>% as.matrix()
D_sm<-cbind(D_emo1w[,1:3], Y_sm) #* <- ANALYSIS DATASET HERE *#

#* visualize all sessions for some IDs *#
matplot(x=seq(min(t),max(t),length=tres),
        y=D_sm %>%
          filter(ID == "080") %>%
          select(-1:-3) %>% t(),
        type='l',lwd=1.5,
        xlab="Time(sec)",ylab='Intensity',main="ID 080",
        lty=1:nrow(D_sm %>% filter(ID == "080") %>%
                   select(-1:-3) %>% t())
)
```



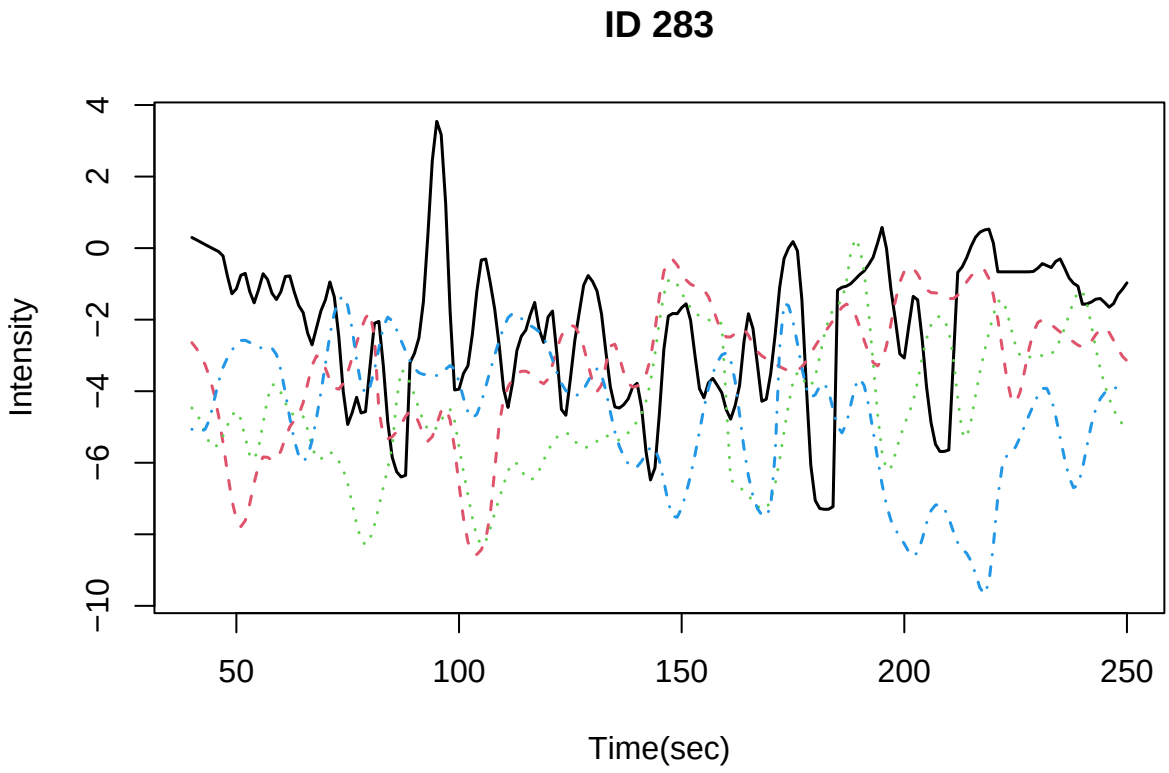

```
###
matplot(x=seq(min(t),max(t),length=tres),
        y=D_sm %>%
          filter(ID == "054") %>%
          select(-1:-3) %>% t(),
        type='l',lwd=1.5,
        xlab="Time(sec)",ylab='Intensity',main="ID 054",
        lty=1:nrow(D_sm %>% filter(ID == "054") %>%
                   select(-1:-3) %>% t())
)
```




```
###
matplot(x=seq(min(t),max(t),length=tres),
        y=D_sm %>%
          filter(ID == "230") %>%
          select(-1:-3) %>% t(),
        type='l',lwd=1.5,
        xlab="Time(sec)",ylab='Intensity',main="ID 230",
        lty=1:nrow(D_sm %>% filter(ID == "230") %>%
                   select(-1:-3) %>% t())
)
```




```
###
matplot(x=seq(min(t),max(t),length=tres),
        y=D_sm %>%
          filter(ID == "283") %>%
          select(-1:-3) %>% t(),
        type='l',lwd=1.5,
        xlab="Time(sec)",ylab='Intensity',main="ID 283",
        lty=1:nrow(D_sm %>% filter(ID == "283") %>%
                   select(-1:-3) %>% t())
)
```



At this point, the emotion data are finalized. Each smoothed trajectory or line of emotion will be used as a functional covariate in a function-on-scalar regression.

Linkage to strategies database

Here, we link the emotional intensity functional data to the strategies database. First, we aggregate the strategies database from ID-attempt level to ID-session level. In other words, each row of the data will be a particular ID during a particular session, along with their session accuracy (% of correct attempts), and perhaps session sophistication metrics, for ex: max sophistication, mean sophistication, modal sophistication, breadth.

At time of writing, this database is under embargo and cannot be shared publicly.

Code to pre-process the strategies database (applies some filters) and prepare for linkage as follows:

```
setwd("C:/Users/pcher/Box/EMO Strategies/Cleaned FaceReader Data")
math_dat<-read_xlsx("ArithStrat_allmergeddata_02_28_2022.xlsx",
                    na="NA") %>%
  #* apply filters to attempts *#
  filter(!is.na(ArithStrat)) %>%       #  filter NAs
  filter(!(PROB_TYPE %in% c(Prob_Type = "Composing Number","Equalize",
                            "Compare","Counting", "Number Comparison"))) %>%
  filter(!(Correctness == 'NA')) %>%

  #* create an ID to match emotion data *#
  mutate(ID = substrRight(CHILD_ID, 3)) %>%

  #* assign order to strategies and re-define as new variable Y *#
```

```
  mutate(Y = ordered(ArithStrat,
                     levels=c("Wild Guess", "Reasonable Guess",
                              "Trial & Error","Makes a set", "Counting All",
                              "Counting On - Concrete",
                              "Counting On - Abstract",
                              "Jump Strategy","Combination",
                              "Derived Combination","Compensation",
                              "Decomposition"))) %>%
  #* collapse some levels of Y together into new variable YC (Y Collapsed) #
  mutate(YC = fct_collapse(Y,
                           "Wild Guess" = "Wild Guess",
                           "RG & TE" = c("Reasonable Guess",
                                         "Trial & Error"),
                           "Makes a set" = "Makes a set",
                           "Counting All" = "Counting All",
                           "Counting On" =
                             c("Counting On - Concrete",
                               "Counting On - Abstract"),
                           "Jump Strategy" = "Jump Strategy",
                           "Combination" = "Combination",
                           "Derived Combination" = "Derived Combination",
                           "Comp & Decomp" = c("Compensation","Decomposition")
  )) %>%

  #* make a numeric correctness variable (1 = correct OR correct w support) *#
  mutate(Correct_num = if_else(
         Correctness %in% c('Correct','Correct with Support'),1,0)) %>%
  #Correctness %in% c('Correct'),1,0)) %>%
  group_by(ID,GRA,SESSION) %>%   # GRA,

  #* make session-level metrics *#
  summarise(avg_acc = mean(Correct_num),
            acc_mod = mean(Correct_num)-1e-8, #<-avoid 100% correct
            tot_att = n(),
            tot_acc = sum(Correct_num),
            max_soph = max(as.numeric(YC)),
            breadth = length(unique(YC)))
#* rename a column to match emotions *#
colnames(math_dat)[3]<-"Session"

#* examine resultant data *#
head(math_dat)
```

```
## # A tibble: 6 x 9
## # Groups:   ID, GRA [1]
##   ID    GRA      Session avg_acc acc_mod tot_att tot_acc max_soph breadth
##   <chr> <chr>      <dbl>   <dbl>   <dbl>   <int>   <dbl>    <dbl>   <int>
## 1 016   Courtney       4   0.833   0.833      42      35       NA       5
## 2 016   Courtney       5   0.867   0.867      45      39       NA       4
## 3 016   Courtney       6   0.75    0.750      36      27        9       8
## 4 016   Courtney       7   0.75    0.750      28      21        7       3
## 5 016   Courtney       8   0.88    0.880      25      22       NA       6
## 6 016   Courtney      10   0.842   0.842      19      16        9       4
```

```
table(math_dat$ID,math_dat$Session)
```

```
##
##       1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
##   016 1 1 1 1 1 1 1 1 1  1  1  1  0  0  0  0
##   019 1 1 1 1 1 1 1 1 0  0  0  0  0  0  0  0
##   023 1 1 1 1 1 1 1 1 1  1  1  1  0  0  0  0
##   030 1 1 1 1 1 1 1 1 1  1  1  1  1  0  0  0
##   035 1 1 1 1 1 1 1 1 1  1  1  1  1  1  0  0
##   046 1 1 1 1 1 1 1 1 1  1  1  0  1  1  0  0
##   049 1 1 1 1 1 1 1 1 0  1  0  1  1  1  0  0
##   054 1 1 1 1 1 1 1 1 1  0  1  1  1  1  0  0
##   066 1 1 1 1 1 1 1 1 1  1  1  1  1  0  0  0
##   072 1 1 1 1 1 1 1 1 1  1  1  0  1  1  1  0
##   077 1 1 1 1 1 1 1 1 1  1  1  1  1  1  0  0
##   080 1 1 1 1 1 1 1 1 1  1  0  0  0  0  0  0
##   098 1 1 1 1 1 1 1 1 1  1  2  1  1  0  0  0
##   102 1 1 1 1 1 1 1 0 1  1  1  0  1  0  0  0
##   110 1 1 1 1 1 1 1 0 1  1  0  1  1  1  0  0
##   113 1 1 1 1 1 1 1 1 1  1  1  1  0  0  0  0
##   126 1 1 0 1 1 1 1 1 1  1  1  1  0  0  0  0
##   131 0 1 1 1 0 1 1 0 1  1  1  1  0  0  0  0
##   138 1 1 1 1 1 1 1 1 1  1  1  1  1  1  1  1
##   147 1 0 1 1 1 1 1 1 1  1  1  1  1  1  1  0
##   157 1 1 1 1 1 1 1 1 0  0  0  0  0  0  0  0
##   167 0 1 1 1 1 1 0 1 1  1  1  1  0  0  0  0
##   171 1 0 1 1 1 1 1 1 1  0  1  0  0  0  0  0
##   178 1 1 1 1 1 1 1 0 1  0  0  1  0  0  0  0
##   191 1 1 1 1 1 1 1 1 1  1  0  1  0  1  1  1
##   194 0 1 0 1 2 0 1 1 1  1  1  1  1  1  1  1
##   204 1 1 1 1 1 0 1 1 1  1  1  1  1  1  1  1
##   212 1 1 1 1 1 1 1 1 1  1  1  1  1  0  0  0
##   213 1 1 1 1 1 1 1 1 1  1  0  1  1  0  0  0
##   230 1 1 1 1 1 1 1 1 1  1  1  1  1  1  1  1
##   240 1 1 1 1 1 1 1 1 1  1  1  1  1  1  1  1
##   245 1 1 1 1 1 1 1 0 1  0  0  1  1  1  1  1
##   256 1 1 1 1 1 1 1 1 1  0  0  1  1  1  1  0
##   261 1 1 1 1 1 0 1 1 1  1  1  1  1  1  0  0
##   276 1 1 1 0 1 1 1 1 0  1  1  0  0  1  1  0
##   283 1 1 1 1 1 1 1 1 1  0  1  1  0  1  1  1
##   293 1 1 1 1 1 1 1 1 0  1  1  1  1  2  0  1
##   302 1 1 1 1 1 1 1 0 1  2  0  1  1  1  0  0
##   312 1 1 1 1 1 1 2 1 1  1  1  1  1  0  0  0
##   315 1 1 1 1 1 1 1 1 0  0  0  0  0  1  0  0
```

Here, we perform the linkage to the emotion data and check the result:

```
#* link (merge) together *#
D_sm_linked<-merge(math_dat, #<-aggregated strategies data
                   D_sm,     #<-smoothed emotion data
                   by=c("ID","Session")) %>%
              arrange(ID,GRA,Session)
#* recast a few variables as factors *#
D_sm_linked$ID<-as.factor(D_sm_linked$ID)
D_sm_linked$GRA<-as.factor(D_sm_linked$GRA)
```

```
D_sm_linked$Gender<-as.factor(D_sm_linked$Gender)

#* take a peek at first 10 columns of the result *#
glimpse(D_sm_linked[,1:10])
```

```
## Rows: 67
## Columns: 10
## $ ID       <fct> 046, 046, 046, 046, 049, 049, 049, 054, 054, 054, 054, 066, 0~
## $ Session  <dbl> 7, 3, 14, 13, 6, 3, 14, 4, 2, 11, 12, 2, 5, 6, 10, 11, 9, 2, ~
## $ GRA      <fct> Candace, Courtney, Jason, Taryn, Jason, Sara, Sara, Courtney,~
## $ avg_acc  <dbl> 0.8095238, 0.6666667, 0.5263158, 0.6400000, 0.6000000, 0.5714~
## $ acc_mod  <dbl> 0.8095238, 0.6666667, 0.5263158, 0.6400000, 0.6000000, 0.5714~
## $ tot_att  <int> 21, 30, 19, 25, 10, 7, 12, 16, 31, 21, 12, 23, 25, 30, 32, 27~
## $ tot_acc  <dbl> 17, 20, 10, 16, 6, 4, 8, 12, 27, 13, 8, 15, 15, 19, 23, 13, 1~
## $ max_soph <dbl> 8, 8, 8, NA, 5, 4, NA, 9, NA, 9, 6, NA, NA, 8, 8, 8, 7, NA, N~
## $ breadth  <int> 4, 5, 4, 7, 2, 1, 4, 4, 6, 1, 1, 5, 7, 5, 4, 6, 5, 6, 6, 1, 3~
## $ Gender   <fct> Male, Male, Male, Male, Female, Female, Female, Male, Male, M~
```

```
#* emotional intensity starts at column 10
glimpse(D_sm_linked[,1:20])
```

```
## Rows: 67
## Columns: 20
## $ ID       <fct> 046, 046, 046, 046, 049, 049, 049, 054, 054, 054, 054, 066, 0~
## $ Session  <dbl> 7, 3, 14, 13, 6, 3, 14, 4, 2, 11, 12, 2, 5, 6, 10, 11, 9, 2, ~
## $ GRA      <fct> Candace, Courtney, Jason, Taryn, Jason, Sara, Sara, Courtney,~
## $ avg_acc  <dbl> 0.8095238, 0.6666667, 0.5263158, 0.6400000, 0.6000000, 0.5714~
## $ acc_mod  <dbl> 0.8095238, 0.6666667, 0.5263158, 0.6400000, 0.6000000, 0.5714~
## $ tot_att  <int> 21, 30, 19, 25, 10, 7, 12, 16, 31, 21, 12, 23, 25, 30, 32, 27~
## $ tot_acc  <dbl> 17, 20, 10, 16, 6, 4, 8, 12, 27, 13, 8, 15, 15, 19, 23, 13, 1~
## $ max_soph <dbl> 8, 8, 8, NA, 5, 4, NA, 9, NA, 9, 6, NA, NA, 8, 8, 8, 7, NA, N~
## $ breadth  <int> 4, 5, 4, 7, 2, 1, 4, 4, 6, 1, 1, 5, 7, 5, 4, 6, 5, 6, 6, 1, 3~
## $ Gender   <fct> Male, Male, Male, Male, Female, Female, Female, Male, Male, M~
## $ `1`      <dbl> -0.4849459, -1.3283726, -5.3119456, -2.1310723, -2.7191097, -~
## $ `2`      <dbl> -1.0093779202, -1.3320080101, -4.8987557025, -2.1016113699, -~
## $ `3`      <dbl> -1.7129192, -1.3562745, -4.4193120, -2.1314186, -4.1225555, -~
## $ `4`      <dbl> -2.5698816, -1.4776421, -3.9284464, -2.2534862, -4.7091023, -~
## $ `5`      <dbl> -3.4699918, -1.8752924, -3.4788389, -2.4618434, -5.0788601, -~
## $ `6`      <dbl> -4.2555446, -2.5660302, -3.0845467, -2.6043922, -5.3036053, -~
## $ `7`      <dbl> -4.8176566, -3.1803213, -2.6994103, -2.4766499, -5.4316598, -~
## $ `8`      <dbl> -5.1437801, -3.5549139, -2.2972351, -2.0554754, -5.5007112, -~
## $ `9`      <dbl> -5.2793788, -3.7736575, -1.9727366, -1.4307580, -5.5650587, -~
## $ `10`     <dbl> -5.2633240, -3.9033558, -1.7524058, -0.7404074, -5.6722871, -~
```

Note that 3 sessions out of the 70 could not be linked and so we have 67 sessions left for which linkage was successful.

Next, we retain only the emotion intensity columns and save them as a functional object (functional covariates) for functional regression analysis. Some re-arranging of the data is needed to make the final analysis dataset (D_fin):

```
#* make emotional intensity functional covariate & analysis dataset *#
Y_f<-D_sm_linked[,c(-1:-10)] %>% as.matrix()
D_fin<-data.frame(D_sm_linked[,1:10],I(Y_f)) %>%
       arrange(ID,Session)
#* take a peek at first 10 columns of the analysis file *#
```

```r
glimpse(D_fin[,1:9])
```

```
## Rows: 67
## Columns: 9
## $ ID       <fct> 046, 046, 046, 046, 049, 049, 049, 054, 054, 054, 054, 066, 0~
## $ Session  <dbl> 3, 7, 13, 14, 3, 6, 14, 2, 4, 11, 12, 2, 5, 6, 9, 10, 11, 2, ~
## $ GRA      <fct> Courtney, Candace, Taryn, Jason, Sara, Jason, Sara, Jason, Co~
## $ avg_acc  <dbl> 0.6666667, 0.8095238, 0.6400000, 0.5263158, 0.5714286, 0.6000~
## $ acc_mod  <dbl> 0.6666667, 0.8095238, 0.6400000, 0.5263158, 0.5714286, 0.6000~
## $ tot_att  <int> 30, 21, 25, 19, 7, 10, 12, 31, 16, 21, 12, 23, 25, 30, 19, 32~
## $ tot_acc  <dbl> 20, 17, 16, 10, 4, 6, 8, 27, 12, 13, 8, 15, 15, 19, 15, 23, 1~
## $ max_soph <dbl> 8, 8, NA, 8, 4, 5, NA, NA, 9, 9, 6, NA, NA, 8, 7, 8, 8, NA, N~
## $ breadth  <int> 5, 4, 7, 4, 1, 2, 4, 6, 4, 1, 1, 5, 7, 5, 5, 4, 6, 6, 6, 1, 3~
```

Functional Regression Analysis

Baseline model

Here, we specify and estimate a baseline (non-functional) regression model and a function-on-scalar regression. Emotional intensity is a complex covariate, but the baseline model is nested within the functional model, so the two models can be tested via Likelihood Ratio Test and compared via AIC or BIC. We note that our final sample size is 67; it doesn't matter how many time points we use for encoded emotions since the entire curve is used as a single covariate using the lf() regression term.

The response variable is not functional (ie is a scalar) and it is the average accuracy = proportion of correct attempts per session. Because the response is bounded between 0 and 1, a beta regression must be used. We note that we use the "modified" accuracy (average accuracy - 1e-8) to avoid 100% accuracy, which cannot be accommodated by the beta regression.

Baseline (non-functional) model, which includes only the subject random intercept:

```r
library(refund)
library(mgcv)
library(lmtest)

#* baseline accuracy *#
fit0<-pfr(acc_mod ~ re(ID),
          family=betar(),
          method="REML",data=D_fin)
summary(fit0)
```

```
##
## Family: Beta regression(4.557)
## Link function: logit
##
## Formula:
## acc_mod ~ s(x = ID, bs = "re")
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.9111     0.2106   4.325 1.52e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
```

```
##         edf Ref.df Chi.sq p-value
## s(ID) 10.13     14  43.41  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.00726   Deviance explained = 55.8%
## -REML = -36.627  Scale est. = 1         n = 67
```

A model with only the student random intercepts explains 55.8% of the variability in session accuracy.

Now we add the joy emotional intensity functional covariate through lf() and compare vs. the baseline model using a Likelihood Ratio Test (LRT):

```
#* emotion functional covariate *#
fit1<-pfr(acc_mod ~ re(ID) +
            lf(Y_f,argvals = seq(min(t),max(t),length=ncol(Y_f))),
          family=betar(),
          method="REML",data=D_fin)
summary(fit1)
```

```
##
## Family: Beta regression(5.073)
## Link function: logit
##
## Formula:
## acc_mod ~ s(x = ID, bs = "re") + s(x = Y_f.tmat, by = L.Y_f)
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.3817     0.2915   1.309     0.19
##
## Approximate significance of smooth terms:
##                     edf Ref.df Chi.sq  p-value
## s(ID)             8.143 14.000  21.32 0.000775 ***
## s(Y_f.tmat):L.Y_f 4.864  5.663  18.03 0.006749 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  -0.0773   Deviance explained = 65.1%
## -REML = -29.974  Scale est. = 1         n = 67
```

```
#* Likelihood Ratio Test
lmtest::lrtest(fit0,fit1)
```

```
## Likelihood ratio test
##
## Model 1: acc_mod ~ s(x = ID, bs = "re")
## Model 2: acc_mod ~ s(x = ID, bs = "re") + s(x = Y_f.tmat, by = L.Y_f)
##      #Df LogLik     Df  Chisq Pr(>Chisq)
## 1 12.960 52.226
## 2 17.121 59.578 4.1616 14.702    0.00536 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

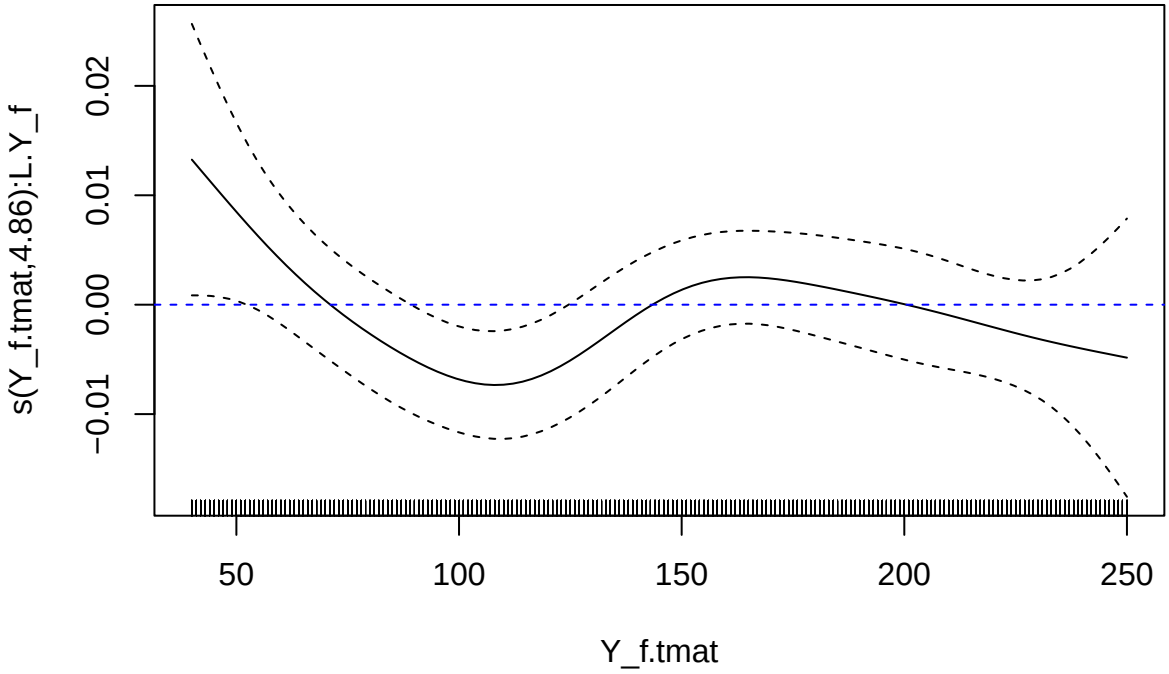
```
#* AIC comparison
AIC(fit0,fit1)
```

```
##            df       AIC
## fit0 12.95983 -78.53307
## fit1 17.12147 -84.91224
```

Joy emotional intensity over 40-250 seconds explains an additional 9.3% of the variability in accuracy. The LRT reveals the contribution of the functional emotional intensity covariate is statistically significant (p=0.0054); AIC of the model with emotional intensity is lower (-84.91 vs. -78.53), indicating the preferred model includes emotional intensity.

We now examine the effect function, or the effect of joy on accuracy over 40-250 seconds. This figure is Figure 1 Panel A:

```
#* plot the effect of joy on session accuracy
plot(fit1,select=2)
abline(h=0,col='blue',lty=2)
```



Here, it appears that more intense initial joy correlates with higher session accuracy, whereas more intense joy during the 2nd minute of the session correlates with lower session accuracy.

Now, we can add the session effect to account for passage of time in the intervention and dosage of instruction:

```
#* baseline + Session effect *#
fit2<-pfr(acc_mod ~ re(ID) + scale(Session),
          family=betar(),
          method="REML",data=D_fin)
summary(fit2)
```

```
##
## Family: Beta regression(5.028)
## Link function: logit
```

```
##
## Formula:
## acc_mod ~ s(x = ID, bs = "re") + scale(Session)
##
## Parametric coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)      0.9015     0.1879   4.798 1.60e-06 ***
## scale(Session)   0.4314     0.1108   3.895 9.81e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##         edf Ref.df Chi.sq  p-value
## s(ID) 9.397     14  31.32 2.72e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  -0.0408   Deviance explained = 62.8%
## -REML = -42.528  Scale est. = 1         n = 67
```

```
#* emotion functional covariate + Session effect *#
fit2em<-pfr(acc_mod ~ re(ID) + scale(Session) +
            lf(Y_f,argvals = seq(min(t),max(t),length=ncol(Y_f))),
          family=betar(),
          method="REML",data=D_fin)
summary(fit2em)
```

```
##
## Family: Beta regression(5.381)
## Link function: logit
##
## Formula:
## acc_mod ~ s(x = ID, bs = "re") + scale(Session) + s(x = Y_f.tmat,
##     by = L.Y_f)
##
## Parametric coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)      0.6032     0.2983   2.022  0.04314 *
## scale(Session)   0.3382     0.1217   2.778  0.00547 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                     edf Ref.df Chi.sq  p-value
## s(ID)             8.348 14.000 21.810 0.000813 ***
## s(Y_f.tmat):L.Y_f 4.306  4.983  9.017 0.098936 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  -0.0684   Deviance explained =   68%
## -REML = -32.17  Scale est. = 1         n = 67
```

```
#* Likelihood Ratio Test
lmtest::lrtest(fit2,fit2em)
```

```
## Likelihood ratio test
```

```
##
## Model 1: acc_mod ~ s(x = ID, bs = "re") + scale(Session)
## Model 2: acc_mod ~ s(x = ID, bs = "re") + scale(Session) + s(x = Y_f.tmat,
##     by = L.Y_f)
##      #Df LogLik     Df Chisq Pr(>Chisq)
## 1 13.424 57.649
## 2 17.833 62.189 4.4092 9.079    0.05915 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#* AIC comparison
AIC(fit2,fit2em)
```

```
##              df       AIC
## fit2   13.42360 -88.45084
## fit2em 17.83285 -88.71134
```

If we include the session effect, the emotional intensity functional covariate is no longer technically statistically significant (LRT p-value=0.0592). AICs are nearly identical, which indicates that the inclusion of emotional intensity is not supported by the data, adjusted for session.