Introduction to Statistics with R

Session R06: Eye Tracking

Marvin Schmitt

List files

```
filenames = list.files(path = "R06_notes_dataset")
filenames
```

```
##  [1] "R06_trial_1.csv"  "R06_trial_10.csv" "R06_trial_11.csv" "R06_trial_12.csv"
##  [5] "R06_trial_13.csv" "R06_trial_14.csv" "R06_trial_15.csv" "R06_trial_16.csv"
##  [9] "R06_trial_17.csv" "R06_trial_18.csv" "R06_trial_19.csv" "R06_trial_2.csv"
## [13] "R06_trial_20.csv" "R06_trial_21.csv" "R06_trial_22.csv" "R06_trial_23.csv"
## [17] "R06_trial_24.csv" "R06_trial_25.csv" "R06_trial_26.csv" "R06_trial_27.csv"
## [21] "R06_trial_28.csv" "R06_trial_29.csv" "R06_trial_3.csv"  "R06_trial_30.csv"
## [25] "R06_trial_31.csv" "R06_trial_32.csv" "R06_trial_33.csv" "R06_trial_34.csv"
## [29] "R06_trial_35.csv" "R06_trial_36.csv" "R06_trial_37.csv" "R06_trial_38.csv"
## [33] "R06_trial_39.csv" "R06_trial_4.csv"  "R06_trial_40.csv" "R06_trial_41.csv"
## [37] "R06_trial_42.csv" "R06_trial_43.csv" "R06_trial_44.csv" "R06_trial_45.csv"
## [41] "R06_trial_46.csv" "R06_trial_47.csv" "R06_trial_48.csv" "R06_trial_49.csv"
## [45] "R06_trial_5.csv"  "R06_trial_50.csv" "R06_trial_51.csv" "R06_trial_52.csv"
## [49] "R06_trial_53.csv" "R06_trial_54.csv" "R06_trial_55.csv" "R06_trial_56.csv"
## [53] "R06_trial_57.csv" "R06_trial_58.csv" "R06_trial_59.csv" "R06_trial_6.csv"
## [57] "R06_trial_60.csv" "R06_trial_61.csv" "R06_trial_62.csv" "R06_trial_63.csv"
## [61] "R06_trial_64.csv" "R06_trial_65.csv" "R06_trial_66.csv" "R06_trial_67.csv"
## [65] "R06_trial_68.csv" "R06_trial_69.csv" "R06_trial_7.csv"  "R06_trial_70.csv"
## [69] "R06_trial_8.csv"  "R06_trial_9.csv"
```

Construct custom text

```
group = "R seminar group"
paste("Hello", "dear", group, "!!", sep="___")
```

```
## [1] "Hello___dear___R seminar group___!!"
```

```r
filename = "data_1.csv"
paste("data/", filename, sep="")
```

```
## [1] "data/data_1.csv"
```

for loops

```
for (letter in c("A", "b", "C")){
  print(letter)
}
```

```
## [1] "A"
## [1] "b"
## [1] "C"
```

```
for (i in 1:10){
  print(i ** 2)
}
```

```
## [1] 1
## [1] 4
## [1] 9
## [1] 16
## [1] 25
## [1] 36
## [1] 49
## [1] 64
## [1] 81
## [1] 100
```

```r
values = 1:10
values_sum = 0
for (value in values){
  values_sum = values_sum + value
}
print(values_sum)
```

```
## [1] 55
```

```r
print(sum(values))      # check that result is equal to sum(...)
```

```
## [1] 55
```

Putting it all together

```
filenames = list.files(path = "R06_notes_dataset")
for (filename in filenames){
  full_filename = paste("R06_notes_dataset", filename, sep="")
  print(full_filename)
}
```

```
## [1] "R06_notes_datasetR06_trial_1.csv"
## [1] "R06_notes_datasetR06_trial_10.csv"
## [1] "R06_notes_datasetR06_trial_11.csv"
## [1] "R06_notes_datasetR06_trial_12.csv"
## [1] "R06_notes_datasetR06_trial_13.csv"
## [1] "R06_notes_datasetR06_trial_14.csv"
## [1] "R06_notes_datasetR06_trial_15.csv"
## [1] "R06_notes_datasetR06_trial_16.csv"
## [1] "R06_notes_datasetR06_trial_17.csv"
## [1] "R06_notes_datasetR06_trial_18.csv"
## [1] "R06_notes_datasetR06_trial_19.csv"
## [1] "R06_notes_datasetR06_trial_2.csv"
## [1] "R06_notes_datasetR06_trial_20.csv"
## [1] "R06_notes_datasetR06_trial_21.csv"
## [1] "R06_notes_datasetR06_trial_22.csv"
## [1] "R06_notes_datasetR06_trial_23.csv"
## [1] "R06_notes_datasetR06_trial_24.csv"
## [1] "R06_notes_datasetR06_trial_25.csv"
## [1] "R06_notes_datasetR06_trial_26.csv"
## [1] "R06_notes_datasetR06_trial_27.csv"
```

```r
filenames = list.files(path = "R06_notes_dataset")
first_full_filename = paste("R06_notes_dataset/", filenames[1], sep="")
df = read.csv(first_full_filename) # start off with first data set
for (filename in filenames[-1]){   # exclude first data set
  full_filename = paste("R06_notes_dataset/", filename, sep="") # construct nam
  df_temp = read.csv(full_filename)  # read in the current "temporary" data
  df = rbind(df, df_temp)    # append current data
}
```

Remove the temporaty data and **select** only relevant columns.

```
remove(df_temp)
```

```
df = df %>%
  select(trial, TIMESTAMP, TRIAL_START_TIME, sound, condition, direction,
         AVERAGE_GAZE_X, AVERAGE_GAZE_Y)
```

Inspecting the (huge) data

```
str(df)
```

```
## 'data.frame':    176784 obs. of  8 variables:
##  $ trial           : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ TIMESTAMP       : num  2780484 2780486 2780488 2780490 2780492 ...
##  $ TRIAL_START_TIME: int  2780484 2780484 2780484 2780484 2780484 2780484 2780484 2780484 2780484 2780
##  $ sound           : chr  "sieden.wav" "sieden.wav" "sieden.wav" "sieden.wav" ...
##  $ condition       : chr  "control" "control" "control" "control" ...
##  $ direction       : chr  "null" "null" "null" "null" ...
##  $ AVERAGE_GAZE_X  : chr  "974.15" "972.35" "971.75" "971.80" ...
##  $ AVERAGE_GAZE_Y  : chr  "544.75" "544.75" "545.00" "545.10" ...
```

Issues:

Convert coordinates to numbers

```
as.numeric(df$AVERAGE_GAZE_X) %>% head()
```

```
## [1] 974.15 972.35 971.75 971.80 972.00 971.50
```

```
na_mask = as.numeric(df$AVERAGE_GAZE_X) %>% is.na()
df[na_mask, ] %>% head()
```

```
##      trial TIMESTAMP TRIAL_START_TIME      sound condition direction
## 1510     1   2783502          2780484 sieden.wav   control      null
## 1511     1   2783504          2780484 sieden.wav   control      null
## 1512     1   2783506          2780484 sieden.wav   control      null
## 1513     1   2783508          2780484 sieden.wav   control      null
## 1514     1   2783510          2780484 sieden.wav   control      null
## 1515     1   2783512          2780484 sieden.wav   control      null
##      AVERAGE_GAZE_X AVERAGE_GAZE_Y
## 1510              .              .
## 1511              .              .
## 1512              .              .
## 1513              .              .
## 1514              .              .
## 1515              .              .
```

```
df$AVERAGE_GAZE_X = as.numeric(df$AVERAGE_GAZE_X)
df$AVERAGE_GAZE_Y = as.numeric(df$AVERAGE_GAZE_Y)
```

```
is.na(df) %>% head()
```

```
##      trial TIMESTAMP TRIAL_START_TIME sound condition direction AVERAGE_GAZE_X
## [1,] FALSE     FALSE            FALSE FALSE     FALSE     FALSE          FALSE
## [2,] FALSE     FALSE            FALSE FALSE     FALSE     FALSE          FALSE
## [3,] FALSE     FALSE            FALSE FALSE     FALSE     FALSE          FALSE
## [4,] FALSE     FALSE            FALSE FALSE     FALSE     FALSE          FALSE
## [5,] FALSE     FALSE            FALSE FALSE     FALSE     FALSE          FALSE
## [6,] FALSE     FALSE            FALSE FALSE     FALSE     FALSE          FALSE
##      AVERAGE_GAZE_Y
## [1,]          FALSE
## [2,]          FALSE
## [3,]          FALSE
## [4,]          FALSE
## [5,]          FALSE
## [6,]          FALSE
```

```
is.na(df) %>% sum()
```

```
## [1] 6866
```

```
df = na.omit(df)
is.na(df) %>% sum()
```

```
## [1] 0
```

```
xtabs(~ trial + condition, df)
```

```
##       condition
## trial control horizontal vertical
##     1    2291          0        0
##     2    2519          0        0
##     3    2532          0        0
##     4    2207          0        0
##     5    2366          0        0
##     6    2530          0        0
##     7    2519          0        0
##     8    2524          0        0
##     9    2530          0        0
##     10   2531          0        0
##     11   2517          0        0
##     12   2446          0        0
##     13   2530          0        0
##     14   2527          0        0
##     15   2524          0        0
##     16   2517          0        0
##     17   2533          0        0
##     18   2406          0        0
##     19   2530          0        0
##     20   2489          0        0
##     21   2404          0        0
##     22   2525          0        0
##     23   2435          0        0
##     24   2517          0        0
##     25   2524          0        0
##     26   2396          0        0
##     27   2530          0        0
##     28   2525          0        0
```

```
xtabs(~ condition + direction, df)
```

```
##             direction
## condition      down leftright   null     up
##   control         0         0 123856      0
##   horizontal      0     24778      0      0
##   vertical    14613         0      0  10068
```

Turn factors into `factors`

```
df$sound = as.factor(df$sound)
df$condition = as.factor(df$condition)
df$direction = as.factor(df$direction)
```

Normalize XY coordinates

```
min_X = 0     # customize to match window width!
max_X = 1920     # customize to match window width!
df$AVERAGE_GAZE_X = (df$AVERAGE_GAZE_X - min_X) / (max_X - min_X)

min_Y = 0     # customize to match window height!
max_Y = 1080     # customize to match window height!
df$AVERAGE_GAZE_Y = (df$AVERAGE_GAZE_Y - min_Y) / (max_Y - min_Y)
```

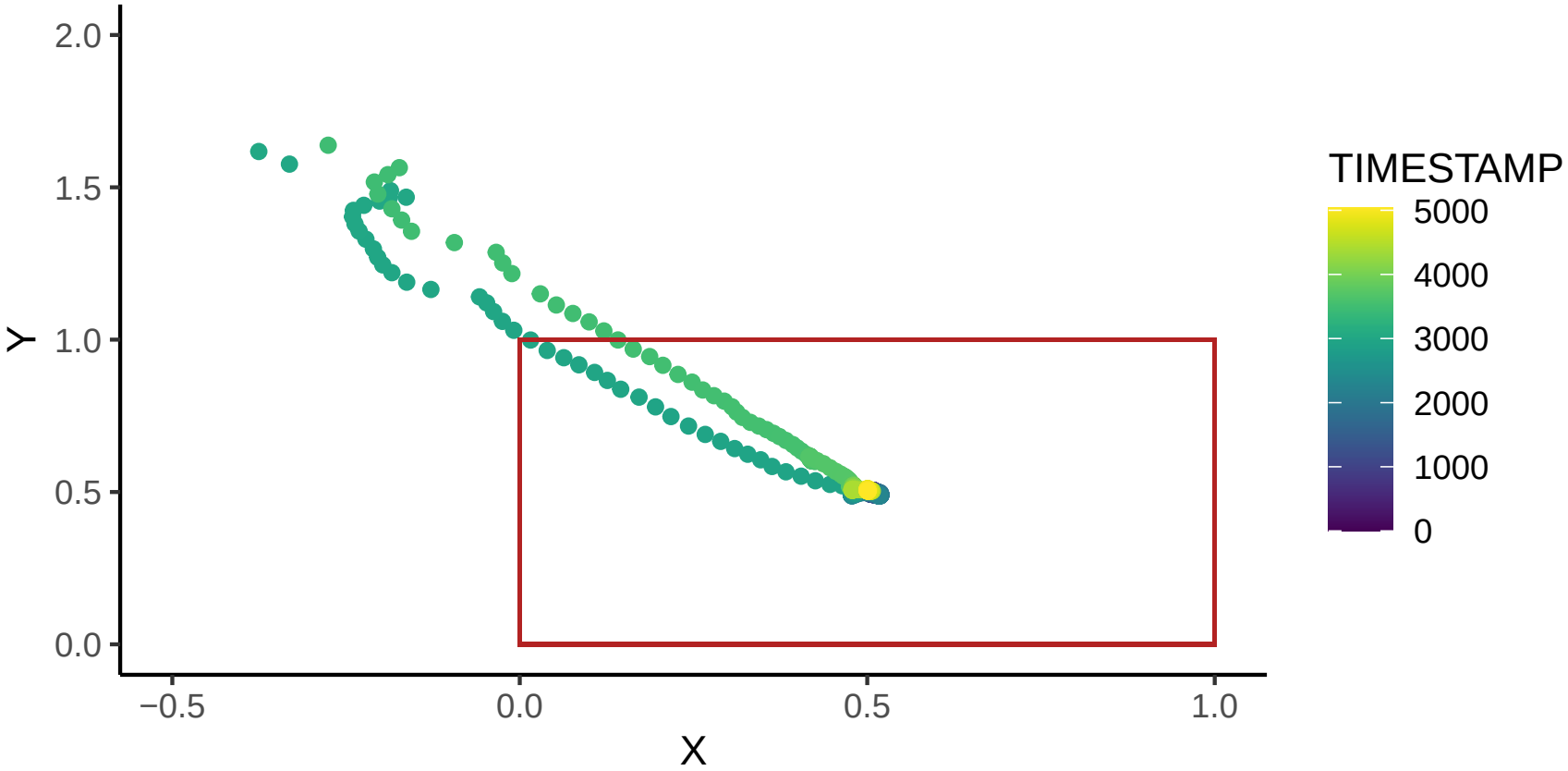
Visualize a path

```
df_path = df %>%
  filter(trial==1)

df_path$TIMESTAMP = df_path$TIMESTAMP - df_path$TRIAL_START_TIME
df_path$TIMESTAMP[1:20]
```

```
##  [1]  0  2  4  6  8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38
```

```
library(viridis)
ggplot(df_path, aes(x=AVERAGE_GAZE_X, y=AVERAGE_GAZE_Y)) +
  geom_point(aes(color=TIMESTAMP)) + xlim(-0.5, 1) + ylim(0, 2) +
  geom_rect(xmin=0, xmax=1, ymin=0, ymax=1, alpha=0.0, color="firebrick")+
  scale_color_viridis() + theme_classic() + labs(x="X", y="Y")
```



The eyetrackingR library

The library eyetrackingR provides numerous functions for common eye-tracking analyses. The documentation with many examples is hosted at http://www.eyetracking-r.com/. The library comes with an example data set word_recognition. That data comes from a 2-alternative forced choice word recognition task of infants who looked at animate vs. inanimate objects.

```
library(eyetrackingR)
data("word_recognition")
nrow(word_recognition)
```

```
## [1] 195912
```

```
colnames(word_recognition)
```

```
##  [1] "ParticipantName"       "Sex"                   "Age"
##  [4] "TrialNum"              "Trial"                 "TimeFromTrialOnset"
##  [7] "Subphase"              "TimeFromSubphaseOnset" "AOI"
## [10] "Animate"               "Inanimate"             "TrackLoss"
## [13] "MCDI_Total"            "MCDI_Nouns"            "MCDI_Verbs"
```

We can turn observational eye-tracking data into a well-specified format for `eyetrackingR` with the function `make_eyetrackingr_data`:

```
data = make_eyetrackingr_data(word_recognition,
                         participant_column = "ParticipantName",
                         trial_column = "Trial",
                         time_column = "TimeFromTrialOnset",
                         trackloss_column = "TrackLoss",
                         aoi_columns = c('Animate','Inanimate'),
                         treat_non_aoi_looks_as_missing = TRUE
)
```

More info is documented at http://www.eyetracking-r.com/vignettes/preparing_your_data

A *response window analyses* assesses eye-movements in a fixed time window, e.g. between 15.5 and 21 seconds after the trial started:

```
response_window <- subset_by_window(data,
                                    window_start_time = 15500,
                                    window_end_time = 21000,
                                    rezero = FALSE) # first frame in window = 0
```

```
## Avg. window length in new data will be 5500
```

We can exclude trials that have more trackloss than a defined threshold:

```
response_window_clean <- clean_by_trackloss(data = response_window,
                                            trial_prop_thresh = .25)
```

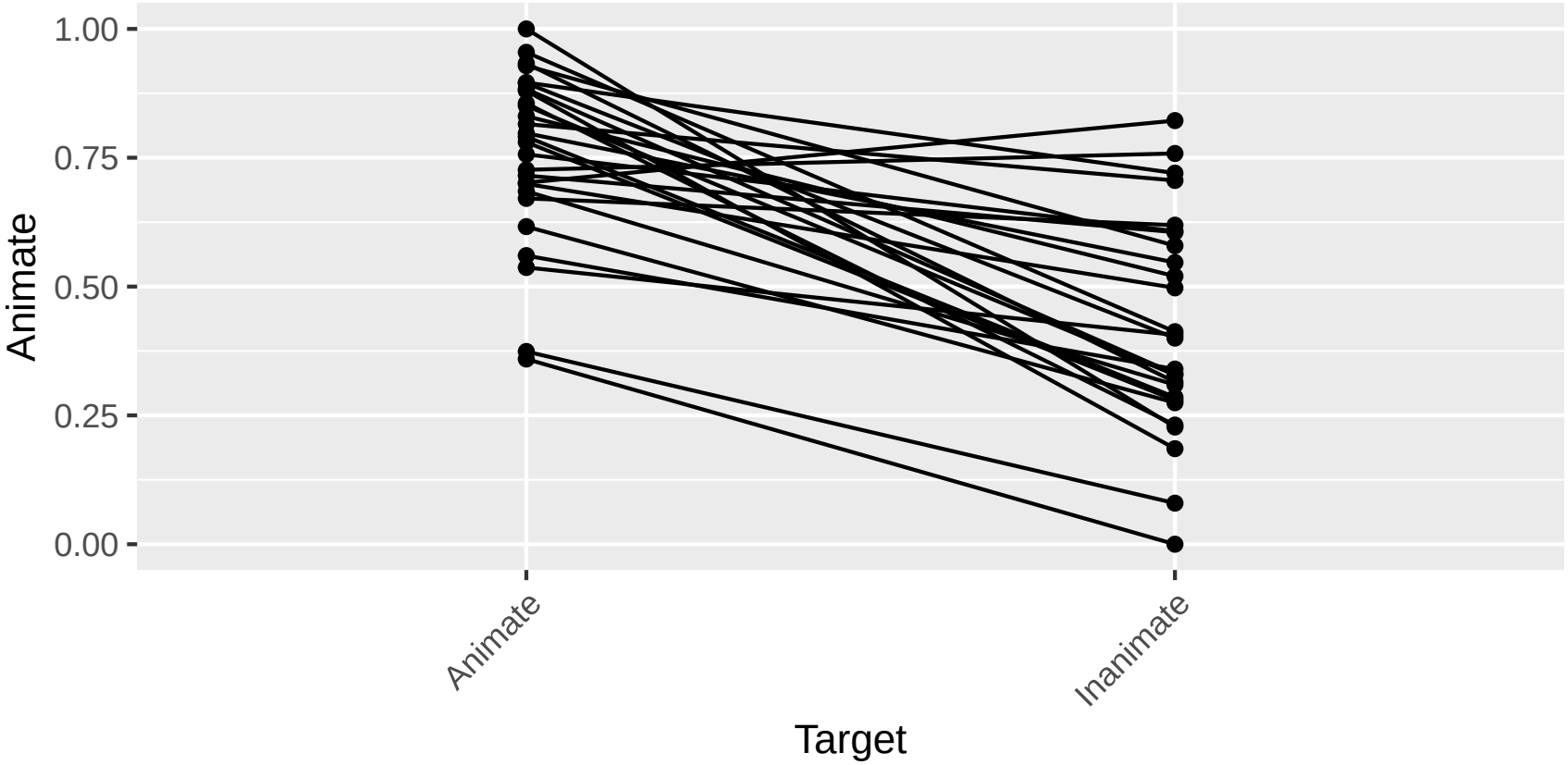
```
## Performing Trackloss Analysis...
## Will exclude trials whose trackloss proportion is greater than : 0.25
##  ...removed  33  trials.
```

We recode the Target variable to give information on congruency with the instruction:

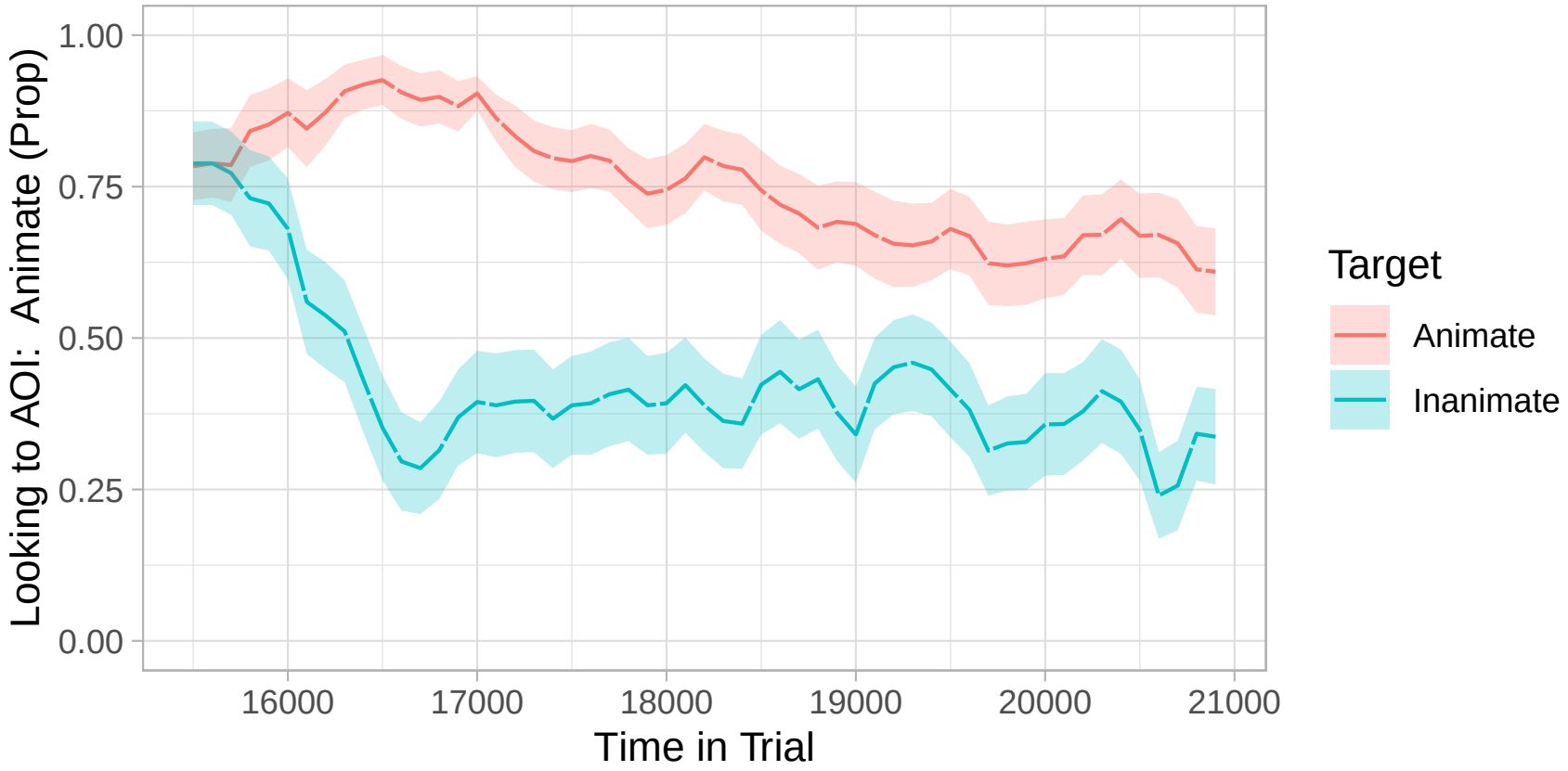
```
response_window_clean$Target = as.factor(
  ifelse(test = grepl('(Spoon|Bottle)',
                      response_window_clean$Trial),
         yes = 'Inanimate',
         no  = 'Animate') )
```

We can briefly visualize the average looking time at the animate objects depending on what object type was named:

```
data_summary = describe_data(response_window_clean, describe_column='Animate',
plot(data_summary)
```



```r
# aggregate across trials within subjects in time analysis
response_time = make_time_sequence_data(response_window_clean, time_bin_size=10
  predictor_columns = c("Target"), aois = "Animate")
# visualize time results
plot(response_time, predictor_column = "Target") +
  theme_light() + coord_cartesian(ylim = c(0,1))
```



Summary

- In order to read data from multiple input files, we can use a combination of
  - `list.files()` to dynamically find all files in a given directory,
  - `paste()` to construct the file names, and
  - `for`-loops to sequentially bind the individual files to a joint data frame.
- For visualization and statistical analyses, the `eyetrackingR` library implements numerous analyses
  - The data set needs to meet a defined format
  - You can use your acquired data manipulation skills (and Google) to prepare your data set accordingly