1st homework prep

```
library("dplyr")
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library("ggplot2")
library("tidyverse")
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v tibble  2.1.3     v purrr   0.3.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

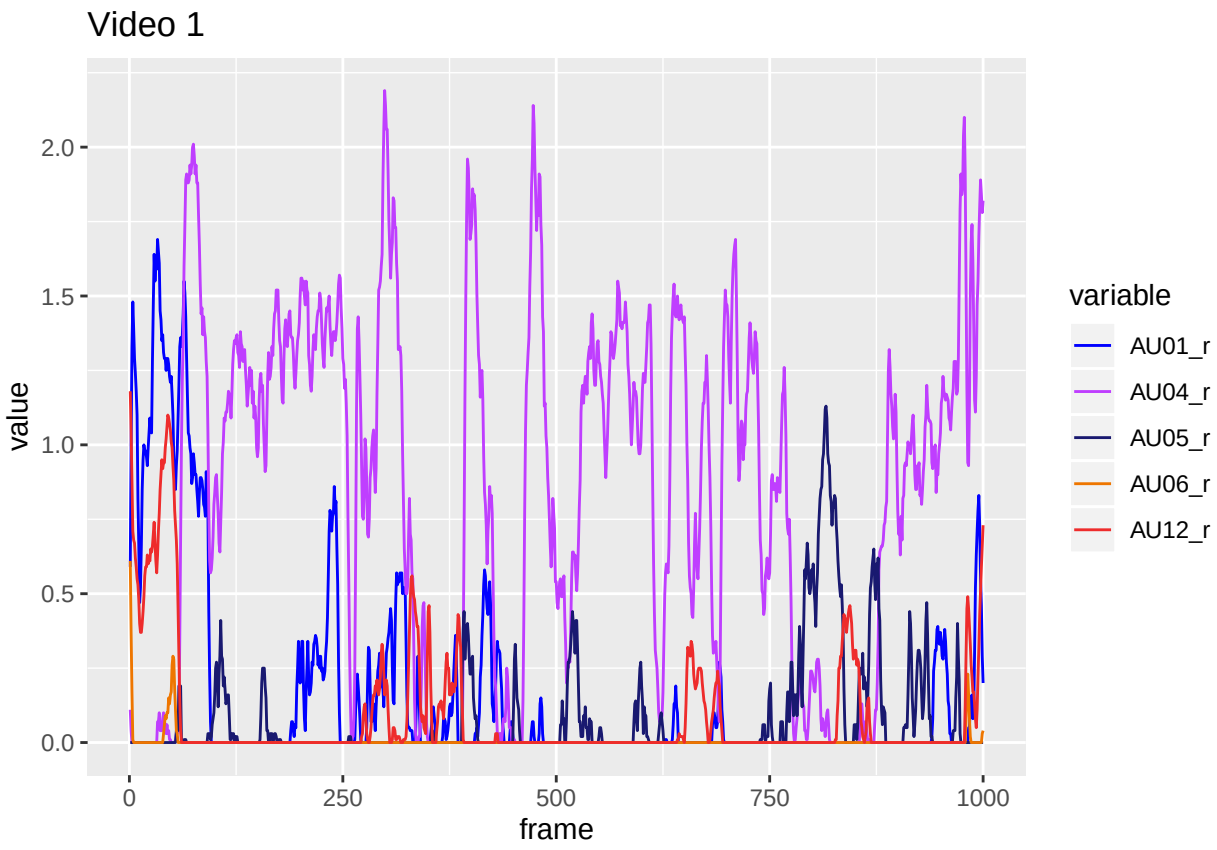
Hyptothesis 1: Relation between positive and negative emotions

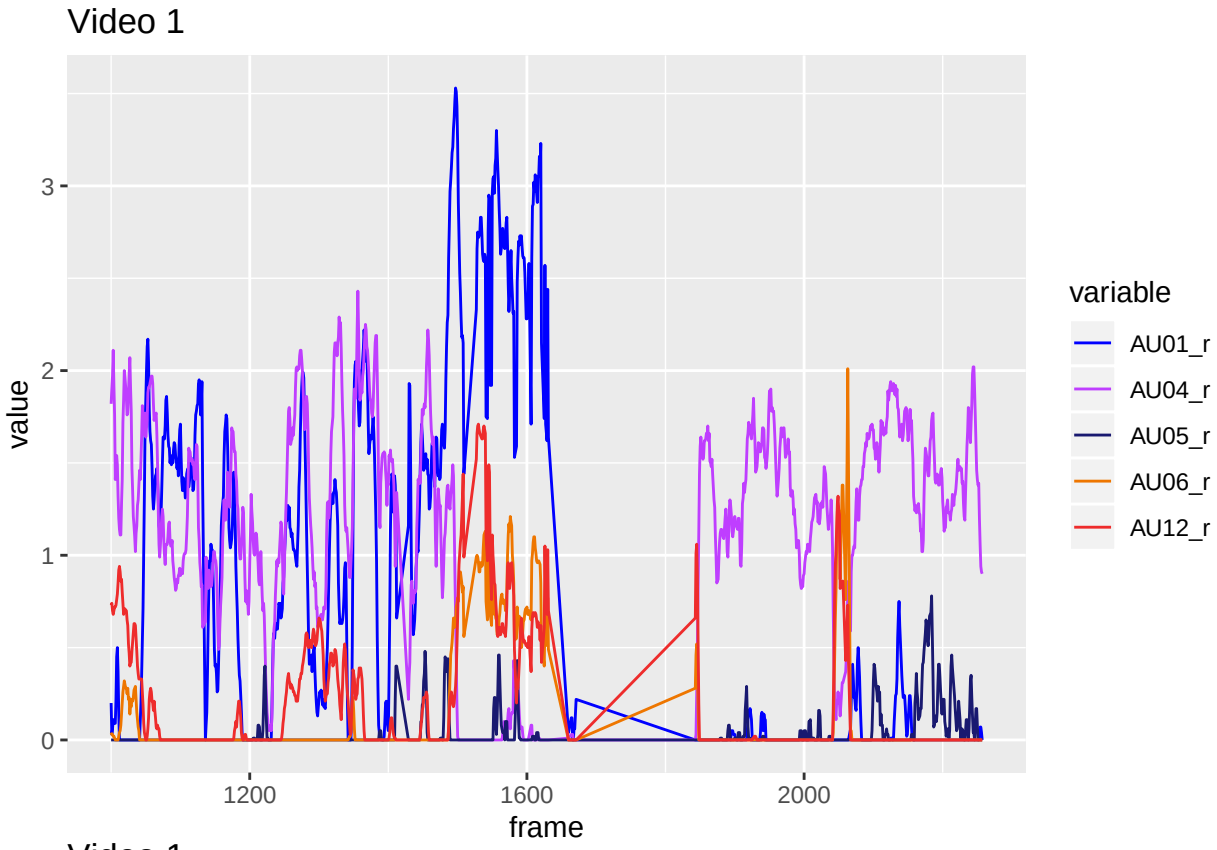
```
video1 <- read.csv("NeuroDesignVid01.csv")
video2 <- read.csv("NeuroDesignVid02.csv")
```

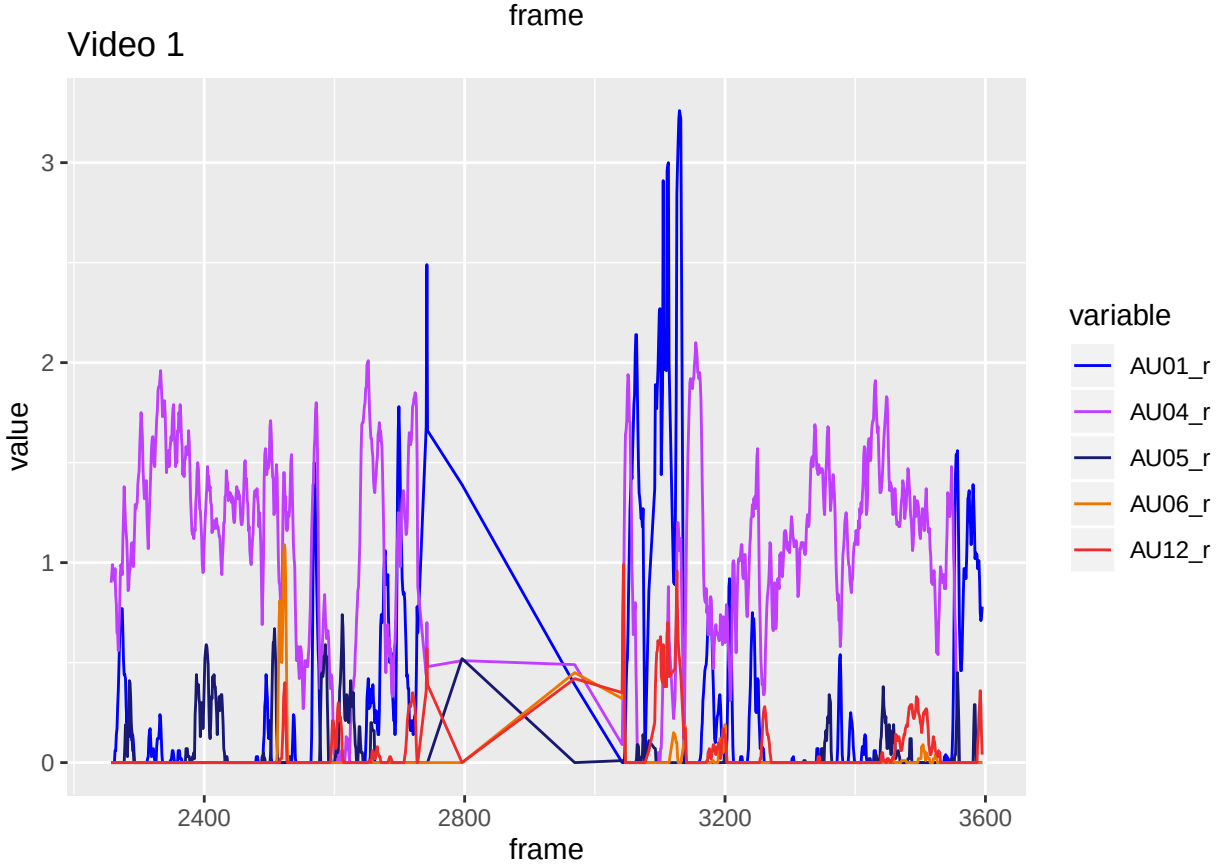
```
# remove all the frames where the face presumably couldn't be captured well enough
video1 %>%
  select(frame, timestamp, confidence, AU01_c, AU01_r, AU04_c, AU04_r, AU05_c, AU05_r, AU06_c, AU06_r, A
  filter(confidence > 0.7) -> video1

video2 %>%
  select(frame, timestamp, confidence, AU01_c, AU01_r, AU04_c, AU04_r, AU05_c, AU05_r, AU06_c, AU06_r, A
  filter(confidence > 0.7) -> video2
```

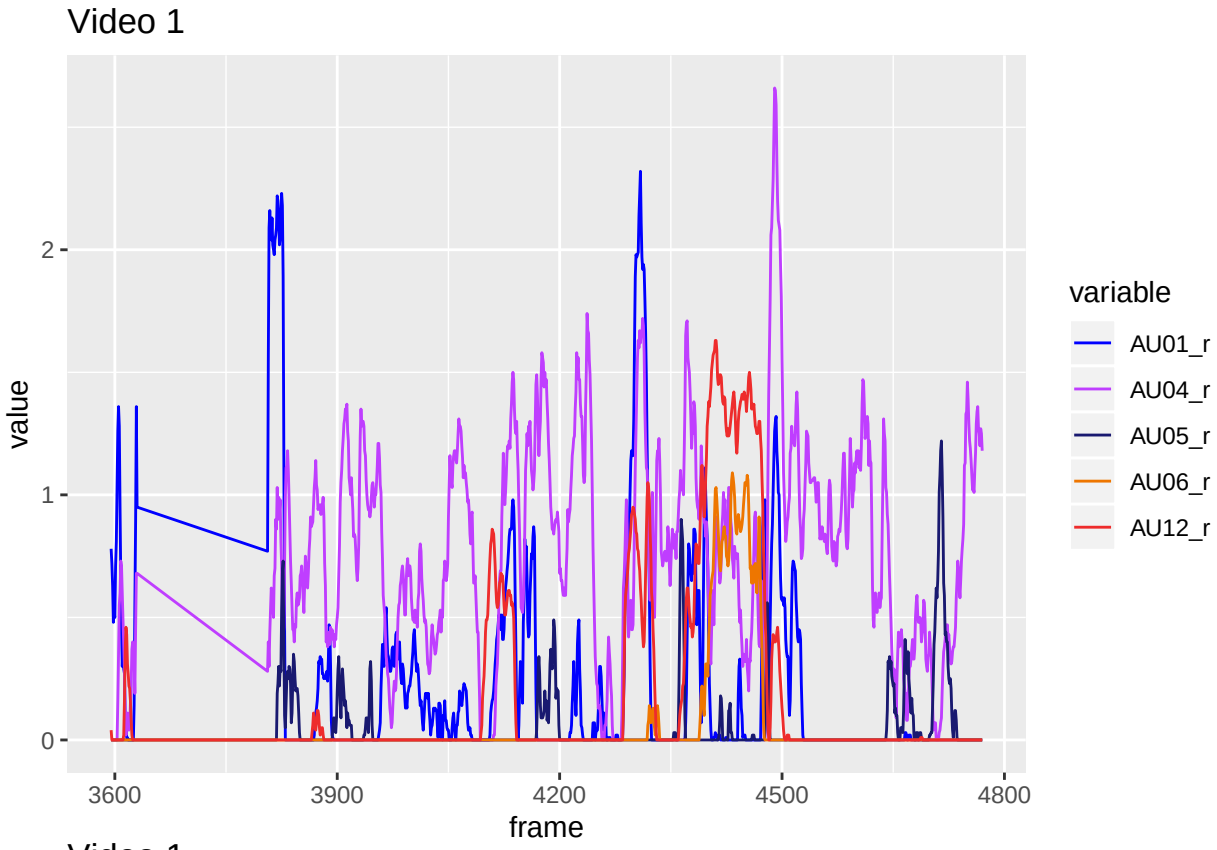
```
# Some data exploration

plot_aus <- function(dataset, title) {
  df1 <- dataset %>%
    select(-timestamp, - confidence, -AU01_c, -AU04_c, -AU05_c, -AU06_c, -AU12_c) %>%
    gather(key = "variable", value = "value", -frame)

  ggplot(df1, aes(x = frame, y = value)) +
    geom_line(aes(color = variable)) +
```

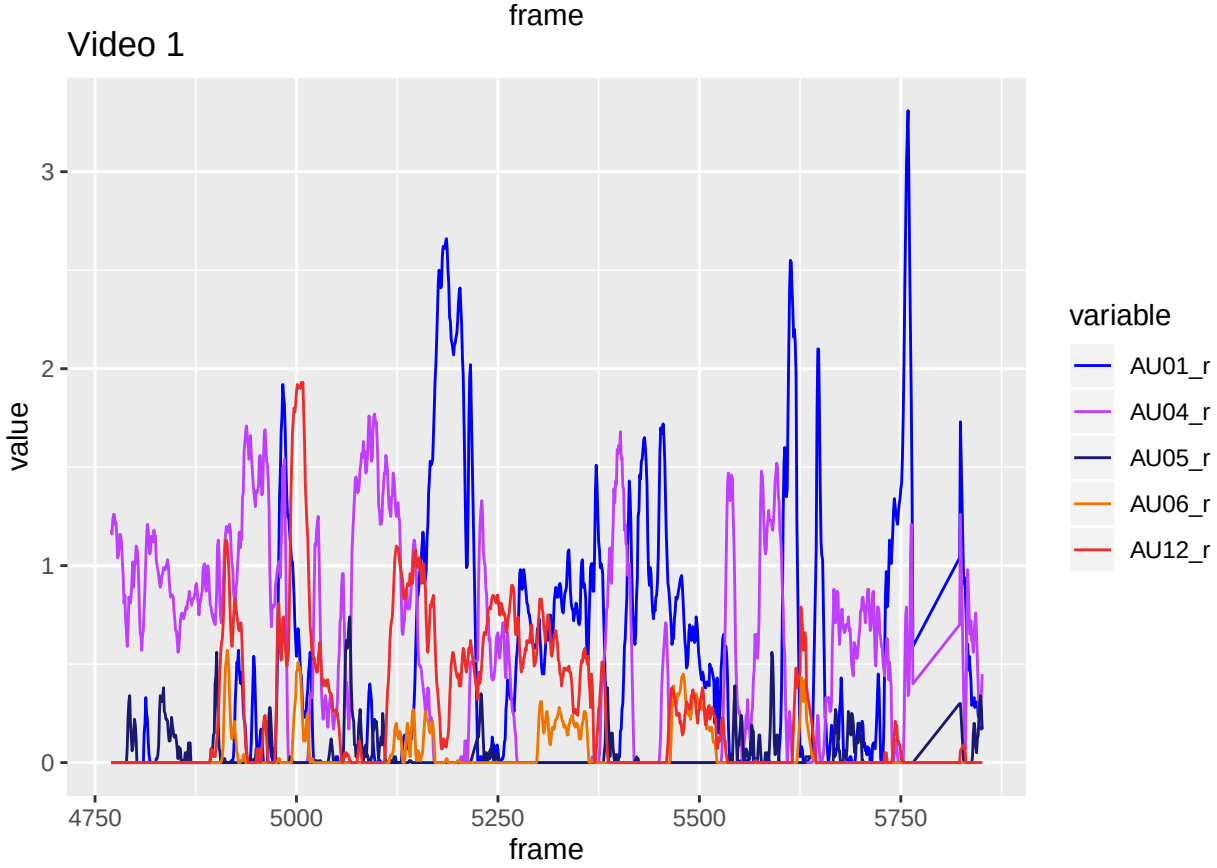
```
    scale_color_manual(values = c("blue1", "darkorchid1", "midnightblue", "darkorange2", "firebrick2")) +
    ggtitle(title)

}

plot_in_steps <- function(dataset, title) {
  for (i in 1:as.integer(nrow(dataset) / 1000)) {
    start = (i - 1) * 1000
    end = i * 1000
    print(plot_aus(dataset[start:end,], title))
  }
}

# AU 01, 04, 05 correspond to happiness
# AU 06, 12 correspond to negative emotions
plot_in_steps(video1, "Video 1")
```



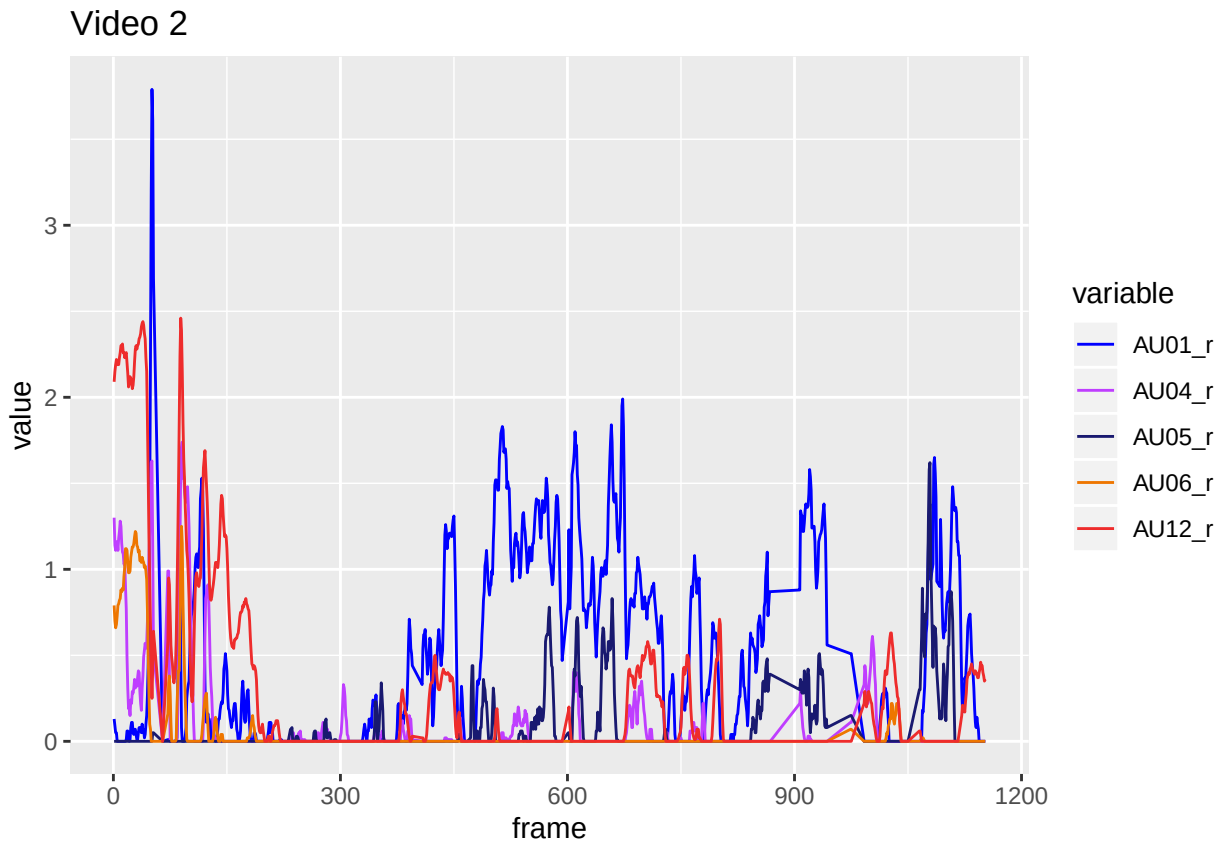

Video 1
value
frame
variable
AU01_r
AU04_r
AU05_r
AU06_r
AU12_r
0
1
2
3
1200
1600
2000

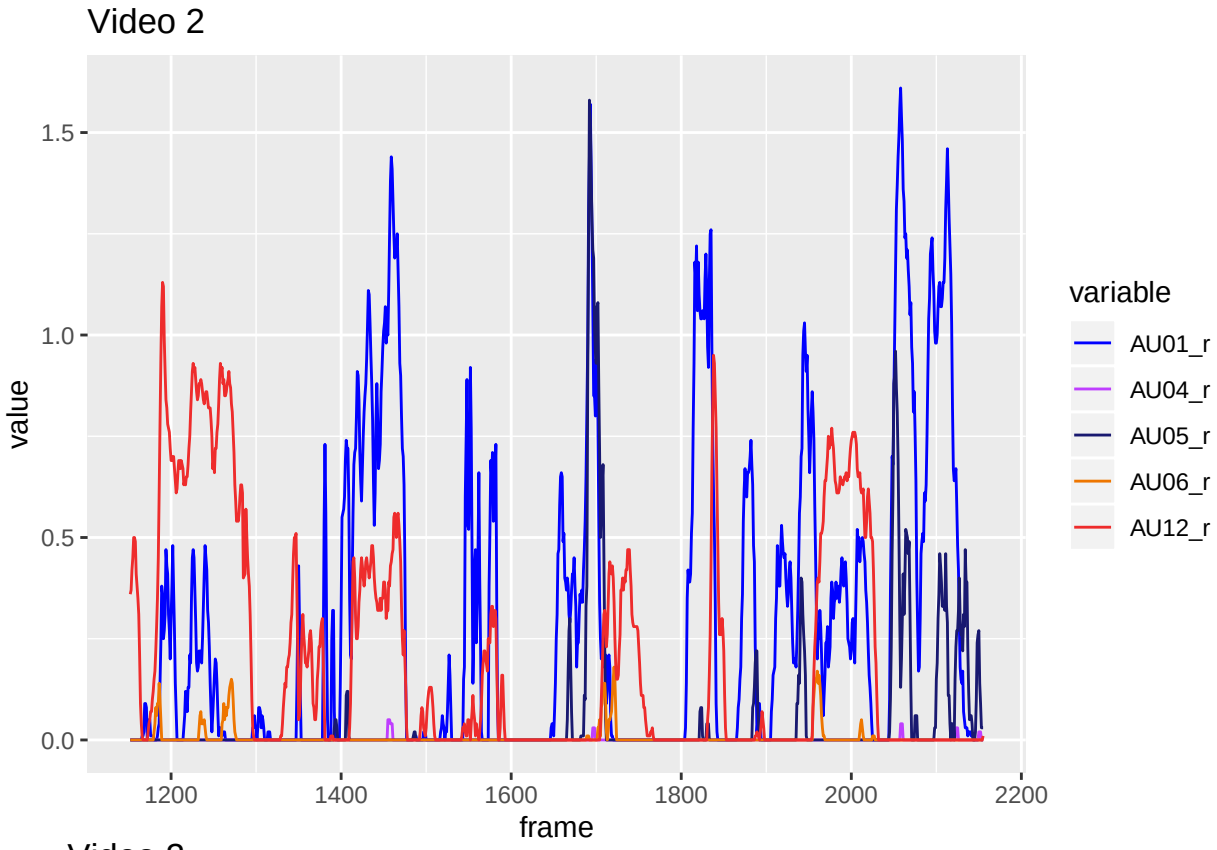

Video 1
value
frame
variable
AU01_r
AU04_r
AU05_r
AU06_r
AU12_r
0
1
2
3
2400
2800
3200
3600

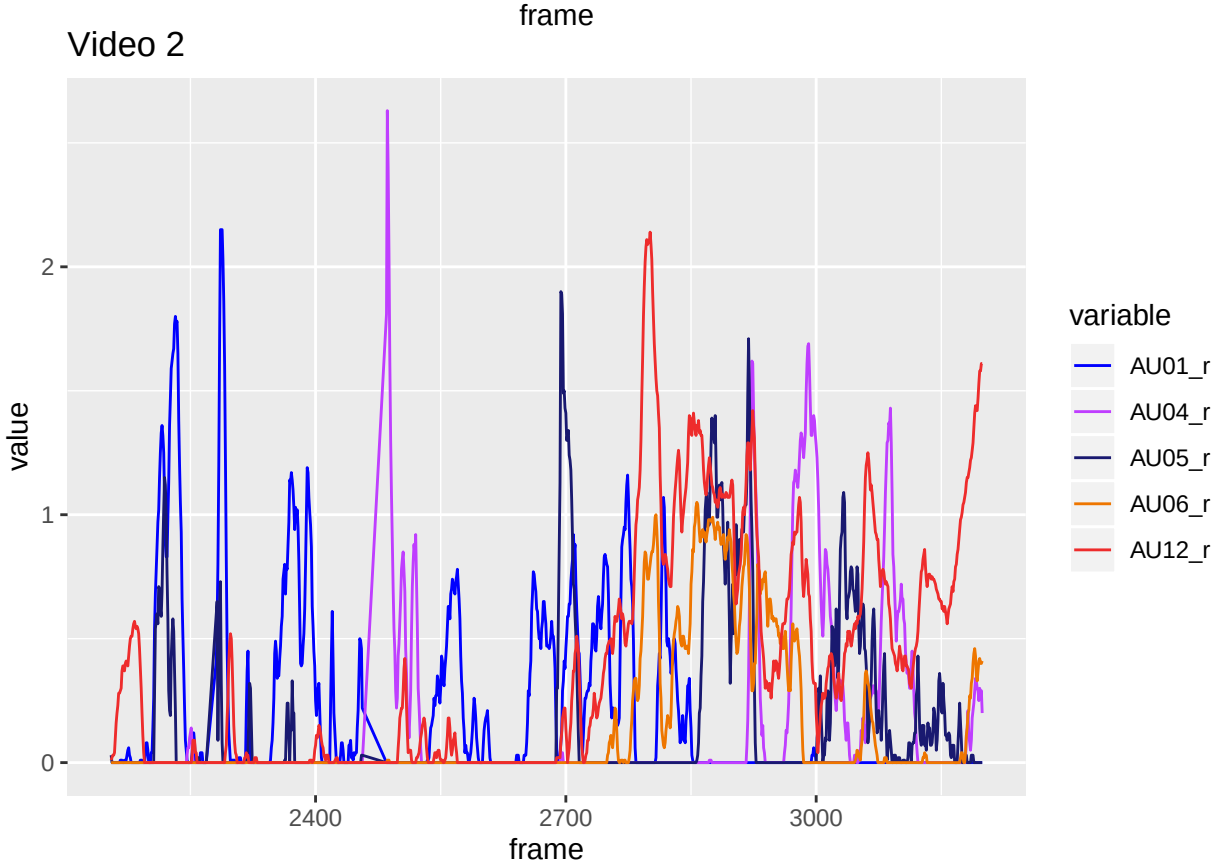

Video 1
value
frame
variable
AU01_r
AU04_r
AU05_r
AU06_r
AU12_r

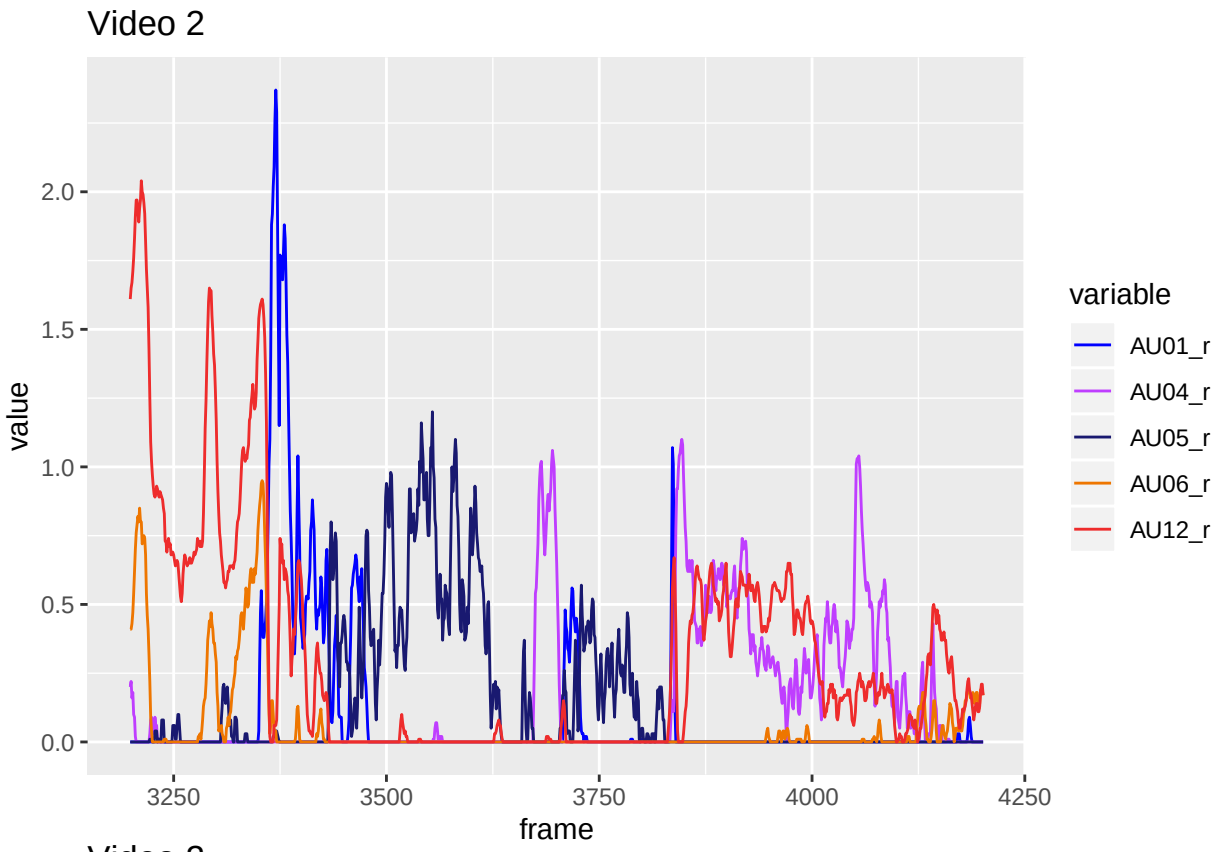

Video 1
value
frame
variable
AU01_r
AU04_r
AU05_r
AU06_r
AU12_r

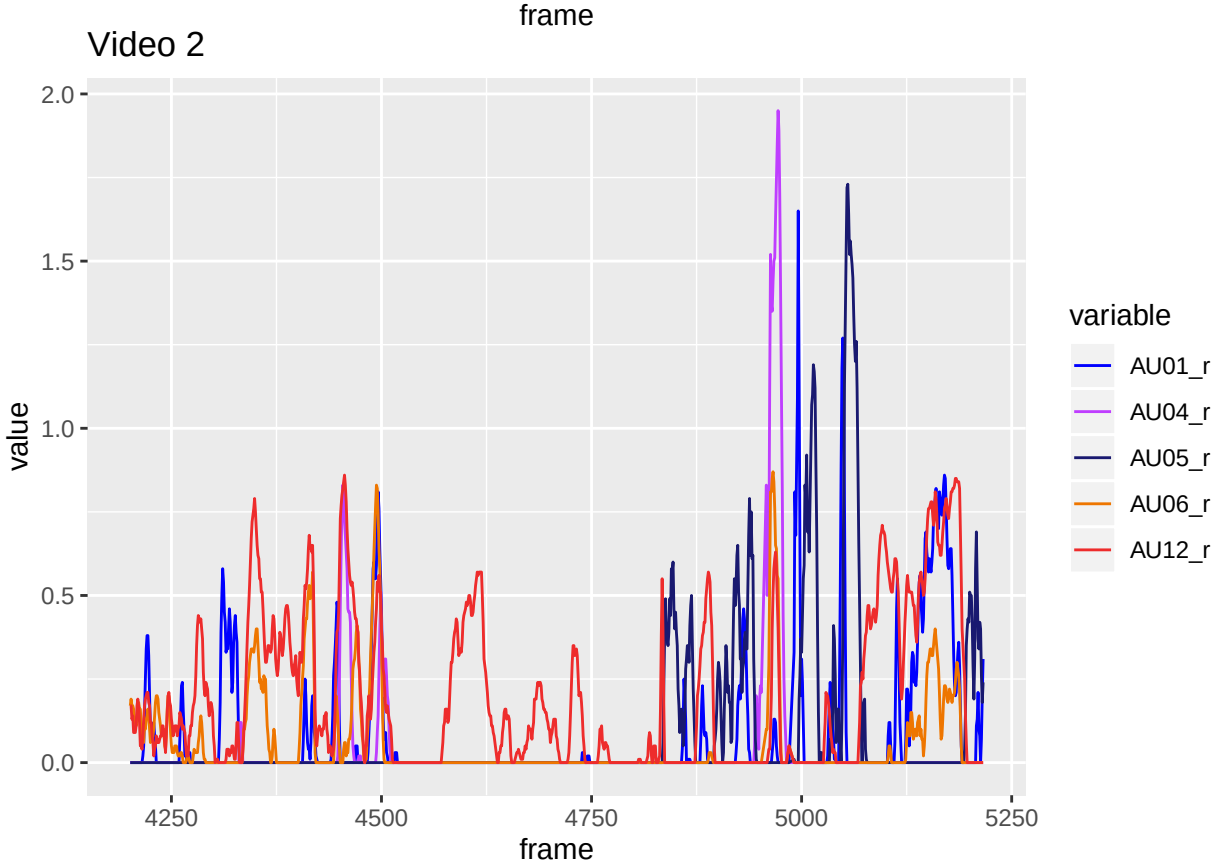
```
plot_in_steps(video2, "Video 2")
```

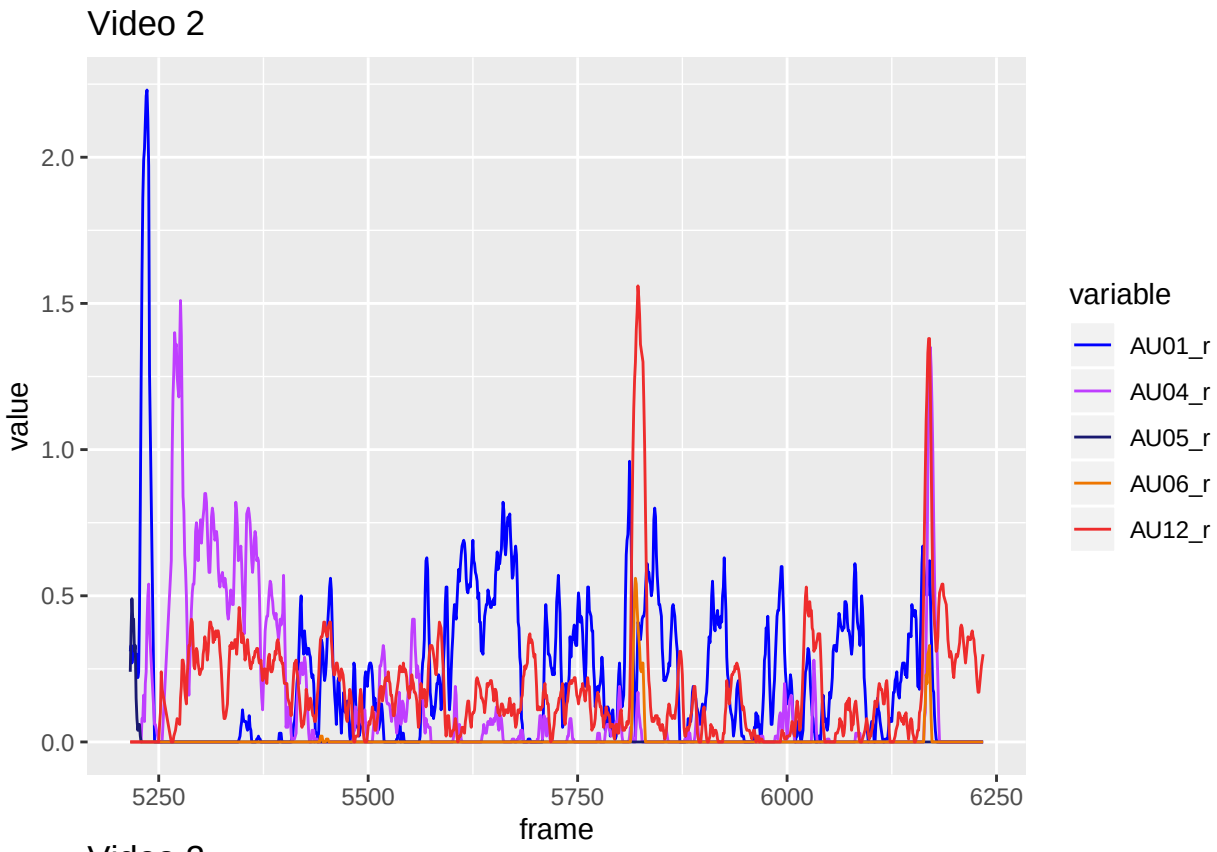


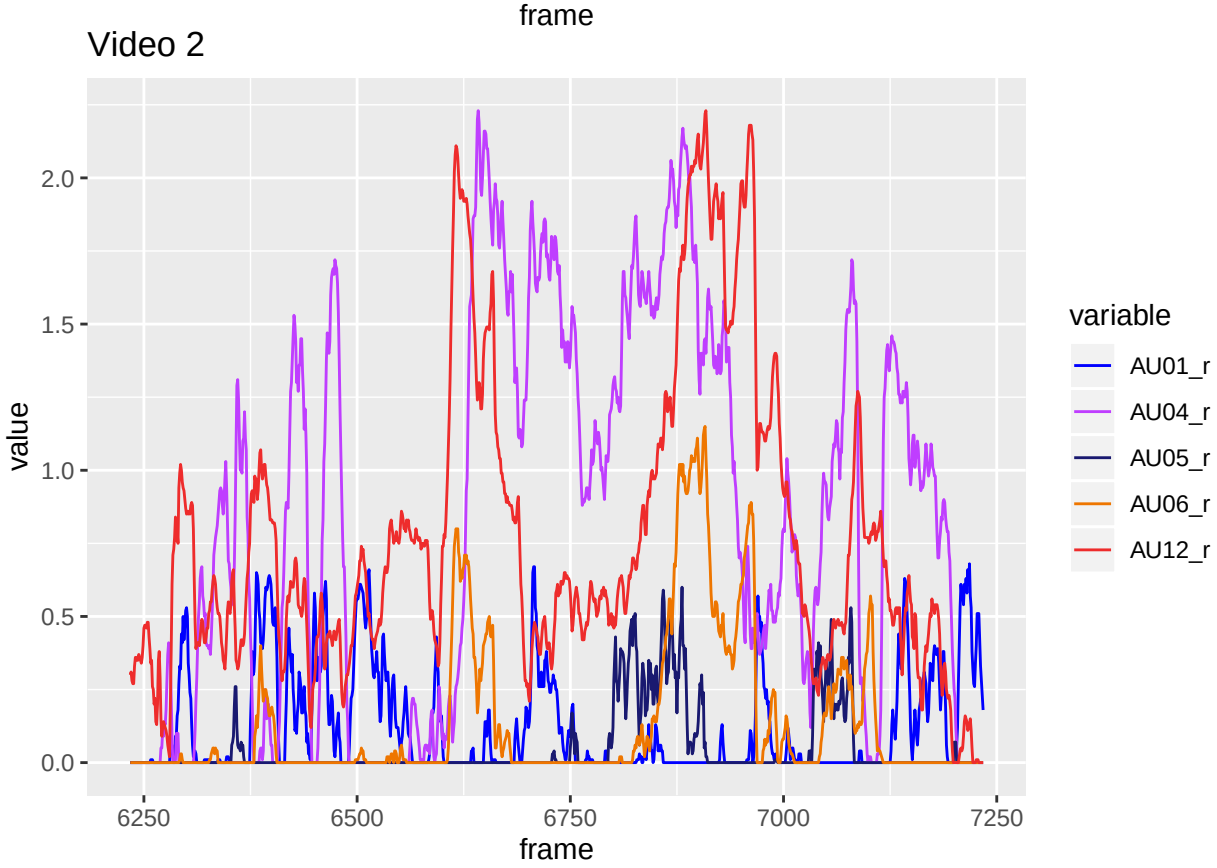

Video 2
value
frame
variable
AU01_r
AU04_r
AU05_r
AU06_r
AU12_r
0.0
0.5
1.0
1.5
1200
1400
1600
1800
2000
2200


Video 2
value
frame
variable
AU01_r
AU04_r
AU05_r
AU06_r
AU12_r
0
1
2
2400
2700
3000


Video 2
value
frame
variable
AU01_r
AU04_r
AU05_r
AU06_r
AU12_r
0.0
0.5
1.0
1.5
2.0
3250
3500
3750
4000
4250


Video 2
value
frame
variable
AU01_r
AU04_r
AU05_r
AU06_r
AU12_r
0.0
0.5
1.0
1.5
2.0
4250
4500
4750
5000
5250


Video 2
value
2.0
1.5
1.0
0.5
0.0
5250
5500
5750
6000
6250
frame
variable
AU01_r
AU04_r
AU05_r
AU06_r
AU12_r


Video 2
value
2.0
1.5
1.0
0.5
0.0
6250
6500
6750
7000
7250
frame
variable
AU01_r
AU04_r
AU05_r
AU06_r
AU12_r

```r
# returns value greater 1 if more positive than negative emotions, a value smaller 1 otherwise
get_relation_emotional_frequencies <- function(dataset) {
  frequencies <- colMeans(select(dataset, AU01_c, AU04_c, AU05_c, AU06_c, AU12_c))

  f <- matrix(frequencies)
  pos_mean <- mean(f[0:3])
  neg_mean <- mean(f[3:5])

  return(pos_mean / neg_mean)
}
```

```r
# Creating artificial datasets because we only have two samples
inflate_and_calc_rel_emotions <- function(df) {
  rel = c()
  for (i in 1:100) {
    new_size <- as.integer(nrow(video1) / 5)
    sample = sample_n(video1, new_size)
    rel = c(rel, get_relation_emotional_frequencies(sample))
  }
  return(rel)
}

rel1 <- inflate_and_calc_rel_emotions(video1)
rel2 <- inflate_and_calc_rel_emotions(video2)
```

```r
t.test(rel1, rel2)
```

```
##
##  Welch Two Sample t-test
##
## data:  rel1 and rel2
## t = 0.71978, df = 196.44, p-value = 0.4725
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.006601695  0.014190299
## sample estimates:
## mean of x mean of y
##  2.141859  2.138065
```

Hypothesis 2: Number of puzzle pieces left

```r
# Creating artificial dataset with number of puzzle pieces left, because the amount of pieces left in t
pieces_left1 <- sample(0:20, 100, replace=TRUE)
pieces_left2 <- sample(0:15, 100, replace=TRUE)
```

```r
mean(pieces_left1)
```

```
## [1] 10.76
```

```r
mean(pieces_left2)
```

```
## [1] 7.79
```

```r
t.test(pieces_left1, pieces_left2)
```

```
##
##  Welch Two Sample t-test
##
## data:  pieces_left1 and pieces_left2
## t = 4.0331, df = 183.17, p-value = 8.072e-05
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  1.517055 4.422945
## sample estimates:
## mean of x mean of y
##     10.76      7.79
```

Hypothesis 3: time needed for successful completion

```
# creating artificial dataset with time spent in seconds
time_needed1 <- sample(120:500, 100, replace=TRUE)
time_needed2 <- sample(100:480, 100, replace=TRUE)

mean(time_needed1)
```

```
## [1] 315.85
```

```
mean(time_needed2)
```

```
## [1] 295.27
```

```
t.test(time_needed1, time_needed2)
```

```
##
##  Welch Two Sample t-test
##
## data:  time_needed1 and time_needed2
## t = 1.3431, df = 196.85, p-value = 0.1808
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -9.638674 50.798674
## sample estimates:
## mean of x mean of y
##    315.85    295.27
```