Synchrony in relationship, example with MONRADO Data

Thomas Gargot

May, 10th 2016

Contents

```
rm(list = ls(all.names = TRUE))
```

Fixed variables

```
FileExtension <- ".MTS.avi_res.csv"

# working directory
# where this report is
setwd("/Users/Ofix/Documents/Fac/internat/Recherche/projets/synchro/synchroData/Git/Monrado/Reports/")

# blue will refer to father
# red will refer to mother
# green to child
colOrderList <- c("blue","red", "green")

ParticipantsList <- c("father", "mother", "child")
```

```
## Create a csv files list with the directories
FullNameList <- list.files("../Data/CSV/raw", full.names=TRUE)
FullNameList

## Create a csv files list without the directories
filesList <- list.files("../Data/CSV/raw", full.names=FALSE)
filesList
```

Functions list

Import Data List

Function that import data from .csv files inside a CSV folder ##### Arguments: List FullNameList with the full name of the .csv

```
importdata <-function(FullnameList){
    data <- c()
    for (i in FullnameList){
        dataAlone <- read.csv(i)
        mydata.nas <- apply(dataAlone[,c(2:5)], 1, function(x){all(is.na(x))})
        dataAlone <- dataAlone[!mydata.nas,]
        print(i)
        data <- rbind(data, dataAlone)
    }
    return (data)
}
```

MeanMotionByTime

Function that takes raw motion history data and compute the mean on a given interval. Intervals don't overlap, so the frequency of the data change (from 25 frames by seconde to 25 frames/interval by second).

Arguments:

- subject : Subject studied (patient, mother, father or therapist)
- indexOfvideos : List of videos studied (element eg 3 or list eg 1:3 or c(1,2,4))
- interval : number of frames in the studied interval
- data : data frame where there is data

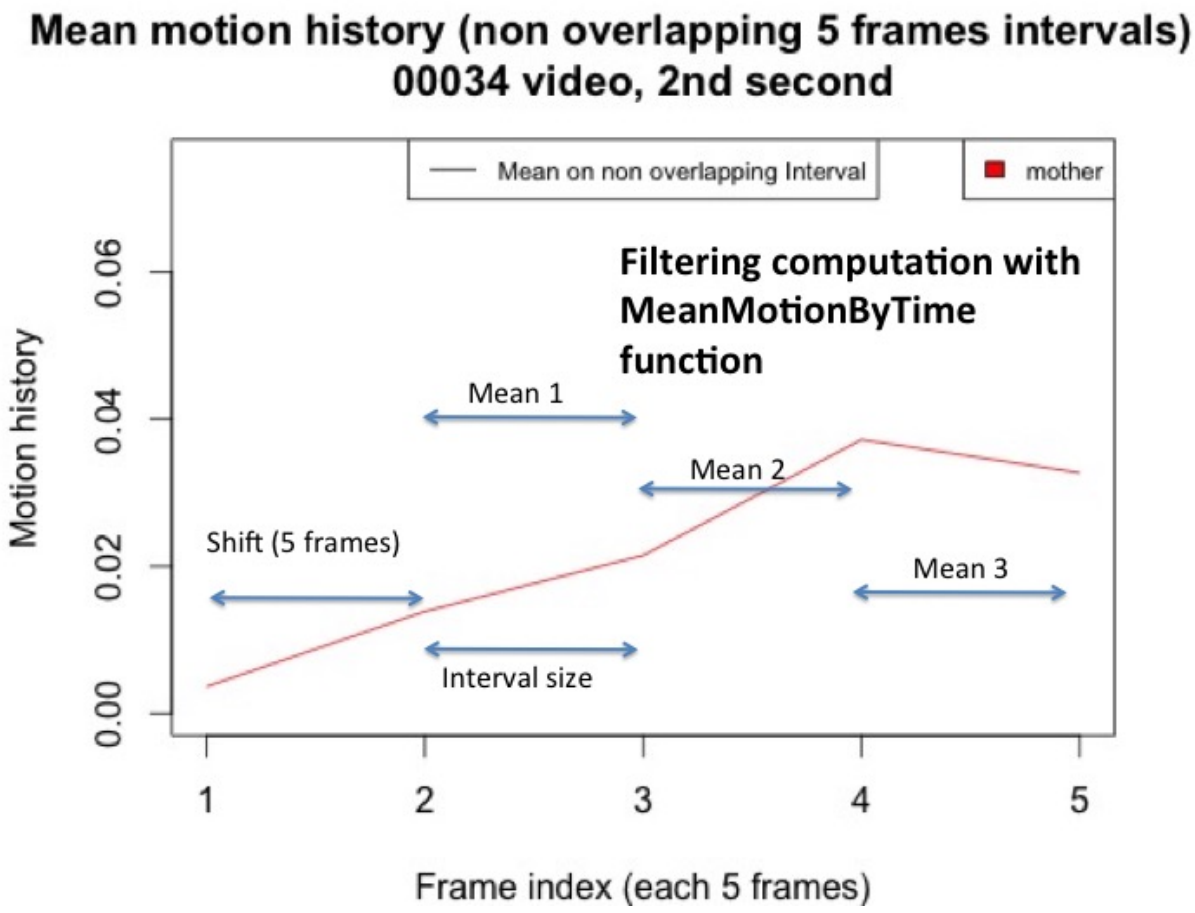


Figure 1:

```
MeanMotionByTime <- function(subject, indexOfvideos=1:NumberOfvideos, interval, data){
  x <- c()
  for (fam in families[indexOfvideos]){
        dataVector <- data[which(data$family==fam), subject]
        ## with ceiling : superior limit of the round
        IntervalNumbersVideo <- ceiling(length(dataVector)/interval)
        for (i in 1:IntervalNumbersVideo){
                borneinf<- 1+(i-1)*interval
                bornesup <-i*interval
```

```
                dataVectorInterval <- dataVector[borneinf:bornesup]
                mean <- mean(dataVectorInterval, na.rm=TRUE)
                x <- c(x, mean)}}
  return (x)}
```

Slidinginterval

Function that takes raw motion history data and compute the mean on a given interval. The interval overlap, so the frequency of the data don't change. It stays at 25 frames/s.

Arguments:

- subject : subject studied (patient, mother, father or therapist)
- indexOfvideos : list of videos studied (element eg. 3 or list eg 1:3 or c(1,2,4))
- interval : number of frames in the studied interval
- data : data frame where there is data

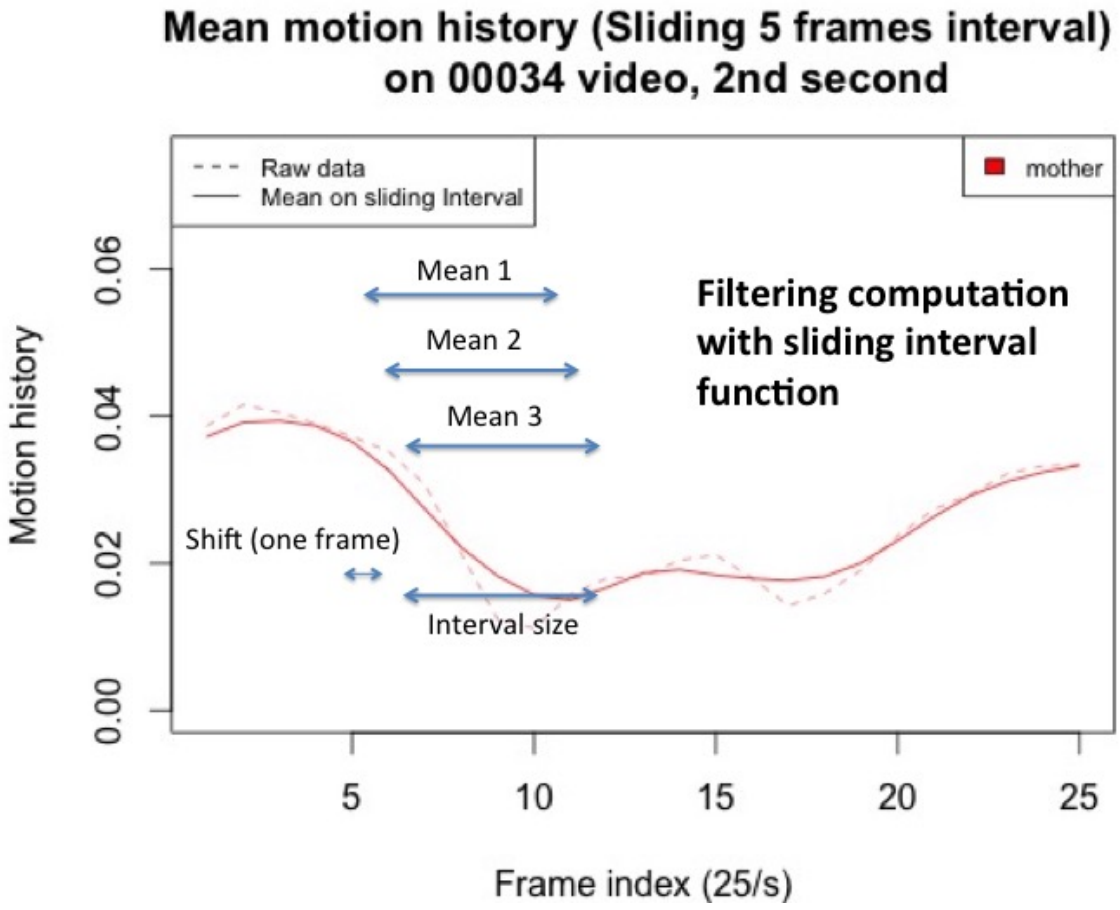


Figure 2:

```
SlidingInterval <- function(subject, indexOfvideos=1:NumberOfvideos, interval, data)
  {x <- c()
  for (file in families[indexOfvideos]){
```

```
      dataVector <- data[which(data$family==file), subject]
      NBofAnalysedFrames <- length(dataVector)-interval+1
          for (i in 1:NBofAnalysedFrames){
              borneinf <- (i)
              bornesup <-(interval-1+i)
              dataVectorInterval <- dataVector[borneinf:bornesup]
              mean <- mean(dataVectorInterval, na.rm=TRUE)
              x <- c(x, mean)}}
  return (x)}
```

Import data

```
data <- importdata(FullNameList)
```

Clean dataframe

Add new columns: compute minutes and log on data frame

```
# Detete No relevant subject here
data$therapist <- NULL

# compute time in minute
data$timeMin <- data$frame/(25*60)

## Create a list of files without the extention of the video
families <- c()
for (i in filesList){
    name <- sub(FileExtension, "", i)
    families <- c(families, name)
}
families
```

```
##  [1] "00034"   "00037"   "00041"   "00048"   "0206"    "1106"    "1606"
##  [8] "BAJE059" "BALE050" "BALU062" "BEAL036" "BEAM031" "BICA"    "BRLO041"
## [15] "COLO022" "DIPE004" "DOMA"    "DRNE"    "FOMA057" "GROP039" "HAJA052"
## [22] "HUMA058" "JAEM046" "JEEO040" "JOCE014" "LACL"    "MAEL048" "MAME20"
## [29] "MAPA029" "MIPH043" "MOSA065" "RAEM049" "RAKA008" "RIEM0"   "SEEM035"
## [36] "SHAN042" "SOGA061" "TIUG032" "VINO"
```

```
NumberOfvideos <- length(families)
NumberOfvideos
```

```
## [1] 39
```

```
# create a list with the simplified dname (whitout extension), make a data frmae of it  and merge 2 dat
a <- data.frame(family = families, unique(data$file))
data <- merge(data, a, by.x="file", by.y="unique.data.file.")

data$fatherShifted <- data$father + min(data$father[which (data$father >0)])/2
data$logFather <- log(data$fatherShifted)

data$motherShifted <- data$mother + min(data$mother[which (data$mother >0)])/2
data$logMother <- log(data$motherShifted)
```

```
data$childShifted <- data$child + min(data$child[which (data$child >0)])/2
data$logChild <- log(data$childShifted)

# Add date TODO
data$file <- NULL

data <- data[,c("family", "frame", "timeMin", "child", "childShifted", "logChild", "father", "fatherShi
```

Presentation of the data

```
str(data)
```

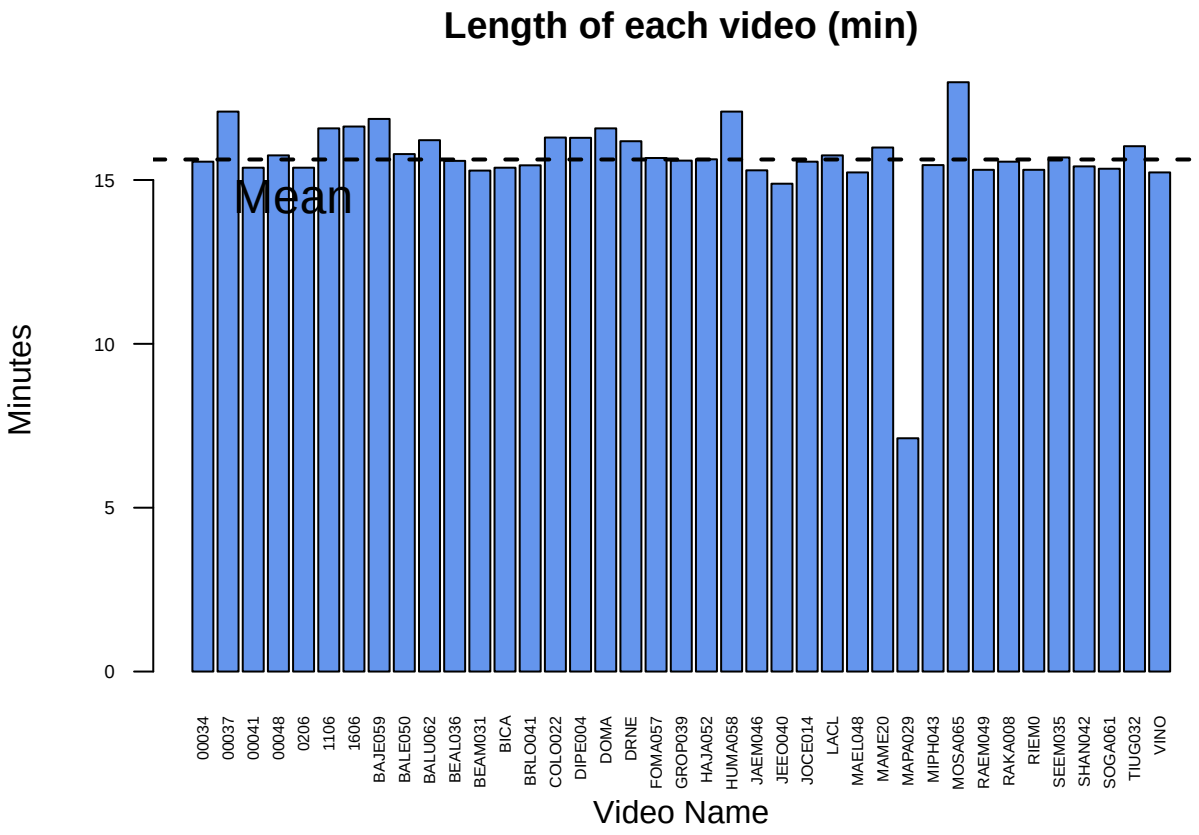
```
## 'data.frame':    914108 obs. of  12 variables:
##  $ family       : Factor w/ 39 levels "00034","00037",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ frame        : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ timeMin      : num  0.000667 0.001333 0.002 0.002667 0.003333 ...
##  $ child        : num  1.62e-04 1.89e-04 7.50e-05 5.36e-05 8.04e-05 ...
##  $ childShifted : num  1.62e-04 1.89e-04 7.55e-05 5.40e-05 8.08e-05 ...
##  $ logChild     : num  -8.73 -8.57 -9.49 -9.83 -9.42 ...
##  $ father       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ fatherShifted: num  NA NA NA NA NA NA NA NA NA NA ...
##  $ logFather    : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ mother       : num  4.00e-04 4.56e-04 2.23e-04 8.85e-05 9.58e-05 ...
##  $ motherShifted: num  4.01e-04 4.57e-04 2.24e-04 8.89e-05 9.61e-05 ...
##  $ logMother    : num  -7.82 -7.69 -8.41 -9.33 -9.25 ...
```

```
summary(data)
```

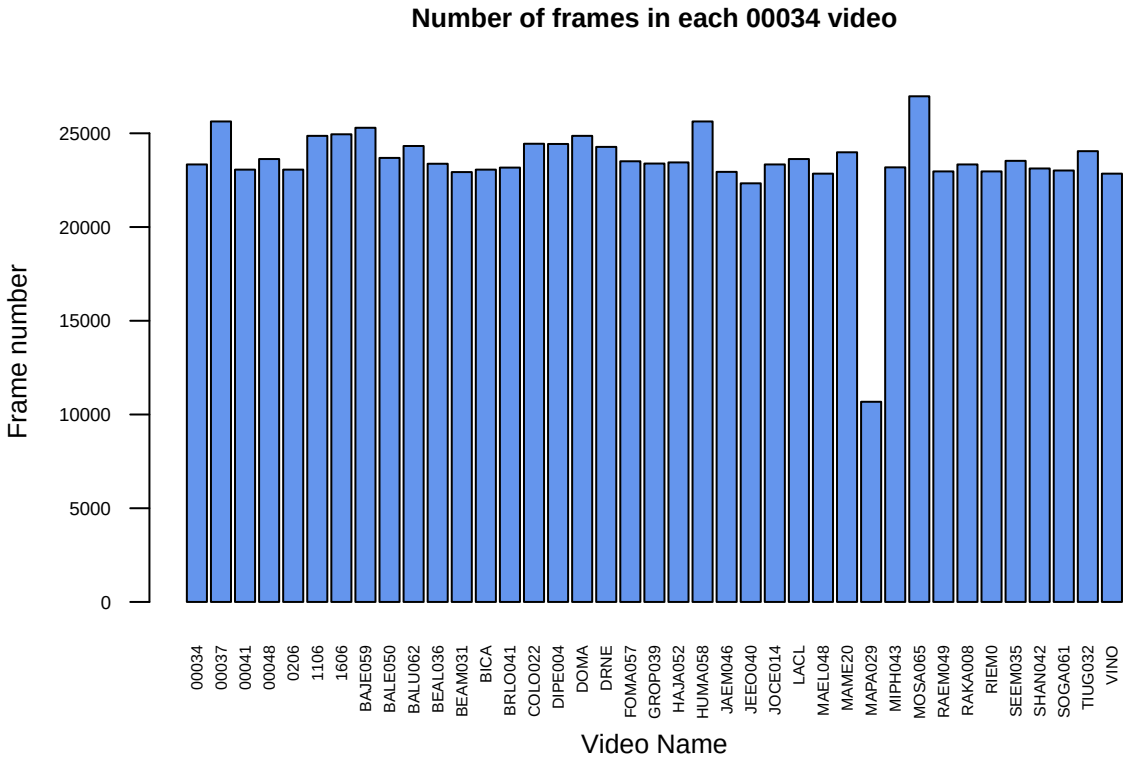
```
##      family           frame          timeMin              child
##  MOSA065: 26975   Min.   :    1   Min.   : 0.000667   Min.   :0.0000000
##  00037  : 25631   1st Qu.: 5860   1st Qu.: 3.906667   1st Qu.:0.0006479
##  HUMA058: 25631   Median :11747   Median : 7.831333   Median :0.0034404
##  BAJE059: 25295   Mean   :11831   Mean   : 7.887165   Mean   :0.0093006
##  1606   : 24947   3rd Qu.:17761   3rd Qu.:11.840667   3rd Qu.:0.0117136
##  1106   : 24863   Max.   :26975   Max.   :17.983333   Max.   :0.9270200
##  (Other):760766
##   childShifted          logChild             father       fatherShifted
##  Min.   :0.0000004   Min.   :-14.66620   Min.   :0.0      Min.   :0.0
##  1st Qu.:0.0006483   1st Qu.: -7.34117   1st Qu.:0.0      1st Qu.:0.0
##  Median :0.0034409   Median : -5.67204   Median :0.0      Median :0.0
##  Mean   :0.0093010   Mean   : -6.19280   Mean   :0.0      Mean   :0.0
##  3rd Qu.:0.0117140   3rd Qu.: -4.44697   3rd Qu.:0.0      3rd Qu.:0.0
##  Max.   :0.9270204   Max.   : -0.07578   Max.   :0.1      Max.   :0.1
##                                          NA's   :742791   NA's   :742791
##    logFather          mother       motherShifted      logMother
##  Min.   :-14.6    Min.   :0.00     Min.   :0.00     Min.   :-14.78
##  1st Qu.: -9.2    1st Qu.:0.00     1st Qu.:0.00     1st Qu.: -8.41
##  Median : -6.8    Median :0.00     Median :0.00     Median : -6.44
##  Mean   : -7.3    Mean   :0.01     Mean   :0.01     Mean   : -7.03
##  3rd Qu.: -5.1    3rd Qu.:0.01     3rd Qu.:0.01     3rd Qu.: -5.09
##  Max.   : -2.0    Max.   :0.96     Max.   :0.96     Max.   : -0.04
##  NA's   :742791   NA's   :147978   NA's   :147978   NA's   :147978
```

The timeMin is calculated with a frame rate of 25/sec.

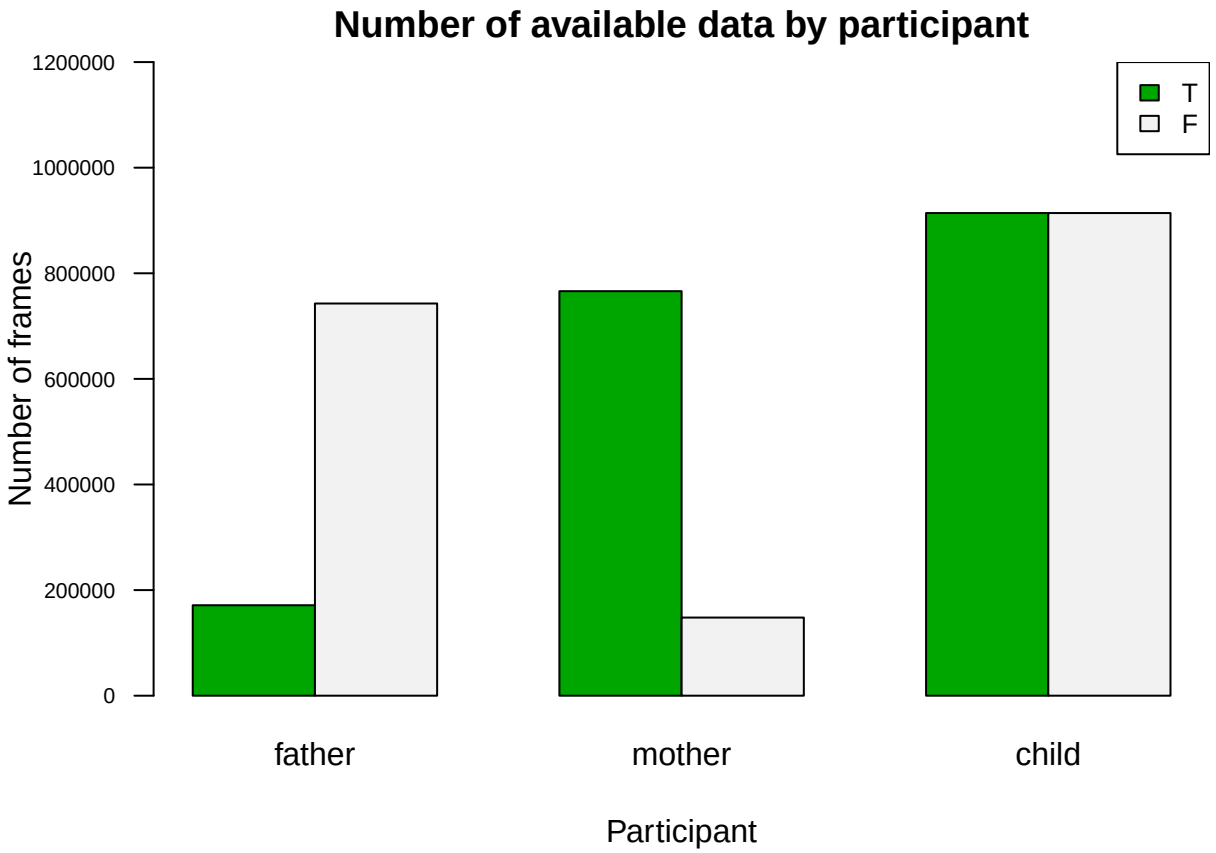
Length of the videos in minutes



Length of the videos in number of frames



Number of Available (True) and Not Available (False) data for each participant

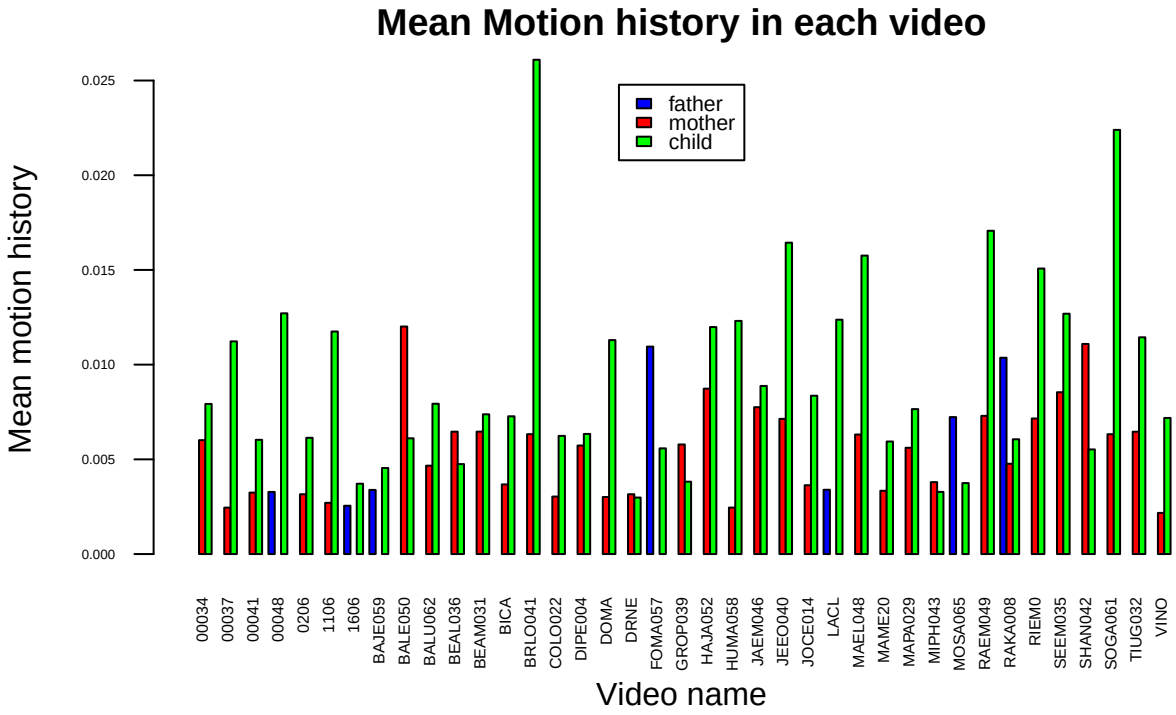


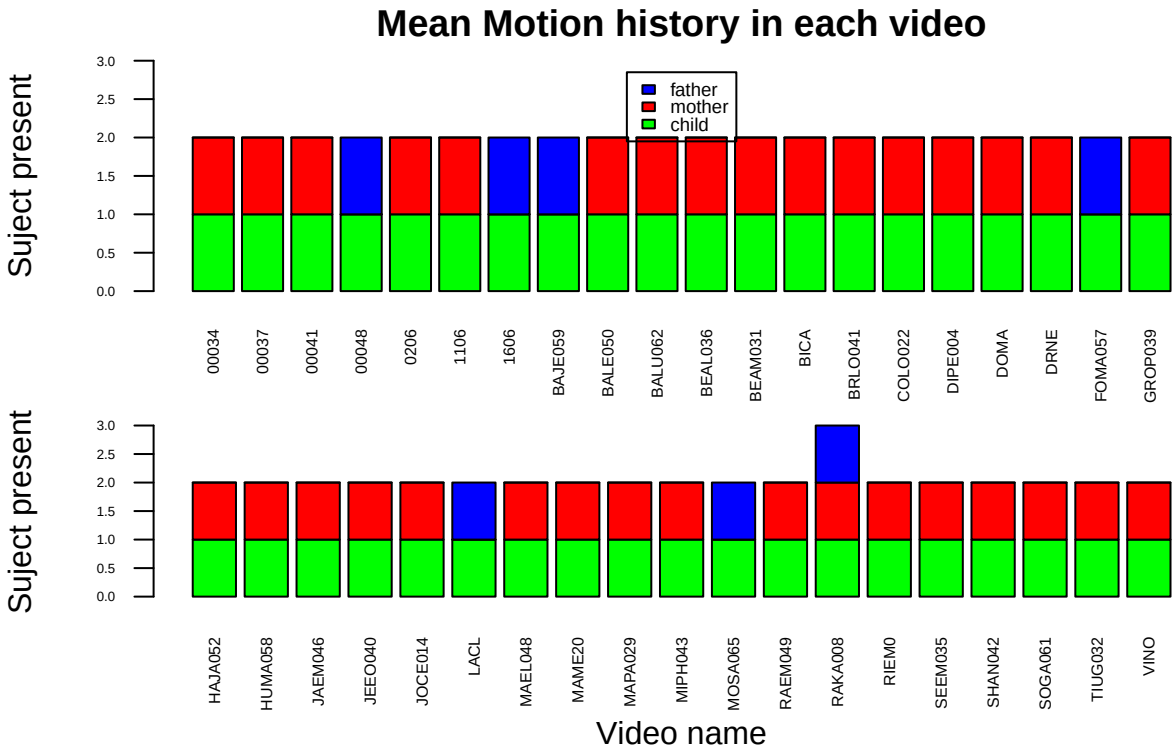
*

All the participants involved are filmed. * All the children are filmed and we have data dor each. * More often there is the other with him/her sometimes, it is the father * In some videos for instance RAKA008, there are 3 subjects

Global Motion history

Mean Motion history by video by participant





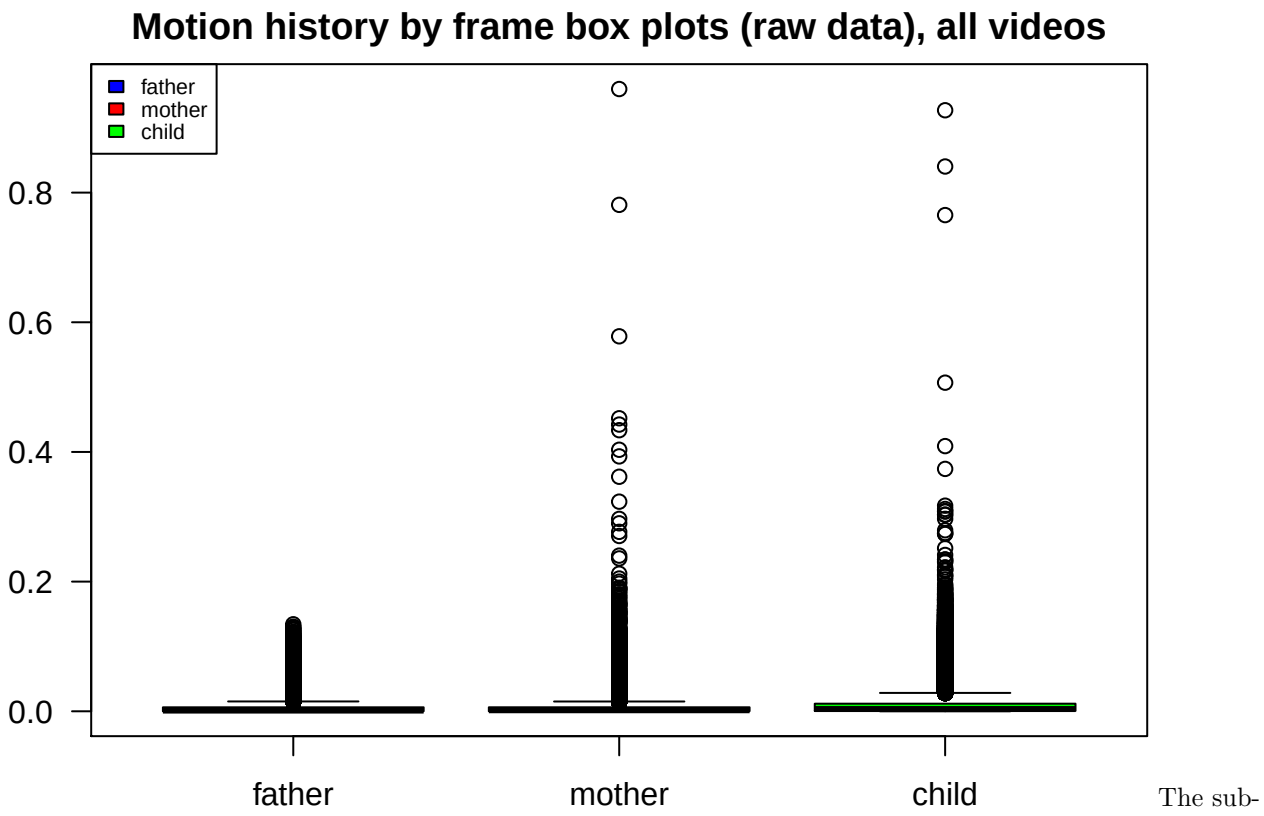
We can see that configurations of subjects are very similar (with always 2 subjects, except RAKA008 with 3 subjects). More often the child is with his mother. Consequently, it makes the comparaisons of the videos easier than in the INCANT study.

Motion history box plots by frame (raw data), all videos

The motion history is not normalized at all. Most of motions are very small but some of them are much big (long tail). This is very usual with this algorithm extraction motion history.

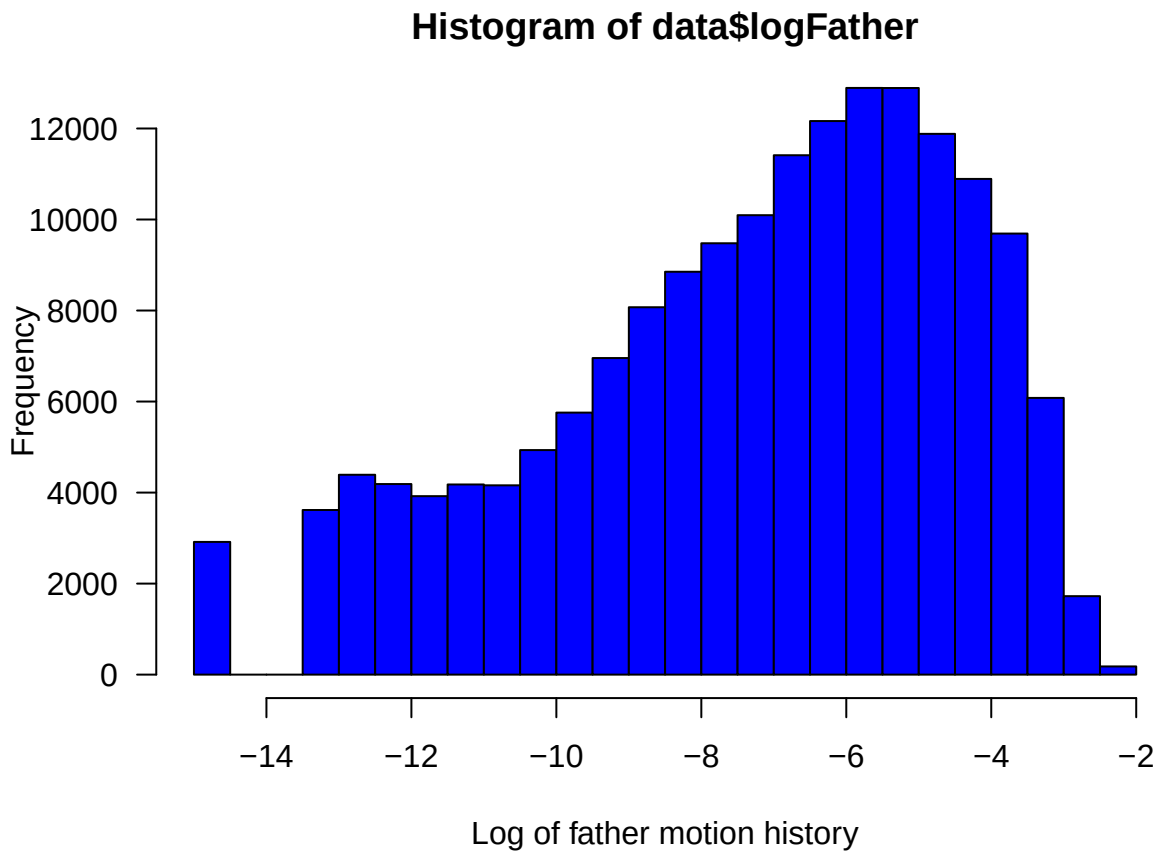
The subjects data are very similar.

```
par(mar=c(3,3,2,2))
boxplot(data$father, data$mother, data$child,
        col=colOrderList,
        names=ParticipantsList,
        main= "Motion history by frame box plots (raw data), all videos", las=1)
par(mar=c(1,0.5,0.5,1))
legend("topleft", ParticipantsList, fill=colOrderList, cex=0.7)
```

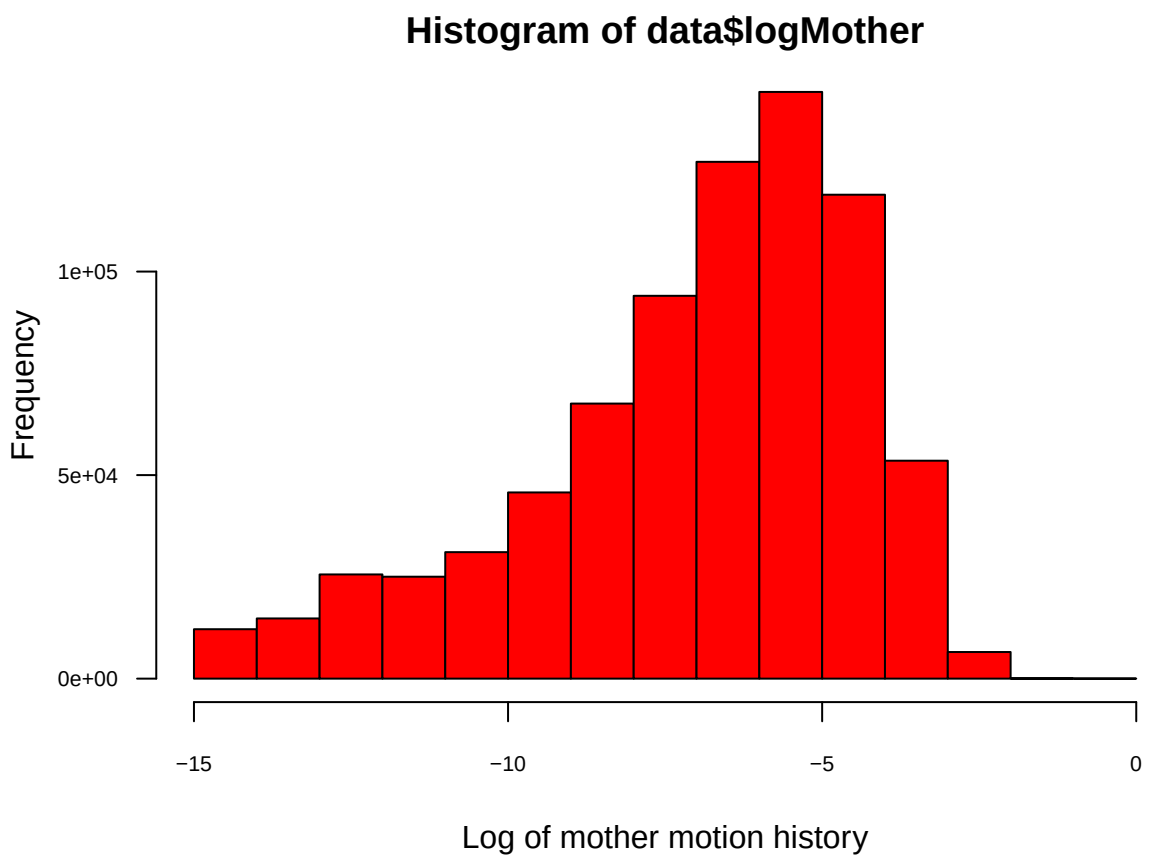


The subjects data distribution are very similar.

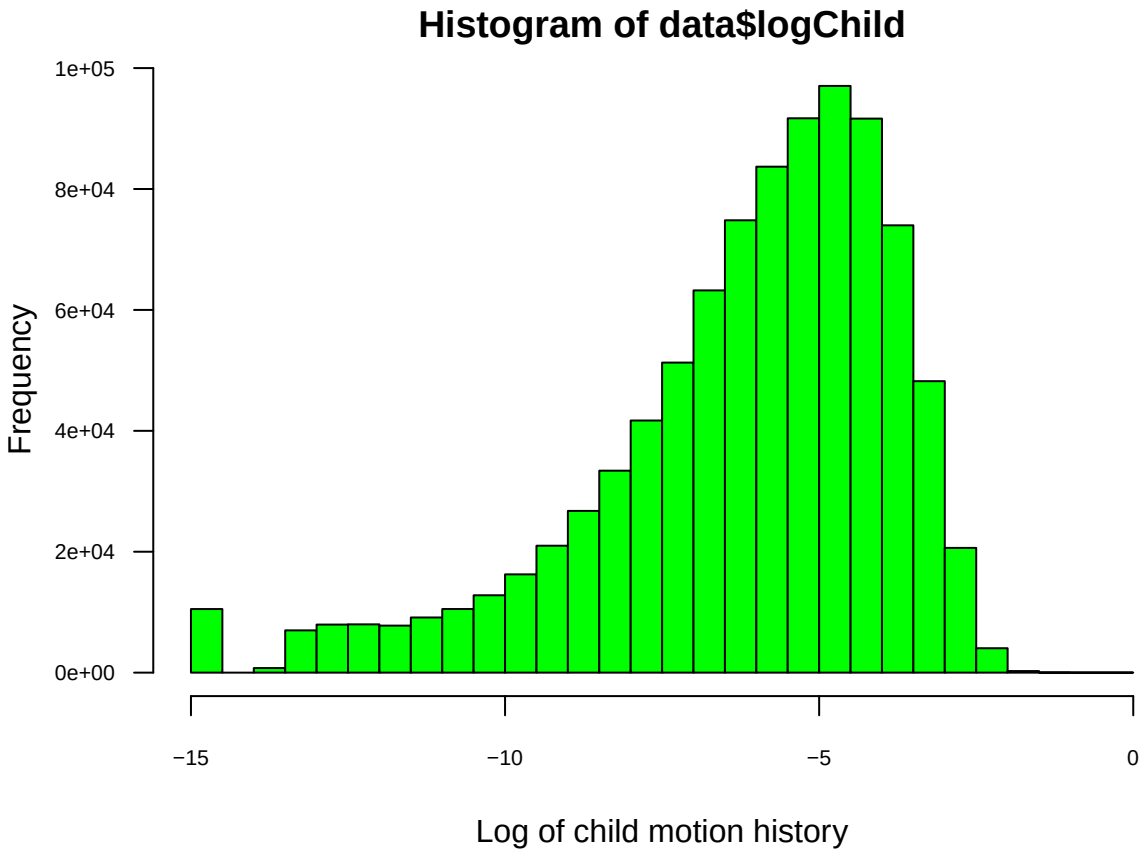
```
par(mar=c(4,4,2,2))
hist(data$logFather, col="blue", las=1, xlab="Log of father motion history")
```




```
hist(data$logMother, col="red", las=1, xlab="Log of mother motion history", cex.axis=0.7)
```



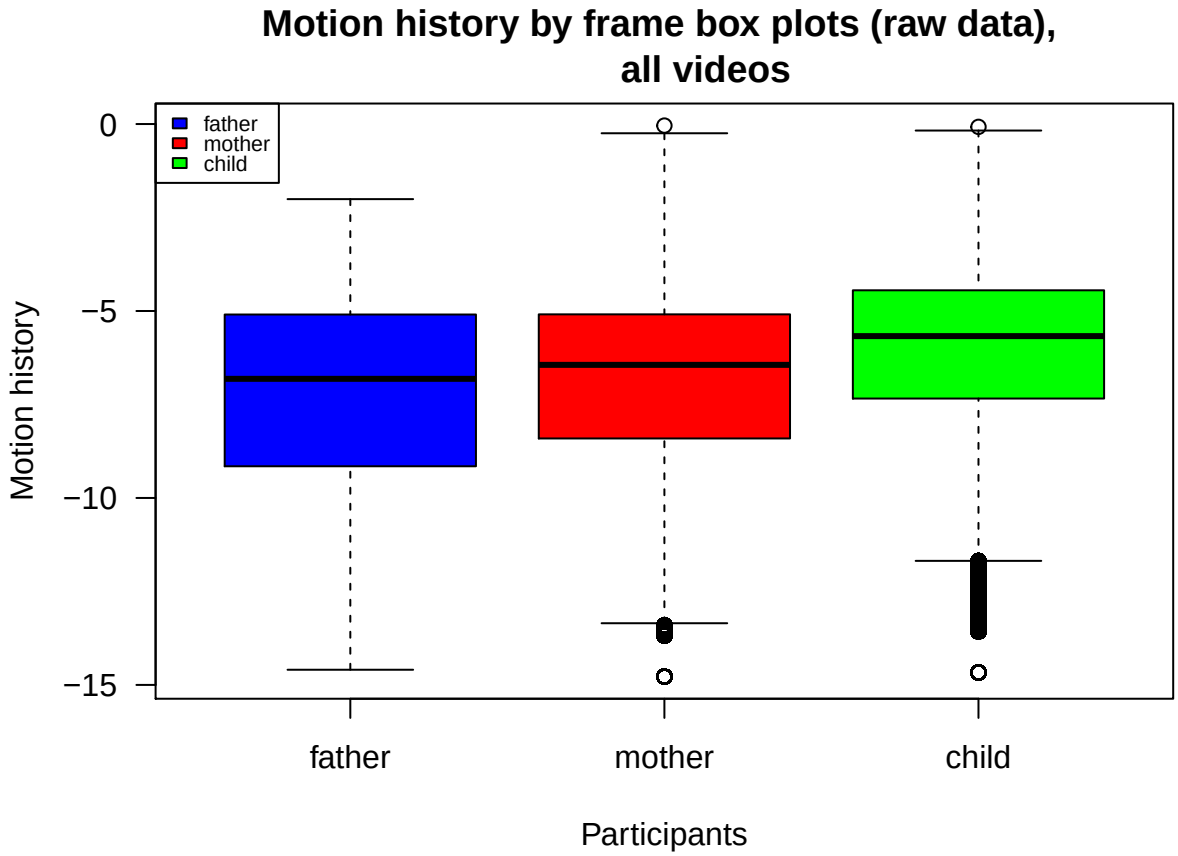

```
hist(data$logChild, col="green", las=1, xlab="Log of child motion history", cex.axis=0.7)
```



When doing the log, we almost normalized the distribution. We couldn't do the log on 0. The result would gave -Inf. We shifted all the distribution to the right by adding the half of the minimum after 0 of the distribution.

$\text{data}childShifted < -data\text{child} + \min(\text{data}child[which(data\text{child} > 0)])/2$

```
par(mar=c(4,4,3,2))
boxplot(data$logFather, data$logMother, data$logChild,
        col=colOrderList,
        names=ParticipantsList,
        main= "Motion history by frame box plots (raw data),
        all videos", las=1, xlab="Participants", ylab="Motion history")
par(mar=c(1,0.5,0.5,1))
legend("topleft", ParticipantsList, fill=colOrderList, cex=0.7)
```



Raw data and mean of Motion History on sliding and non overlapping intervals on 00034 video

It is the first video of 00034.

Raw data

```
rawdataMother <- data[which(data$family=="00034"),]$mother
rawdataChild <- data[which(data$family=="00034"),]$child

summary(rawdataMother)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000000 0.0004453 0.0026660 0.0060140 0.0085460 0.1735000
```

```
summary(rawdataChild)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.000000 0.000537 0.002285 0.007926 0.008199 0.179000
```

Sliding interval

```
## REMINDER:
# SlidingInterval <- function(subject, indexOfvideos=1:NumberOfvideos, interval, data) with :
# subject : subject studied (patient, mother, father or therapist)
# indexOfvideos : list of videos studied (element eg. 3 or list eg 1:3 or c(1,2,4))
# interval : number of frames in the studied interval
# data : data frame where there is data
# repalce by 5 after
slidedMother <- SlidingInterval("mother", 1 , 5, data)
slidedChild <- SlidingInterval("child", 1 , 5, data)

summary(slidedMother)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 5.400e-07 5.367e-04 2.972e-03 6.015e-03 8.495e-03 1.616e-01
```

```
summary(slidedChild)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000000 0.0006375 0.0024400 0.0079270 0.0082810 0.1684000
```

Non overlapping interval

```
motherFive <- MeanMotionByTime("mother", indexOfvideos=1, interval=5, data)
childFive <- MeanMotionByTime("child", indexOfvideos=1, interval=5, data)

summary(childFive)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 2.100e-07 6.356e-04 2.417e-03 7.925e-03 8.280e-03 1.646e-01
```

```
summary(motherFive)
```

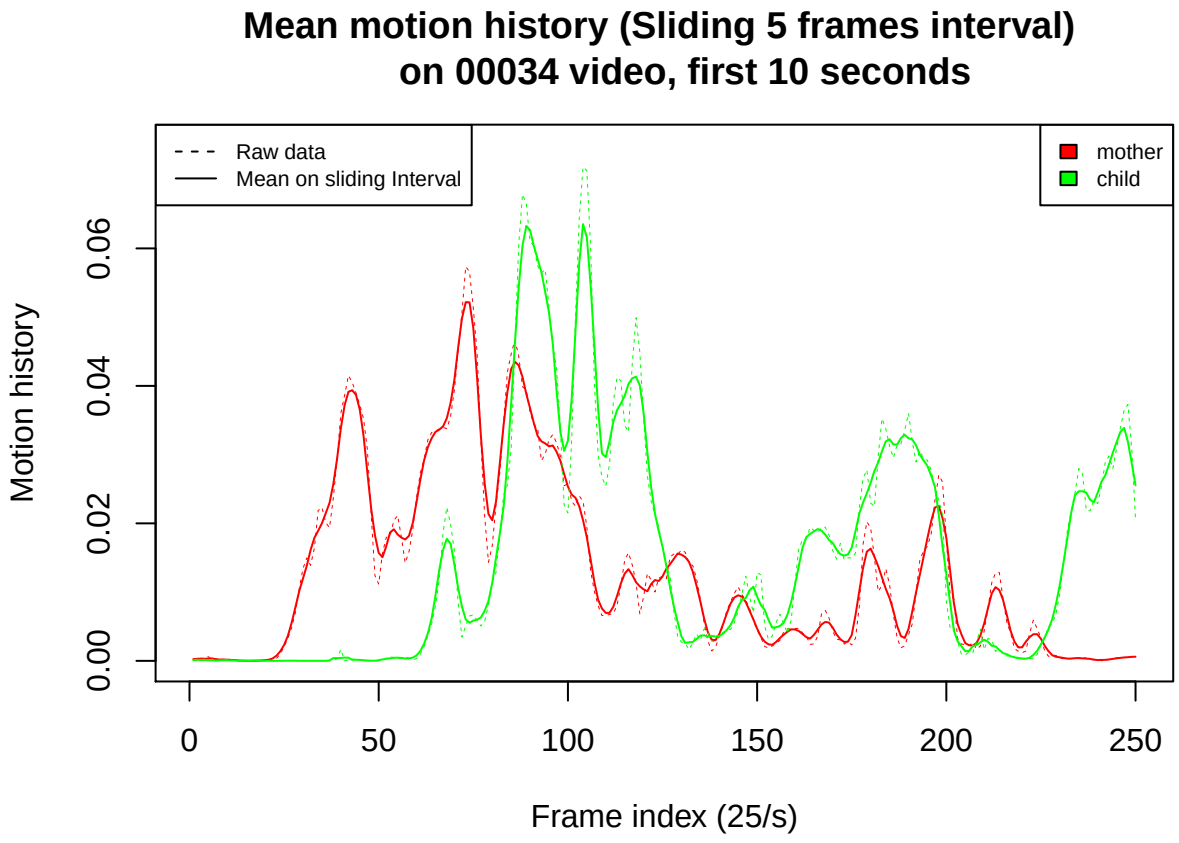
```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 5.400e-07 5.269e-04 3.015e-03 6.013e-03 8.504e-03 1.529e-01
```

Focus on the motion history of the first 10 seconds of the first video 00034

Sliding interval function on a 5 frames interval

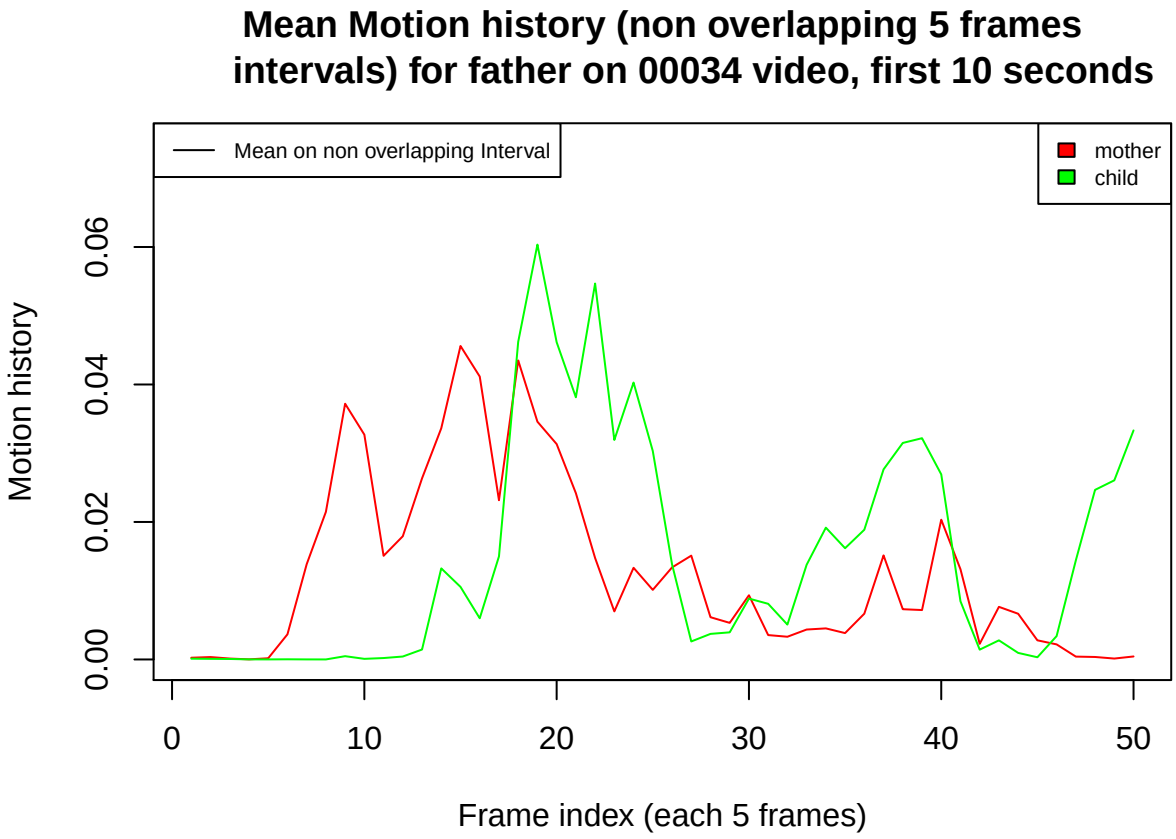
```
par(mar=c(4,4,4,2))
plot(1:250, data$mother[3:252], main="Mean motion history (Sliding 5 frames interval)
     on 00034 video, first 10 seconds ", xlab="Frame index (25/s)",
     ylab="Motion history",
     col="red", type="l", lty=2, lwd=0.5, ylim=c(0, 0.075))
lines(slidedMother[1:250],  col="red", lty=1)
lines(slidedChild[1:250],  col="green", lty=1)
lines(data$child[3:252],  col="green", lty=2, lwd=0.5)
```

```
legend("topleft", c("Raw data", "Mean on sliding Interval"), lty=c(2, 1), cex=0.7)
legend("topright", ParticipantsList[c(2,3)], fill=colOrderList[c(2,3)], cex=0.7)
```



Non overlapping interval function on a 5 frames interval

```
par(mar=c(4,4,4,2))
plot (1:50, motherFive[1:50], type="l", col="red",
main="Mean Motion history (non overlapping 5 frames
      intervals) for father on 00034 video, first 10 seconds",
ylab="Motion history", xlab="Frame index (each 5 frames)", ylim=c(0, 0.075))
lines(childFive[1:50], col="green", lty=1)
legend("topleft", "Mean on non overlapping Interval" , lty=1, cex=0.7)
legend("topright", ParticipantsList[c(2,3)], fill=colOrderList[2:3], cex=0.7)
```



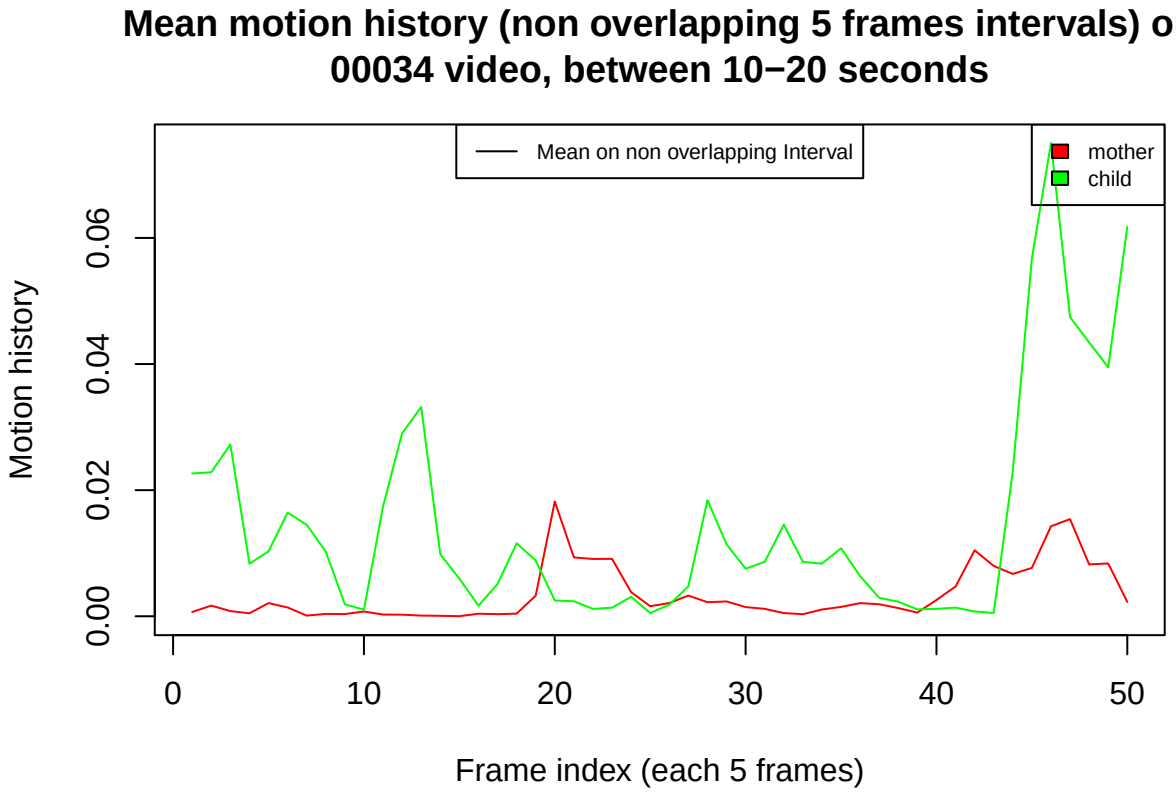
Motion history of the father during 10-20 seconds of the first video 00034

Non overlapping interval function on a 5 frames interval

```
par(mar=c(4,4,4,2))
plot(1:250, data$mother[253:502], main="Mean motion history (Sliding 5 frames
     interval) for father on 00034 video, 10-20 seconds", xlab="Frame index (25/s)",
     ylab="Motion history", col="red", type="l", lty=2, lwd=0.5, ylim=c(0, 0.075))
lines(slidedMother[251:500],  col="red", lty=1)
lines(data$child[253:502],  col="green", lty=2, lwd=0.5)
lines(slidedChild[251:500],  col="green", lty=1)
legend("topleft", c("Raw data", "Mean on sliding Interval") , lty=c(2, 1), cex=0.7)
legend("topright", ParticipantsList[c(2,3)], fill=colOrderList[c(2,3)], cex=0.7)
```

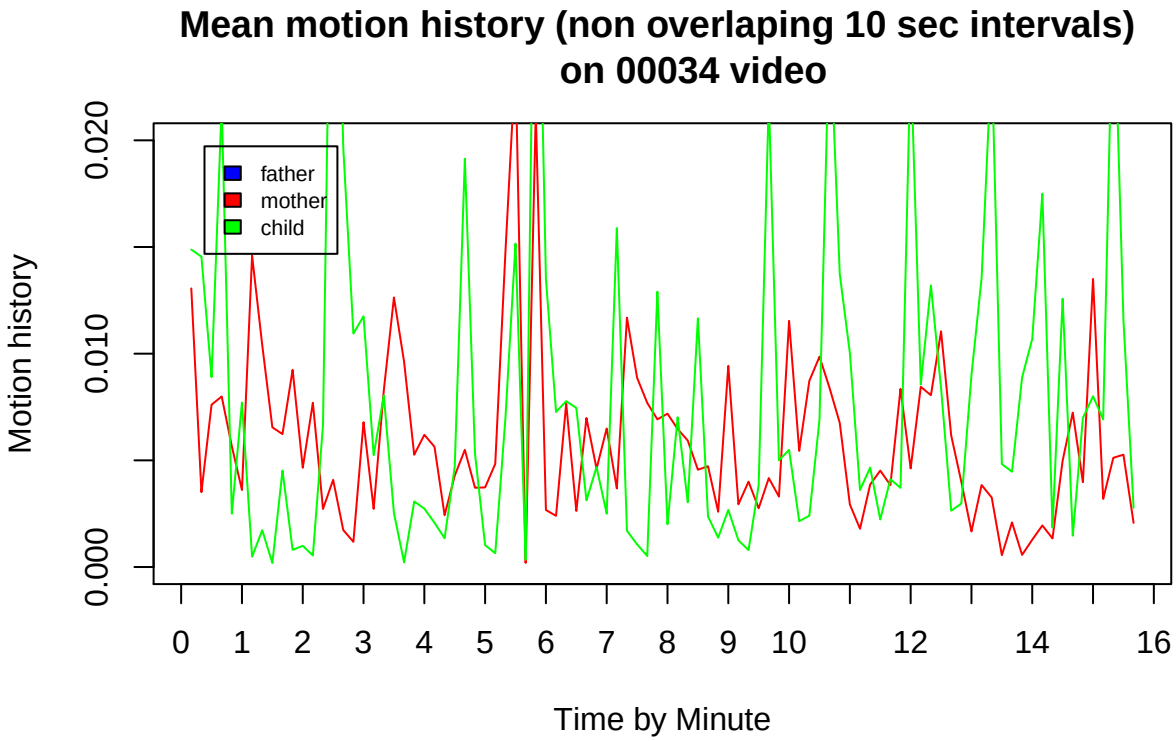
Non overlapping interval function on a 5 frames interval

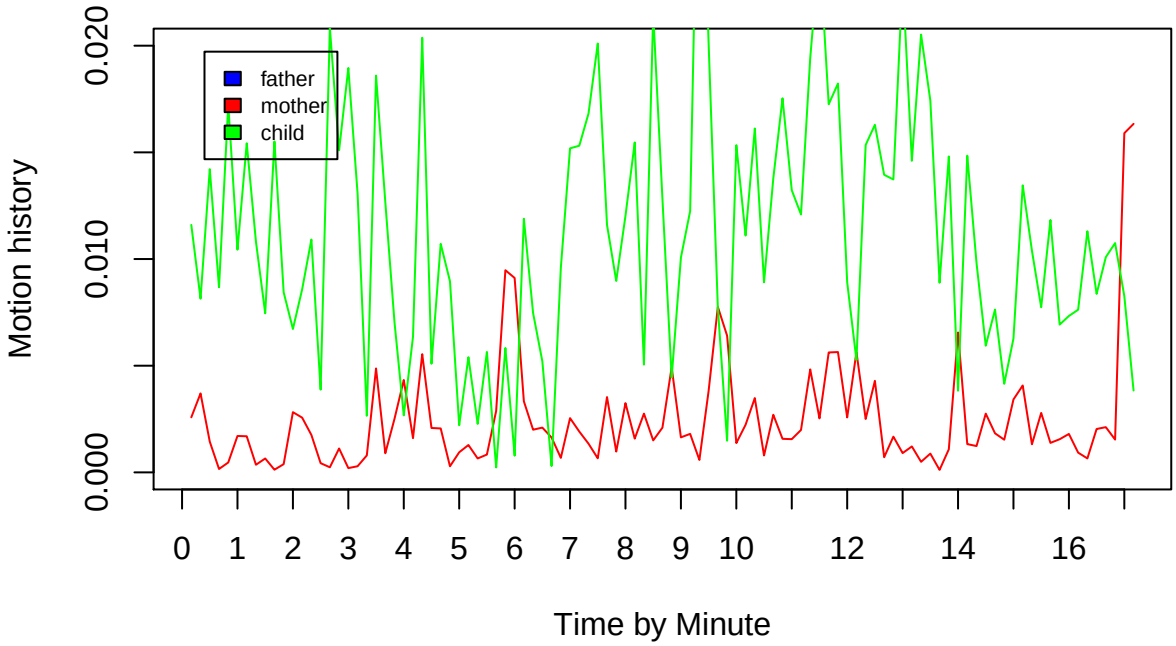
```
plot (1:50, motherFive[51:100], type="l", col="red",
main="Mean motion history (non overlapping 5 frames intervals) on
00034 video, between 10-20 seconds",
ylab="Motion history", xlab="Frame index (each 5 frames)", ylim=c(0, 0.075))
lines(childFive[51:100], col="green", lty=1)
legend("top", "Mean on non overlapping Interval" , lty=1, cex=0.7)
legend("topright", ParticipantsList[c(2,3)], fill=colOrderList[c(2,3)], cex=0.7)
```

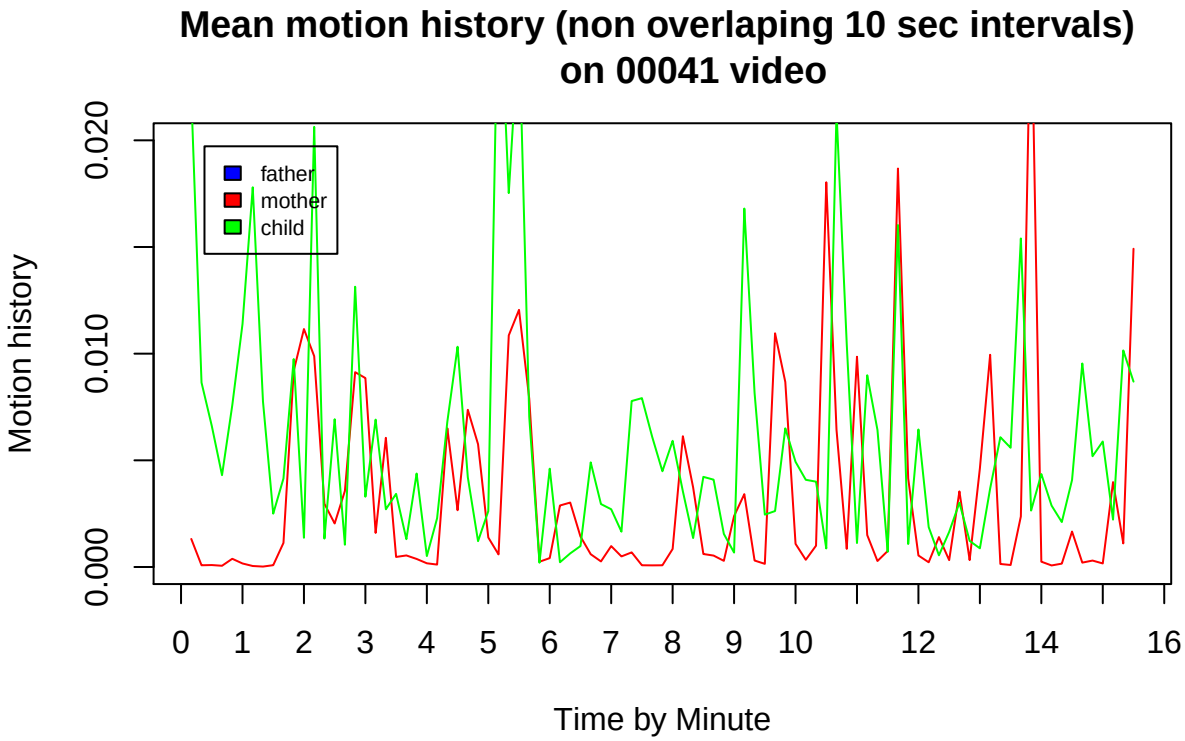


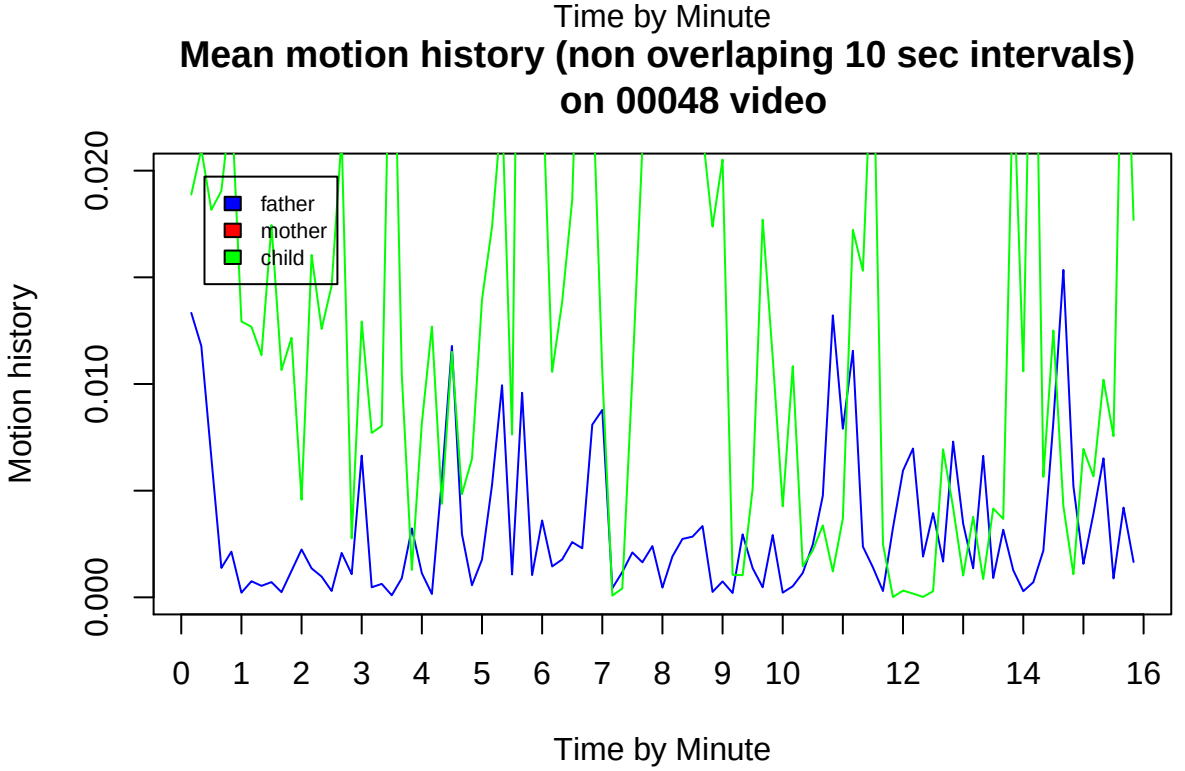
Mean motion history by 10 sec plots

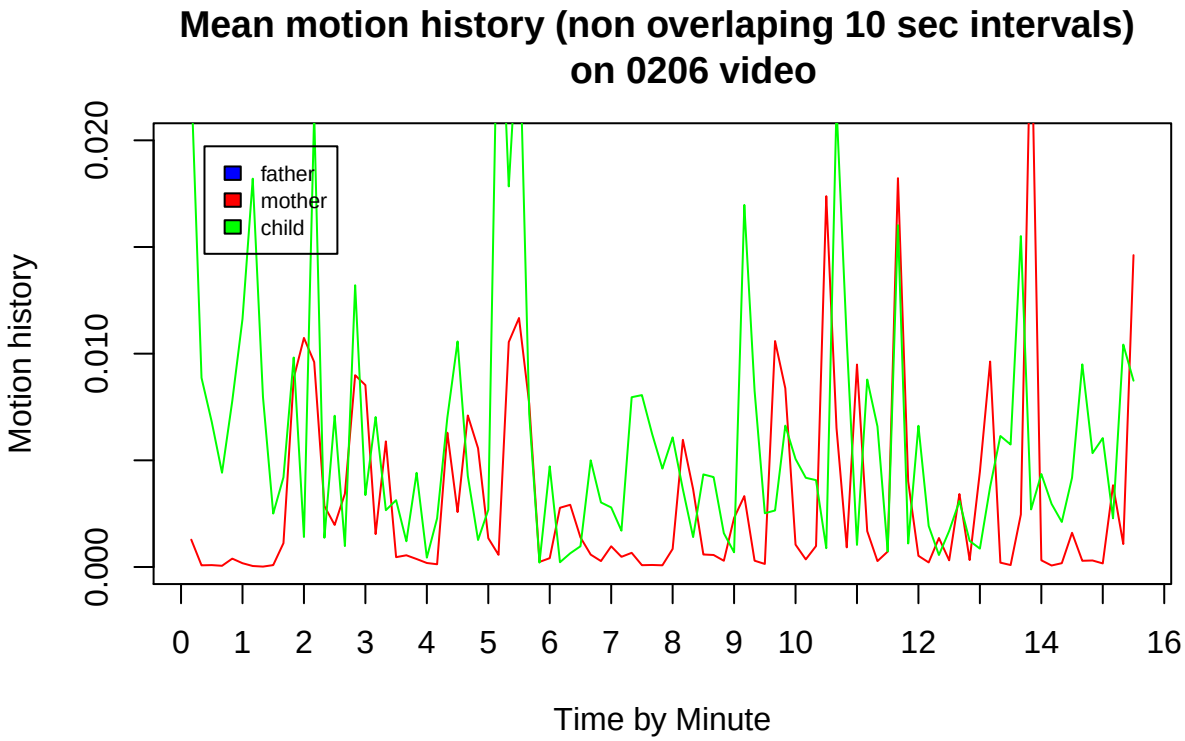
```
for (i in 1:NumberOfvideos){
  fatherMinute<- MeanMotionByTime("father", indexOfvideos=i, interval=250, data)

  motherMinute<- MeanMotionByTime("mother", indexOfvideos=i, interval=250, data)

  childMinute<- MeanMotionByTime("child", indexOfvideos=i, interval=250, data)

  par(mar=c(4,4,4,2))
      plot ((1:length(fatherMinute)/6), fatherMinute, type="l", col="blue",
      main=paste("Mean motion history (non overlaping 10 sec intervals)
      on ", families[i], " video" , sep=""),
      ylab="Motion history", xlab="Time by Minute", ylim=c(0, 20E-03),
      xaxp=c(0, (round(length(fatherMinute)/6)), round((length(fatherMinute)/6))))
      lines((1:length(fatherMinute)/6), motherMinute, col="red")
      lines((1:length(fatherMinute)/6), childMinute, col="green")
      legend("topleft", inset=.05, ParticipantsList[1:3],
             fill=colOrderList[1:3], cex=0.7)}
```

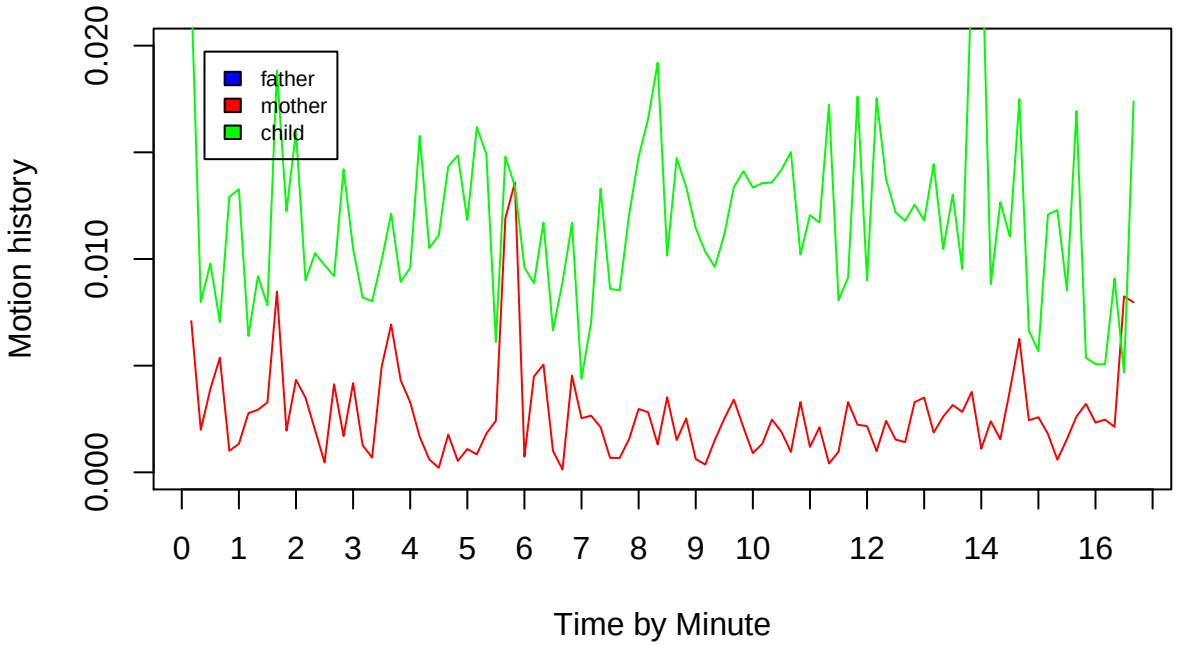

Mean motion history (non overlaping 10 sec intervals)
on 00034 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

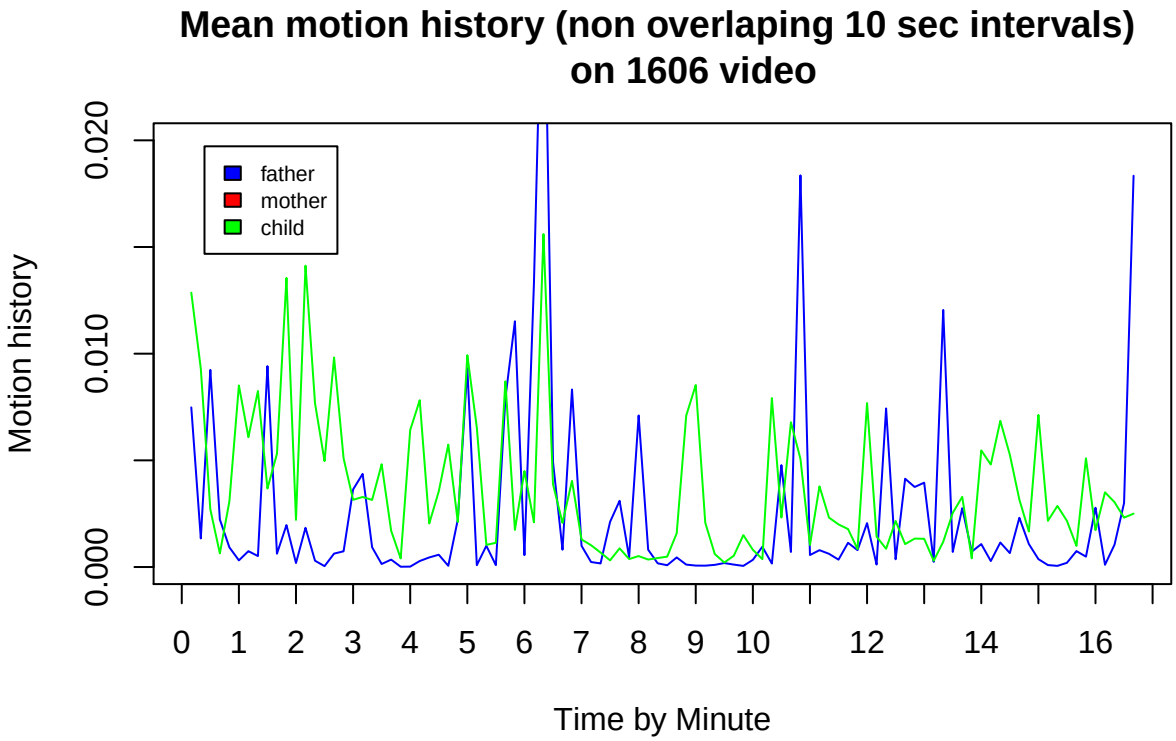

Mean motion history (non overlaping 10 sec intervals)
on 00037 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

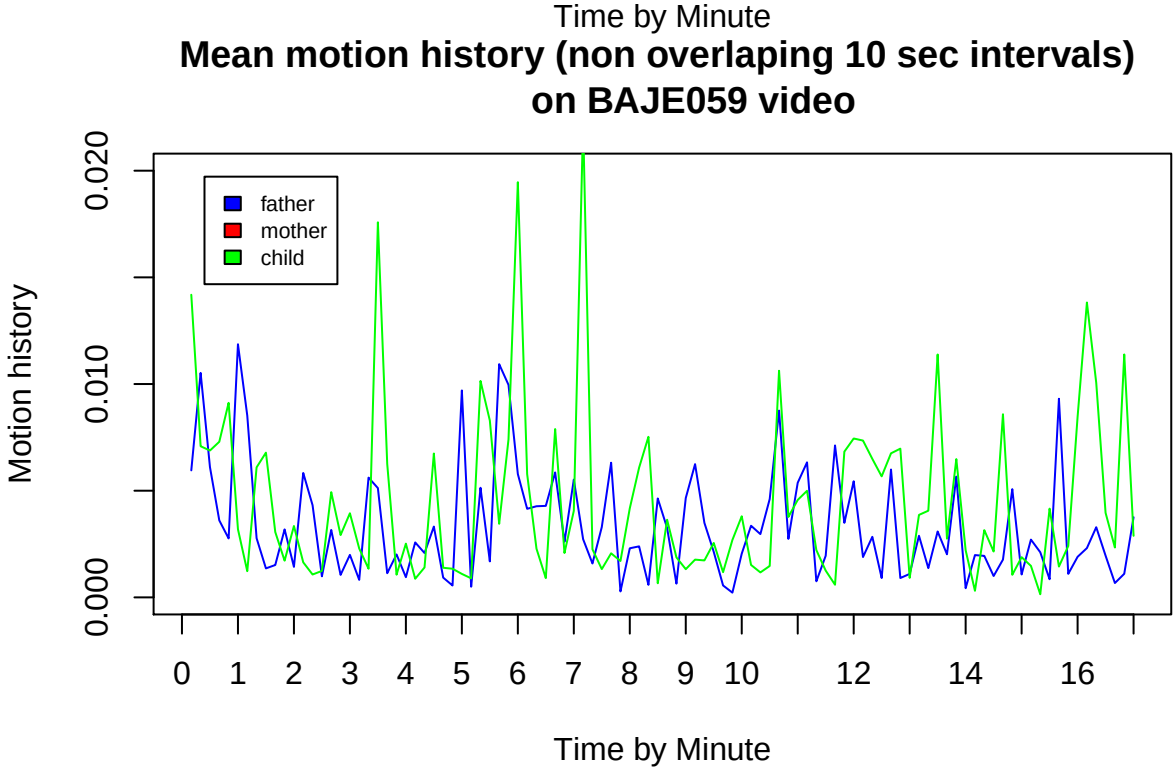

Mean motion history (non overlaping 10 sec intervals)
on 00041 video
Motion history
0.000
0.010
0.020
father
mother
child
0
1
2
3
4
5
6
7
8
9
10
12
14
16
Time by Minute

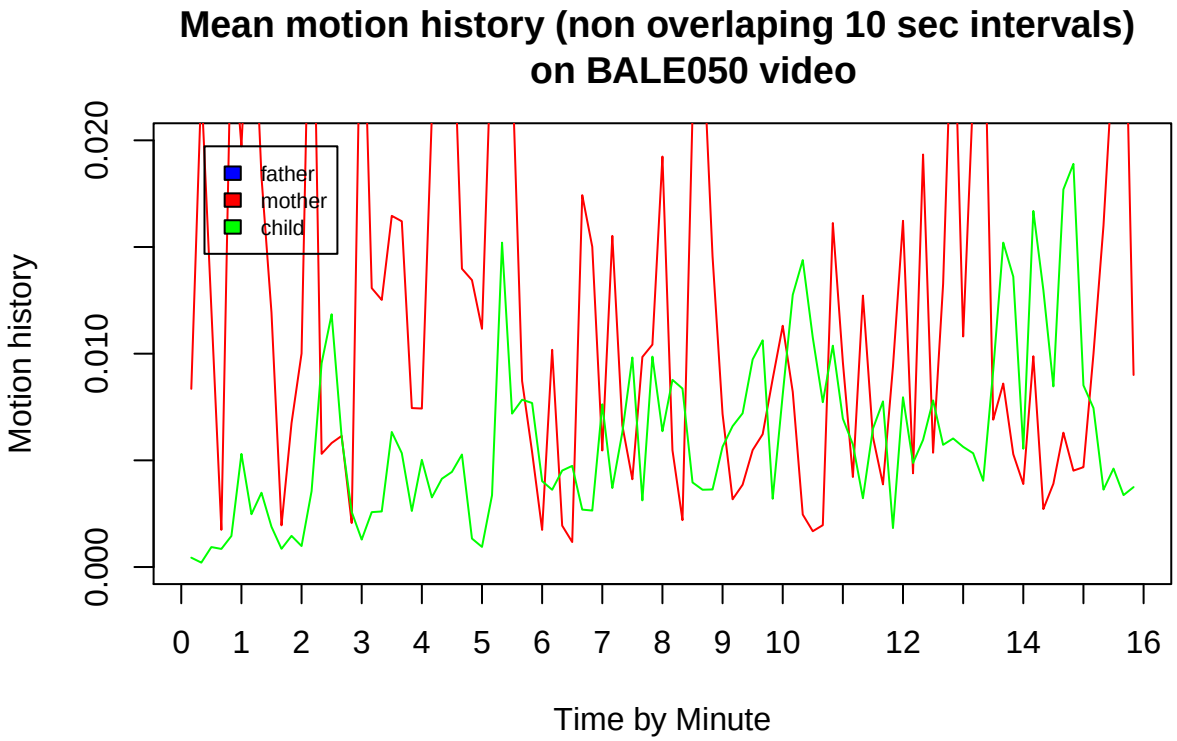

Mean motion history (non overlaping 10 sec intervals)
on 00048 video
Motion history
0.000
0.010
0.020
father
mother
child
0
1
2
3
4
5
6
7
8
9
10
12
14
16
Time by Minute

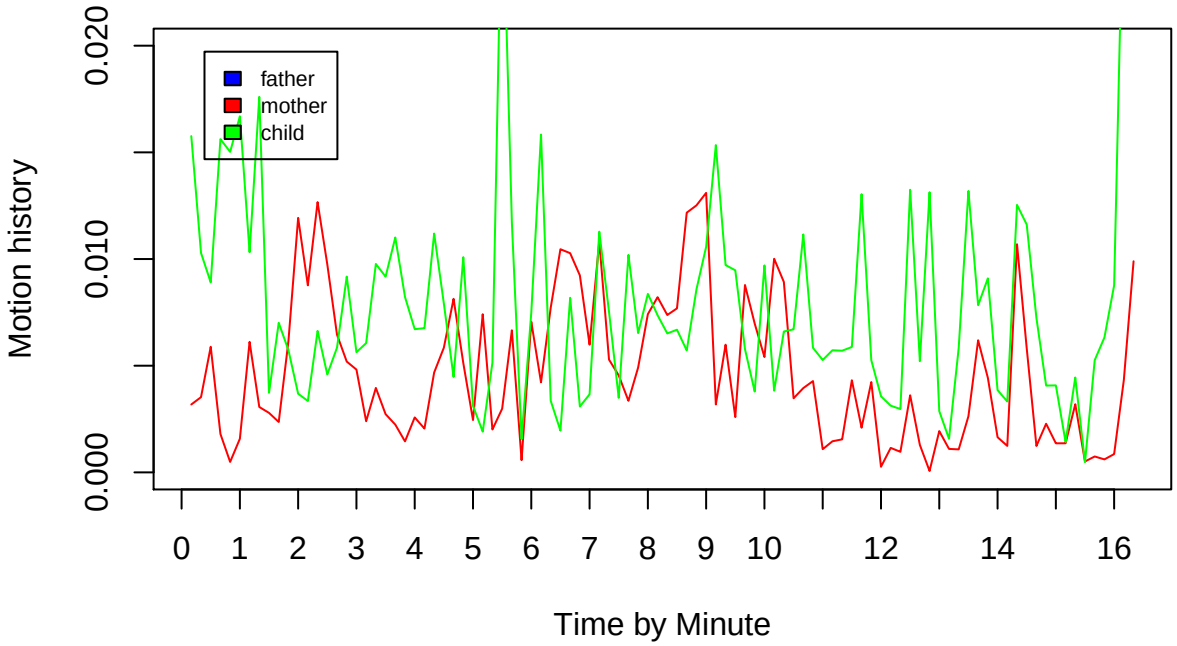

Mean motion history (non overlaping 10 sec intervals)
on 0206 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

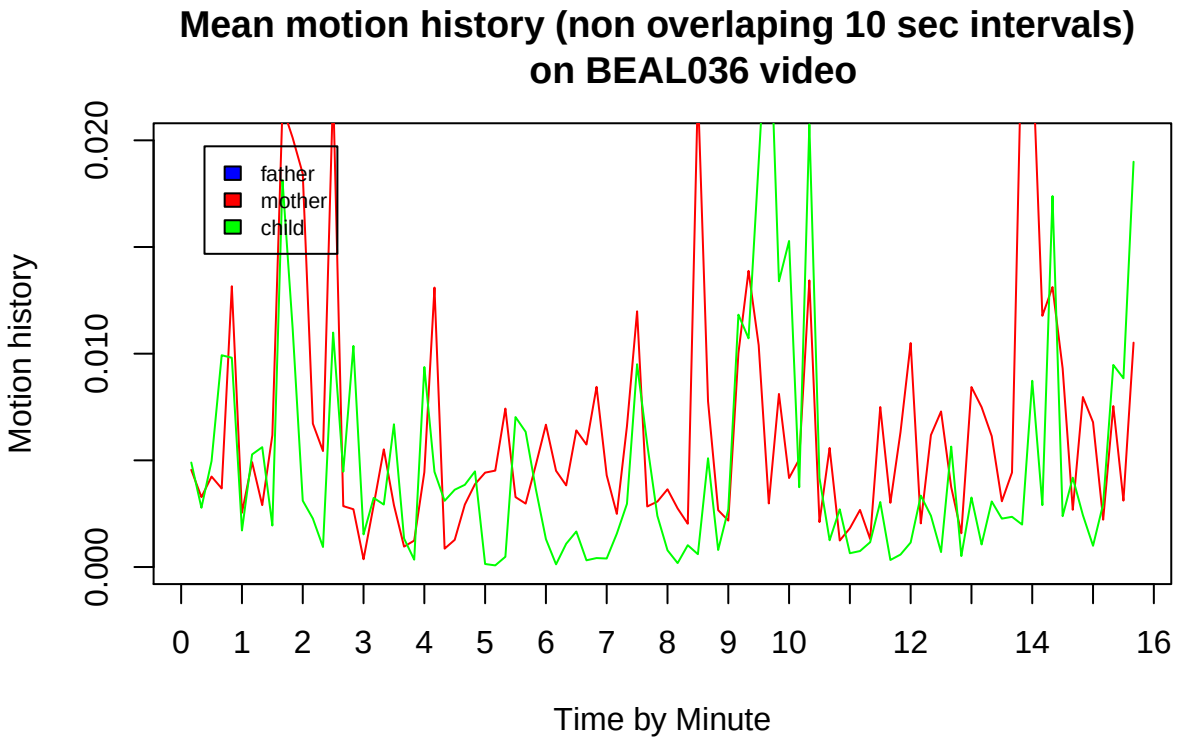

Mean motion history (non overlaping 10 sec intervals)
on 1106 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

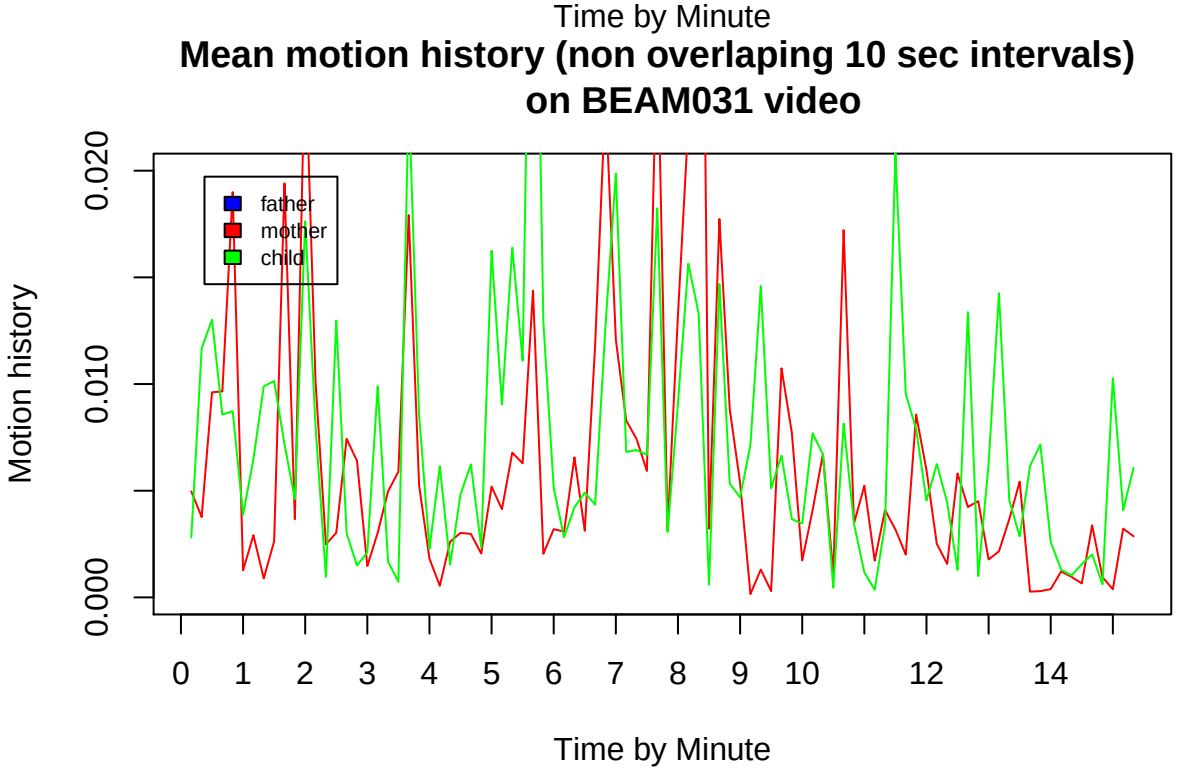

Mean motion history (non overlaping 10 sec intervals)
on 1606 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

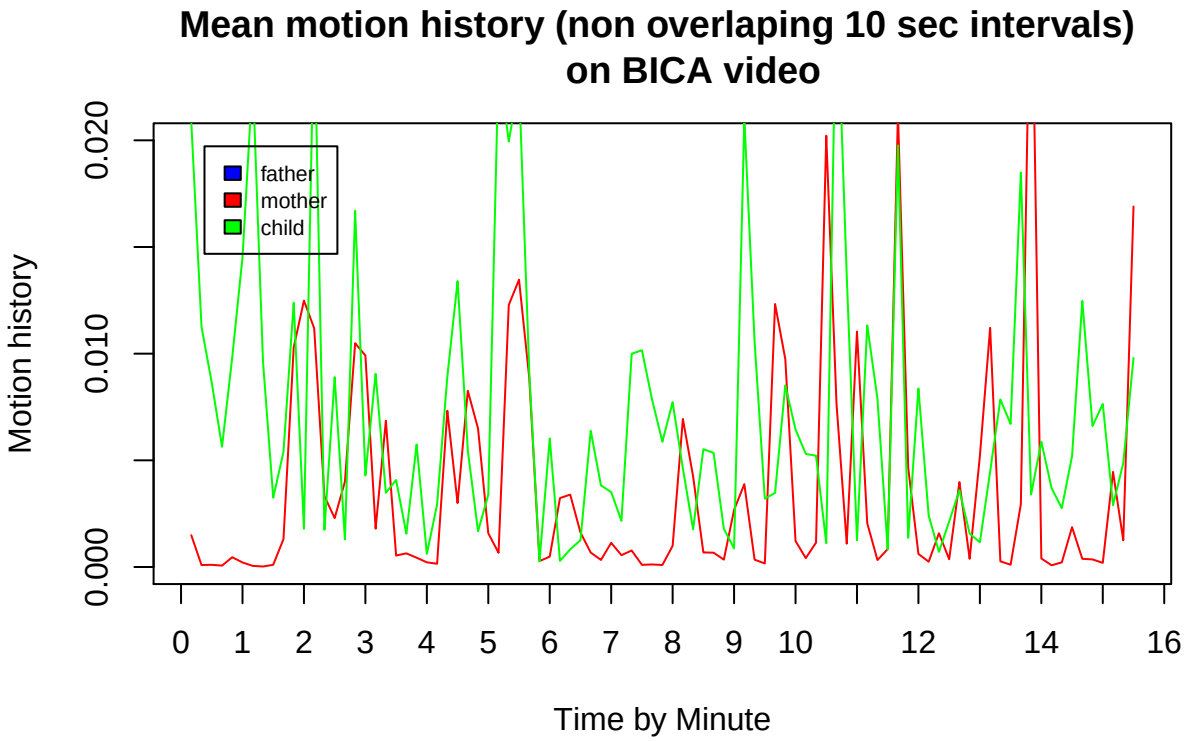

Mean motion history (non overlaping 10 sec intervals)
on BAJE059 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

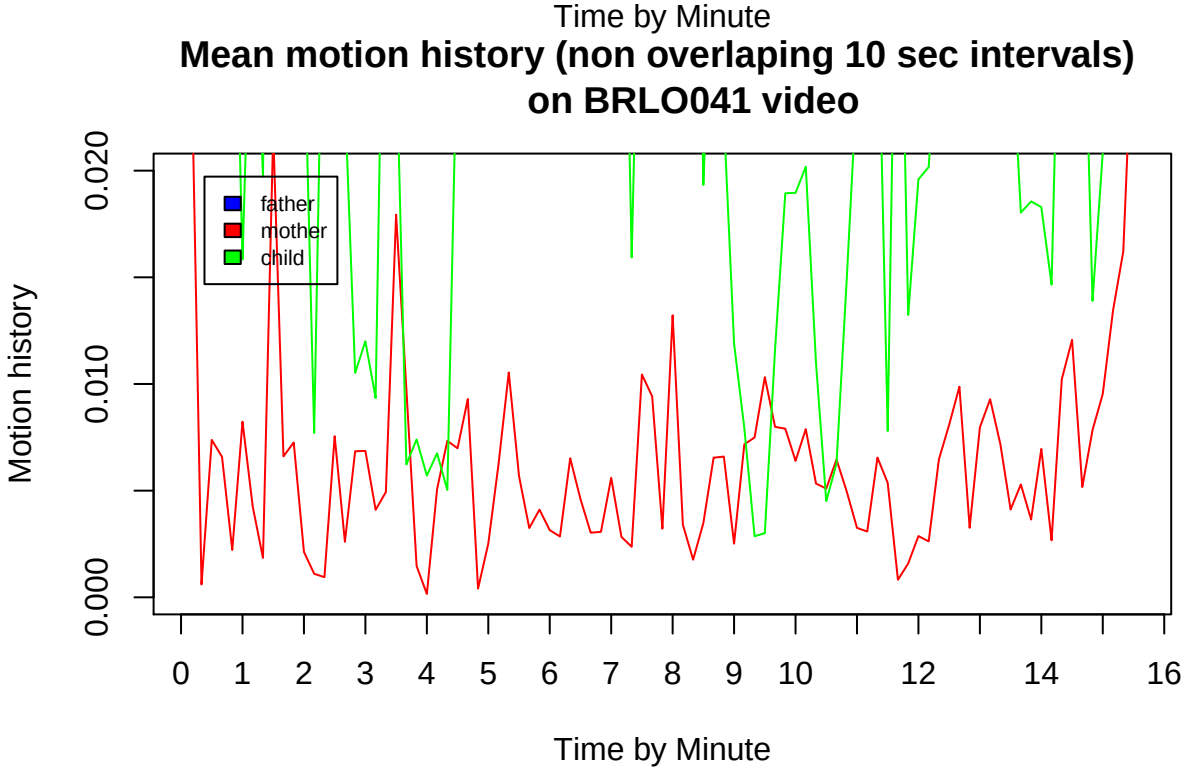

Mean motion history (non overlaping 10 sec intervals)
on BALE050 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

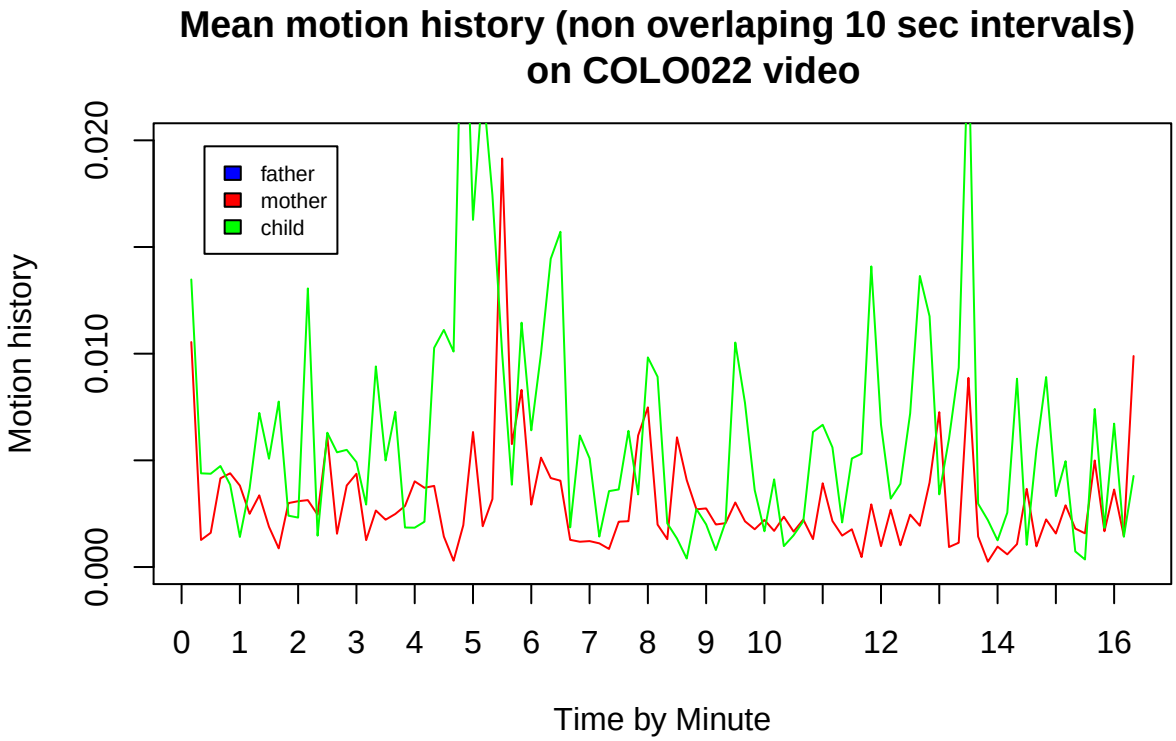

Mean motion history (non overlaping 10 sec intervals)
on BALU062 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

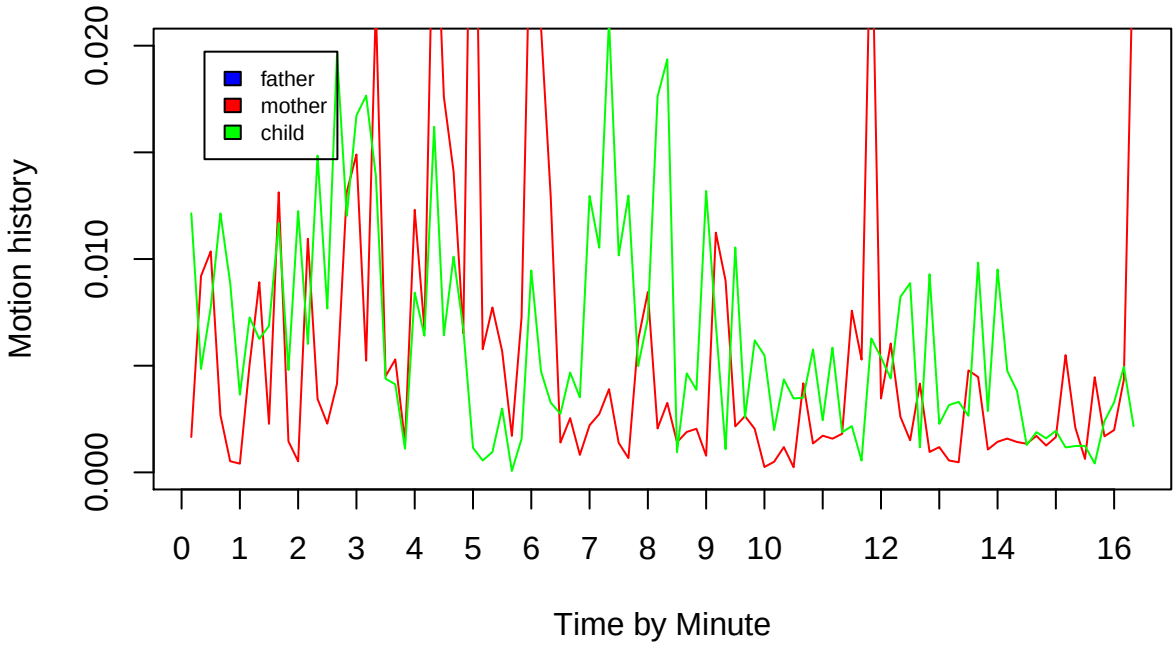

Mean motion history (non overlaping 10 sec intervals)
on BEAL036 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

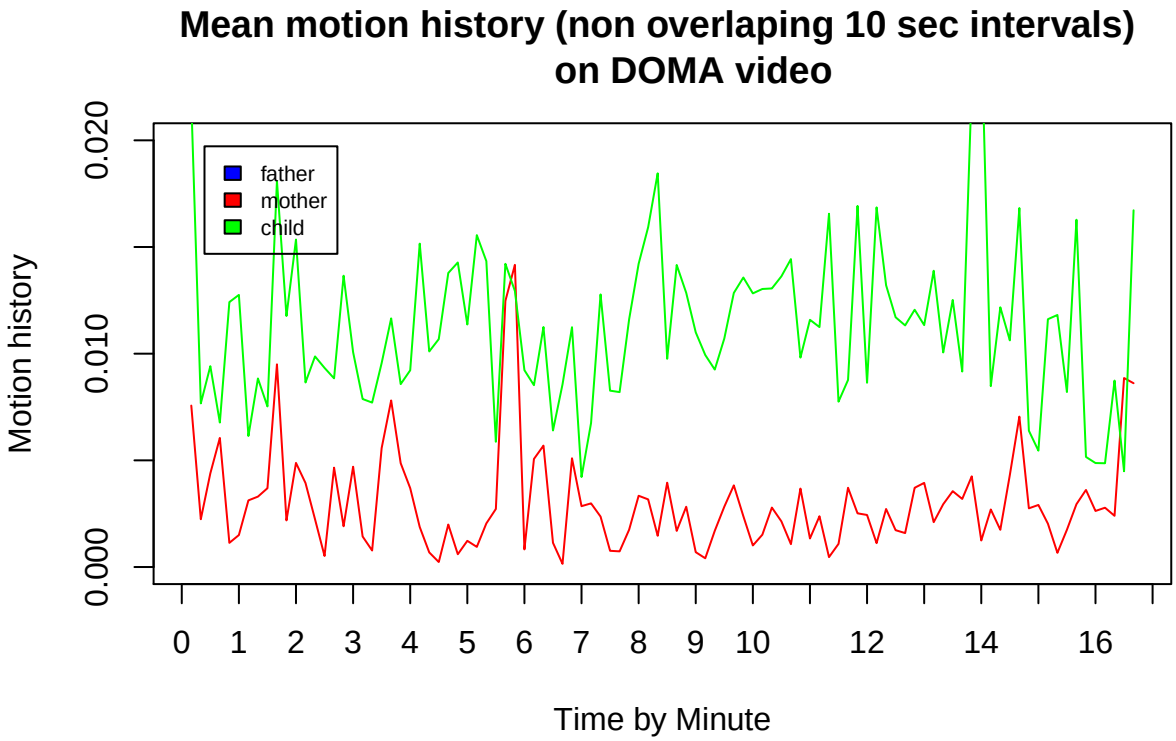

Mean motion history (non overlaping 10 sec intervals)
on BEAM031 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14
Time by Minute

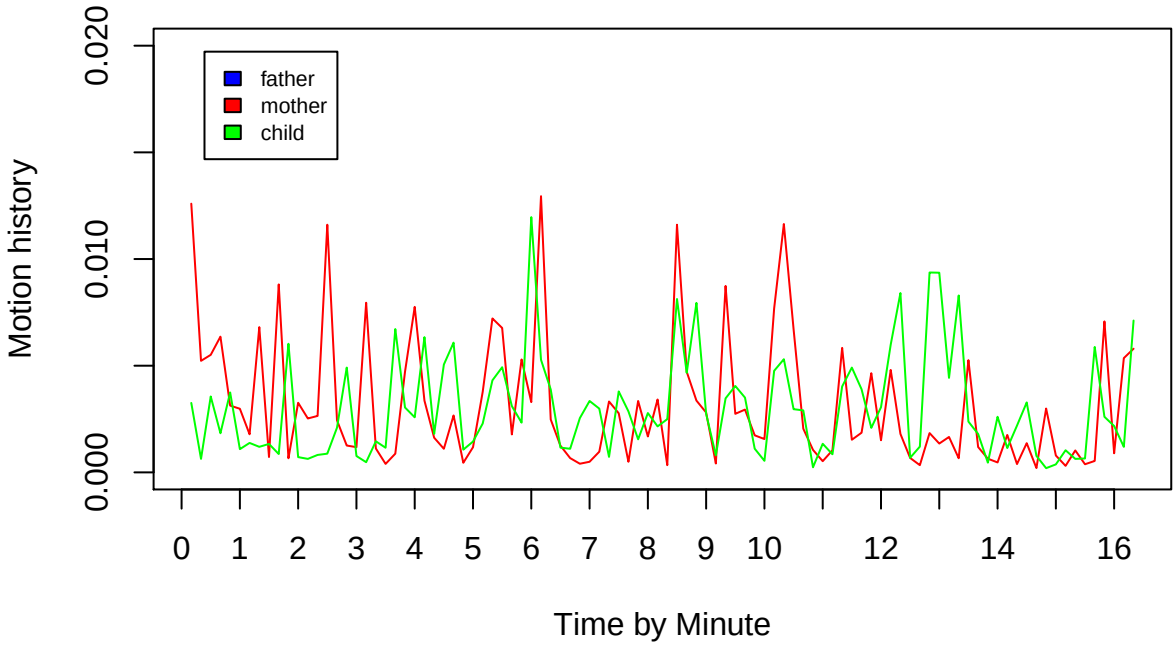

Mean motion history (non overlaping 10 sec intervals)
on BICA video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

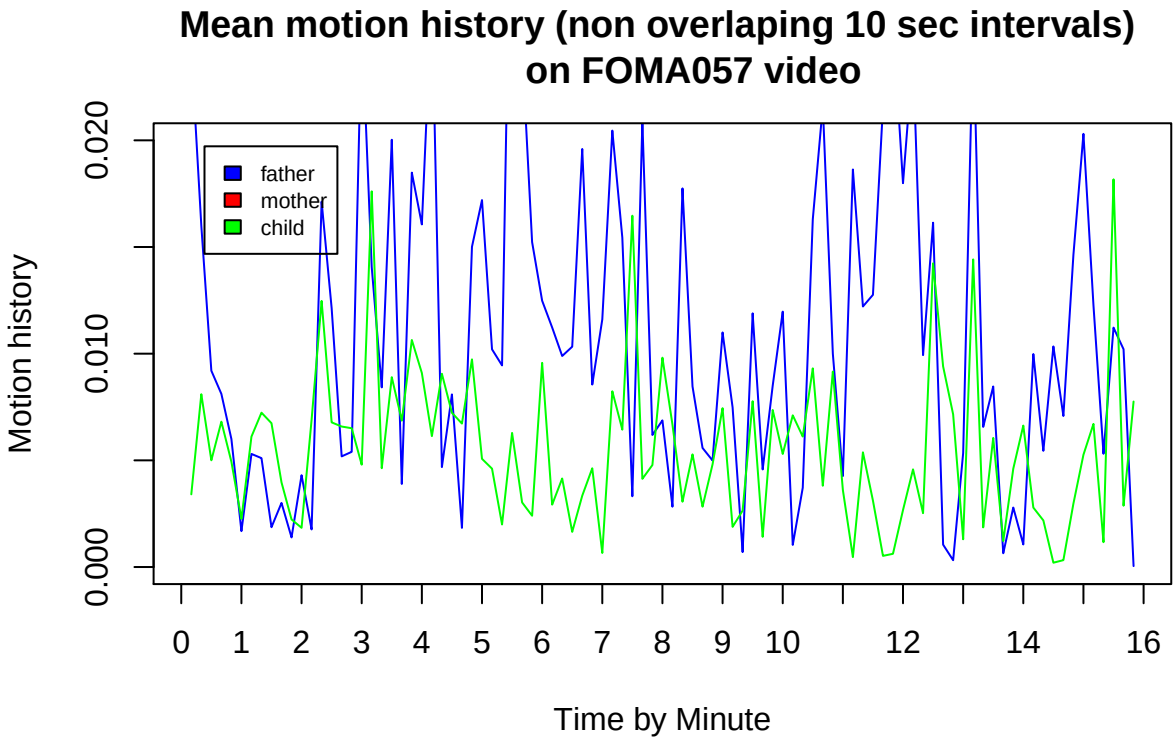

Mean motion history (non overlaping 10 sec intervals)
on BRLO041 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

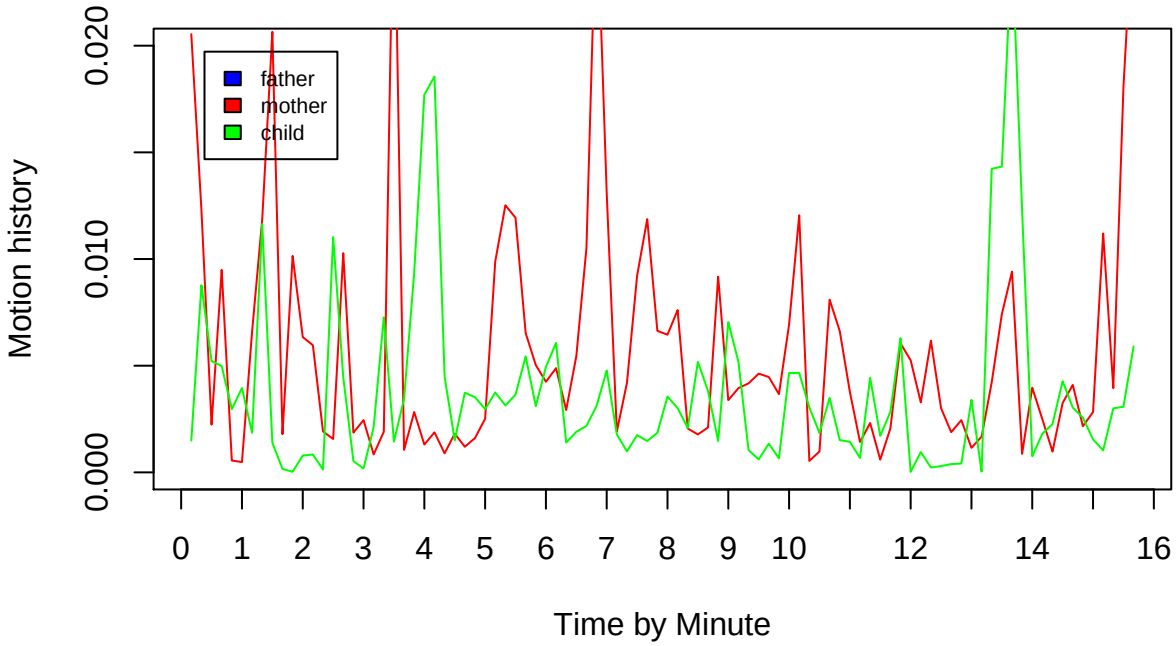

Mean motion history (non overlaping 10 sec intervals)
on COLO022 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

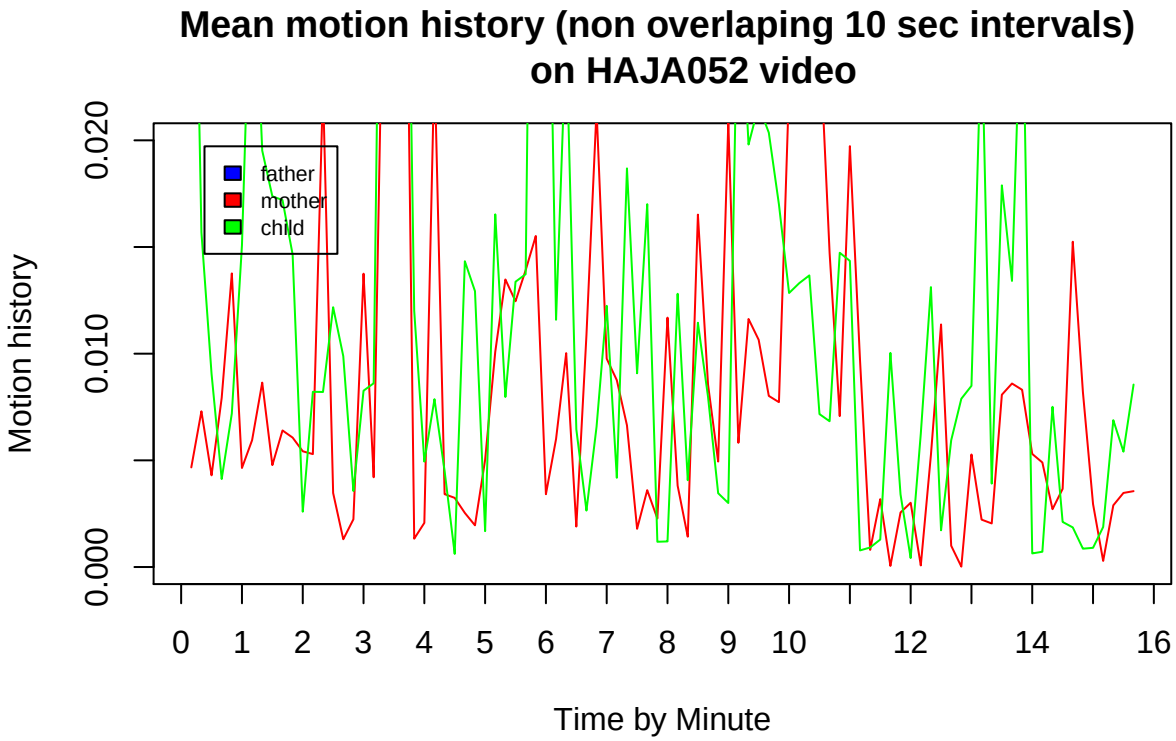

Mean motion history (non overlaping 10 sec intervals)
on DIPE004 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

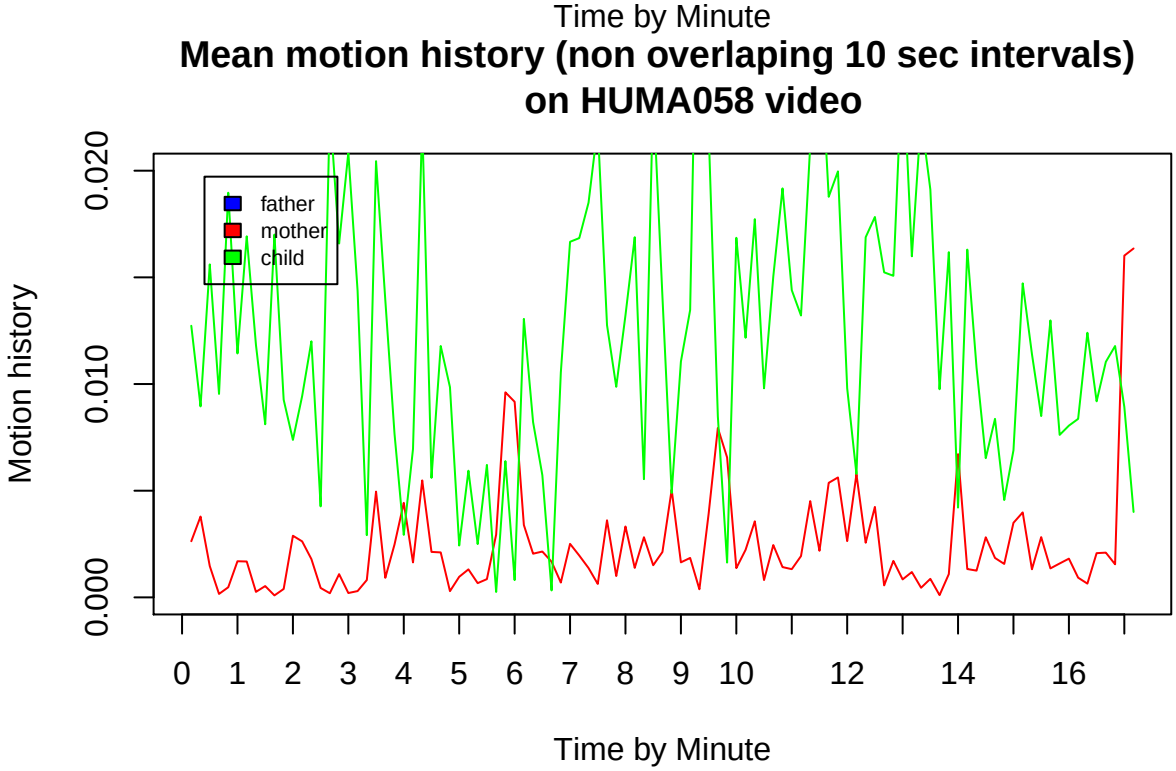

Mean motion history (non overlaping 10 sec intervals)
on DOMA video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

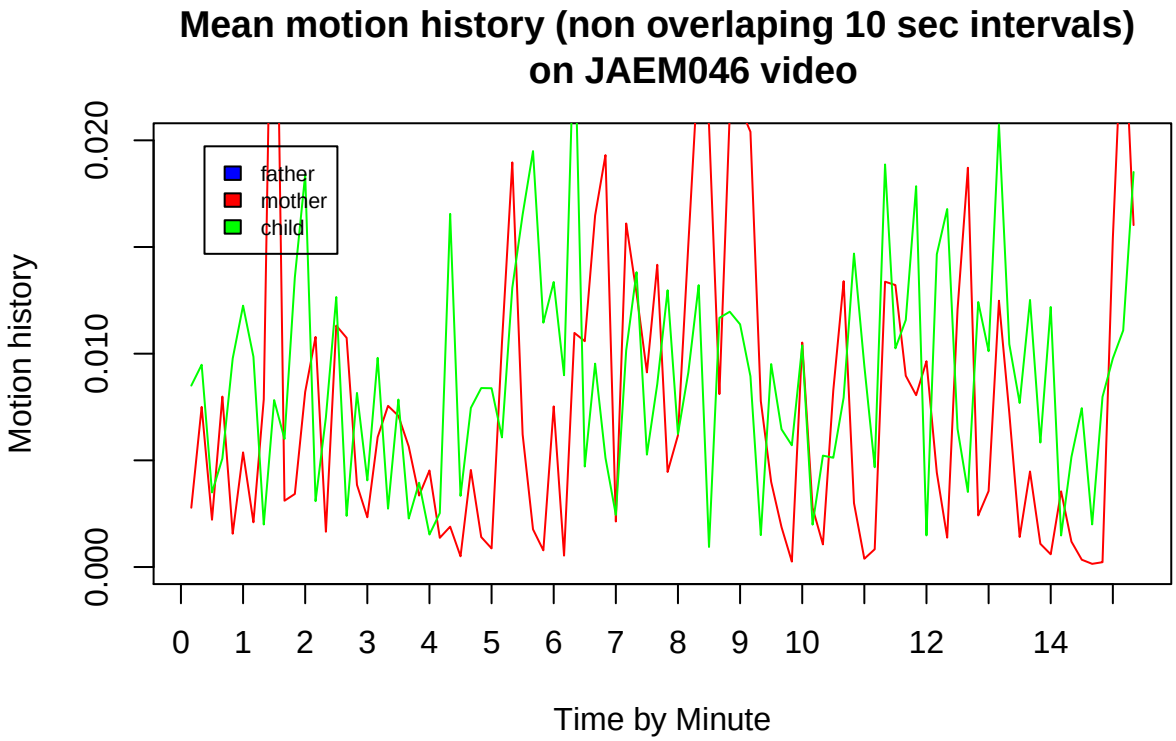

Mean motion history (non overlaping 10 sec intervals)
on DRNE video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

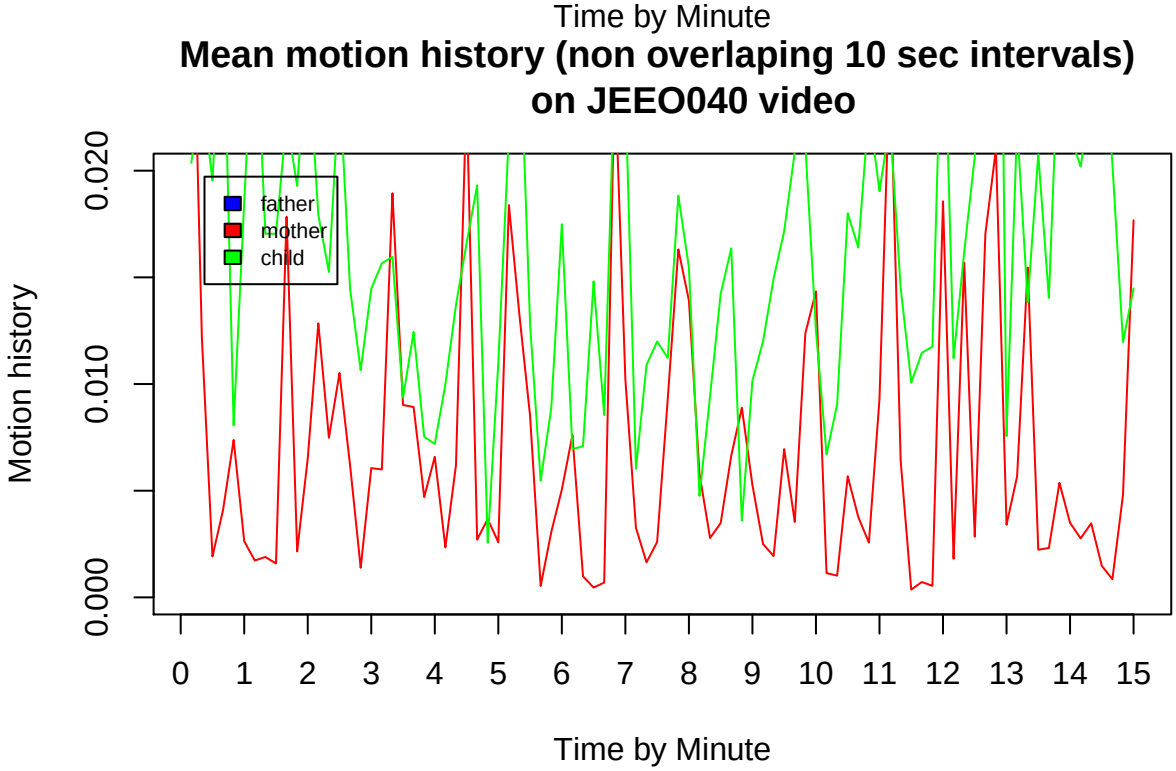

Mean motion history (non overlaping 10 sec intervals)
on FOMA057 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

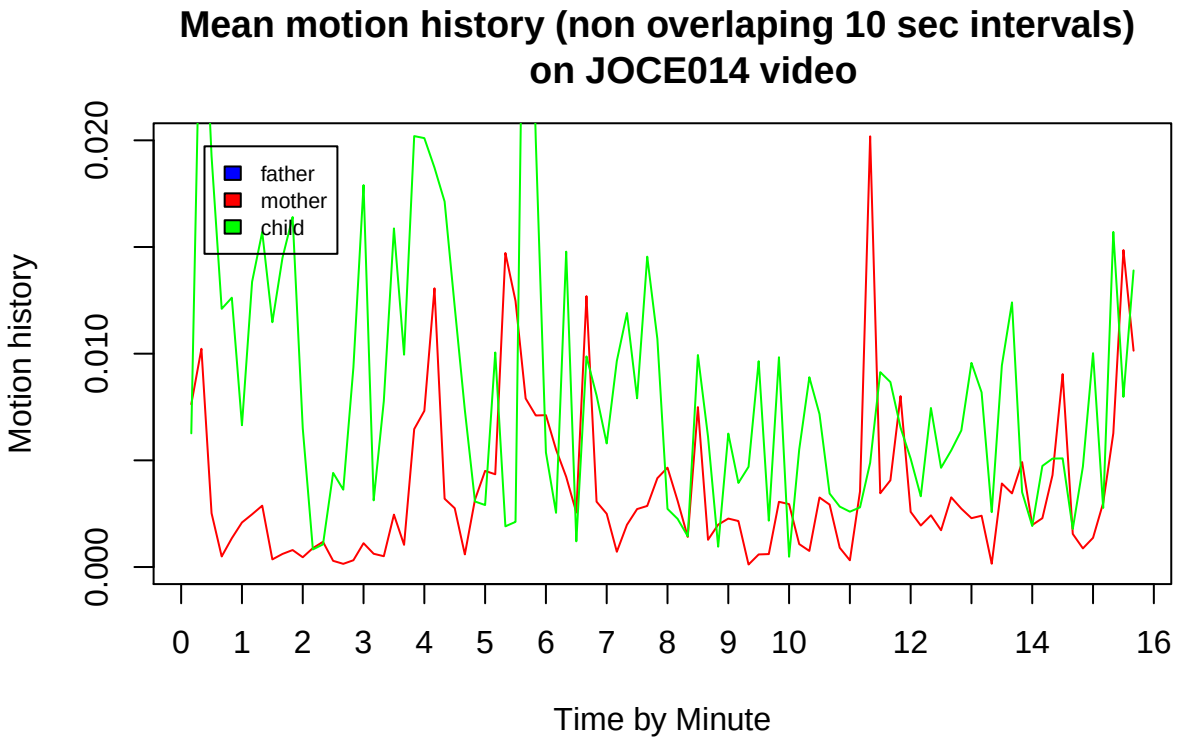

Mean motion history (non overlaping 10 sec intervals)
on GROP039 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

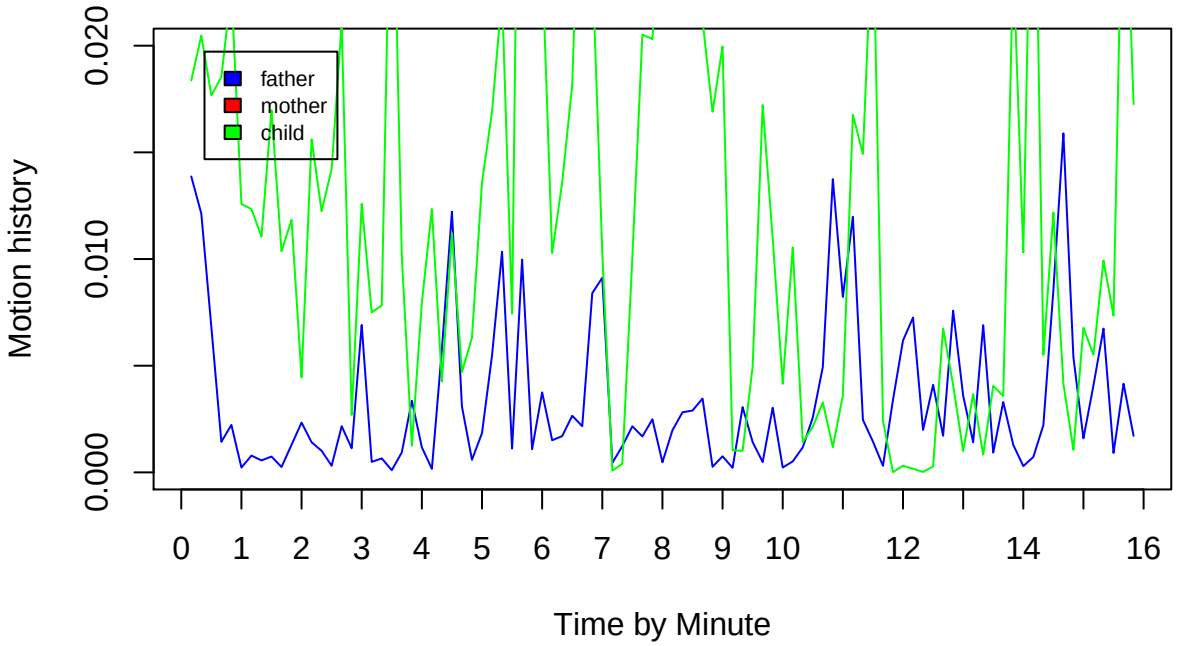

Mean motion history (non overlaping 10 sec intervals)
on HAJA052 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

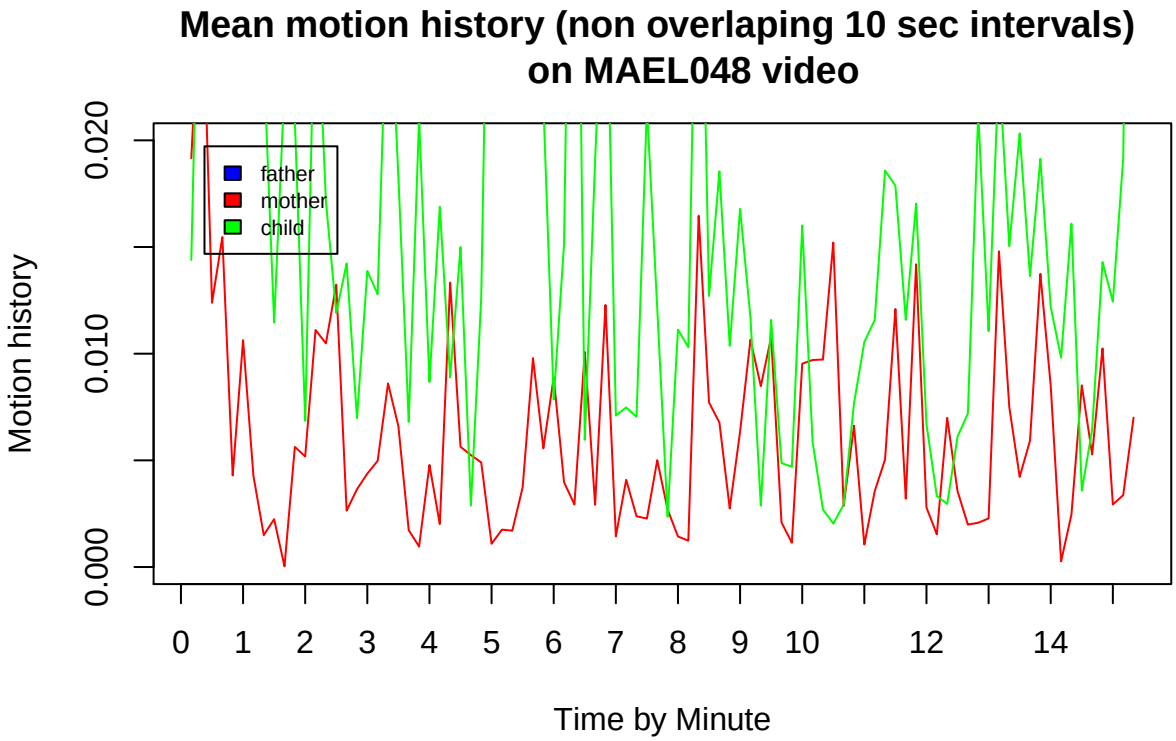

Mean motion history (non overlaping 10 sec intervals)
on HUMA058 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

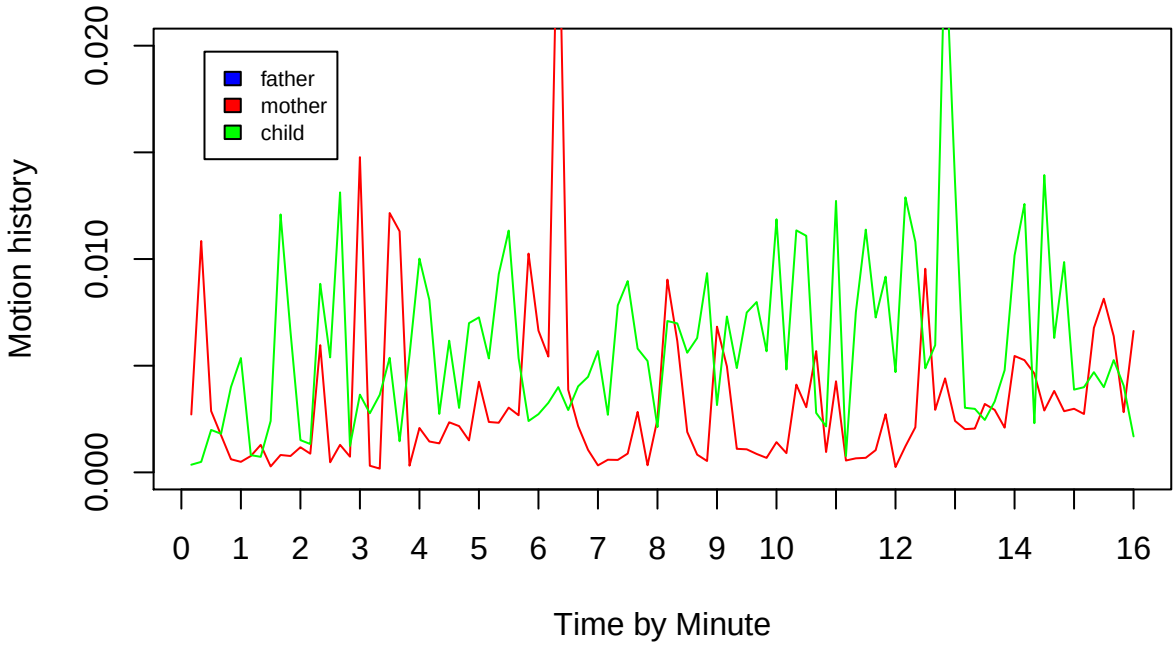

Mean motion history (non overlaping 10 sec intervals)
on JAEM046 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14
Time by Minute

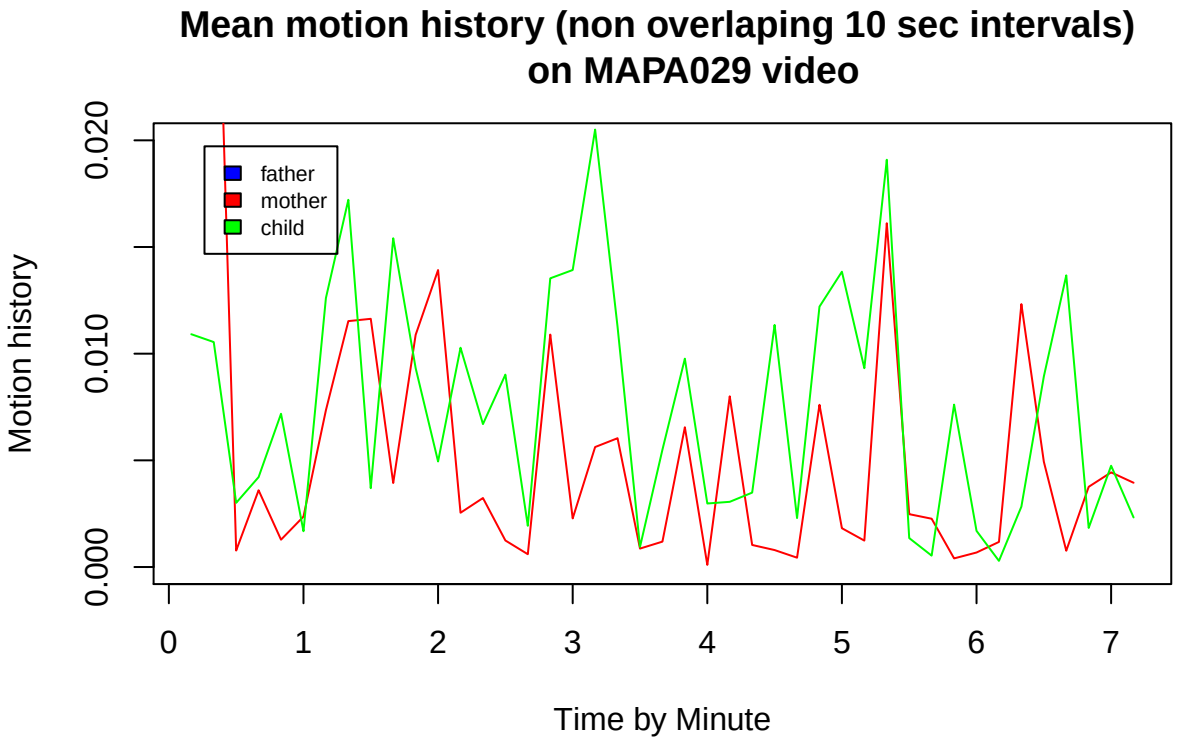

Mean motion history (non overlaping 10 sec intervals)
on JEEO040 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
Time by Minute

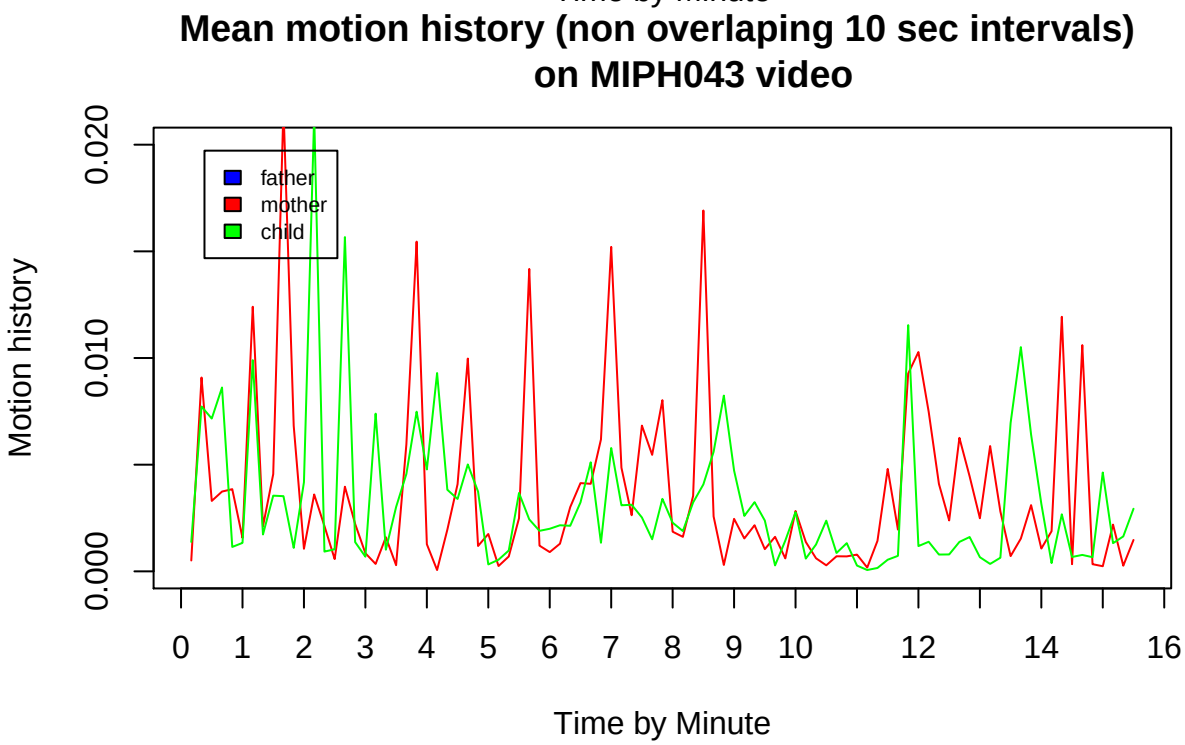

Mean motion history (non overlaping 10 sec intervals)
on JOCE014 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

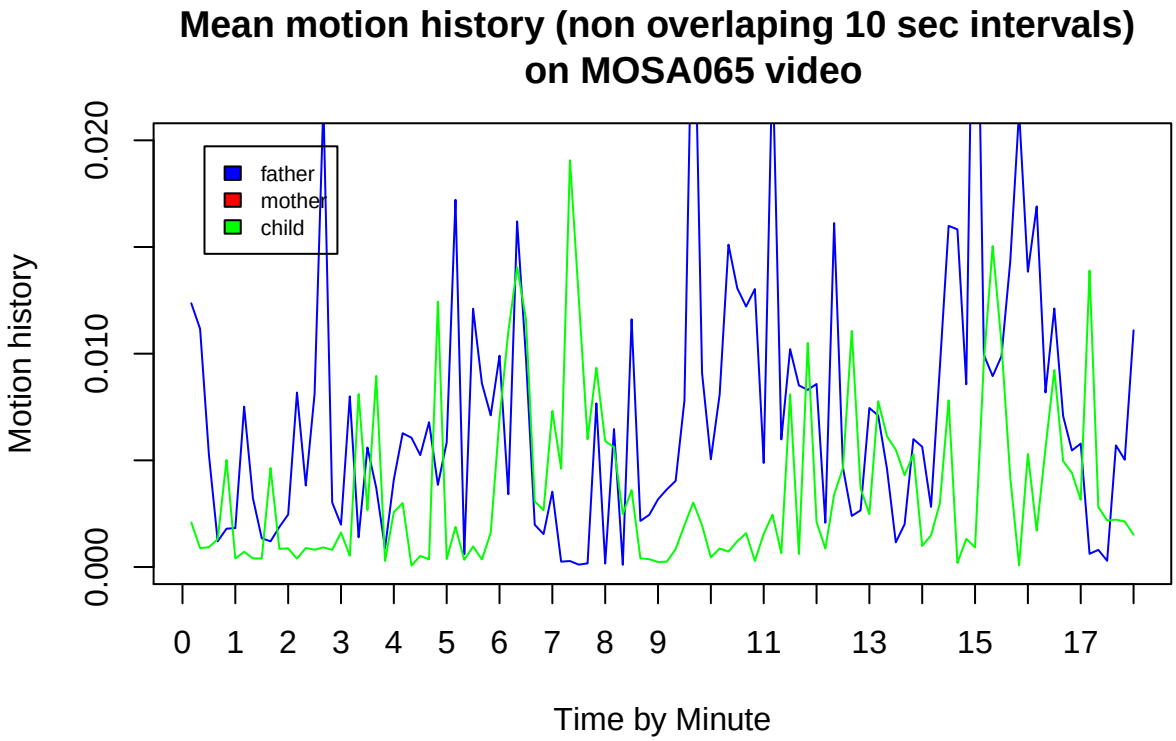

Mean motion history (non overlaping 10 sec intervals)
on LACL video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

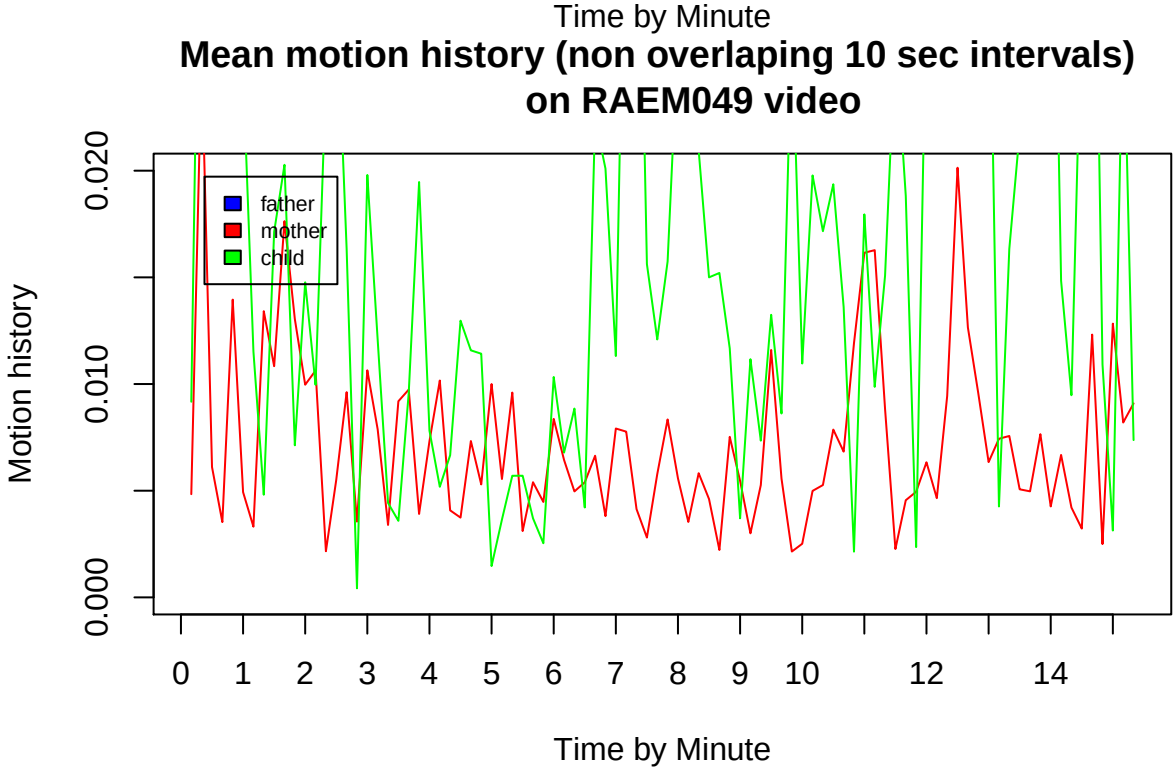

Mean motion history (non overlaping 10 sec intervals)
on MAEL048 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14
Time by Minute

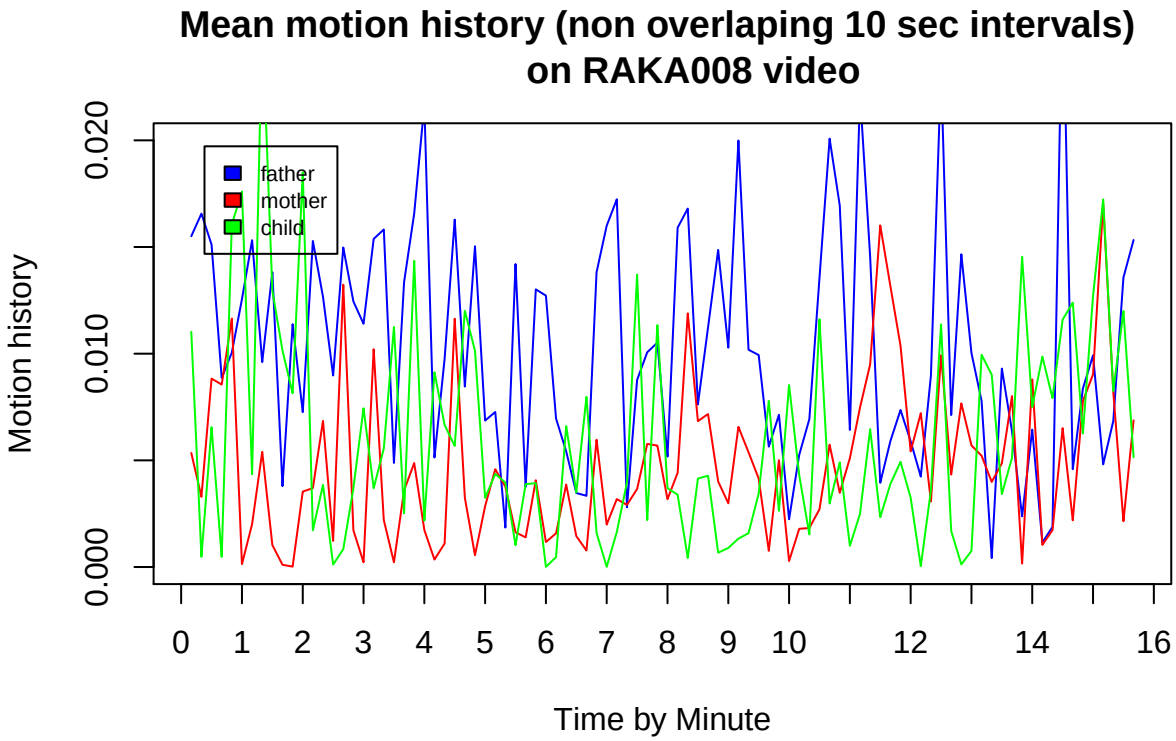

Mean motion history (non overlaping 10 sec intervals)
on MAME20 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

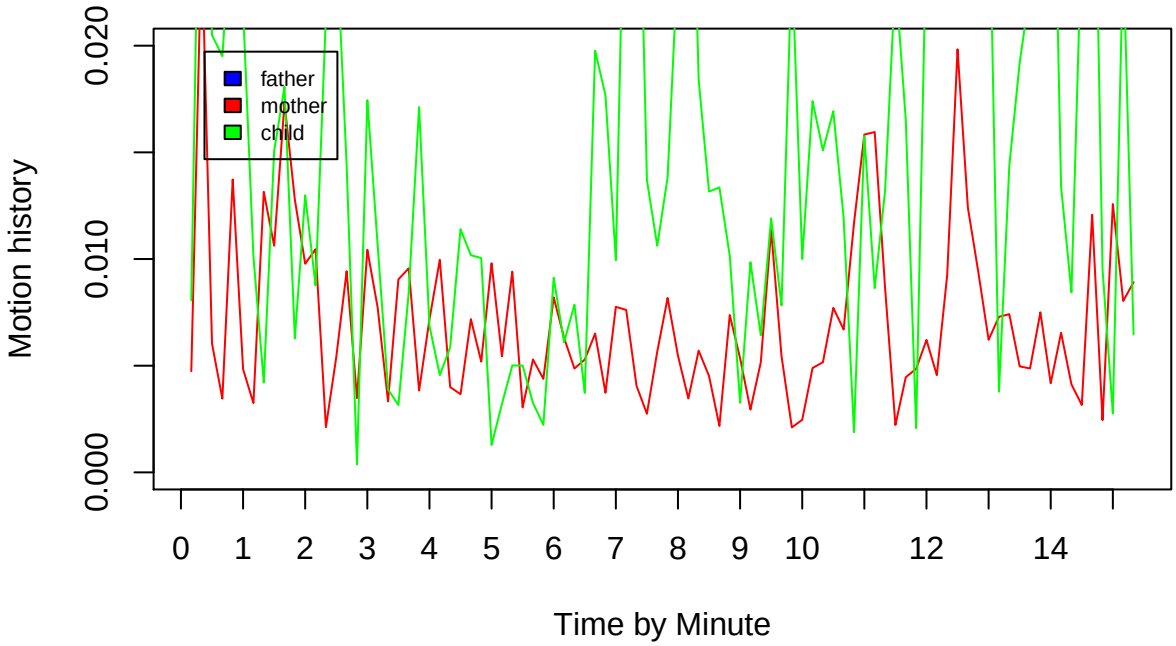

Mean motion history (non overlaping 10 sec intervals)
on MAPA029 video
Motion history
0.000
0.010
0.020
father
mother
child
0
1
2
3
4
5
6
7
Time by Minute

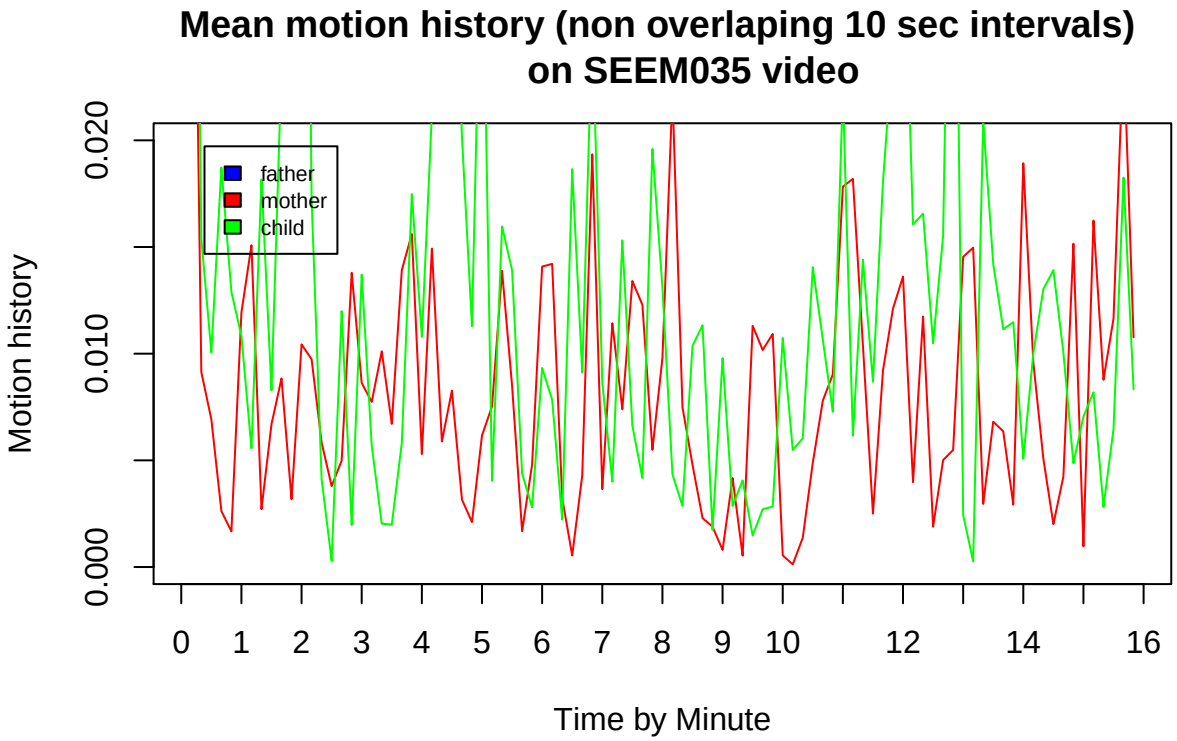

Mean motion history (non overlaping 10 sec intervals)
on MIPH043 video
Motion history
0.000
0.010
0.020
father
mother
child
0
1
2
3
4
5
6
7
8
9
10
12
14
16
Time by Minute

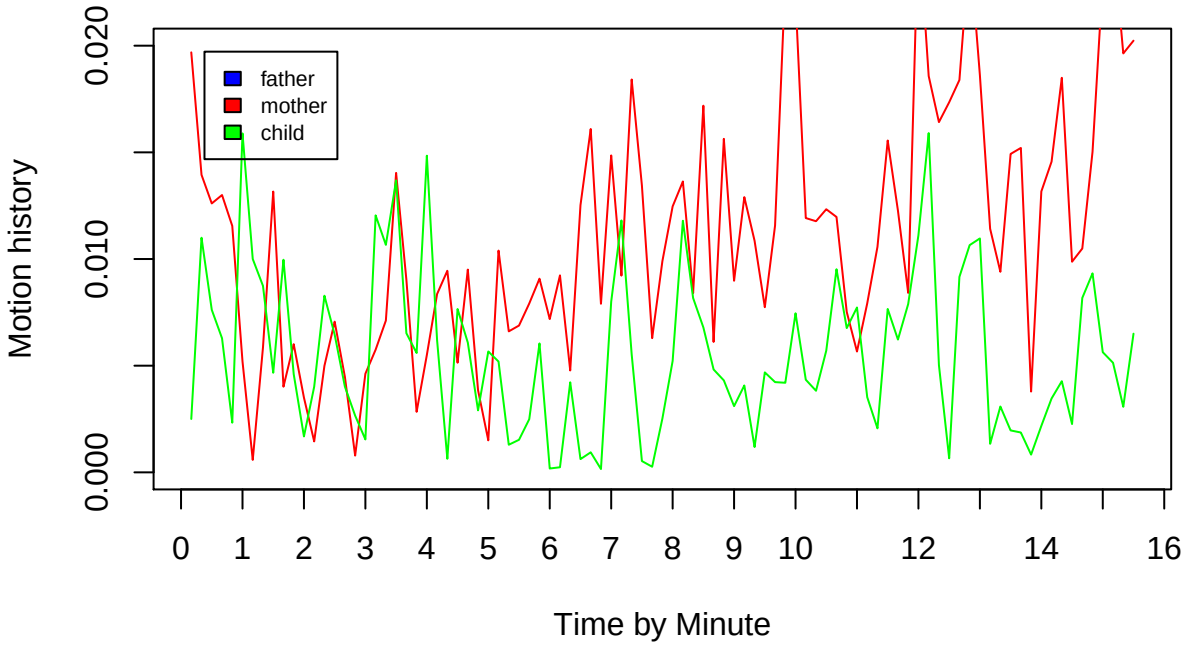

Mean motion history (non overlaping 10 sec intervals)
on MOSA065 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 11 13 15 17
Time by Minute

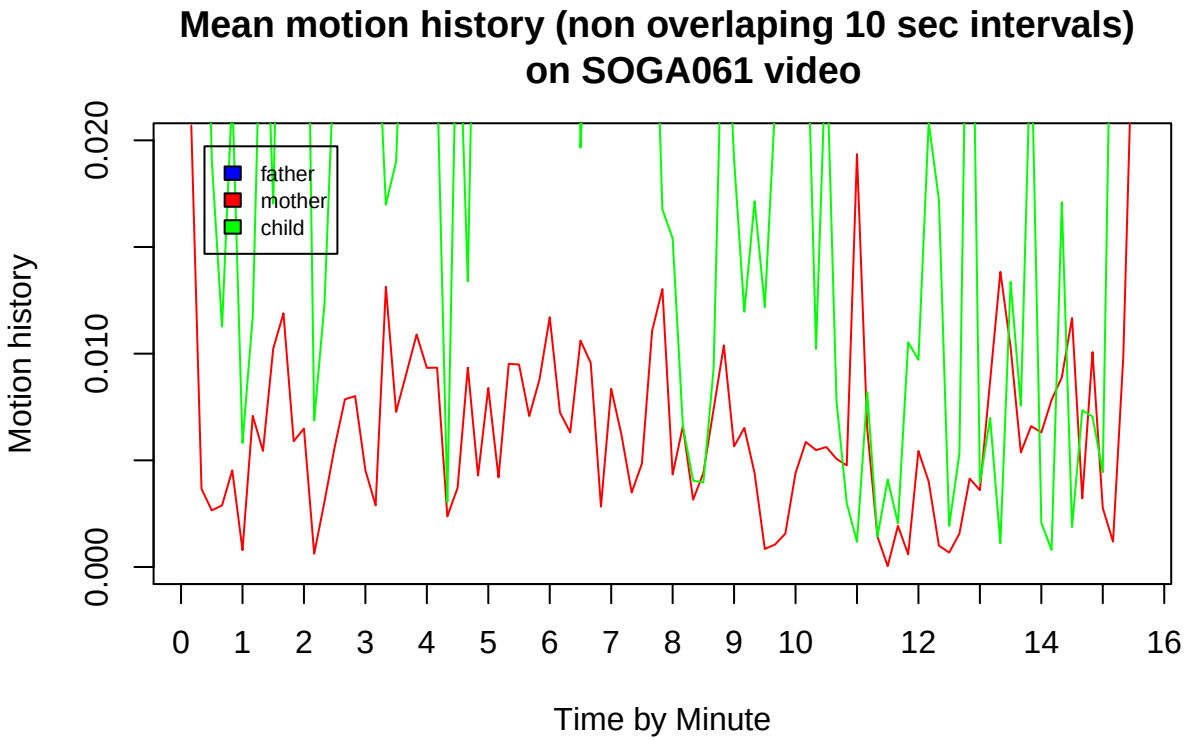

Mean motion history (non overlaping 10 sec intervals)
on RAEM049 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14
Time by Minute

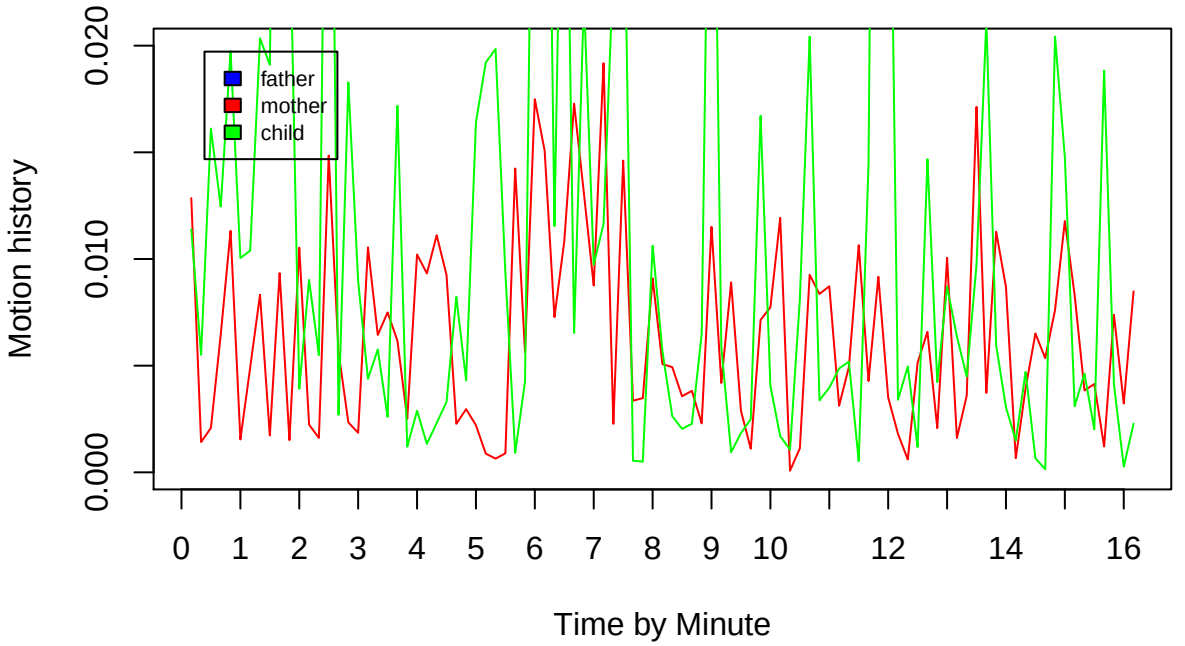

Mean motion history (non overlaping 10 sec intervals)
on RAKA008 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute

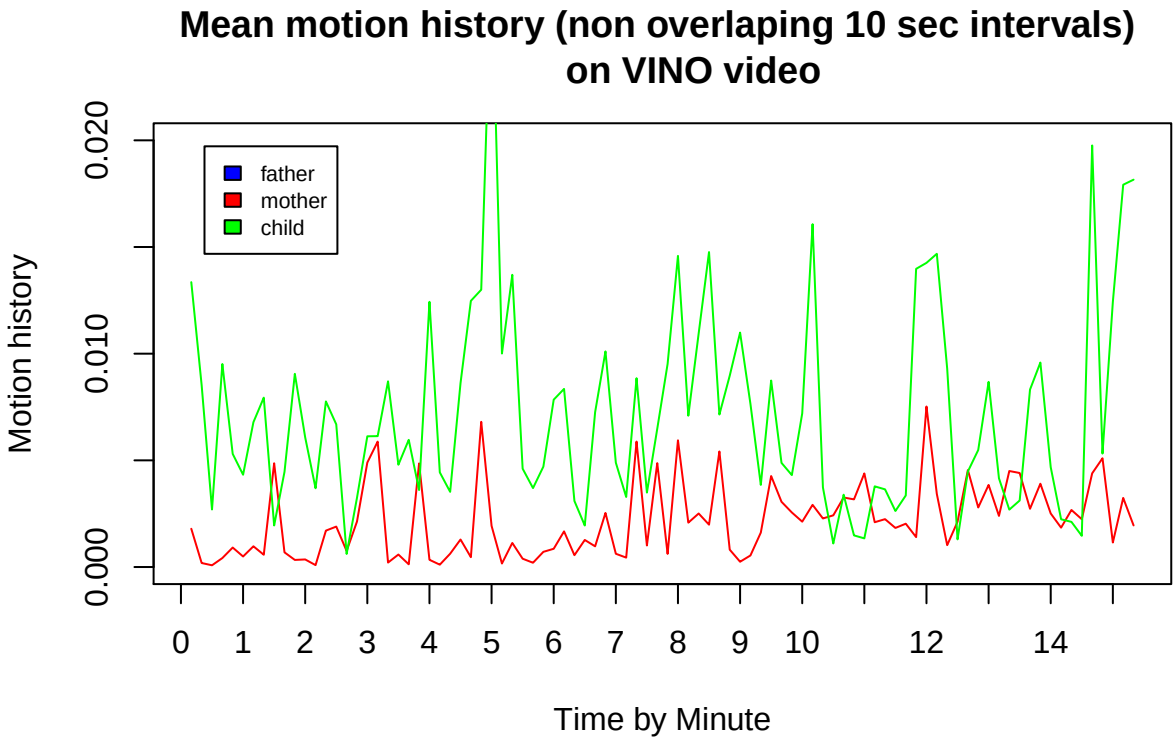

Mean motion history (non overlaping 10 sec intervals)
on RIEM0 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14
Time by Minute


Mean motion history (non overlaping 10 sec intervals)
on SEEM035 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute


Mean motion history (non overlaping 10 sec intervals)
on SHAN042 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute


Mean motion history (non overlaping 10 sec intervals)
on SOGA061 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute


Mean motion history (non overlaping 10 sec intervals)
on TIUG032 video
Motion history
0.000
0.010
0.020
father
mother
child
0 1 2 3 4 5 6 7 8 9 10 12 14 16
Time by Minute



Export no log data in text files

```
videoIndex <- 1
# videoName is the name of current video
for (videoName in unique(data$family)){
# Compute slinding interval for each participant

      print(paste("Computing slidedFather", videoName))
      slidedFather <- SlidingInterval("father", videoIndex, 5, data)
#      print(head(data[which(data$family==videoName),]$father))
#      print(tail(data[which(data$family==videoName),]$father))
#      print(table(is.na(slidedFather)))
#      print(table(is.na(data[which(data$family==videoName),]$father)))

      print(paste("Computing slidedMother", videoName))
      slidedMother <- SlidingInterval("mother", videoIndex, 5, data)
#      print(head(data[which(data$family==videoName),]$mother))
#      print(tail(data[which(data$family==videoName),]$mother))
#      print(table(is.na(data[which(data$family==videoName),]$mother)))
#      print(table(is.na(slidedMother)))

      print(paste("Computing slidedChild", videoName))
      slidedChild <- SlidingInterval("child", videoIndex, 5, data)
#      print(head(data[which(data$family==videoName),]$child))
#      print(tail(data[which(data$family==videoName),]$child))
#      print(table(is.na(data[which(data$family==videoName),]$child)))
#      print(table(is.na(slidedChild)))
```

```
      slidedVideo <- data.frame(
        slidedFather, slidedMother, slidedChild,
        "video"=rep(families[videoIndex], length(slidedFather)),
        frame_index = 1:length(slidedFather))


#      slidedVideo.nas <- apply(slidedVideo, 1, function(x){all(is.na(x))})
#      slidedVideo <- slidedVideo[!slidedVideo.nas,]
#      indexes <-  data.frame ("video"=rep(families[videoIndex], length(slidedFather)),
#        frame_index = 1:length(slidedFather))
            write.csv(slidedVideo, paste("../Data/CSV/filtered/noLog/",videoName, ".slideddata.csv", sep
      videoIndex <-(videoIndex+1)
      }
```

Export log data in text files

```
videoIndex <- 1
# videoName is the name of current video
for (videoName in unique(data$family)){
# Compute slinding interval for each participant
      print(paste("Computing slidedFather", videoName))
      slidedFather <- SlidingInterval("logFather", videoIndex, 5, data)
      print(paste("Computing slidedMother", videoName))
      slidedMother <- SlidingInterval("logMother", videoIndex, 5, data)
      print(paste("Computing slidedChild", videoName))
      slidedChild <- SlidingInterval("logChild", videoIndex, 5, data)

      slidedVideo <- data.frame(
        slidedFather, slidedMother, slidedChild,
        "video"=rep(families[videoIndex], length(slidedFather)),
        frame_index = 1:length(slidedFather))

            write.csv(slidedVideo, paste("../Data/CSV/filtered/log/",videoName, ".log.slideddata.csv", s
      videoIndex <-(videoIndex+1)
      }
```

SyncPy utilisation for creating synchrony dataframe

After extracting filtered motion motion history with mean on sliding interval (overlapping interval) of 5 frames

And after puting this data on a CSV file slideddata.csv

We import this data on python Script with panda module Call_S_Estimator.py

This script will compute the synchrony between each dyad of the interaction and of the whole group

It will return a csv file for each video SSIXXXX.csv with XXXX the name of the video (F1044C, F1044D1, etc) that we can import with R with

this following function

```
print("SSI Files Directory")
```

```
## [1] "SSI Files Directory"
```

```
SSIlogFilesList <- list.files("../Data/CSV/Synchrony/log/S_estimator", full.name=TRUE)
#SSIlogFilesList
```

```
print("SSI Files Directory")
```

```
## [1] "SSI Files Directory"
```

```
SSInoLogFilesList <- list.files("../Data/CSV/Synchrony/noLog/S_estimator", full.name=TRUE)
#SSInoLogFilesList
```

```
SSIlog <- data.frame(video="Name")
for (file in SSIlogFilesList){
    SSIalone <- read.csv(file)
 #   print(str(SSIalone))
    SSIlog <- rbind.fill(SSIlog, SSIalone)}
str(SSIlog)
```

```
## 'data.frame':    1201 obs. of  8 variables:
##  $ video       : chr  "Name" "34" "34" "34" ...
##  $ X           : int  NA 0 1 2 3 4 5 6 7 8 ...
##  $ Interval    : int  NA 1 2 3 4 5 6 7 8 9 ...
##  $ Time_min    : num  NA 0 0.5 1 1.5 2 2.5 3 3.5 4 ...
##  $ SSI_mo_ch   : num  NA 6.91e-03 5.31e-02 2.32e-05 5.74e-04 ...
##  $ SSI_fa_ch   : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ SSI_fa_mo   : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ SSI_fa_mo_ch: num  NA NA NA NA NA NA NA NA NA NA ...
```

```
SSIlog$video <- as.factor(SSIlog$video)
SSIlog <- SSIlog[-which(SSIlog$video=="Name"),]
```

```
SSInoLog <- data.frame(video="Name")
for (file in SSInoLogFilesList){
#    print(file)
    SSIalone <- read.csv(file)
  #  print(str(SSIalone))
```

```
    SSInoLog<- rbind.fill(SSInoLog, SSIalone)}
SSInoLog$video <-as.factor(SSInoLog$video)
SSInoLog <- SSInoLog[-which(SSInoLog$video=="Name"),]
#SSInoLog$action <- rep(NA, nrow(SSInoLog))
#SSInoLog[which(video == & Time_min==)]$action
```

Description of SSIlog data frame

```
str(SSIlog)
```

```
## 'data.frame':    1200 obs. of  8 variables:
##  $ video       : Factor w/ 40 levels "1106","1606",..: 4 4 4 4 4 4 4 4 4 4 ...
##  $ X           : int  0 1 2 3 4 5 6 7 8 9 ...
##  $ Interval    : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Time_min    : num  0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 ...
##  $ SSI_mo_ch   : num  6.91e-03 5.31e-02 2.32e-05 5.74e-04 1.88e-02 ...
##  $ SSI_fa_ch   : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ SSI_fa_mo   : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ SSI_fa_mo_ch: num  NA NA NA NA NA NA NA NA NA NA ...
```

```
#View(SSI)
```

Description of noLogSSI data frame

```
str(SSInoLog)
```

```
## 'data.frame':    1200 obs. of  8 variables:
##  $ video       : Factor w/ 40 levels "1106","1606",..: 4 4 4 4 4 4 4 4 4 4 ...
##  $ X           : int  0 1 2 3 4 5 6 7 8 9 ...
##  $ Interval    : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Time_min    : num  0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 ...
##  $ SSI_mo_ch   : num  0.018627 0.024666 0.04506 0.001527 0.000817 ...
##  $ SSI_fa_ch   : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ SSI_fa_mo   : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ SSI_fa_mo_ch: num  NA NA NA NA NA NA NA NA NA NA ...
```

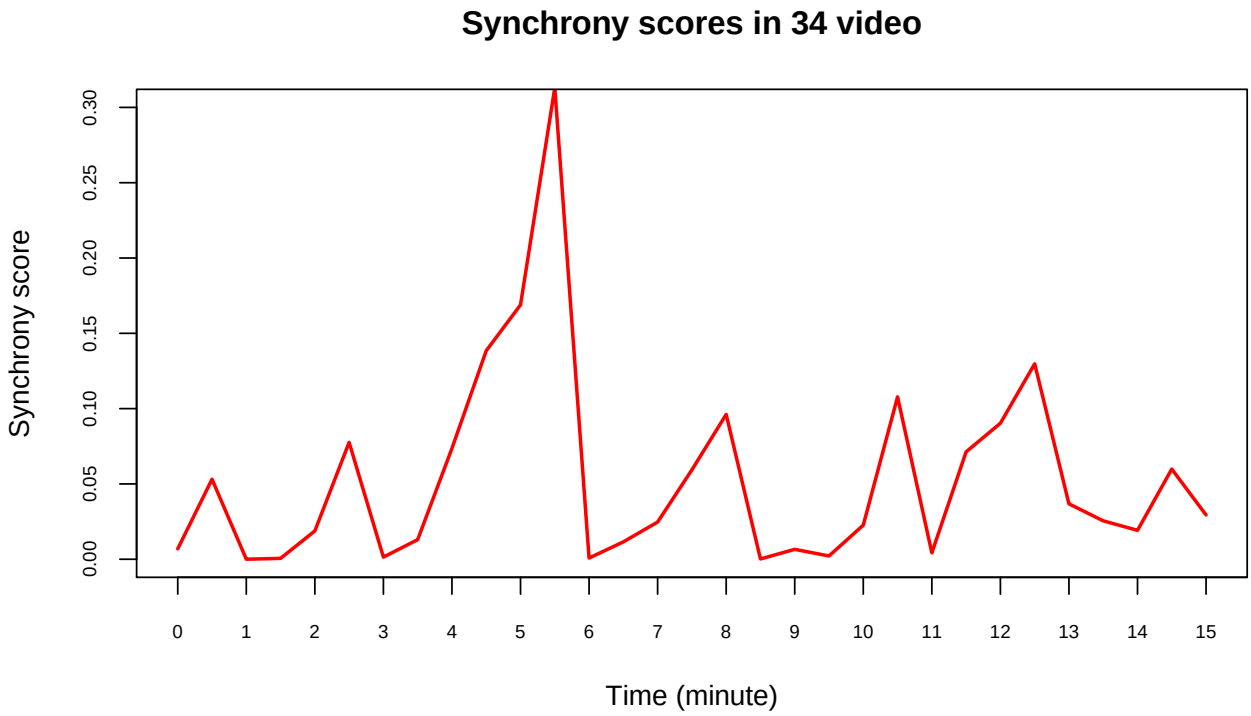
```
#View(SSInoLog)
```

Synchrony scores log for each dyad, triad and for the whole group

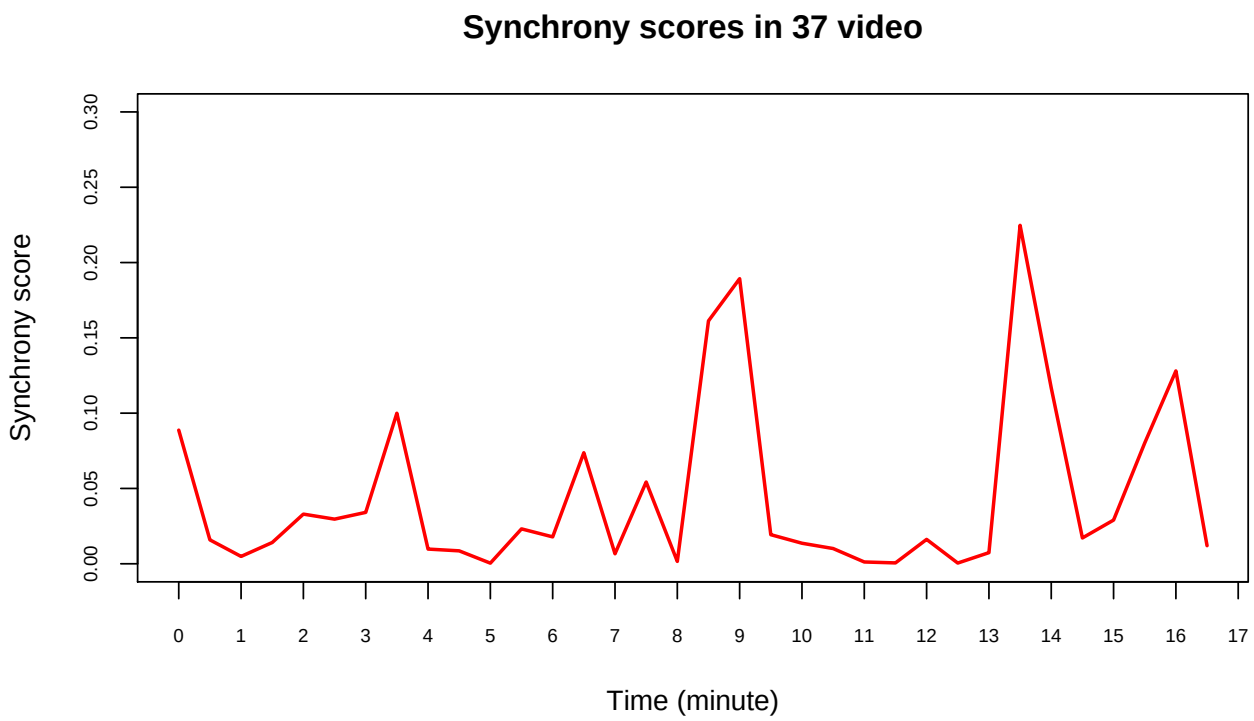
```
par(mar=c(4,4,4,3), mfrow=c(1,1))
for (i in unique(SSIlog$video)){
      if (all(!is.na(SSIlog[which(SSIlog$video==i),]$SSI_mo_ch)==TRUE)){
        print(SSIlog[which(SSIlog$video==i),]$Time_min)
        print(str(SSIlog[which(SSIlog$video==i),]$SSI_mo_ch))
            plot(SSIlog[which(SSIlog$video==i),]$Time_min, SSIlog[which(SSIlog$video==i),]$SSI_mo_ch,
                 ylim=c(0, 0.3), main=paste("Synchrony scores in", i, "video"), xlab = "Time (minute)",

      else if(all(!is.na(SSIlog[which(SSIlog$video==i),]$SSI_fa_ch)==TRUE)){
```

```
    plot(SSIlog[which(SSIlog$video==i),]$Time_min, SSIlog[which(SSIlog$video==i),]$SSI_fa_ch,
              ylim=c(0, 0.3), main=paste("Synchrony scores in", i, "video"), xlab = "Time (minute)", yla
  else{print("error")}}
```

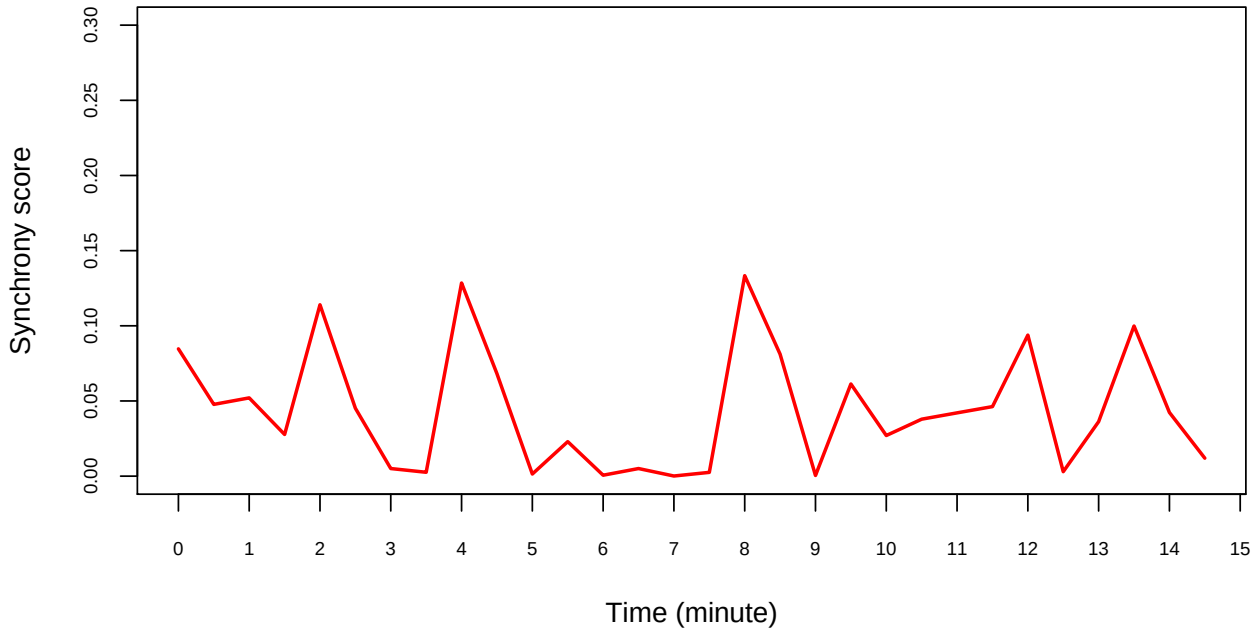
```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0
##  num [1:31] 6.91e-03 5.31e-02 2.32e-05 5.74e-04 1.88e-02 ...
## NULL
```

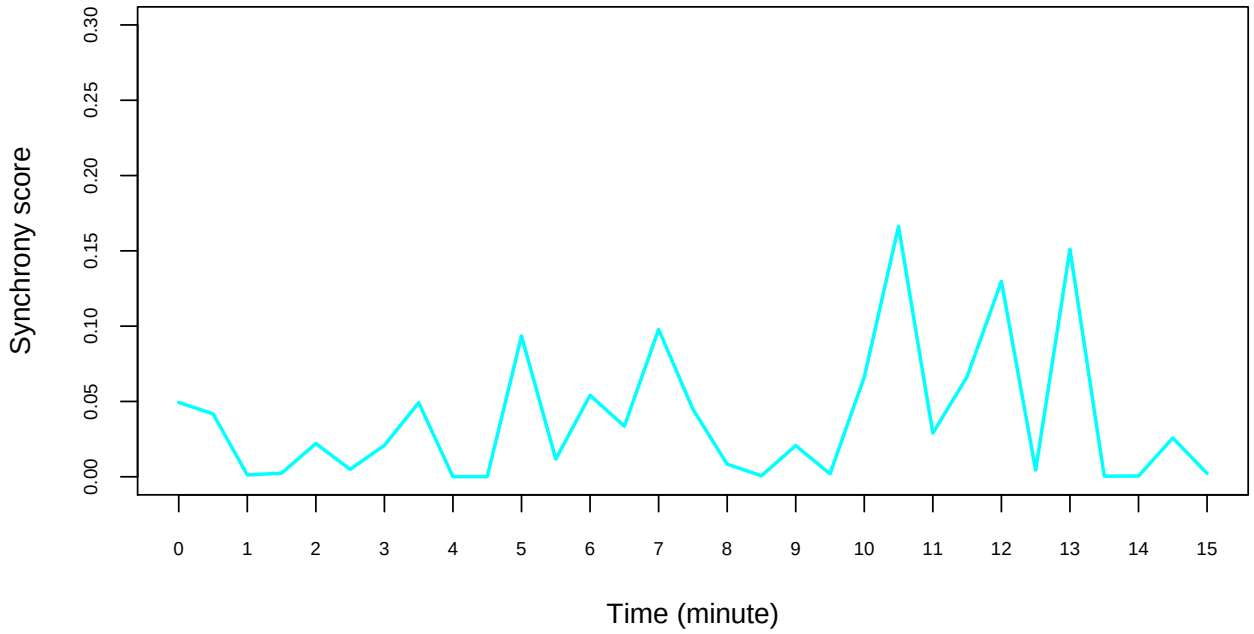



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0 15.5 16.0 16.5
##  num [1:34] 0.08872 0.01586 0.00486 0.01412 0.03297 ...
## NULL
```

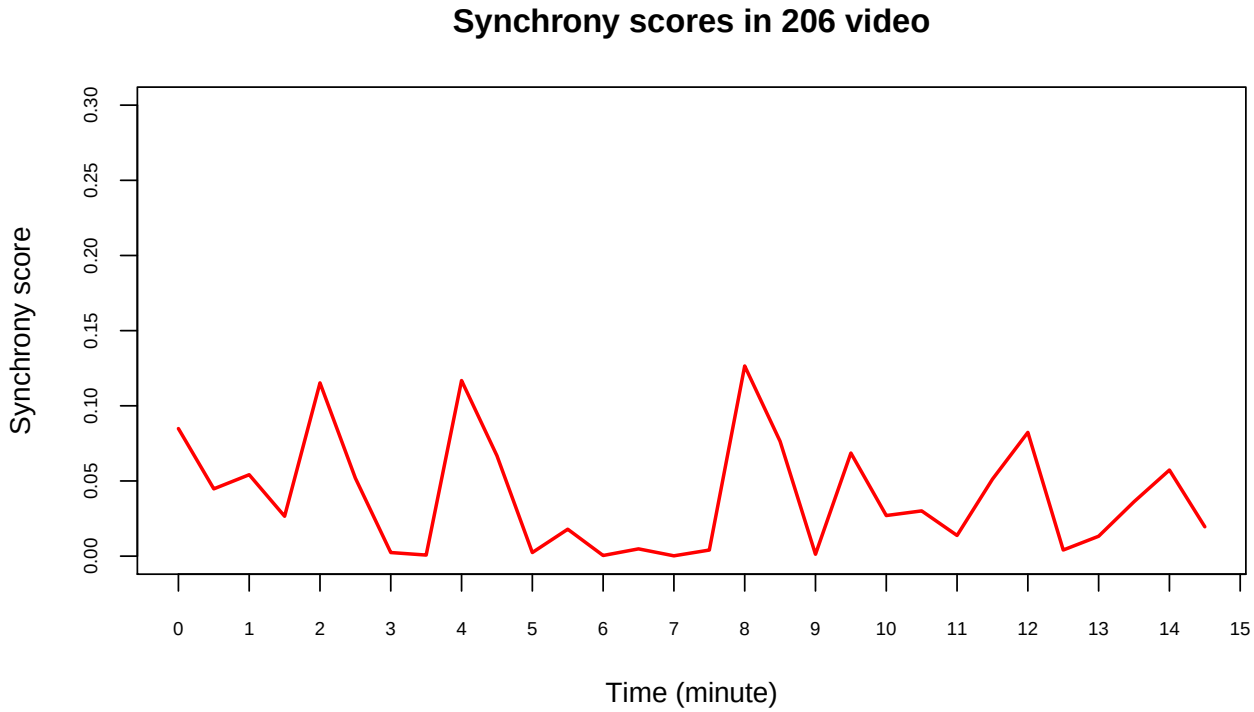



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5
##  num [1:30] 0.0847 0.0477 0.0521 0.0277 0.114 ...
## NULL
```



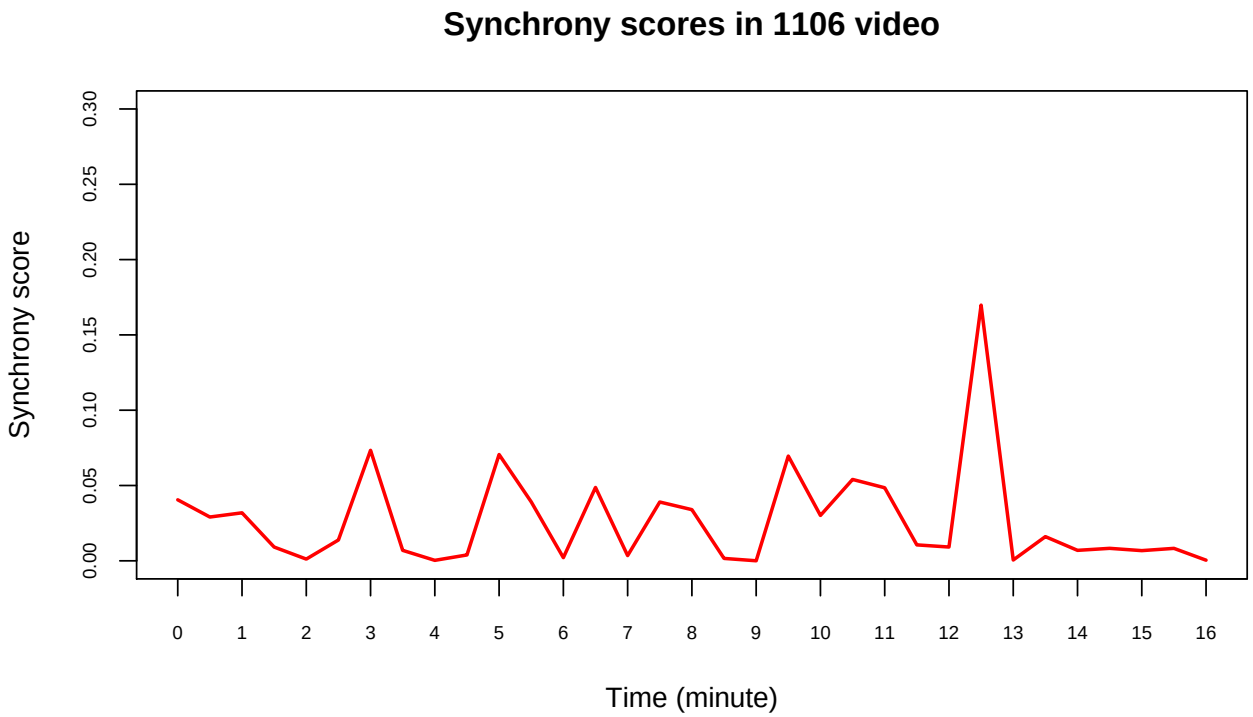


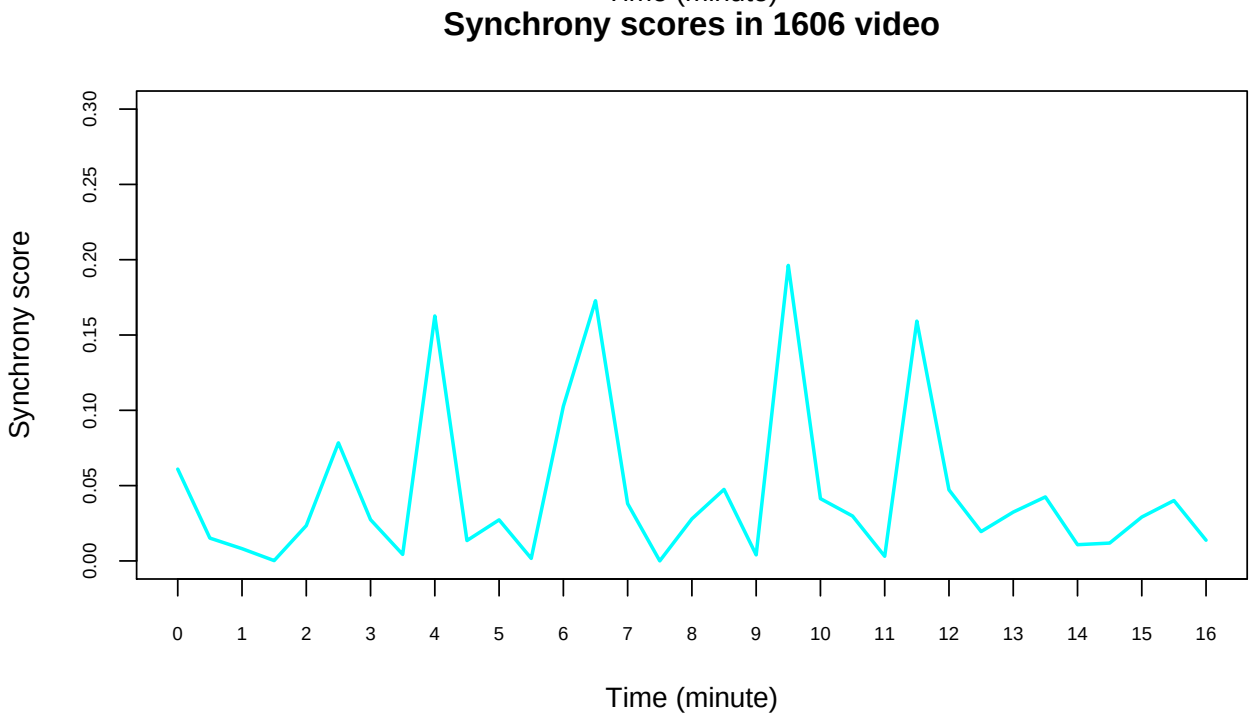

```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5
##  num [1:30] 0.0849 0.0448 0.0542 0.0265 0.1153 ...
## NULL
```

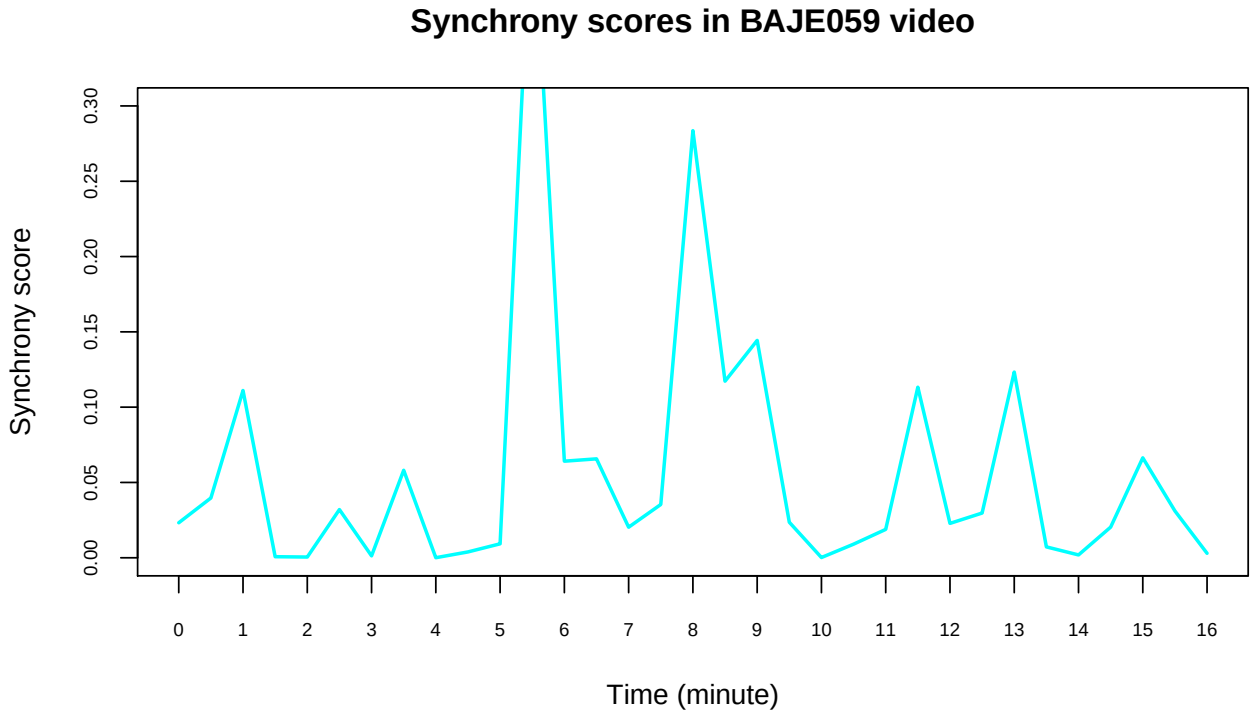



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0 15.5 16.0
```

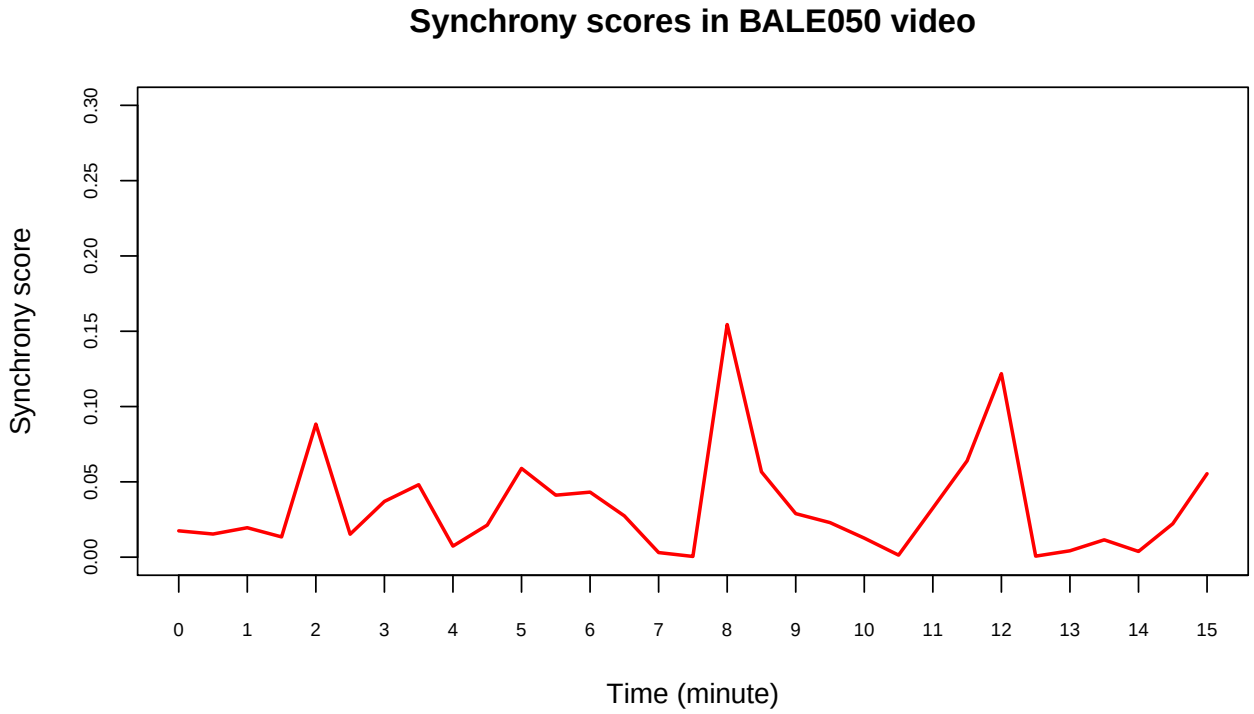
```
##  num [1:33] 0.04051 0.02908 0.03187 0.00913 0.00115 ...
## NULL
```





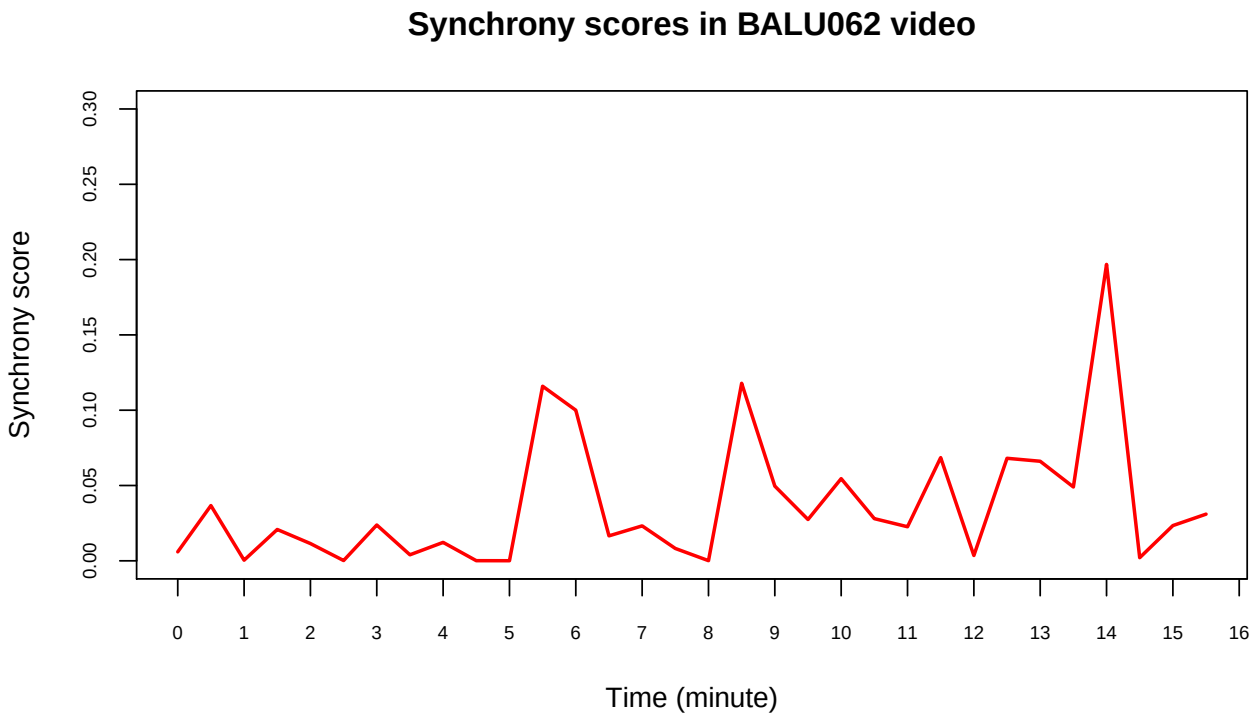



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0
##  num [1:31] 0.0175 0.0154 0.0195 0.0134 0.0883 ...
## NULL
```

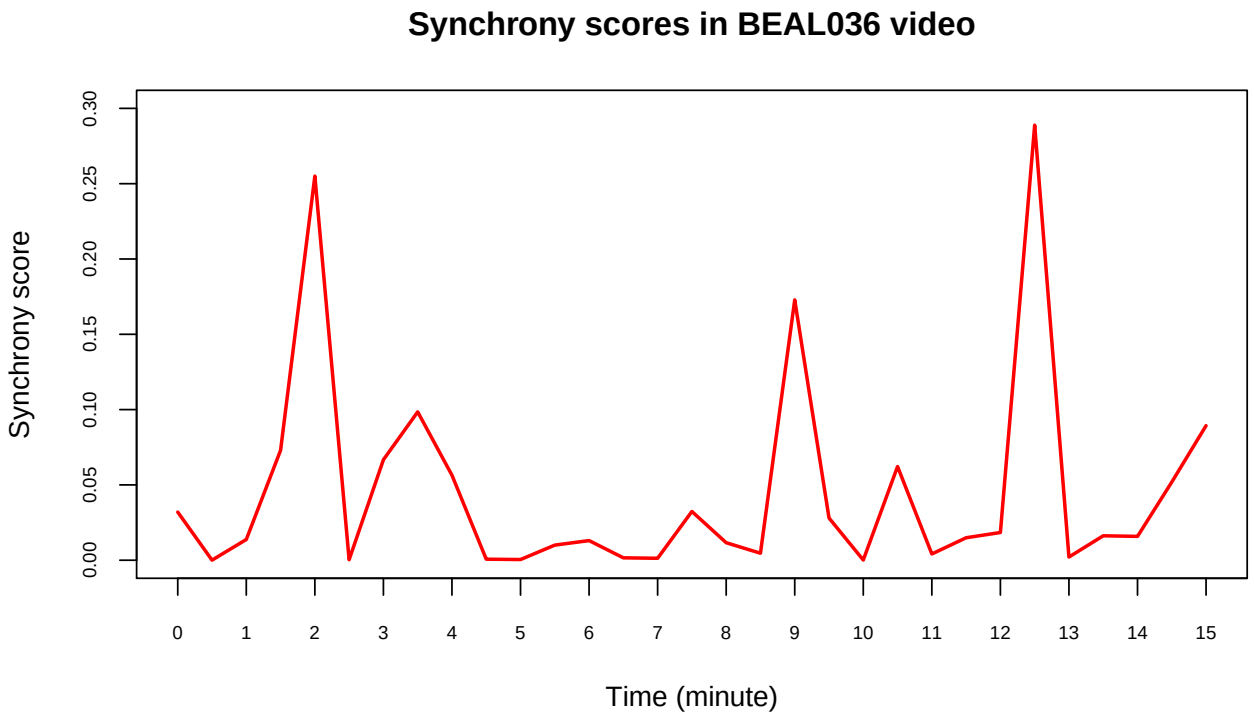



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0 15.5
```

```
##  num [1:32] 0.005957 0.036683 0.000418 0.020801 0.011463 ...
## NULL
```

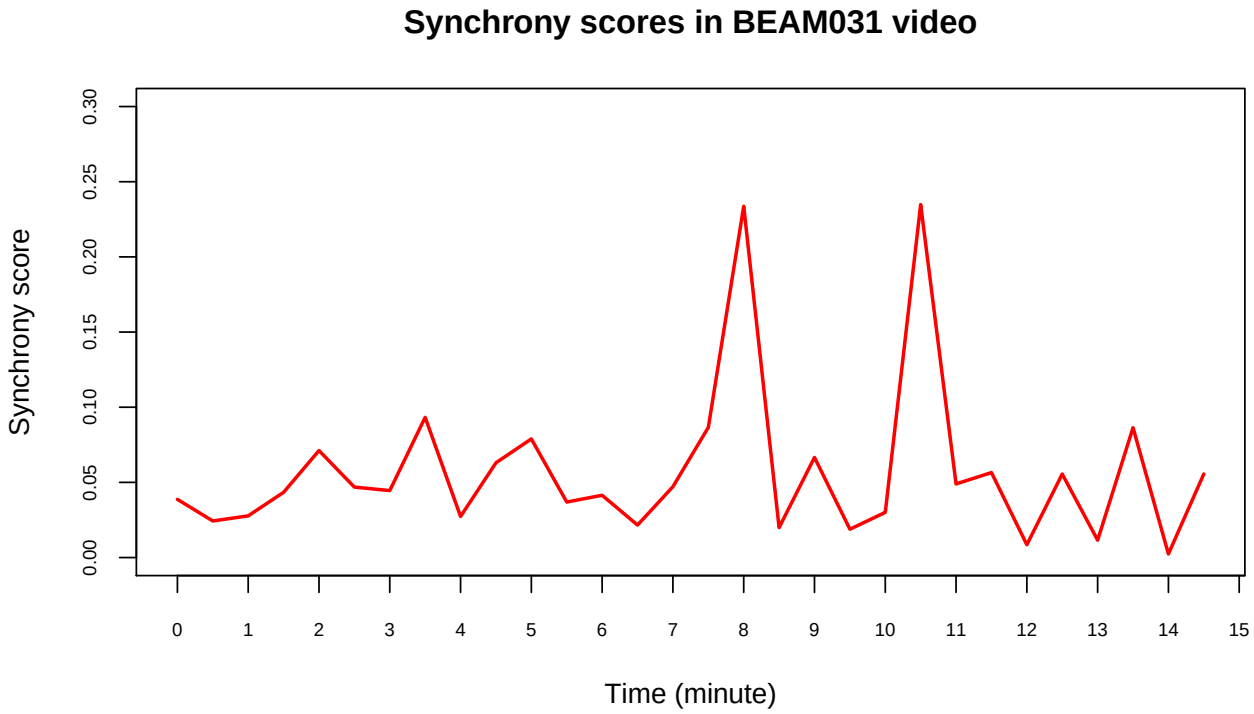



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0
##  num [1:31] 3.20e-02 4.75e-05 1.38e-02 7.30e-02 2.55e-01 ...
## NULL
```

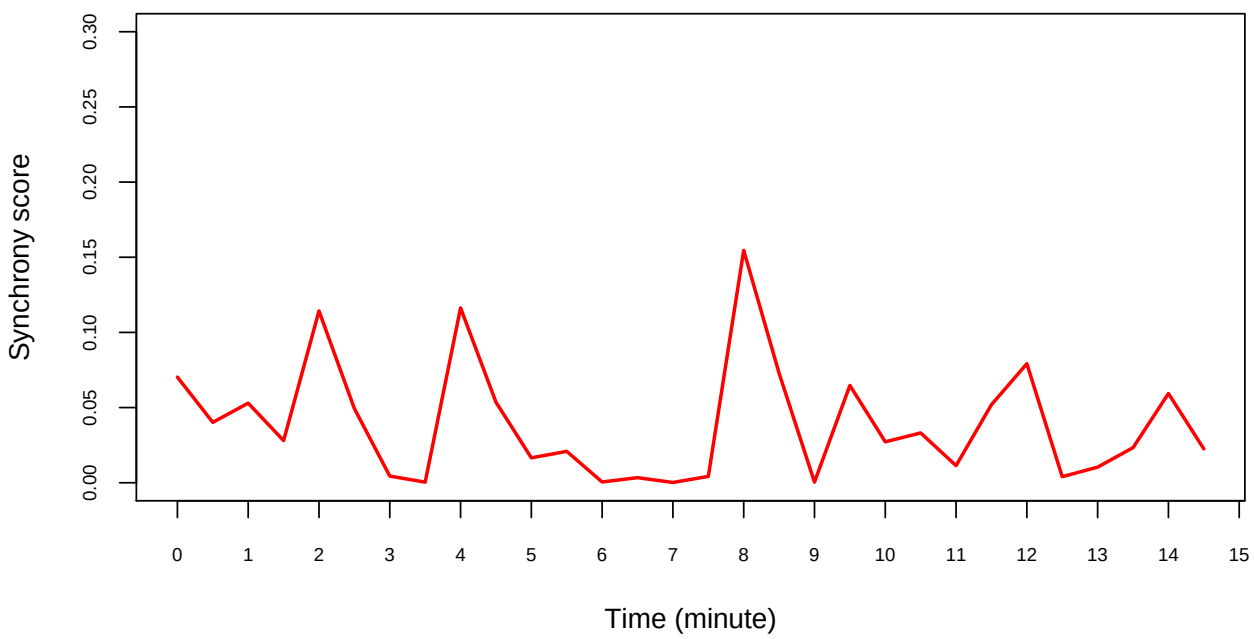



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
```

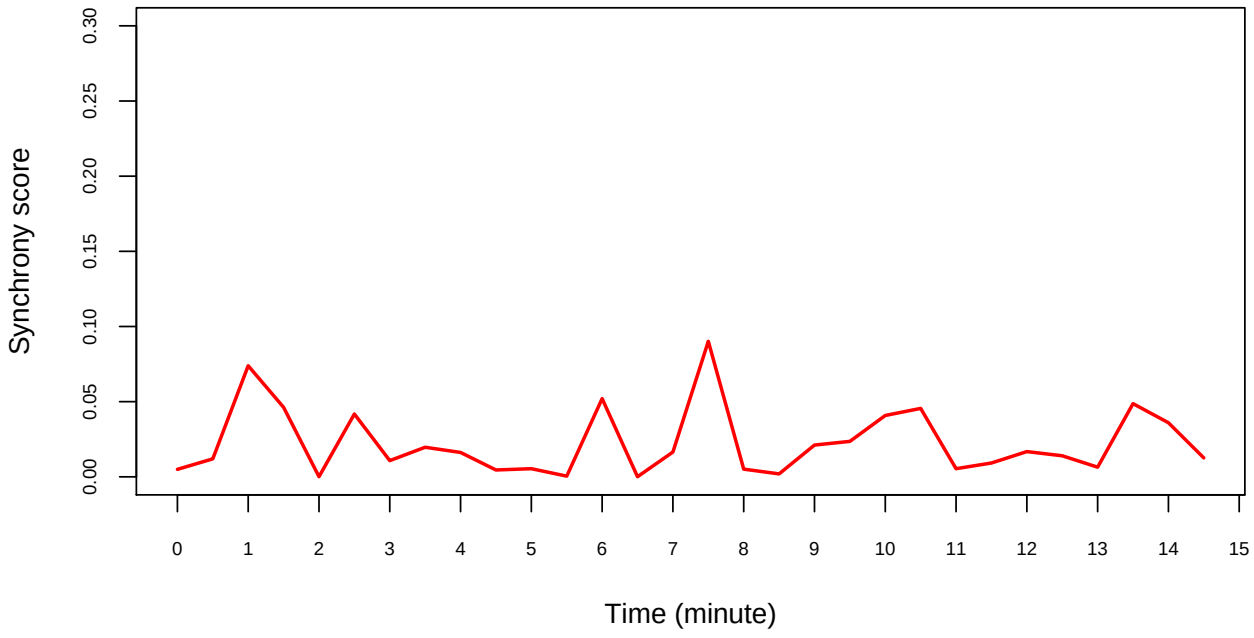
```
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5
##  num [1:30] 0.0387 0.0243 0.0277 0.0434 0.0712 ...
## NULL
```




```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5
##  num [1:30] 0.0703 0.0402 0.053 0.028 0.1143 ...
## NULL
```

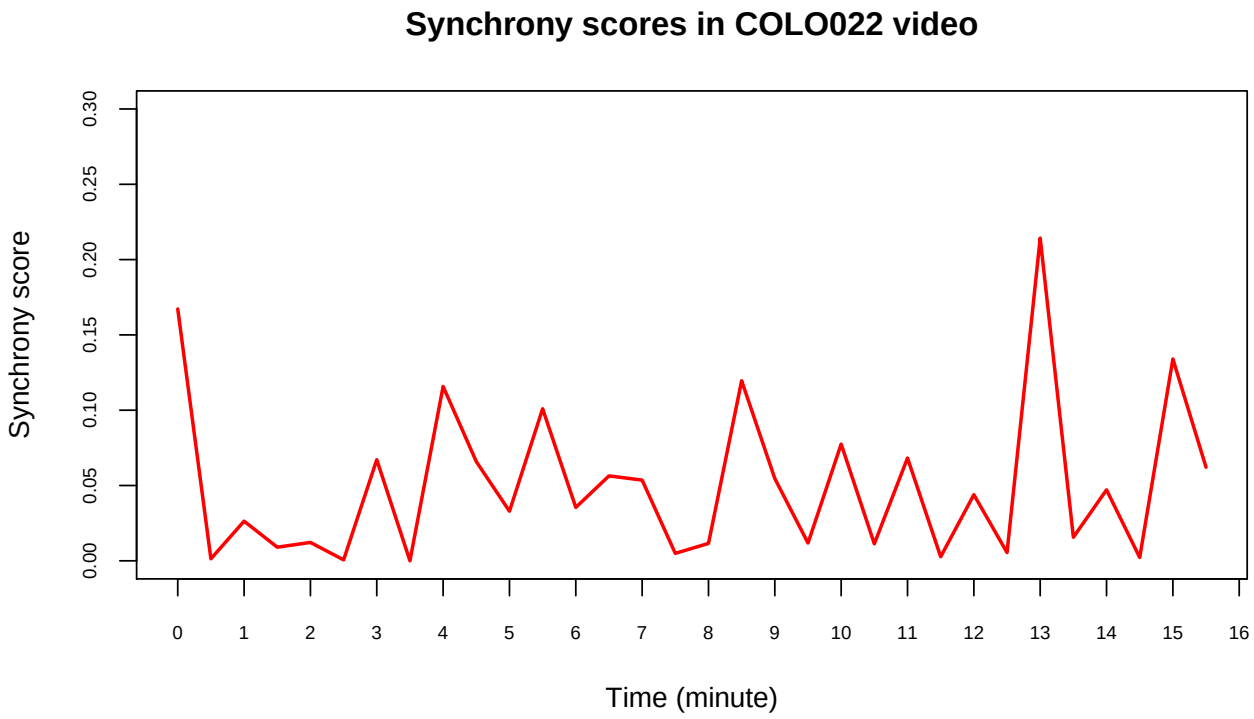



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5
##  num [1:30] 0.004976 0.011956 0.073889 0.046218 0.000101 ...
## NULL
```

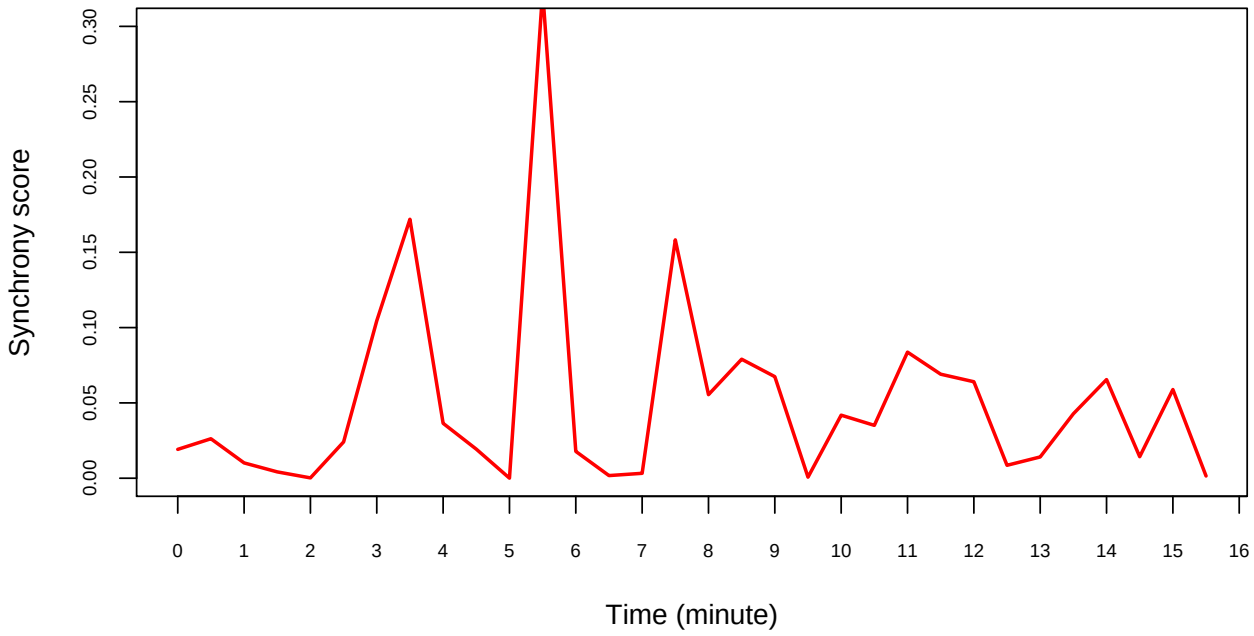



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0 15.5
```

```
##  num [1:32] 0.16722 0.00133 0.02633 0.00904 0.01221 ...
## NULL
```

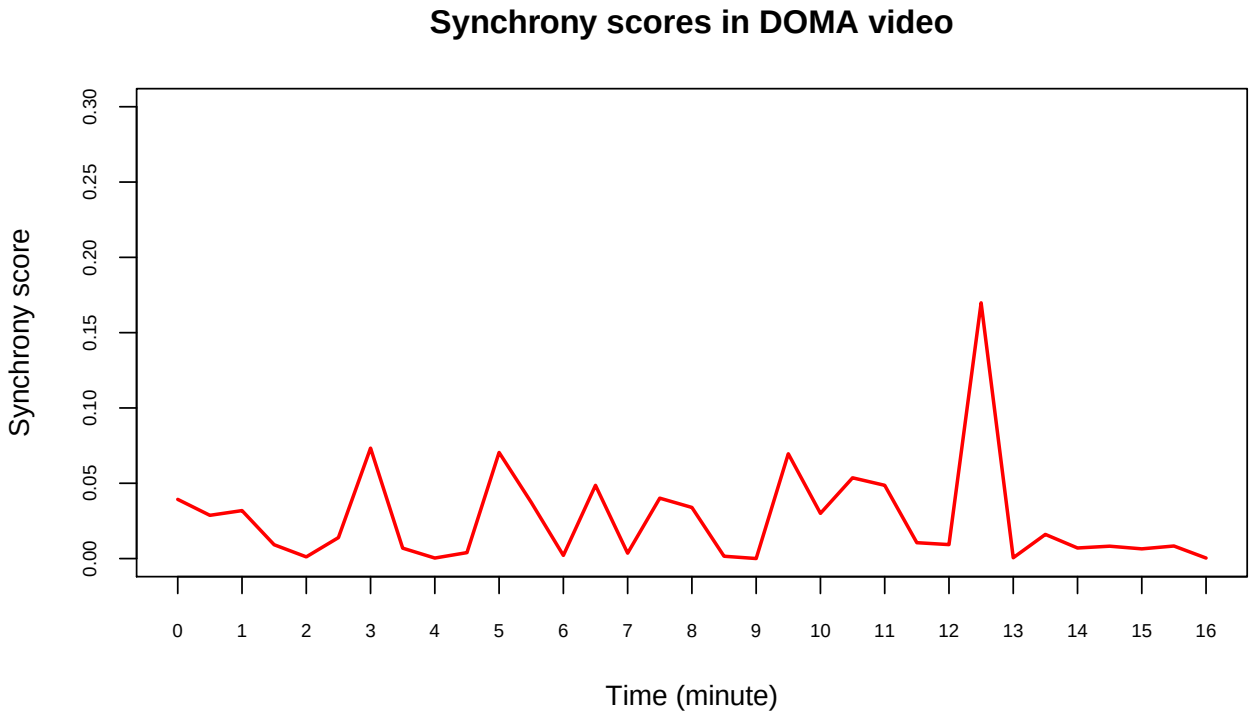



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0 15.5
##  num [1:32] 0.01914 0.026202 0.010165 0.004201 0.000208 ...
## NULL
```

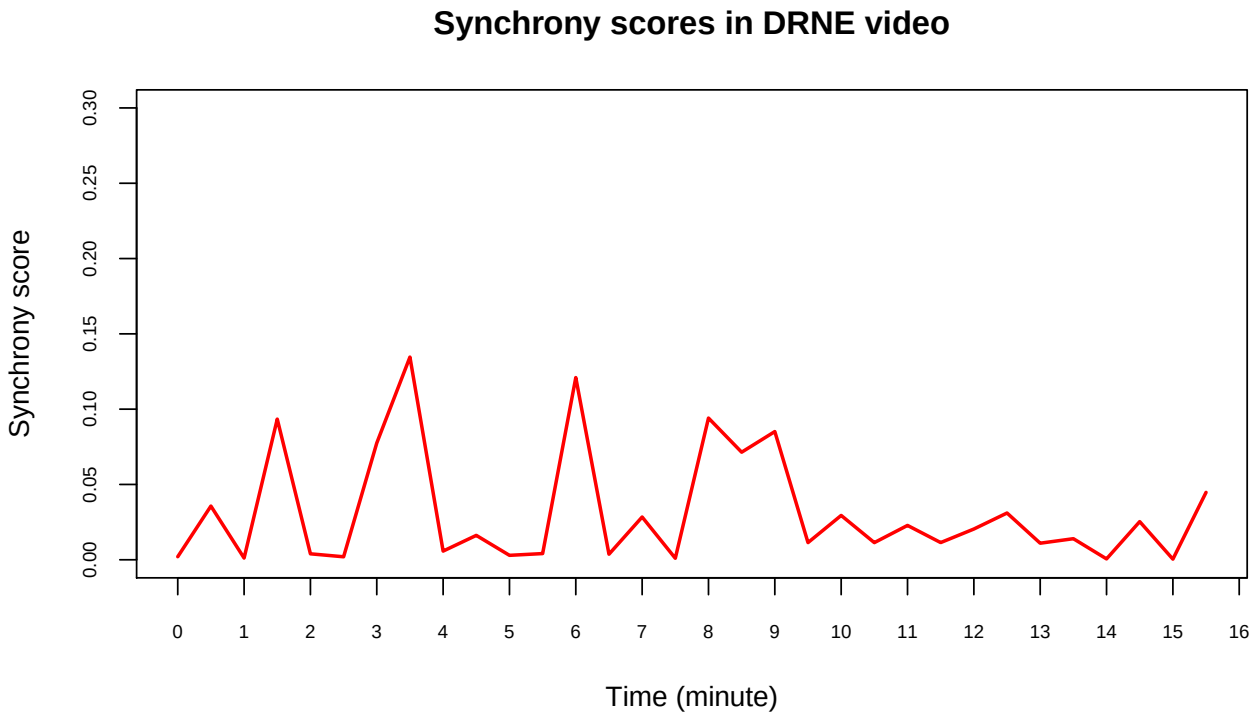


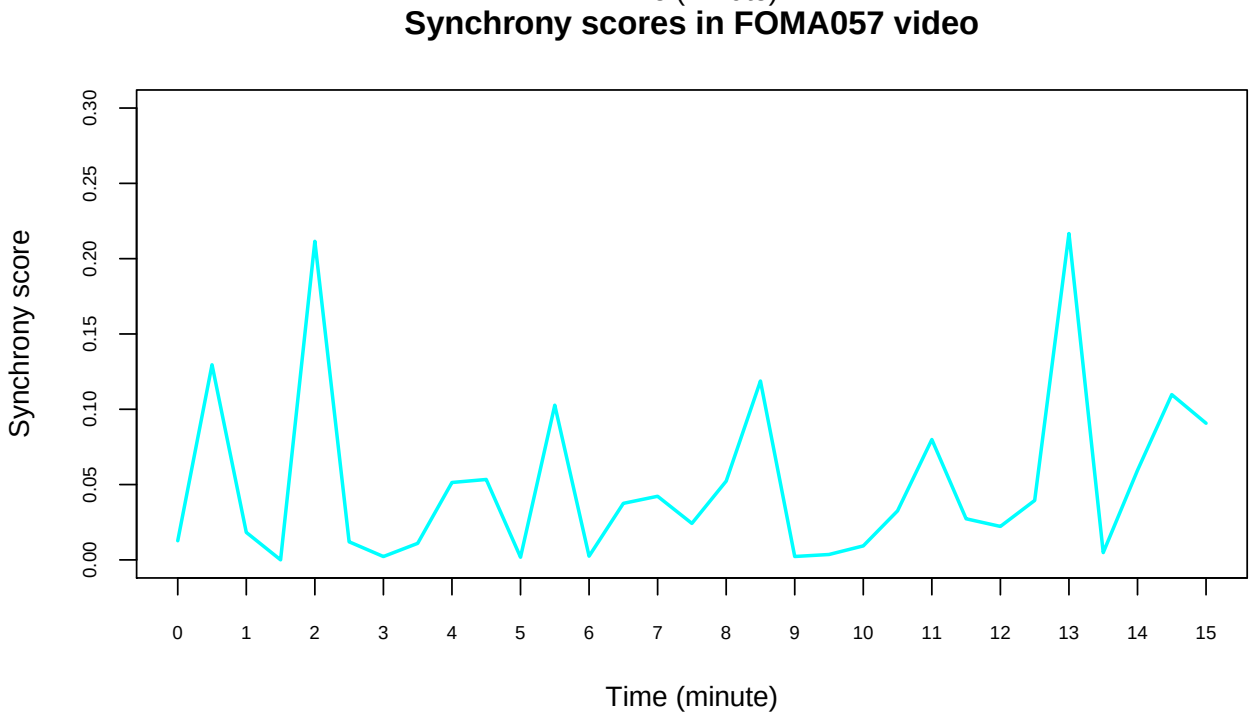

```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
```

```
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0 15.5 16.0
##  num [1:33] 0.0393 0.02872 0.03185 0.00918 0.00113 ...
## NULL
```

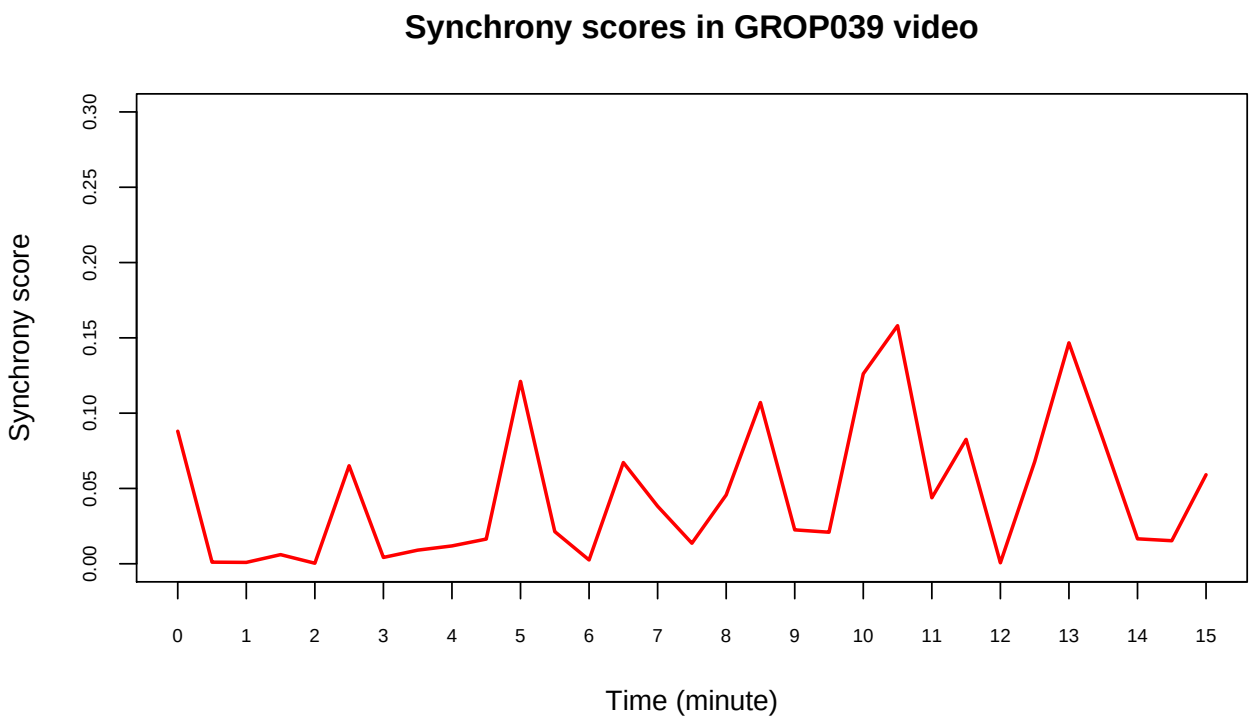



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0 15.5
##  num [1:32] 0.00203 0.03566 0.00112 0.09339 0.00396 ...
## NULL
```

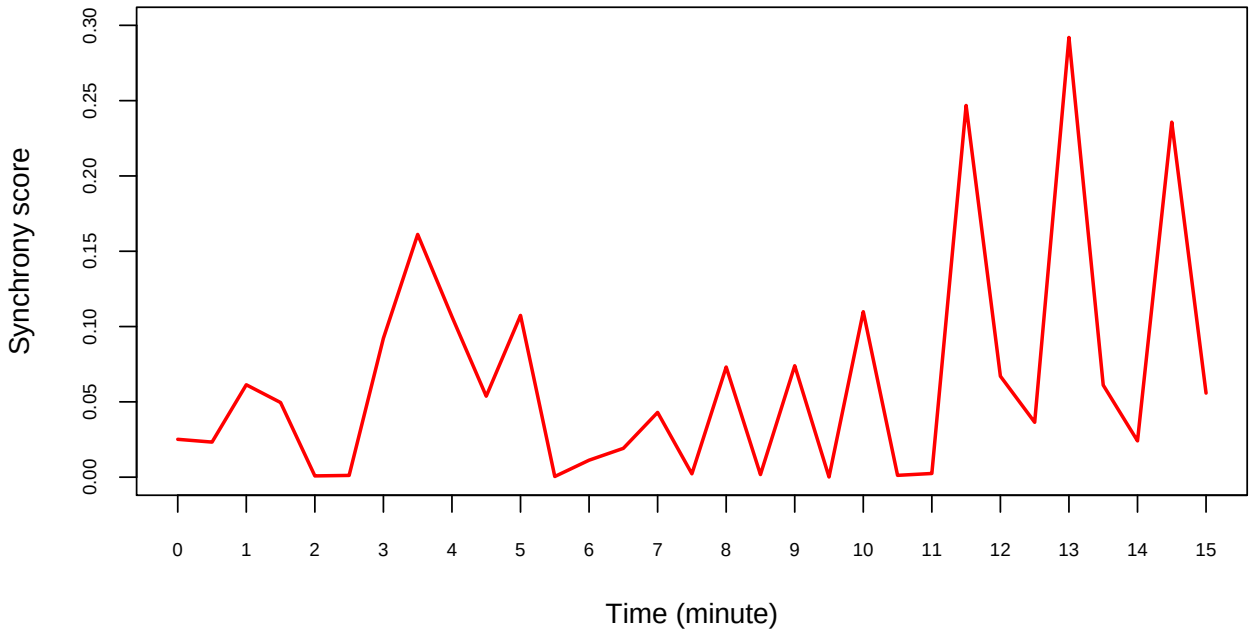





```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0
##  num [1:31] 0.088007 0.001068 0.000906 0.00606 0.000358 ...
## NULL
```

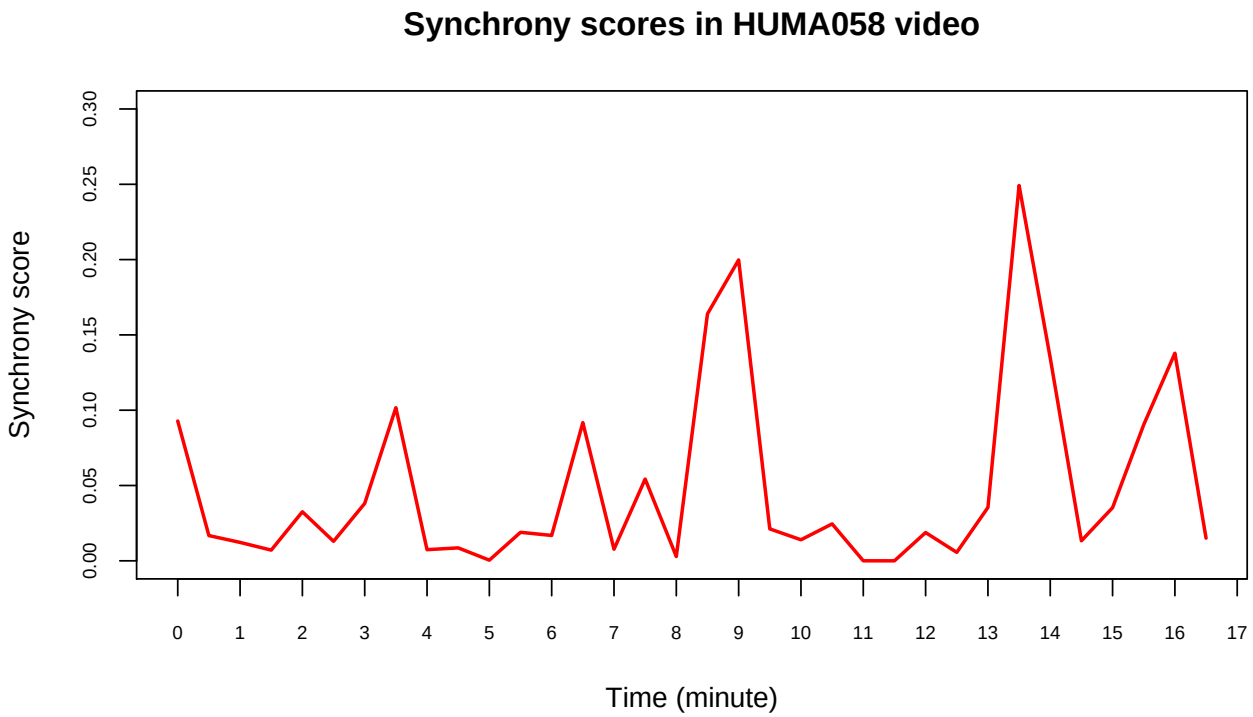



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0
##  num [1:31] 0.025127 0.023286 0.061355 0.049564 0.000826 ...
## NULL
```

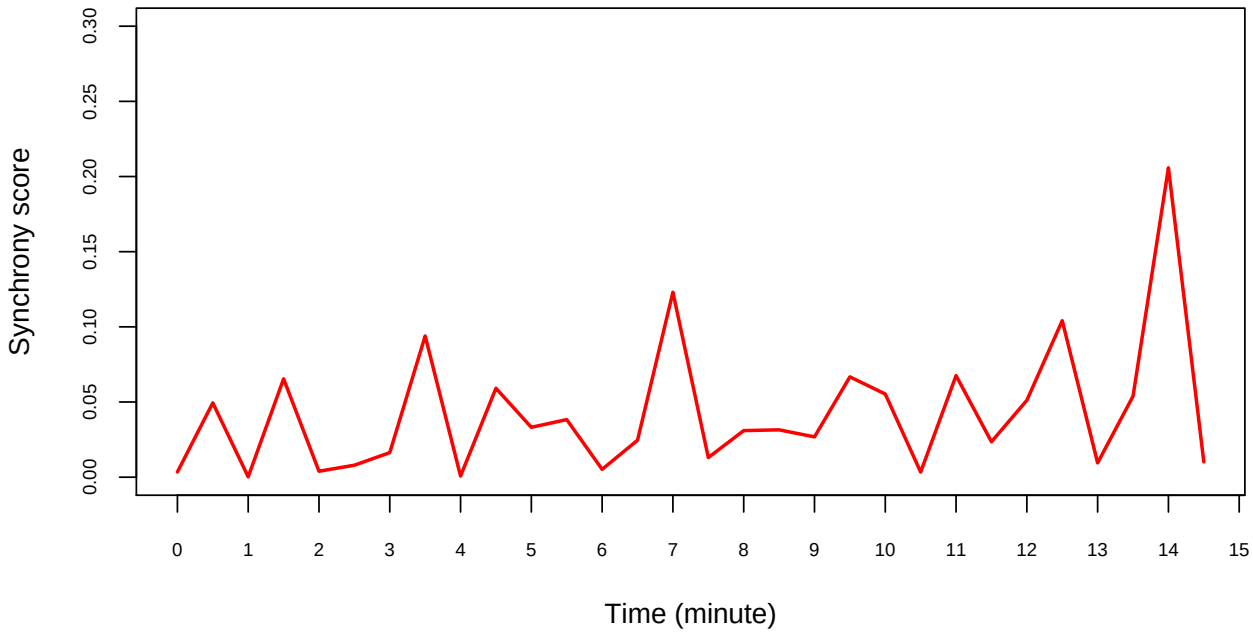



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0 15.5 16.0 16.5
```

```
##  num [1:34] 0.0928 0.0168 0.0122 0.0071 0.0326 ...
## NULL
```

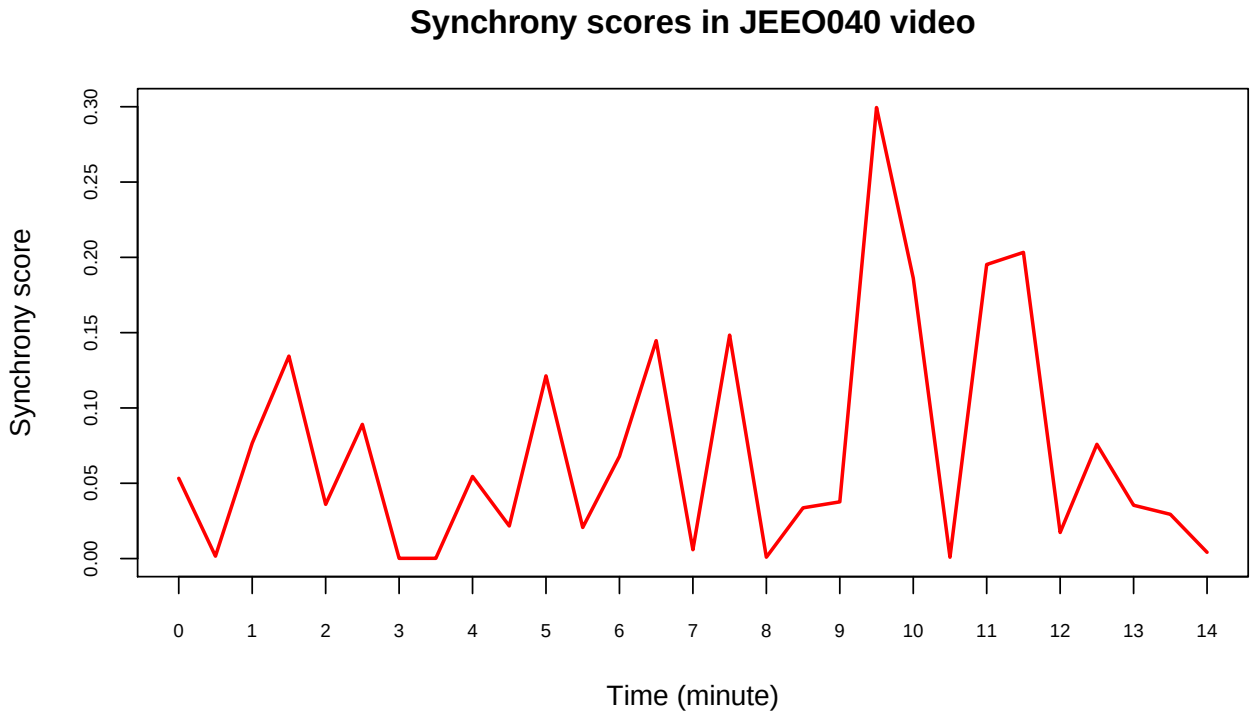



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5
##  num [1:30] 0.003451 0.04943 0.000226 0.06541 0.003947 ...
## NULL
```

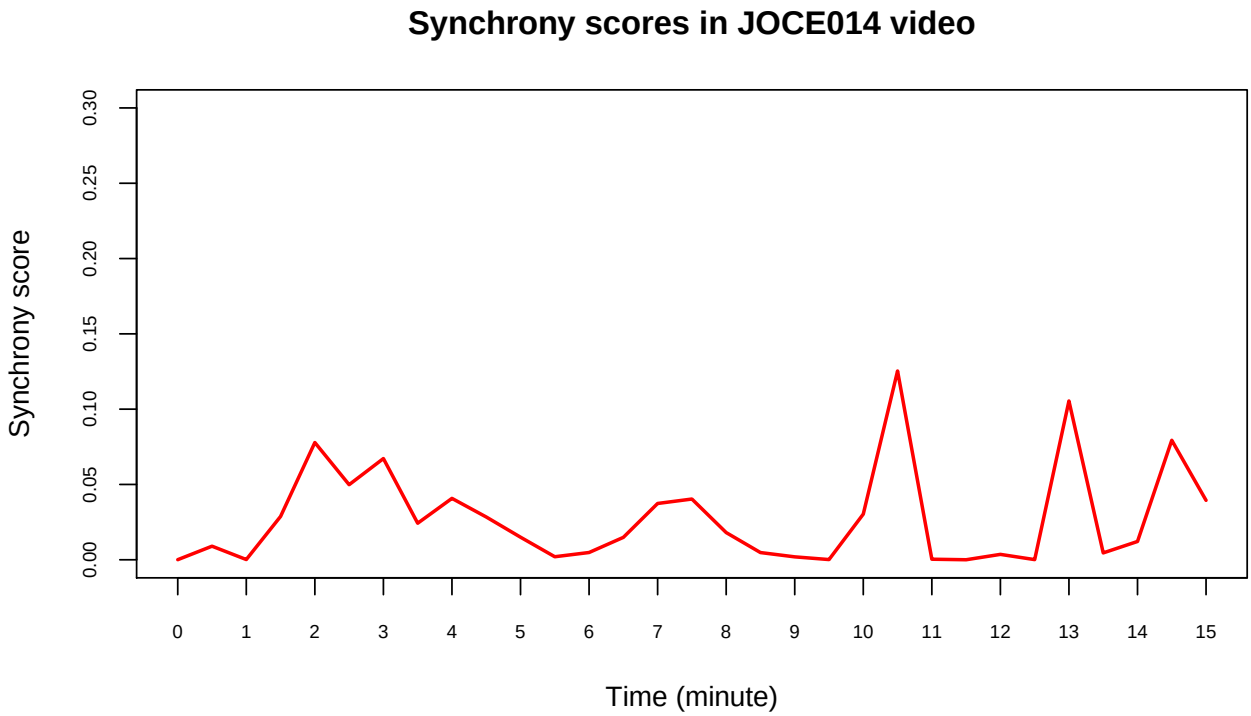


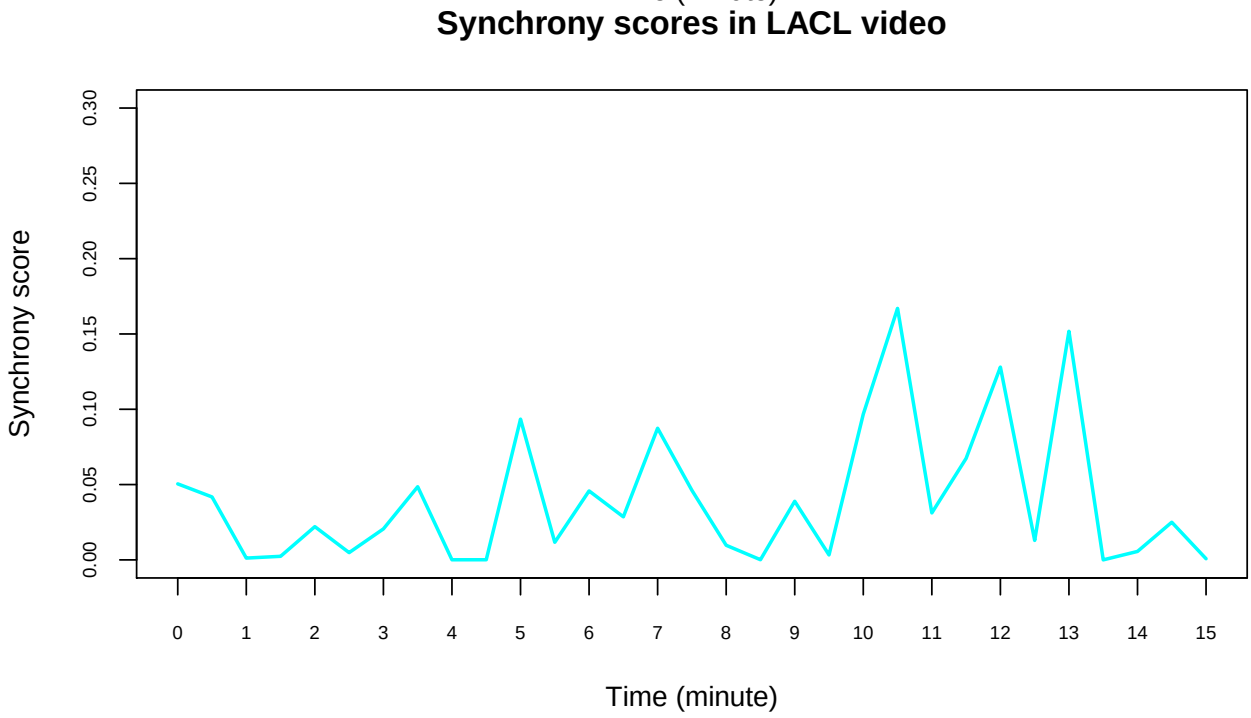

```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
```

```
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0
##  num [1:29] 0.0533 0.0016 0.0766 0.1344 0.036 ...
## NULL
```

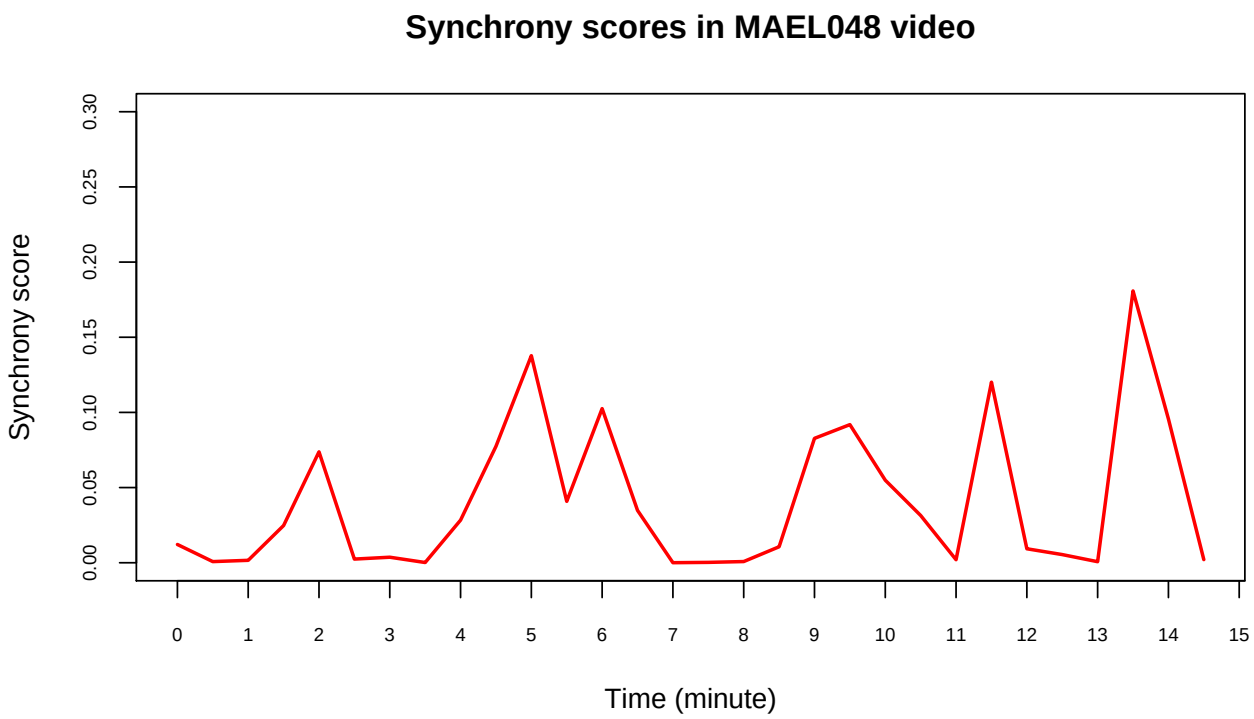



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0
##  num [1:31] 6.03e-05 9.01e-03 1.58e-04 2.88e-02 7.79e-02 ...
## NULL
```

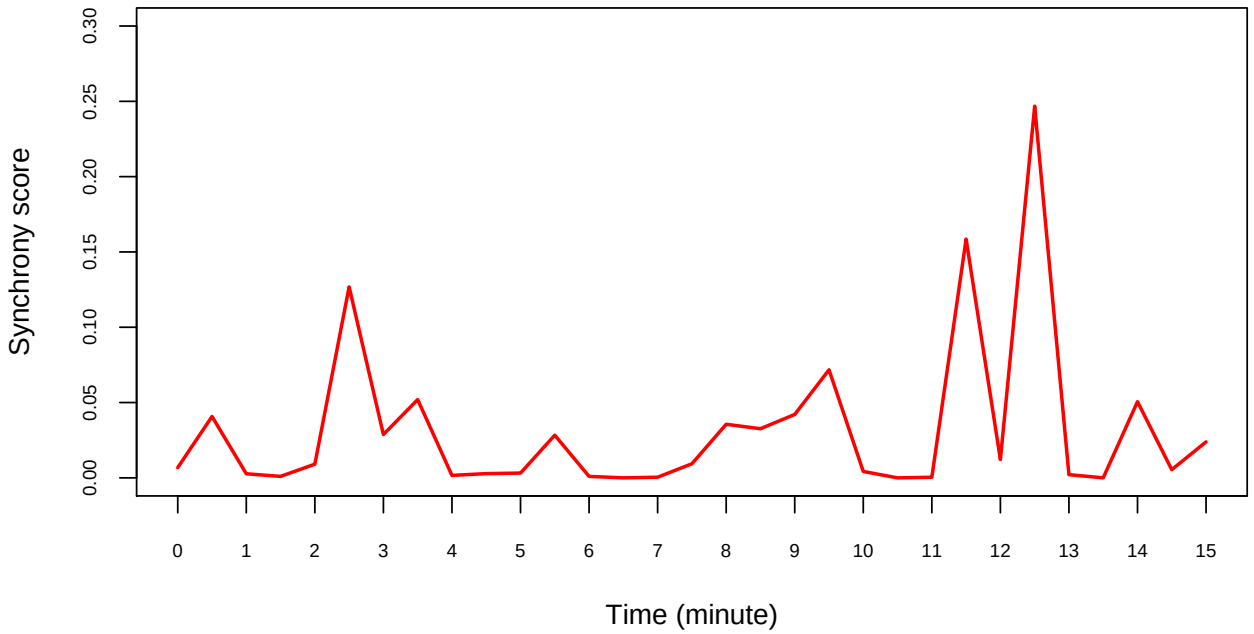




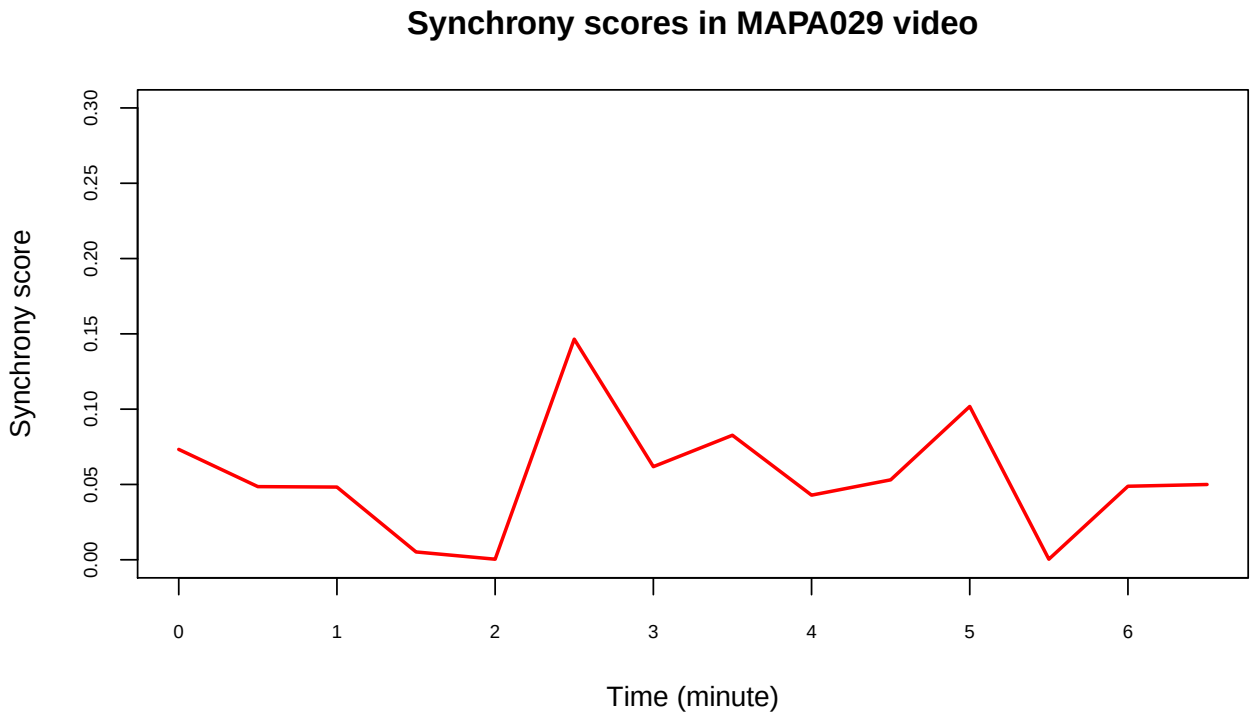

```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5
##  num [1:30] 0.01214 0.00075 0.00162 0.02476 0.07378 ...
## NULL
```



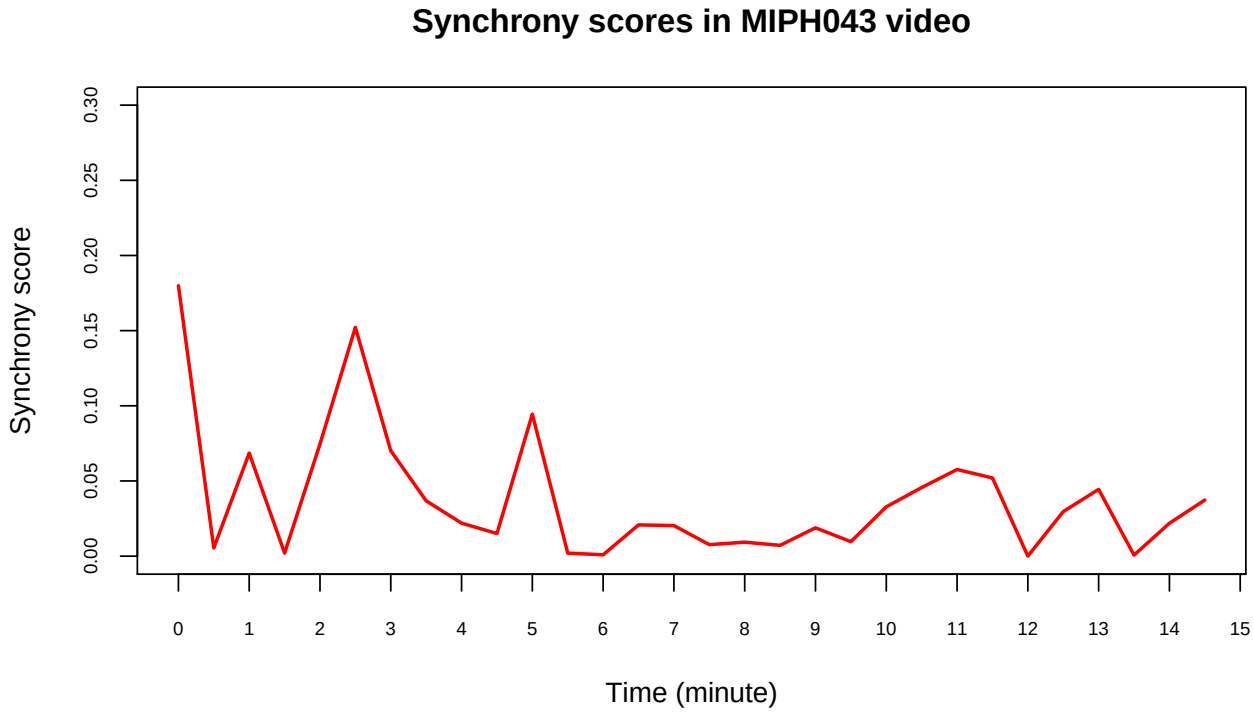

```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0
##  num [1:31] 0.0068 0.040758 0.002717 0.000973 0.009073 ...
## NULL
```

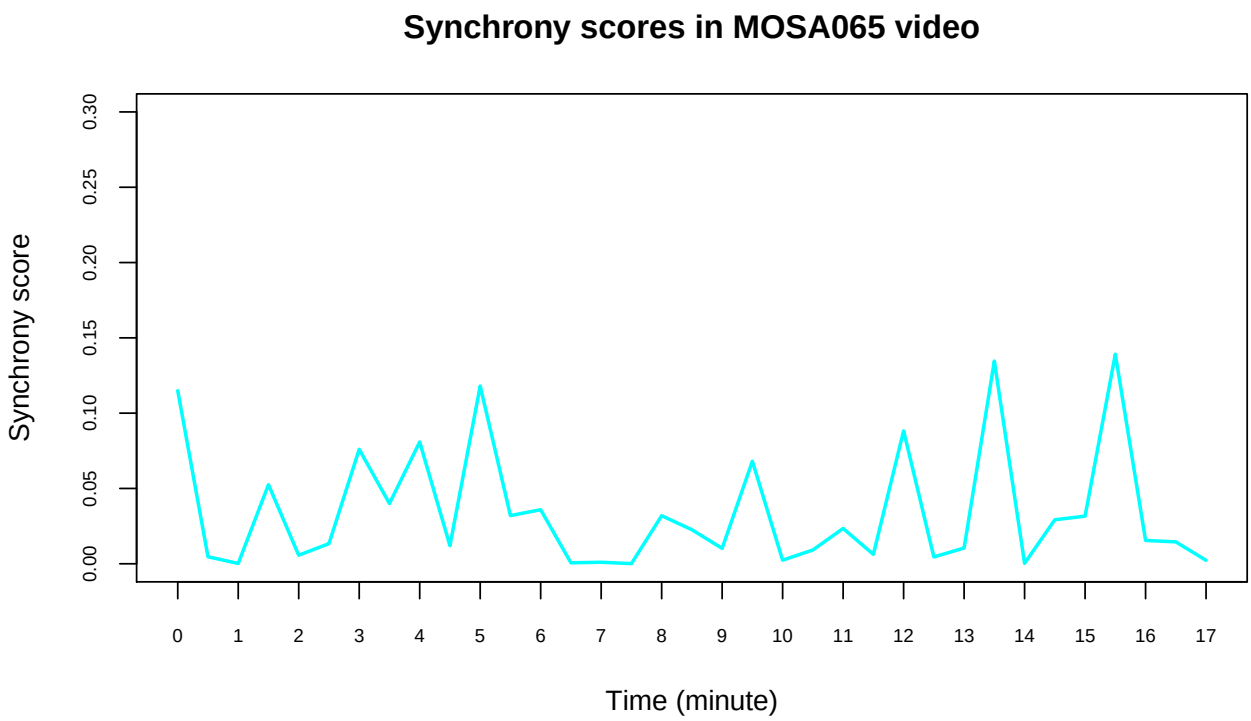



```
##  [1] 0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0 5.5 6.0 6.5
##  num [1:14] 0.073278 0.048566 0.048273 0.005188 0.000367 ...
## NULL
```

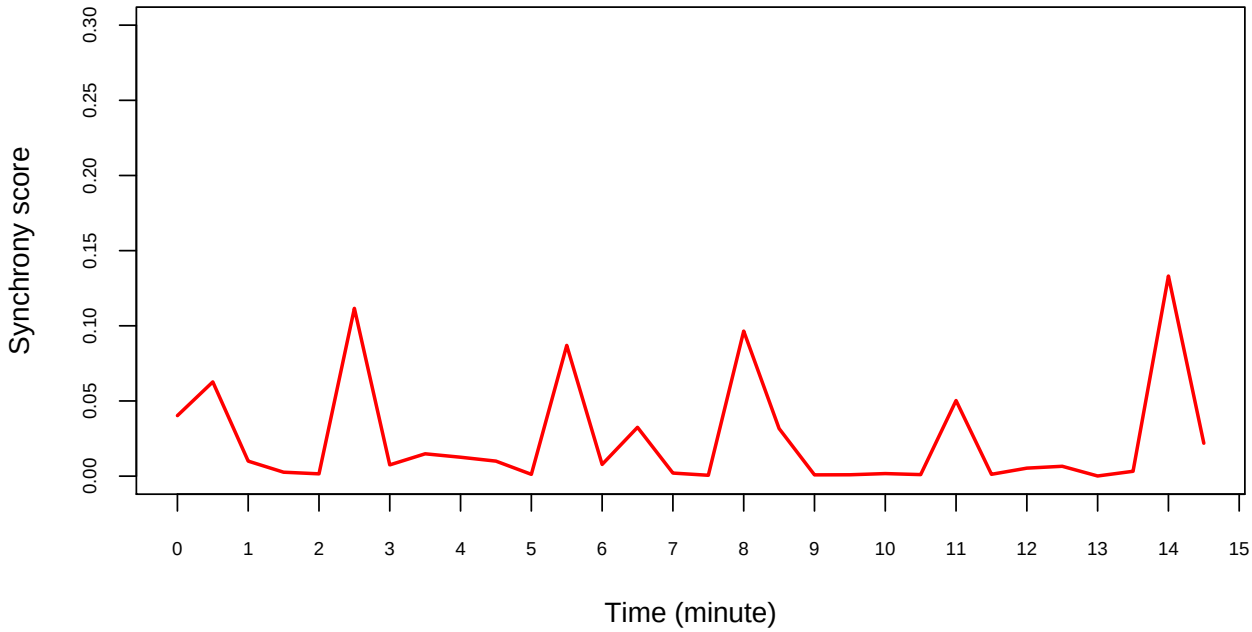



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5
##  num [1:30] 0.17985 0.00546 0.06853 0.00201 0.0746 ...
## NULL
```



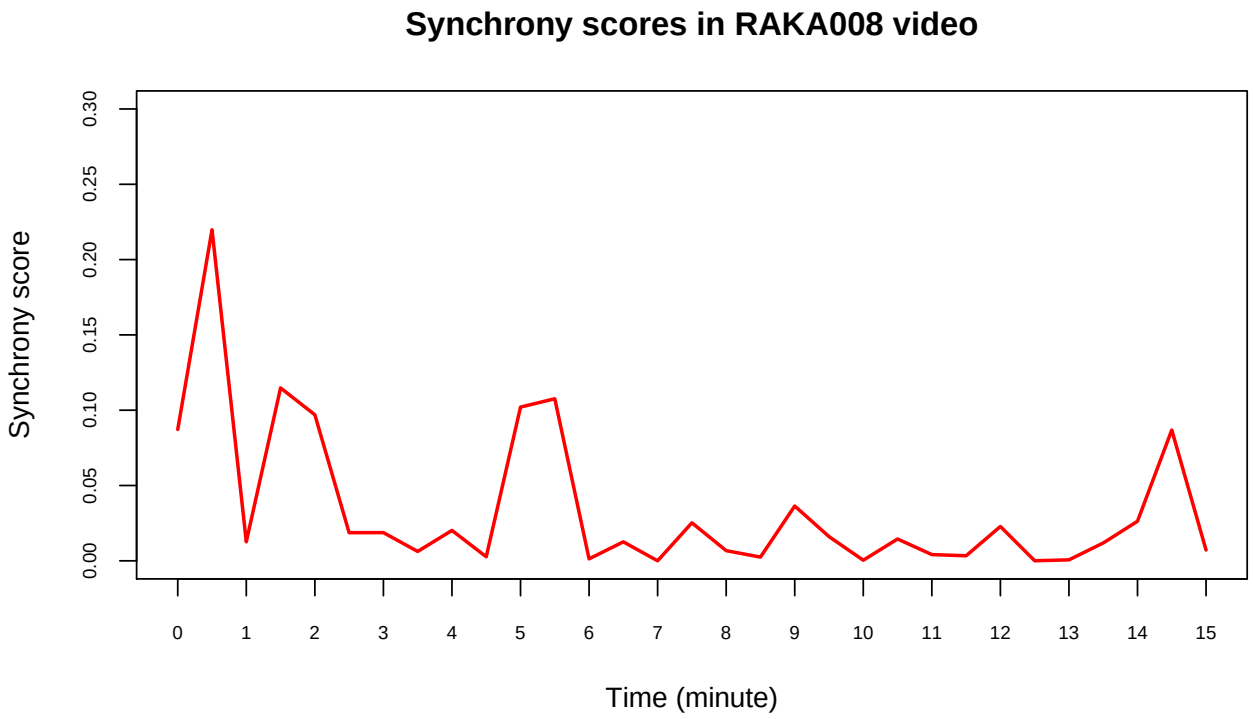



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5
##  num [1:30] 0.04027 0.06268 0.00999 0.00262 0.00154 ...
## NULL
```

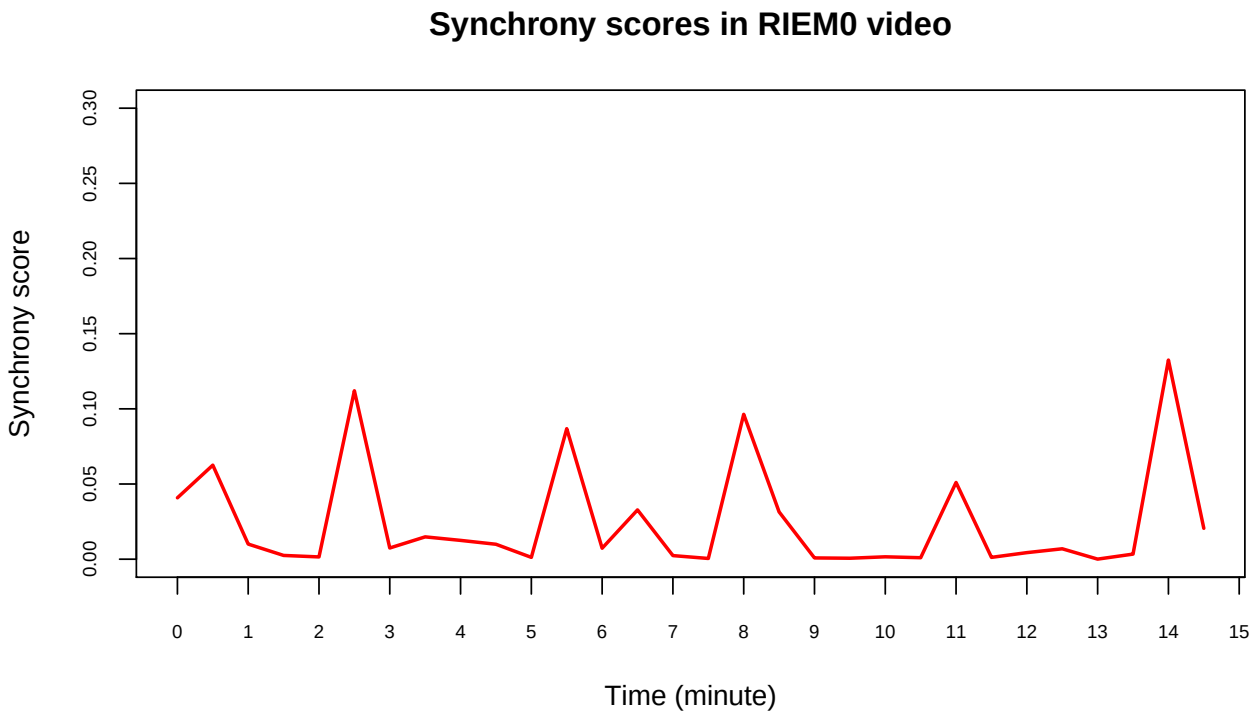



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0
```

```
##  num [1:31] 0.0872 0.2198 0.0126 0.1148 0.097 ...
## NULL
```

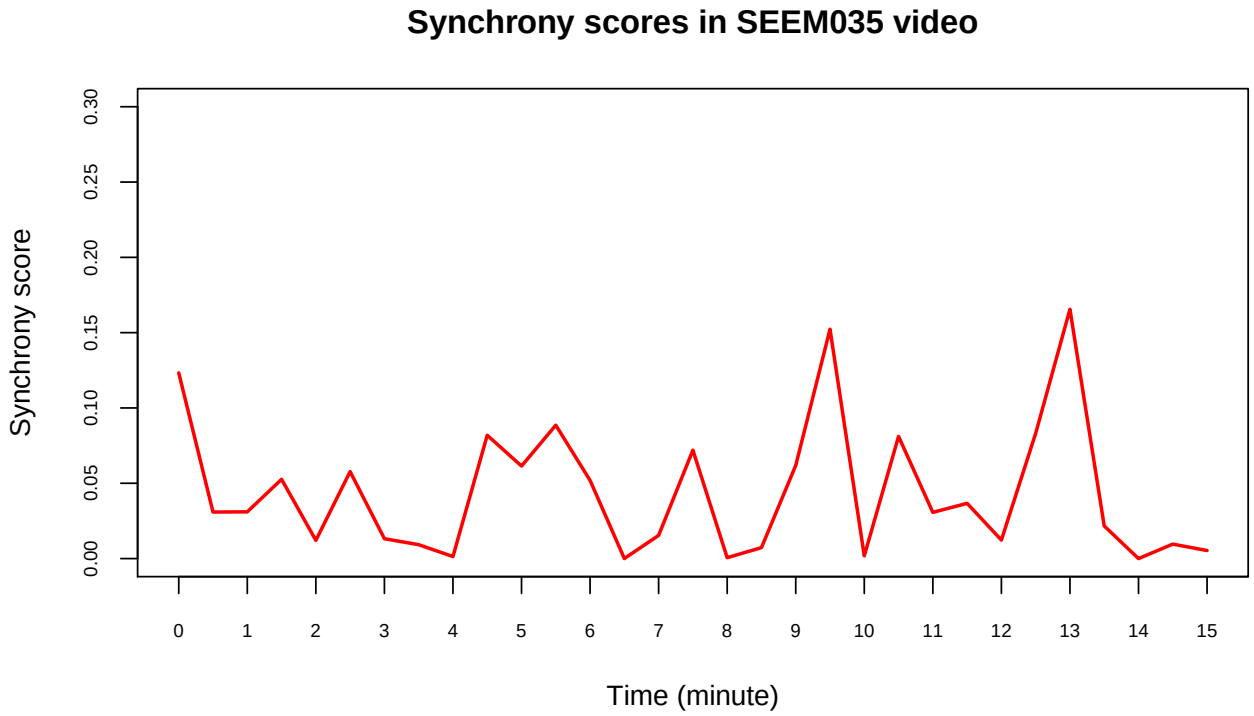



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5
##  num [1:30] 0.04083 0.06254 0.01005 0.00249 0.0015 ...
## NULL
```

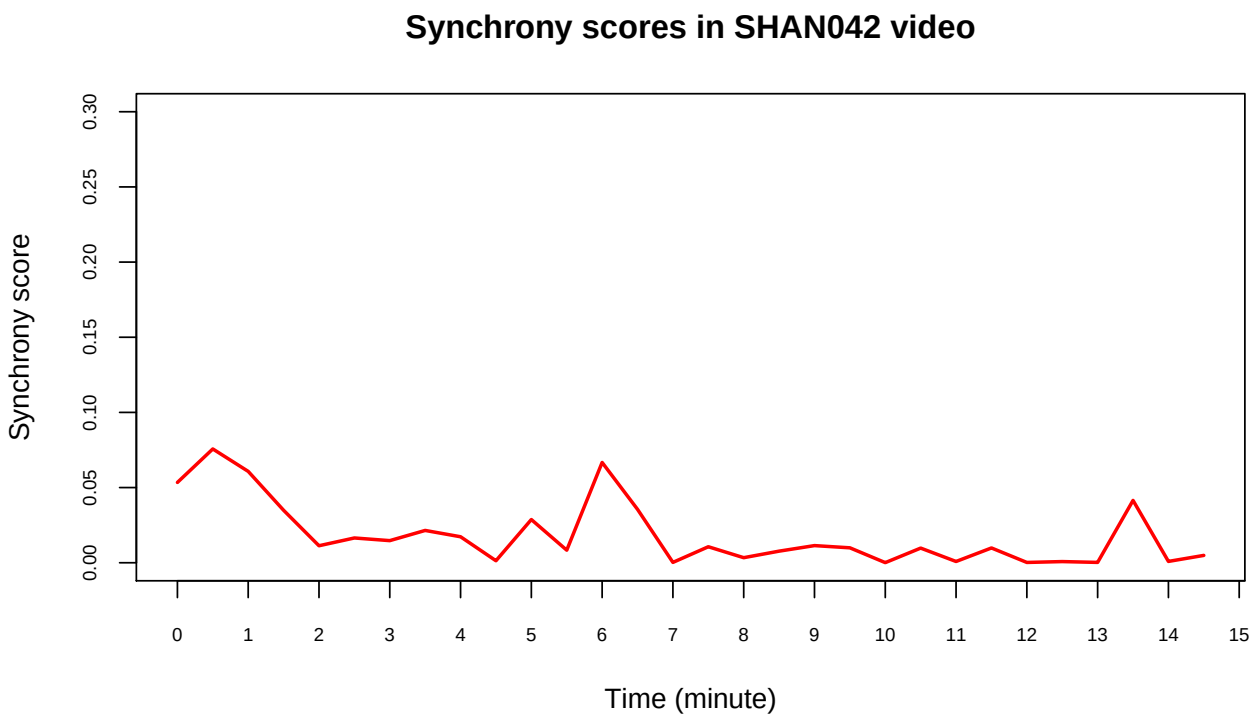



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
```

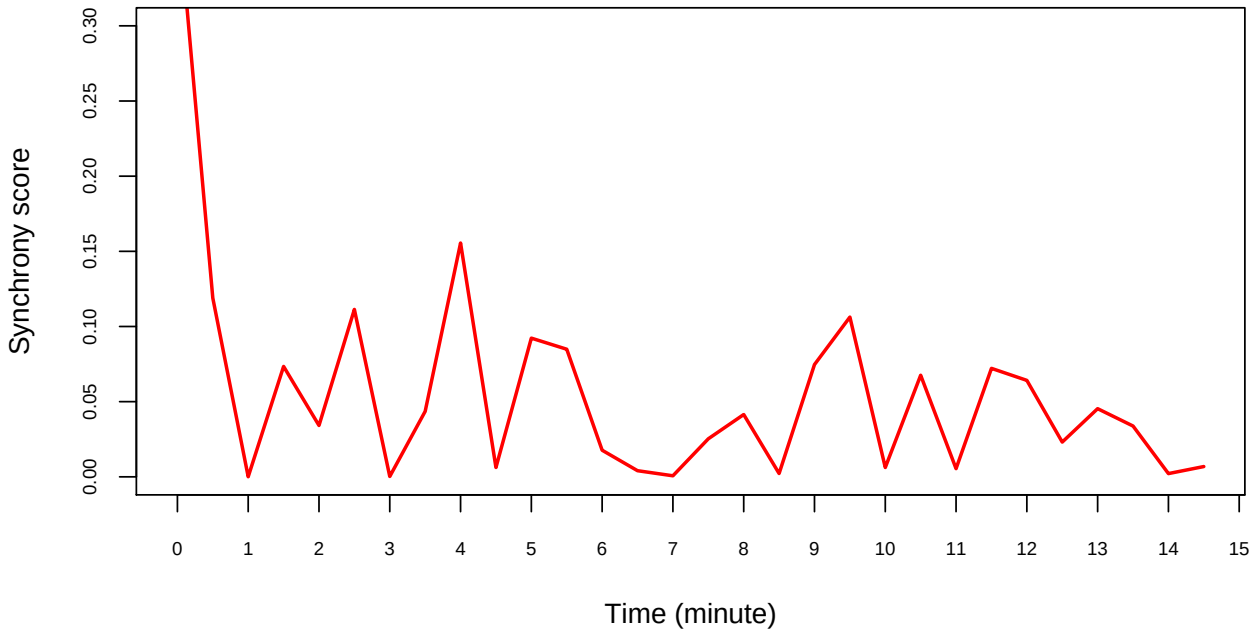
```
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0
##  num [1:31] 0.1233 0.0309 0.031 0.0526 0.0121 ...
## NULL
```




```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5
##  num [1:30] 0.0534 0.0757 0.0608 0.0349 0.0113 ...
## NULL
```

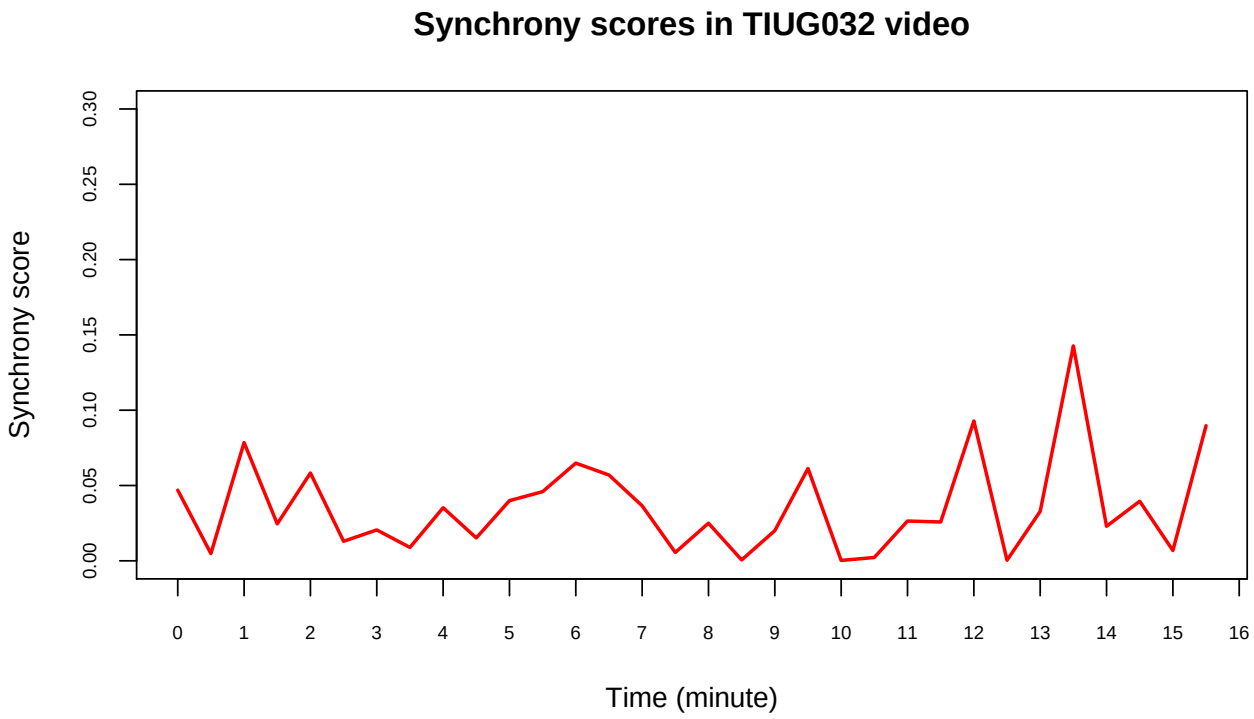



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5
##  num [1:30] 3.84e-01 1.19e-01 5.88e-05 7.34e-02 3.42e-02 ...
## NULL
```

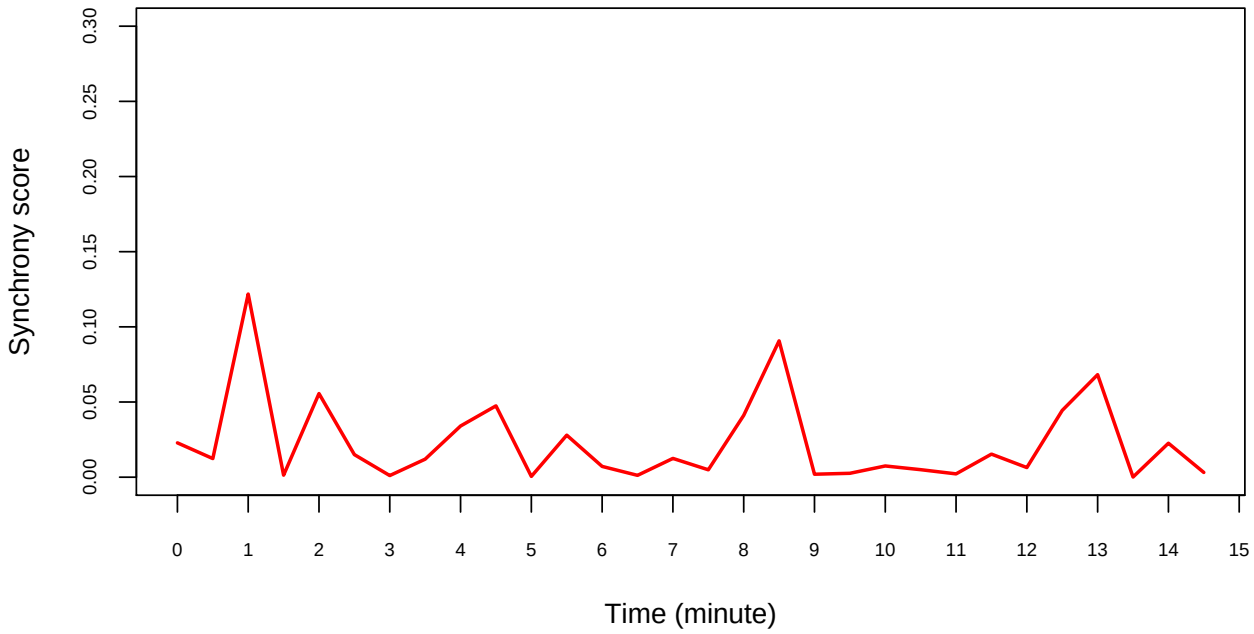



```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5 15.0 15.5
```

```
##  num [1:32] 0.0469 0.00481 0.07851 0.02454 0.0583 ...
## NULL
```



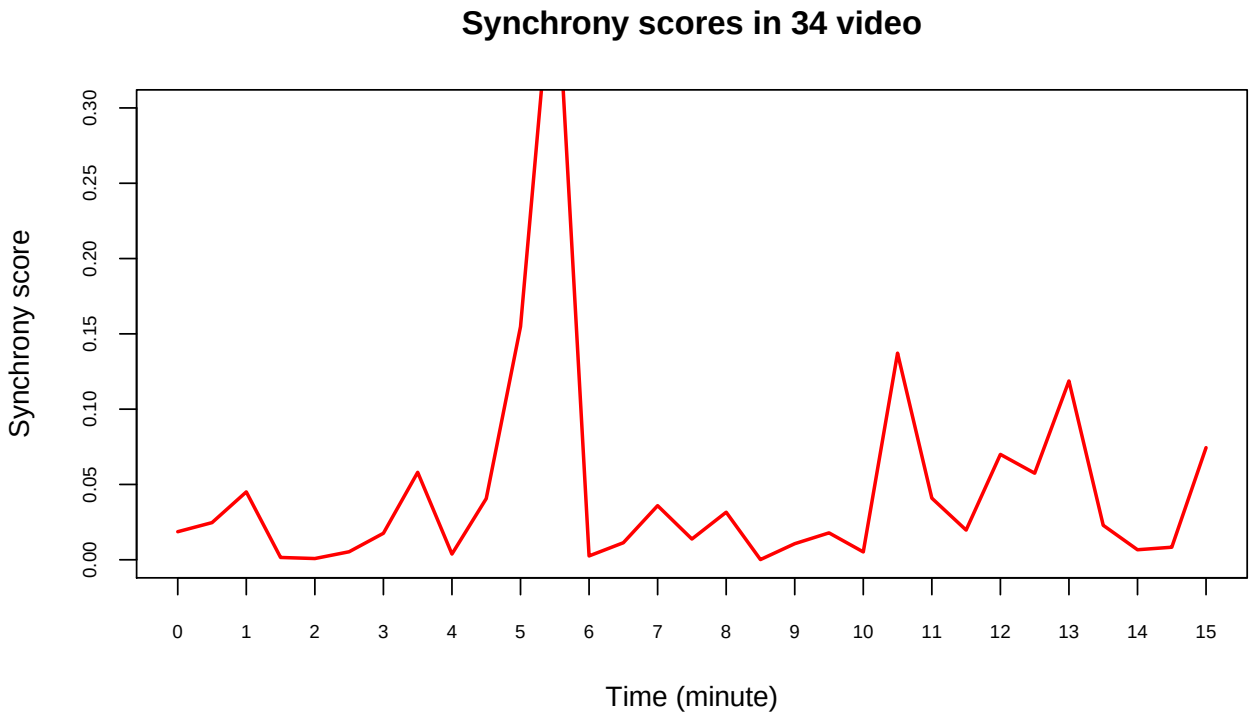

```
##  [1]  0.0  0.5  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5
## [15]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5
## [29] 14.0 14.5
##  num [1:30] 0.02284 0.01238 0.12182 0.00135 0.05563 ...
## NULL
```

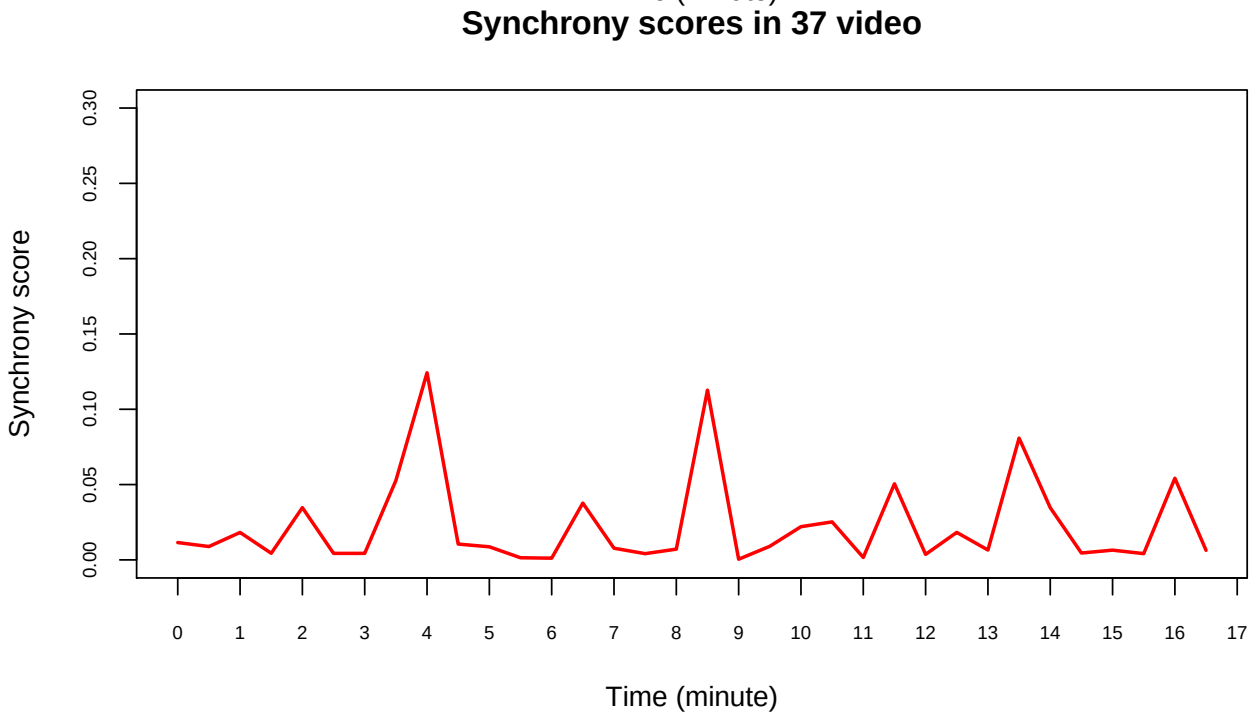


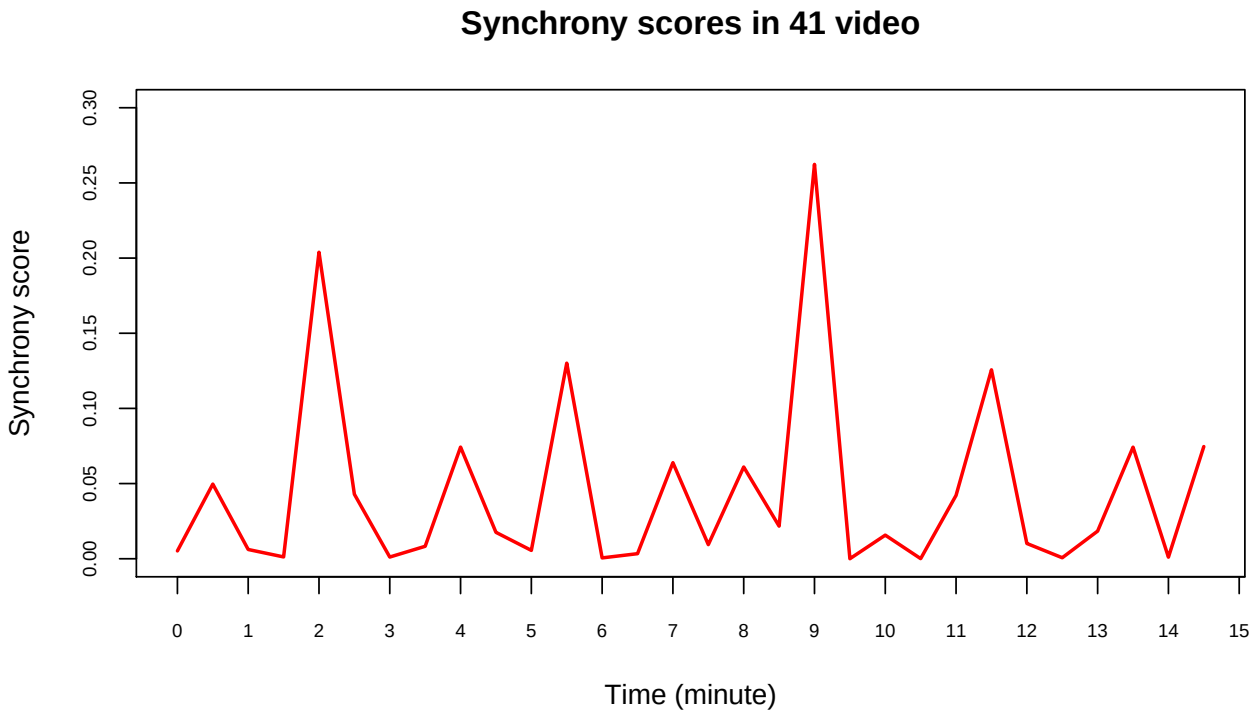
```
#par(mar=c(4,4,4,3), mfrow=c(1,1))

#for (i in unique(SSIlog$video)){
 #       plot(SSIlog[which(SSIlog$video==i),]$Time_min,
#        SSIlog[which(SSIlog$video==i),]$SSI_mo_ch, type="l", ylim=c(0, 0.3), #col=rainbow(4)[1], main=

#        lines(SSIlog[which(SSIlog$video==i),]$SSI_fa_mo_ch, col=rainbow(4)[2], lwd=2)
#        lines(SSIlog[which(SSIlog$video==i),]$SSI_fa_ch, col=rainbow(4)[3], lwd=2)
#
#        str(SSIlog[which(SSIlog$video==i),]$Time_min)
#        str(SSIlog[which(SSIlog$video==i),]$SSI_mo_ch)
#        str(SSIlog[which(SSIlog$video==i),]$SSI_fa_mo_ch)
#        str(SSIlog[which(SSIlog$video==i),]$SSI_fa_ch)}

#abline(h= mean(SSIlog$SSI_mo_ch, na.rm=TRUE), col=rainbow(4)[4], lwd=2, lty=2)

#legend("topleft", inset=.05, c("fa_mo", "fa_mo_ch", "fa_ch",
#"mo_ch"),
#col=rainbow(4), cex=0.6, lwd=2)

#legend ("topright", inset=.05, c(
#      paste ("Mean fa_mo :", round(mean(SSIlog$SSI_fa_mo, na.rm=TRUE),3)),
#      paste ("Mean fa_mo_ch :", round(mean(SSIlog$SSI_fa_mo_ch,na.rm=TRUE),3)),
#      paste ("Mean fa_ch :", round(mean(SSIlog$SSI_fa_ch, na.rm=TRUE),3)),
#      paste ("Mean mo_ch :", round(mean(SSIlog$SSI_mo_ch,na.rm=TRUE),3))),
#col=rainbow(4), cex=0.5, lty=2, lwd=1)}
```

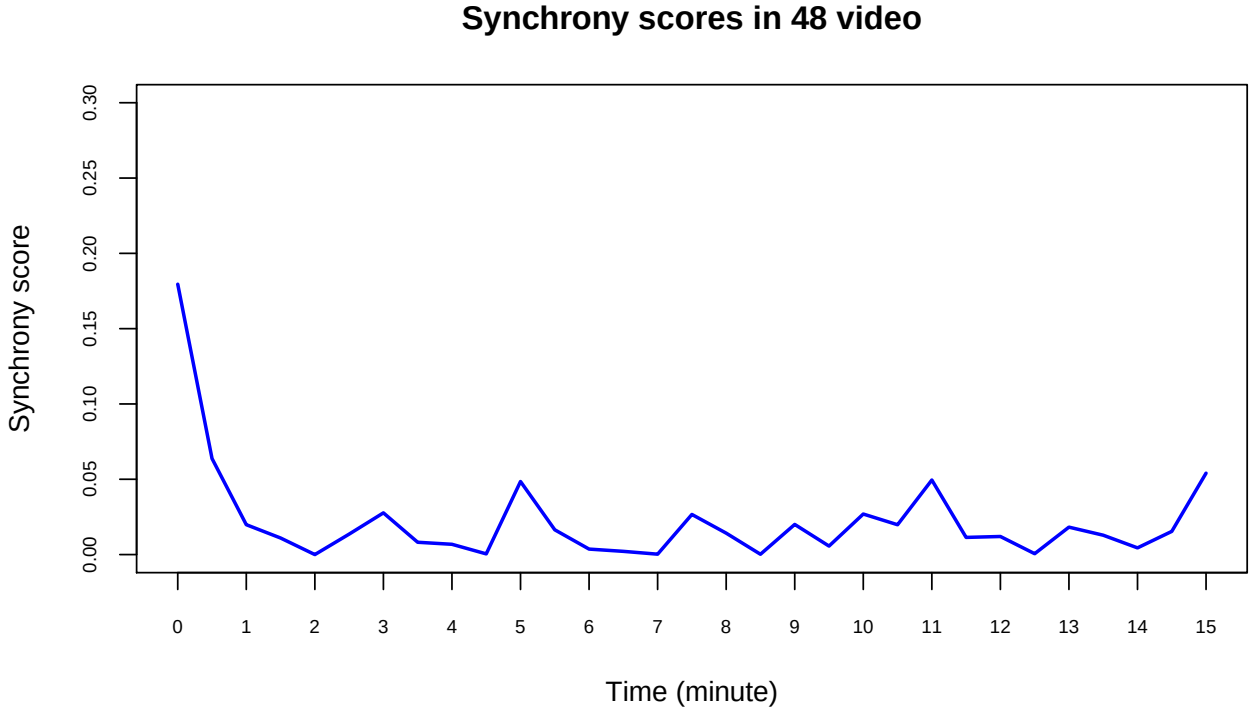
Synchrony scores noLog for each dyad, triad and for the whole group

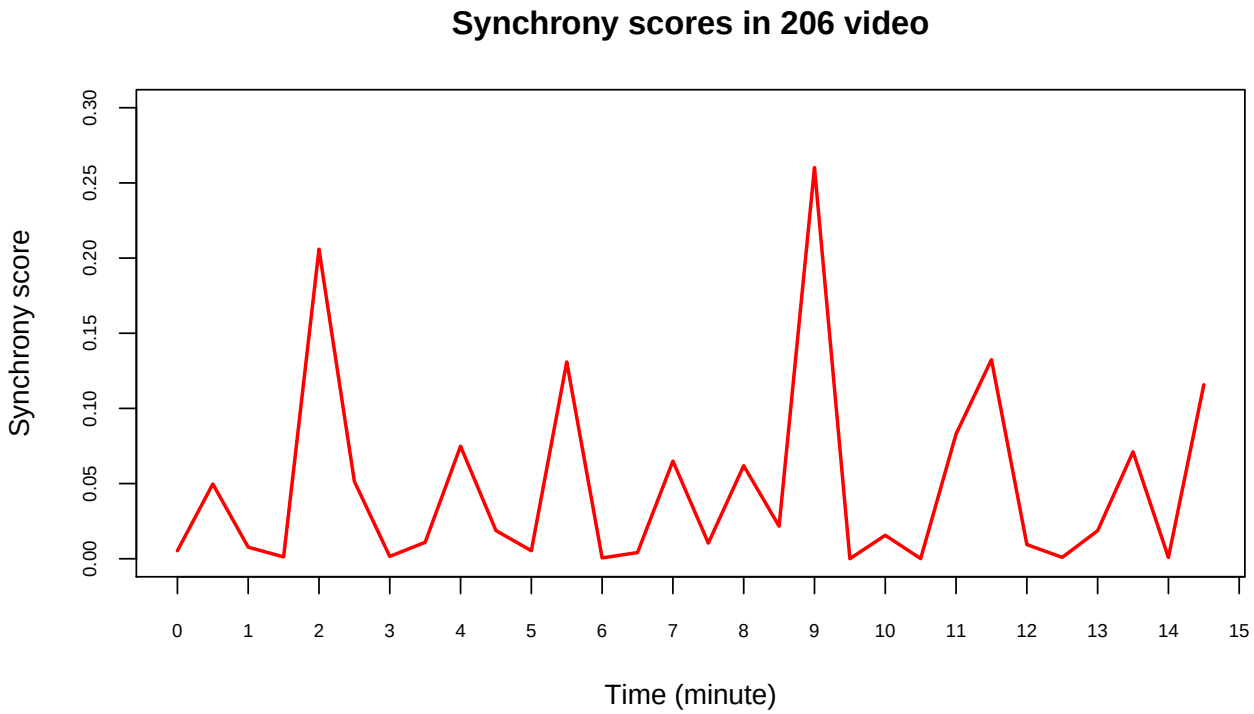
```
par(mar=c(4,4,4,3), mfrow=c(1,1))
for (i in unique(SSInoLog$video)){
      if (all(!is.na(SSInoLog[which(SSIlog$video==i),]$SSI_mo_ch)==TRUE)){
#        print(str(SSInoLog[which(SSIlog$video==i),]$Time_min))
#        print(str(SSInoLog[which(SSInoLog$video==i),]$SSI_mo_ch))
        plot(SSInoLog[which(SSIlog$video==i),]$Time_min, SSInoLog[which(SSInoLog$video==i),]$SSI_mo_ch,
             ylim=c(0, 0.3), main=paste("Synchrony scores in", i, "video"), xlab = "Time (minute)", ylab

      else if(all(!is.na(SSInoLog[which(SSIlog$video==i),]$SSI_fa_ch)==TRUE)){
#    print(str(SSInoLog[which(SSInoLog$video==i),]$SSI_fa_ch))
    plot(SSInoLog[which(SSIlog$video==i),]$Time_min, SSInoLog[which(SSIlog$video==i),]$SSI_fa_ch, ylim=c
      else{print("error")}}
```

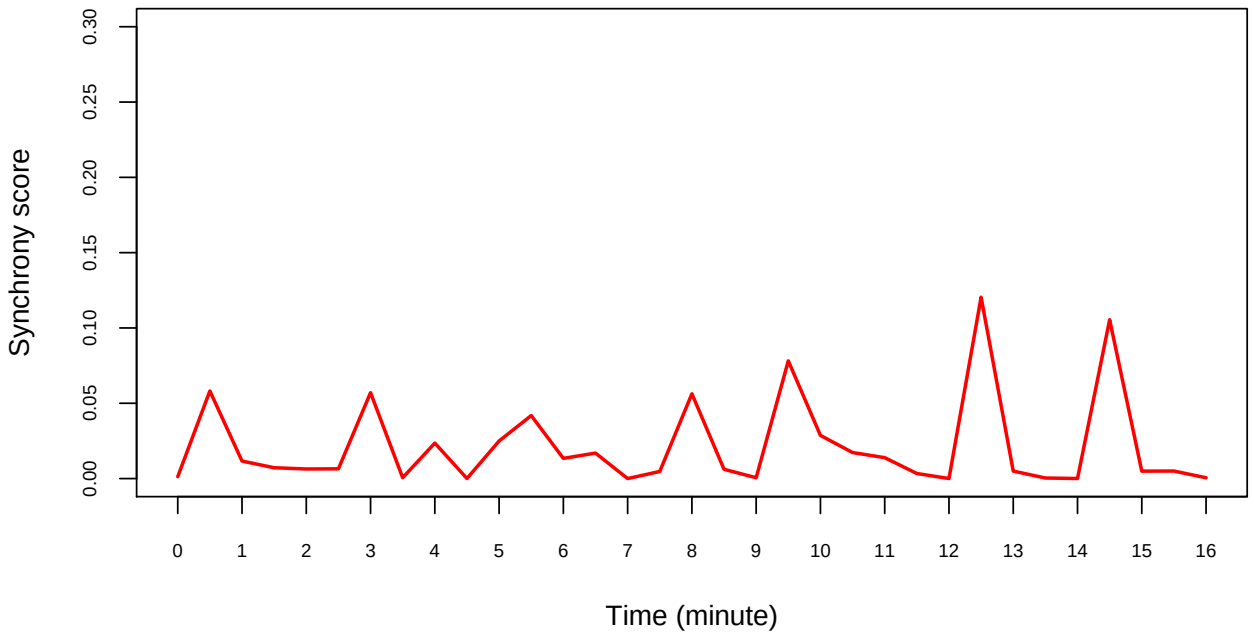

Synchrony scores in 34 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
Time (minute)

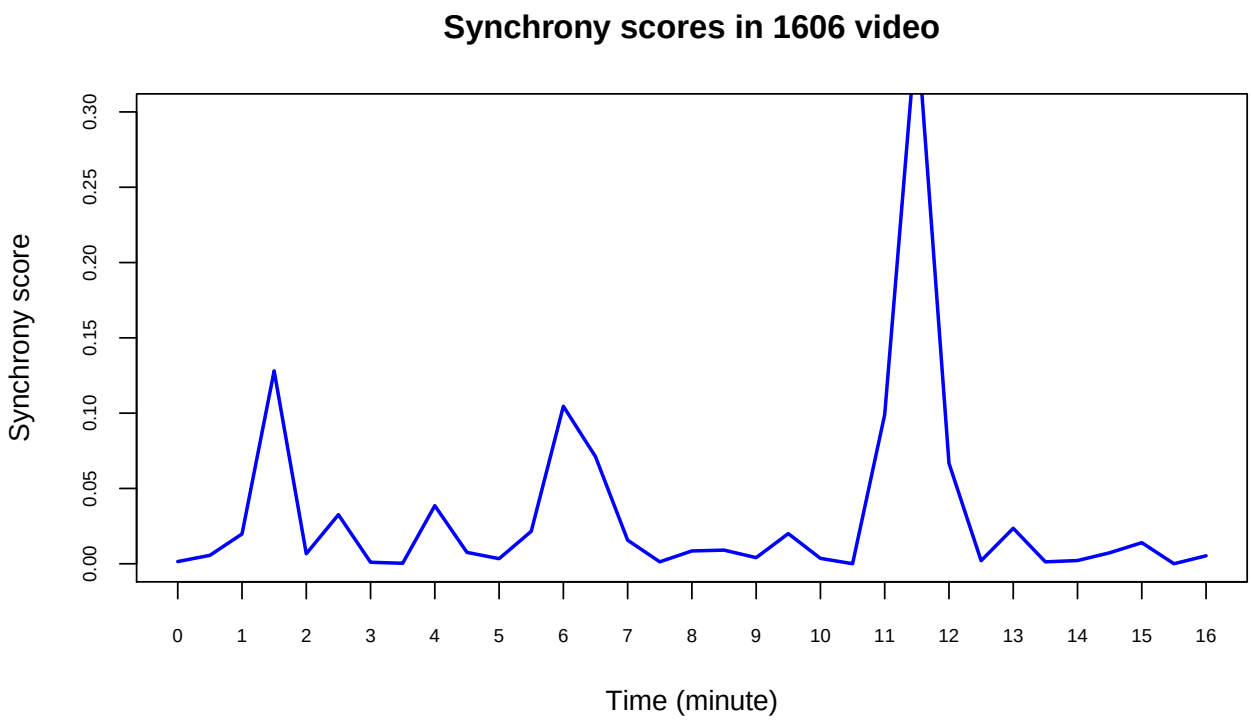

Synchrony scores in 37 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
Time (minute)

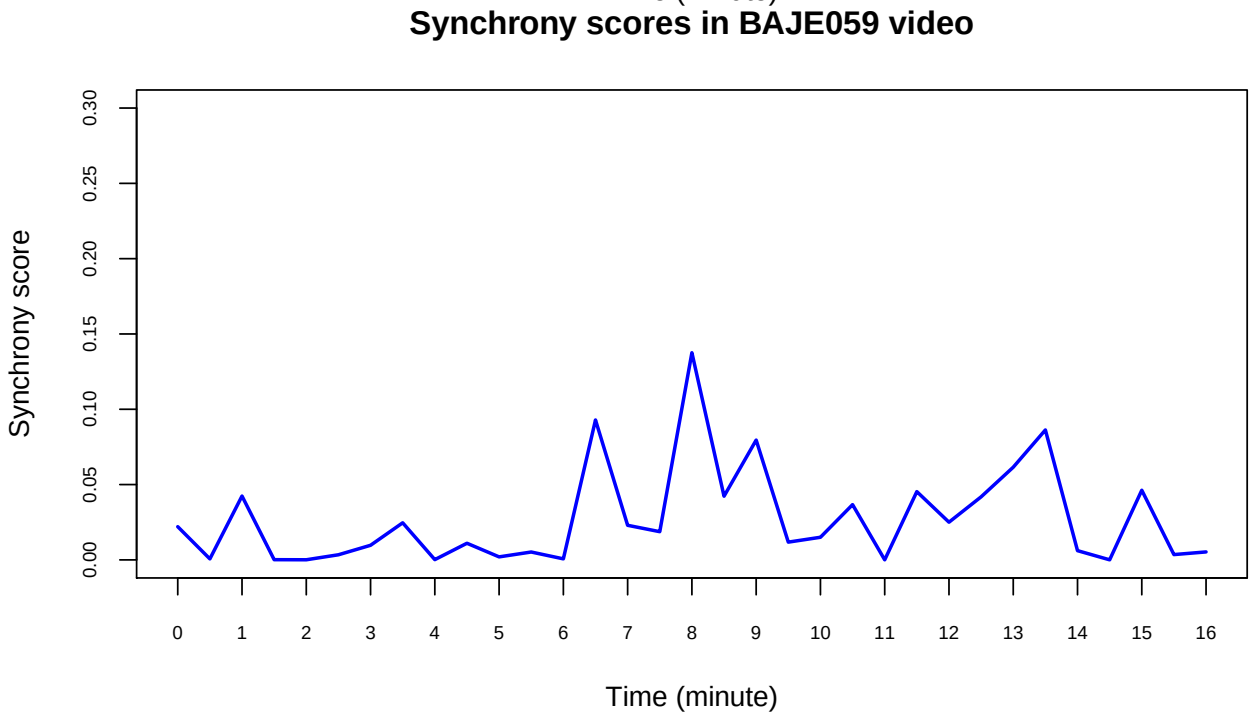

Synchrony scores in 41 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)

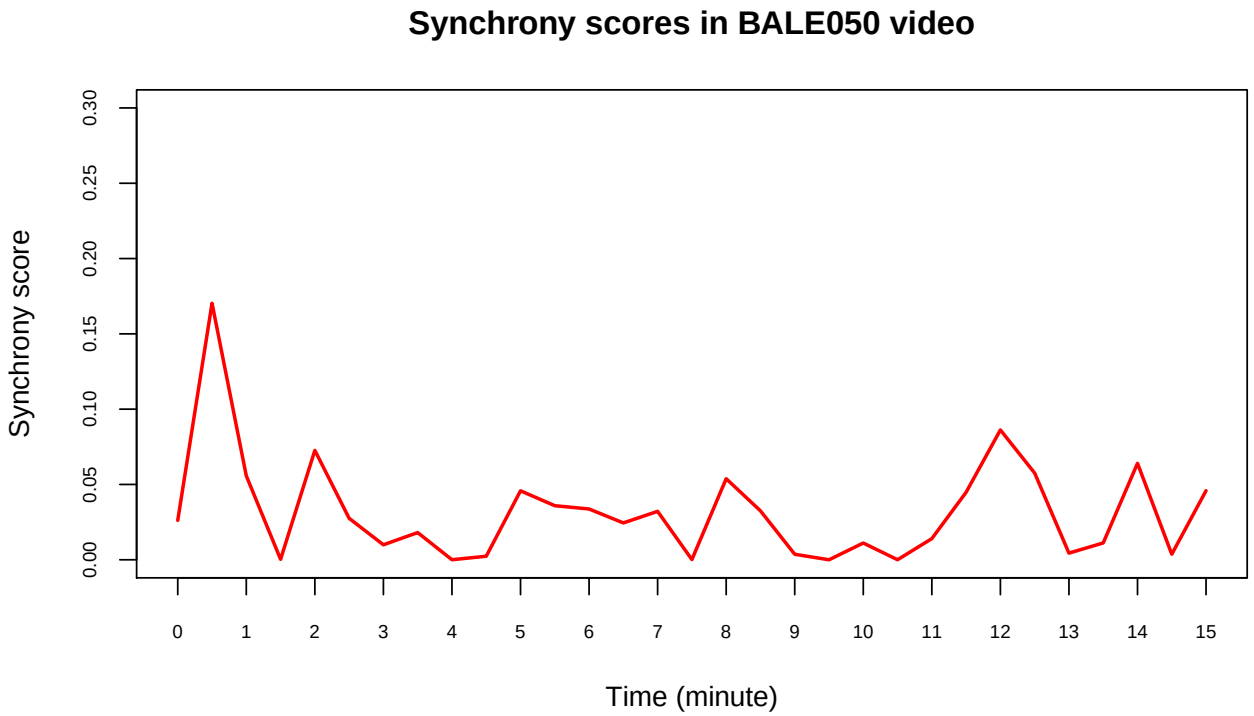

Synchrony scores in 48 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)

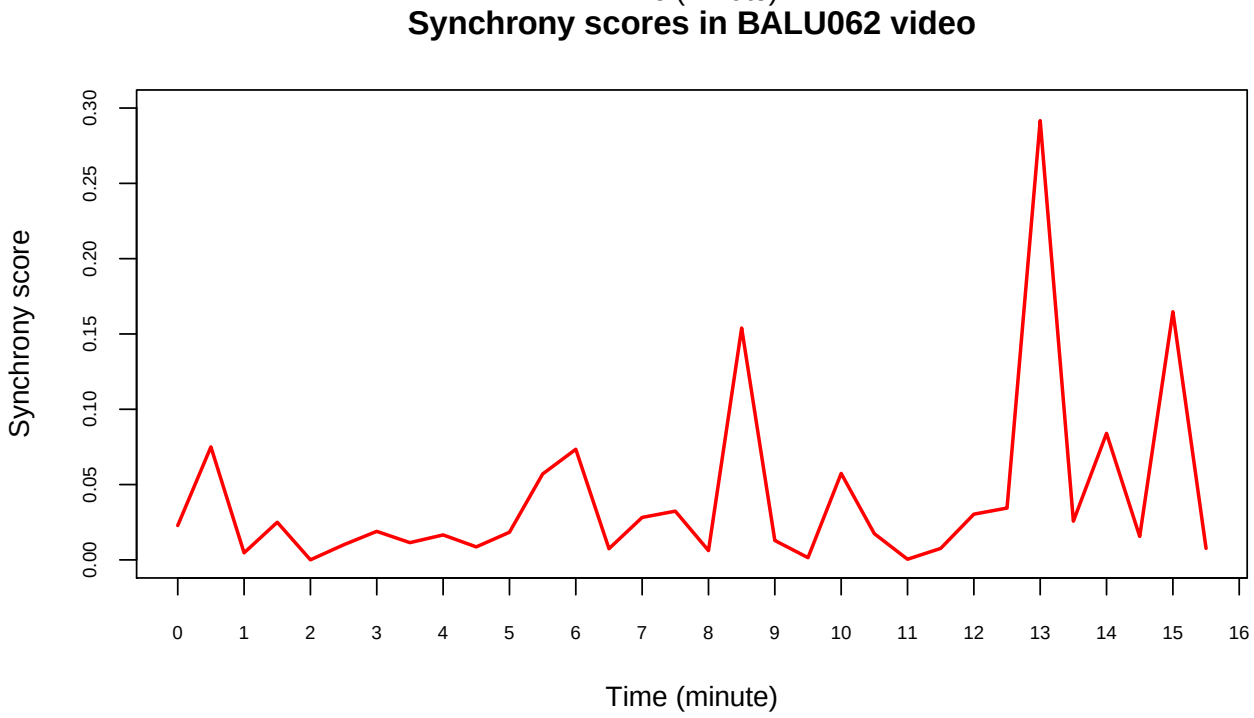

Synchrony scores in 206 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
Time (minute)

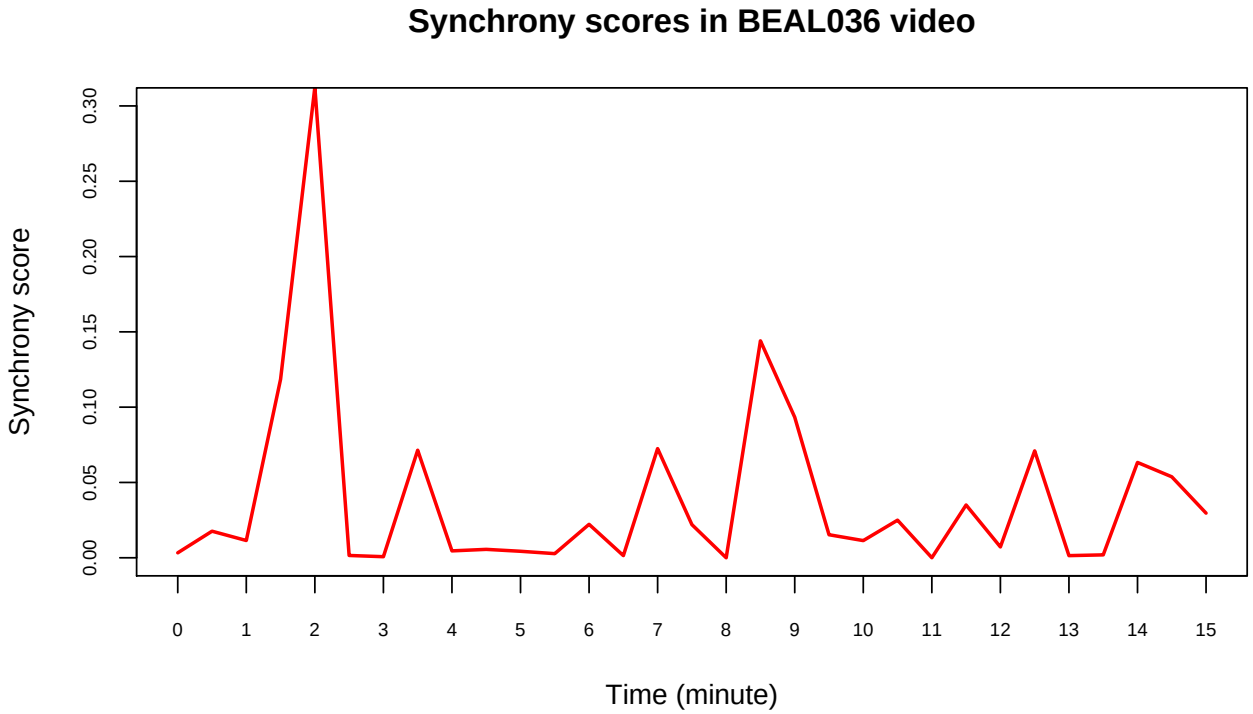

Synchrony scores in 1106 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
Time (minute)

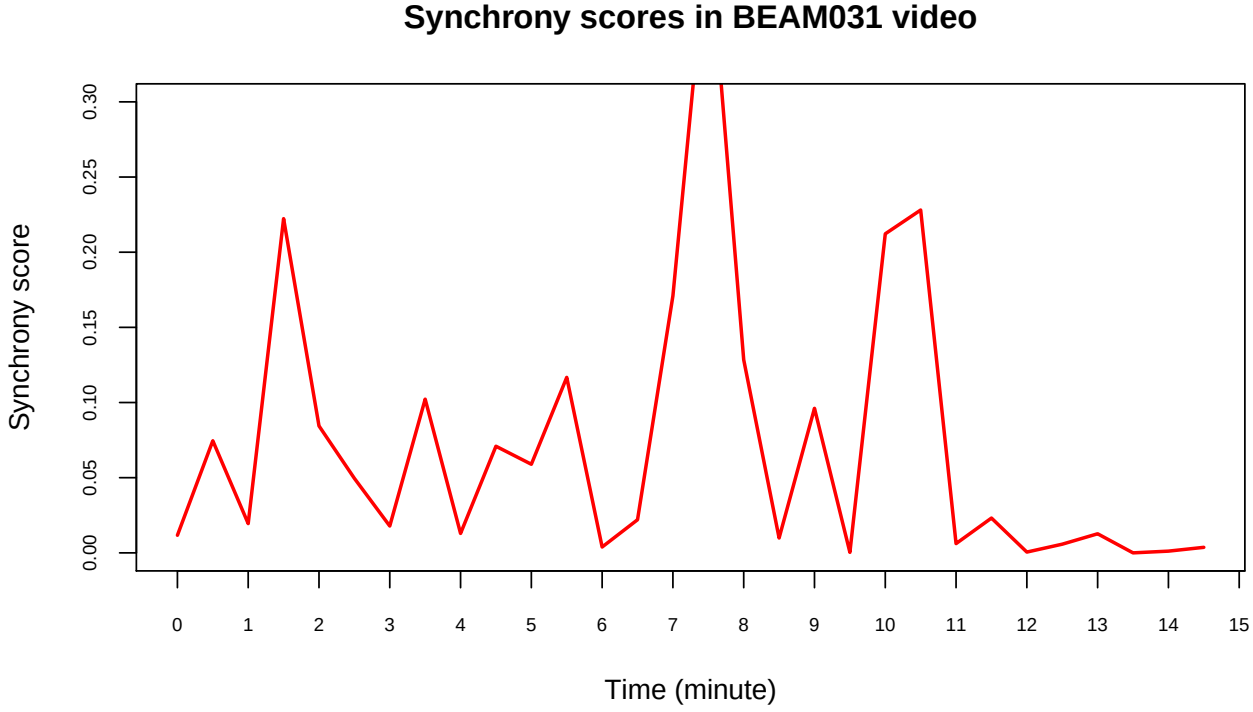

Synchrony scores in 1606 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Time (minute)

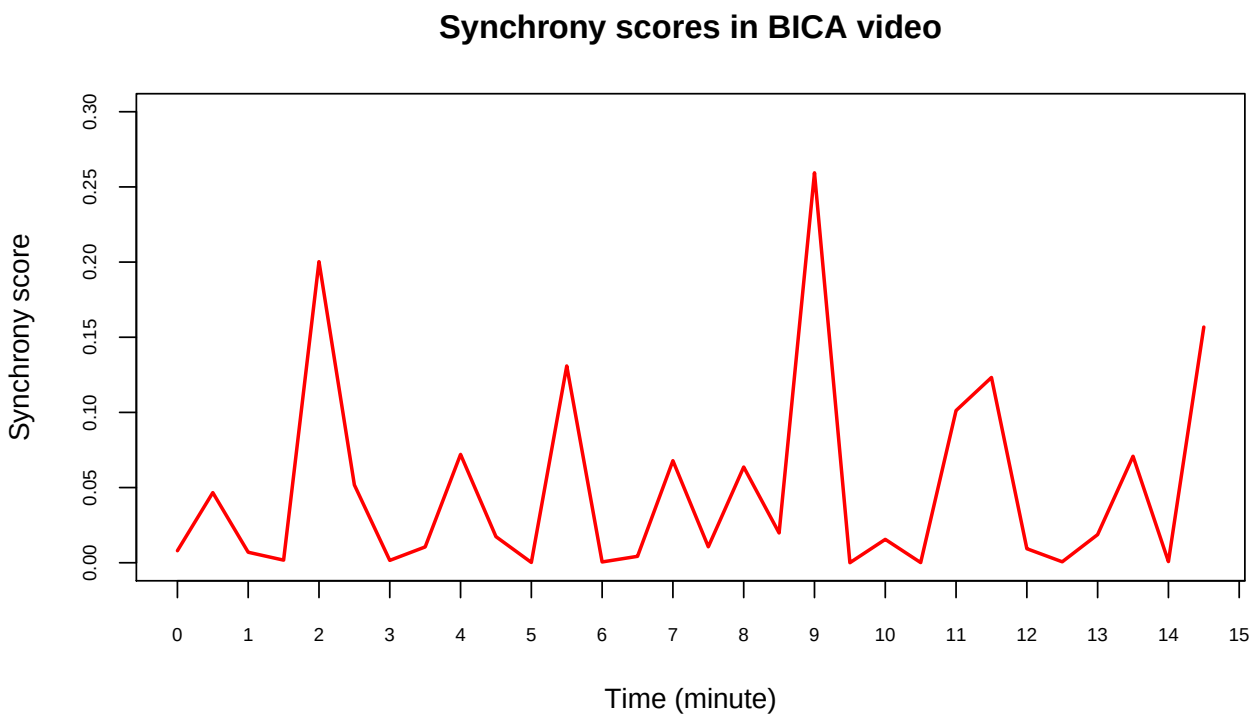

Synchrony scores in BAJE059 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Time (minute)

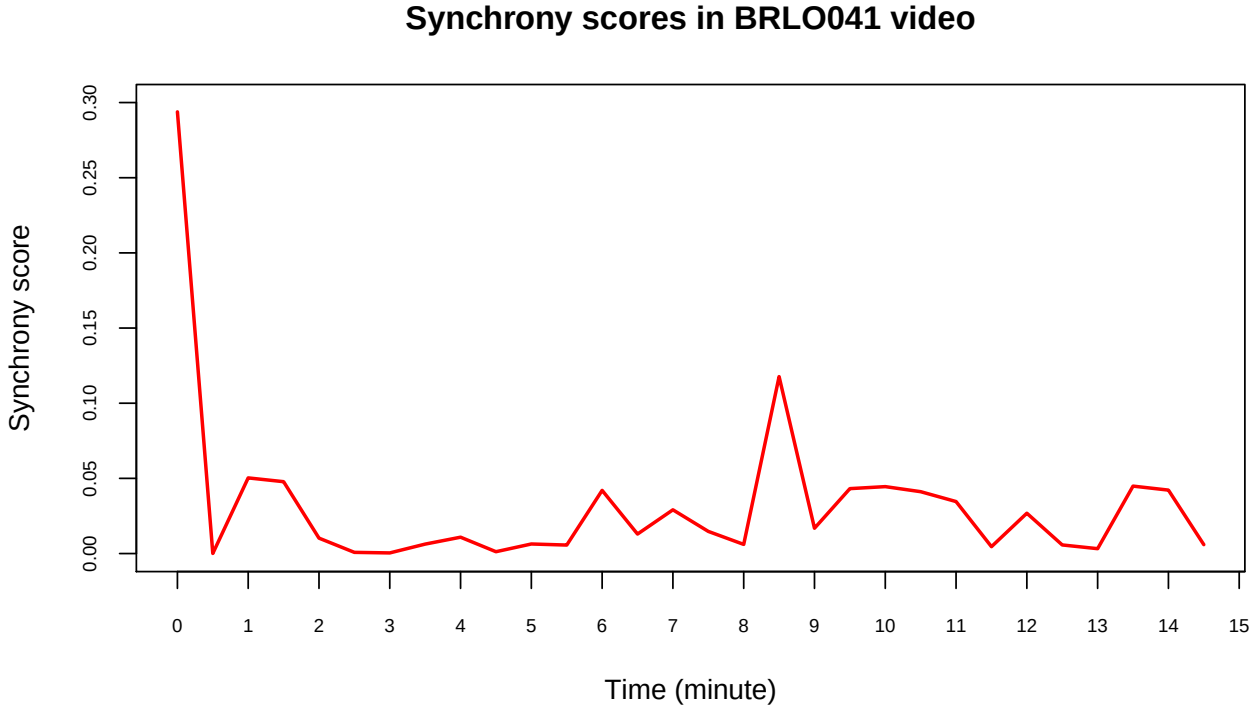

Synchrony scores in BALE050 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)

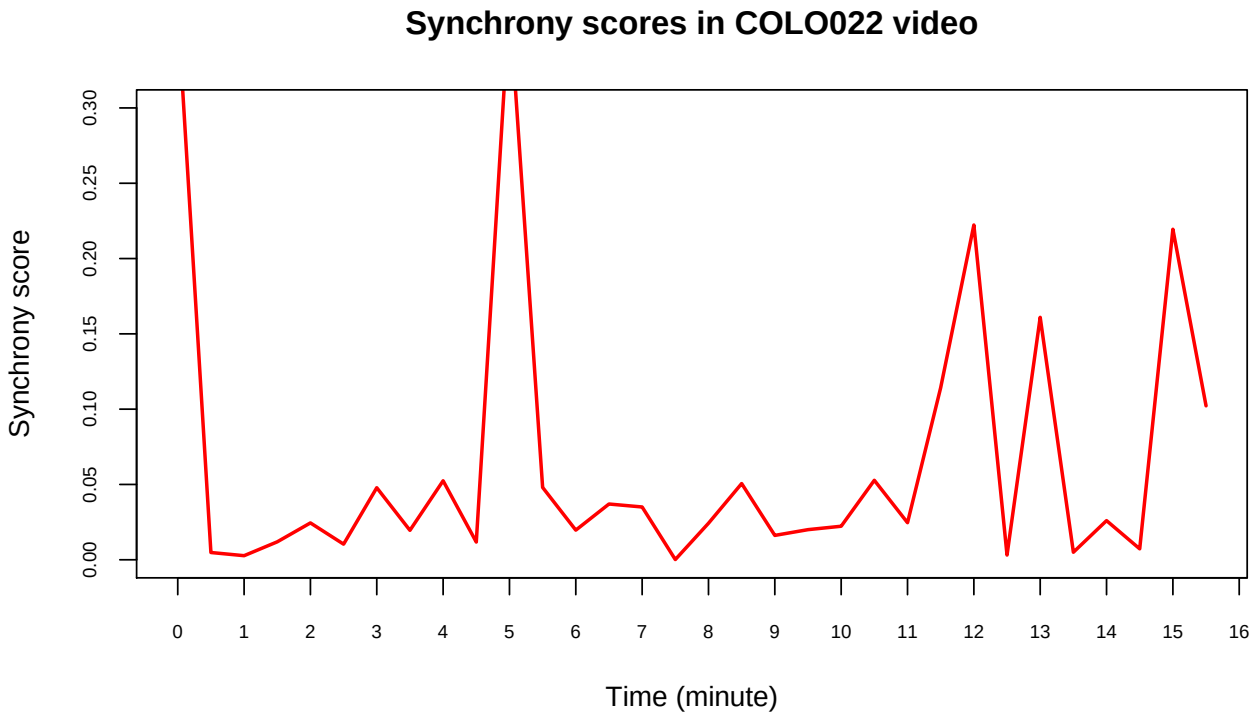

Synchrony scores in BALU062 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Time (minute)

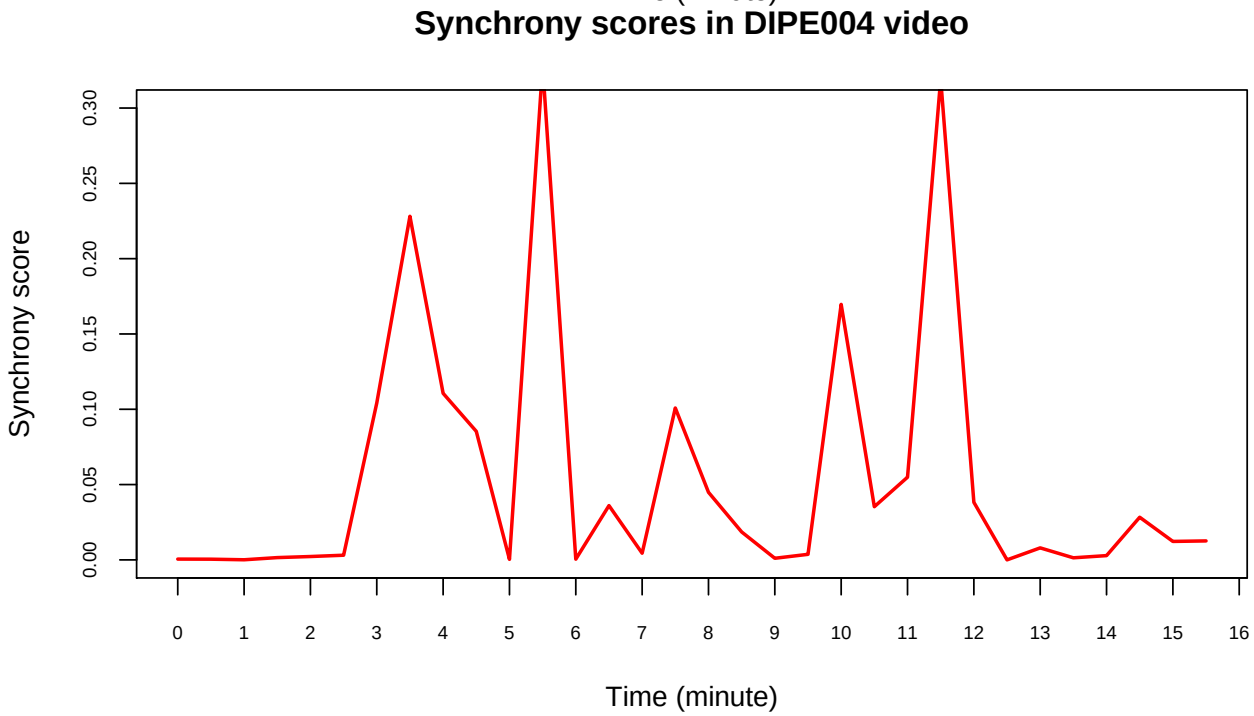

Synchrony scores in BEAL036 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)

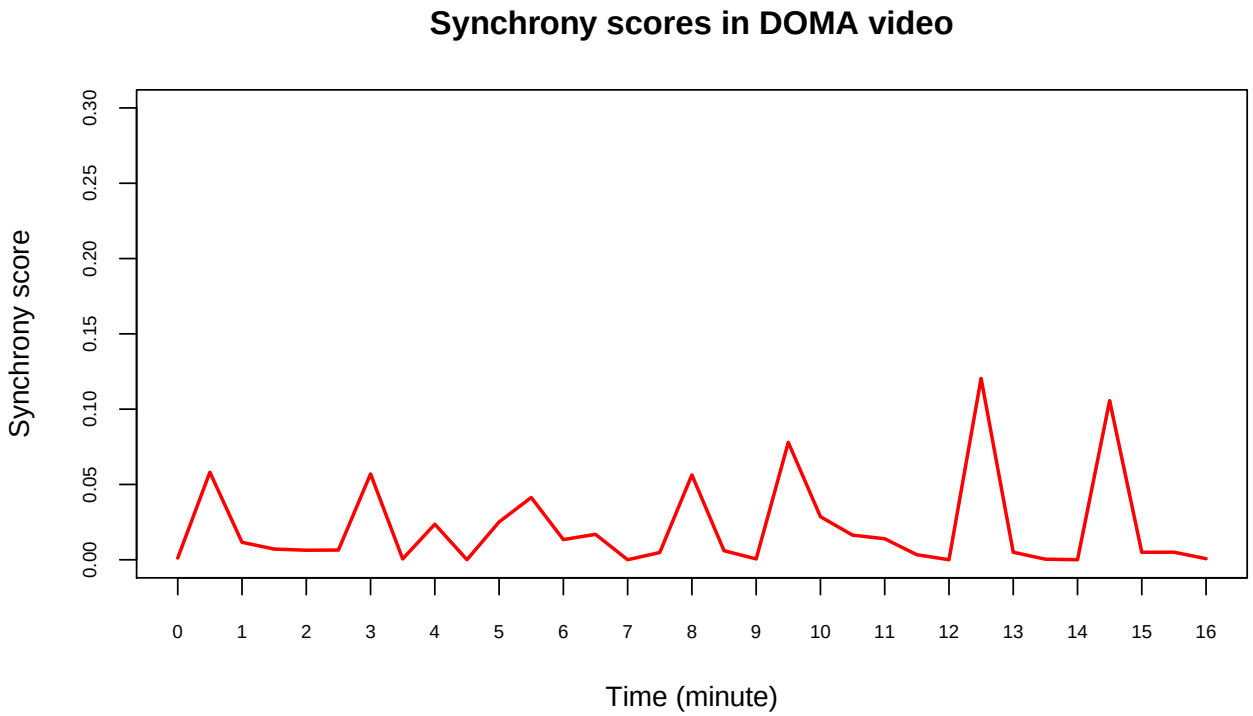

Synchrony scores in BEAM031 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)

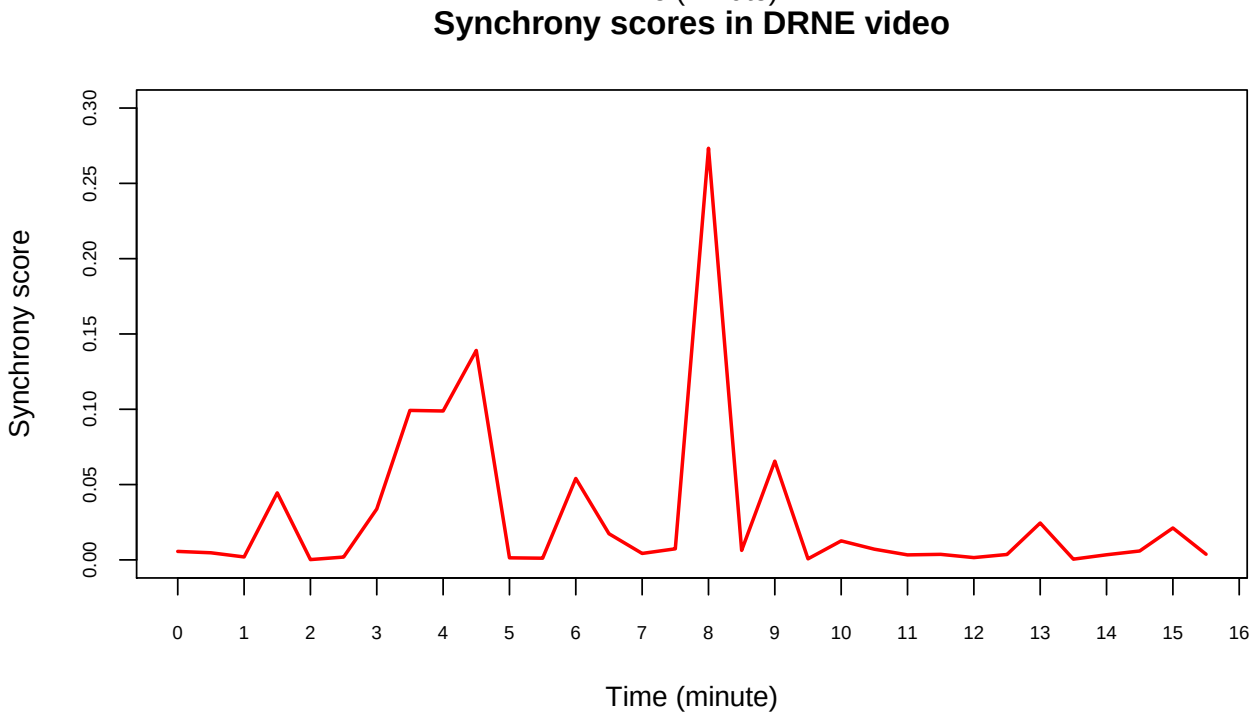

Synchrony scores in BICA video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)

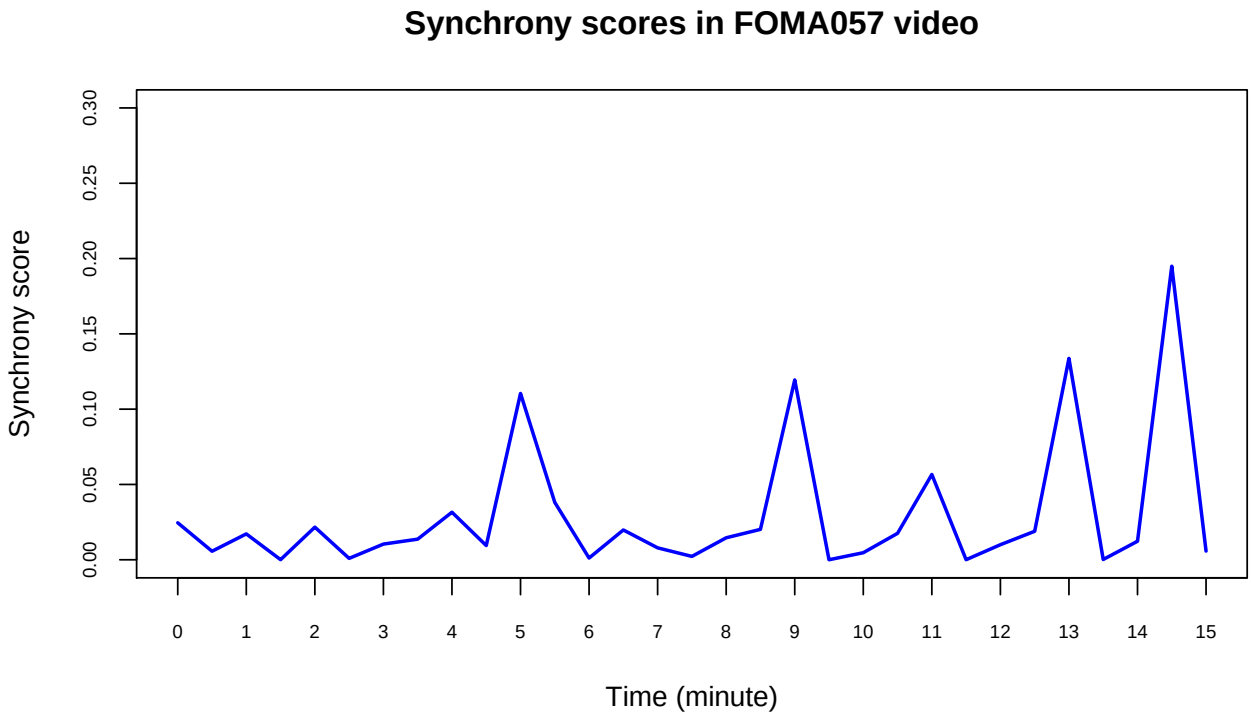

Synchrony scores in BRLO041 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)

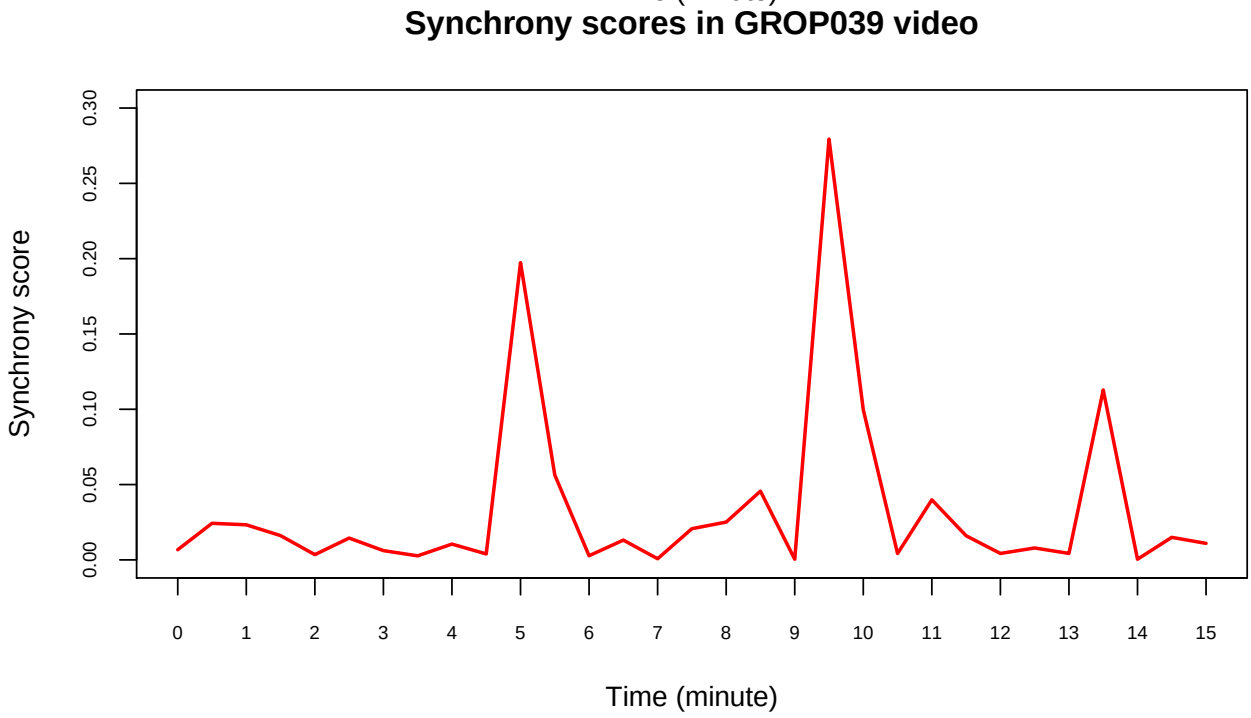

Synchrony scores in COLO022 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Time (minute)

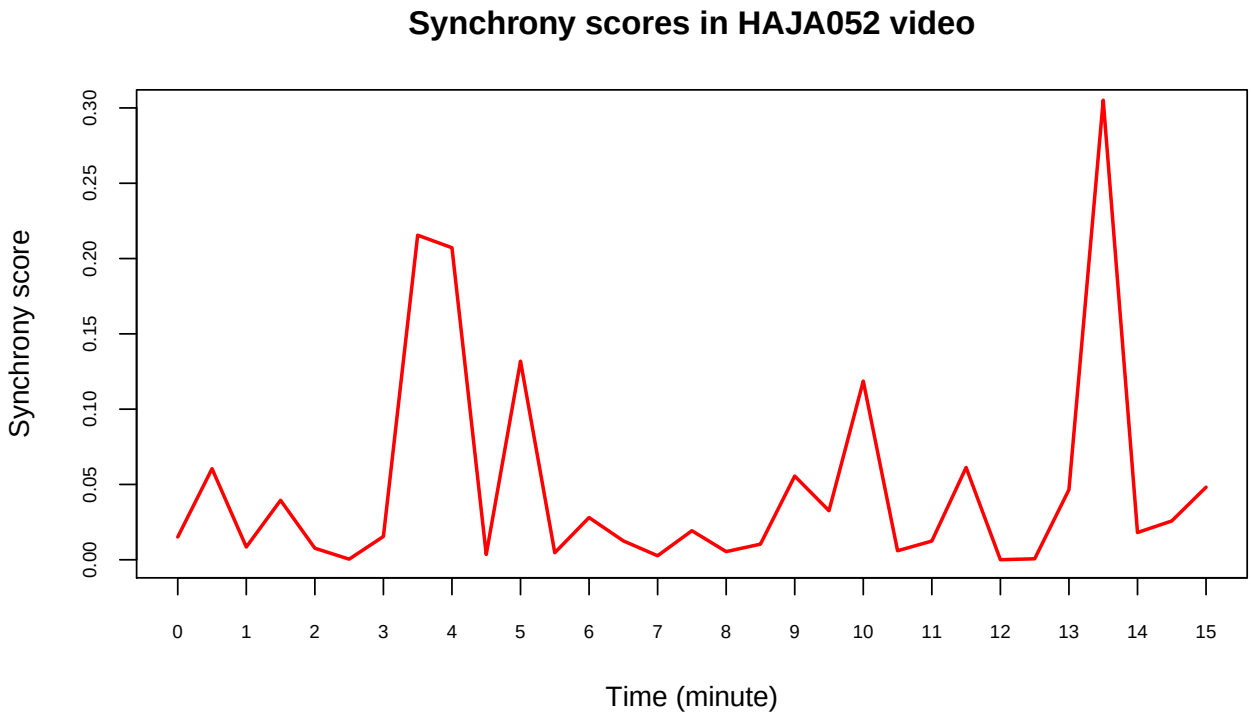

Synchrony scores in DIPE004 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Time (minute)

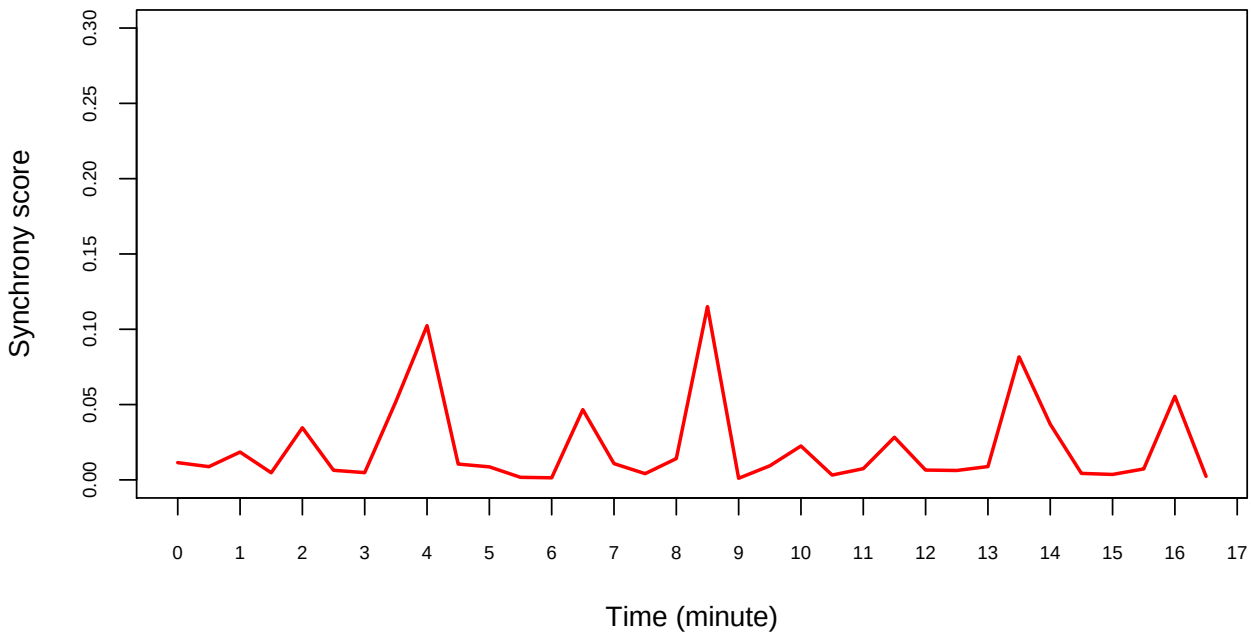

Synchrony scores in DOMA video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
Time (minute)

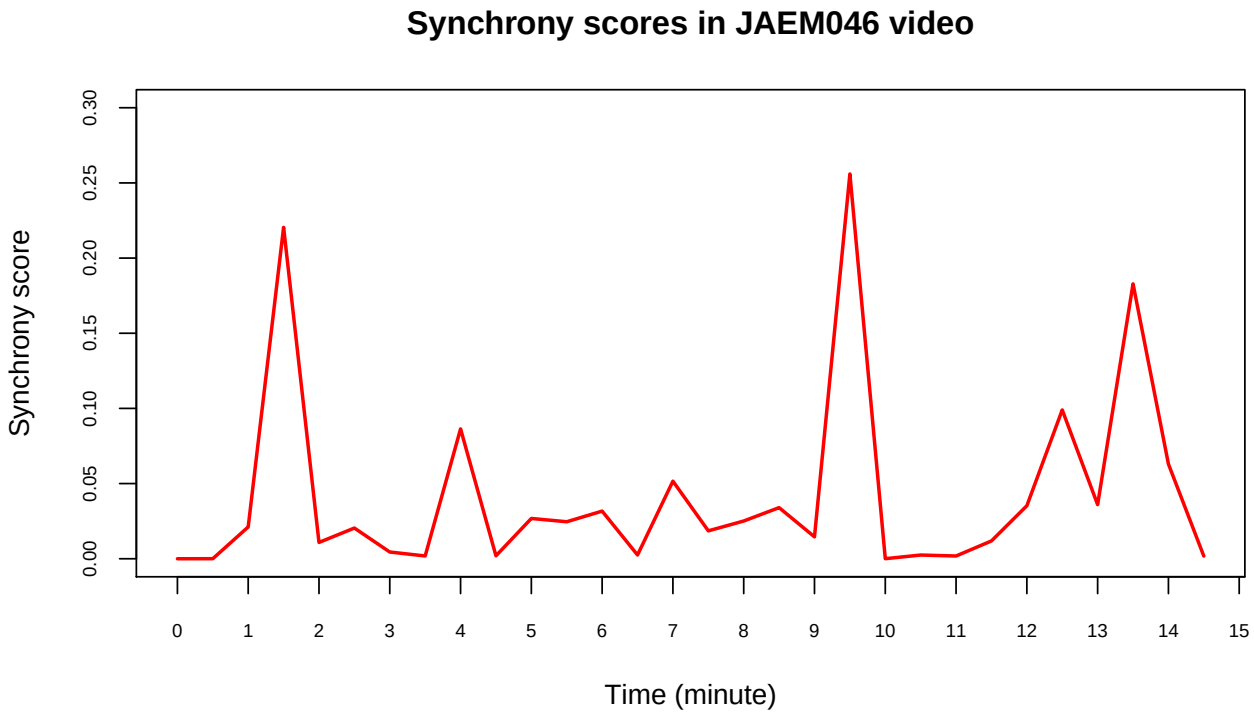

Synchrony scores in DRNE video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
Time (minute)

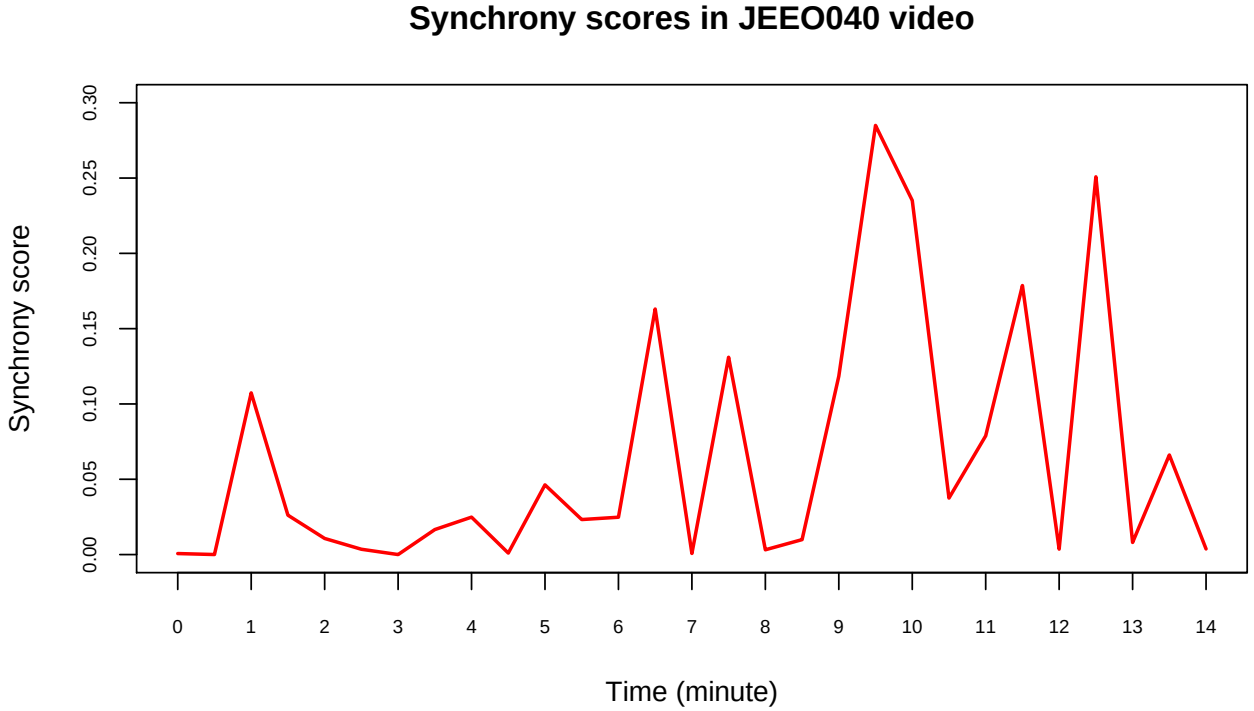

Synchrony scores in FOMA057 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)

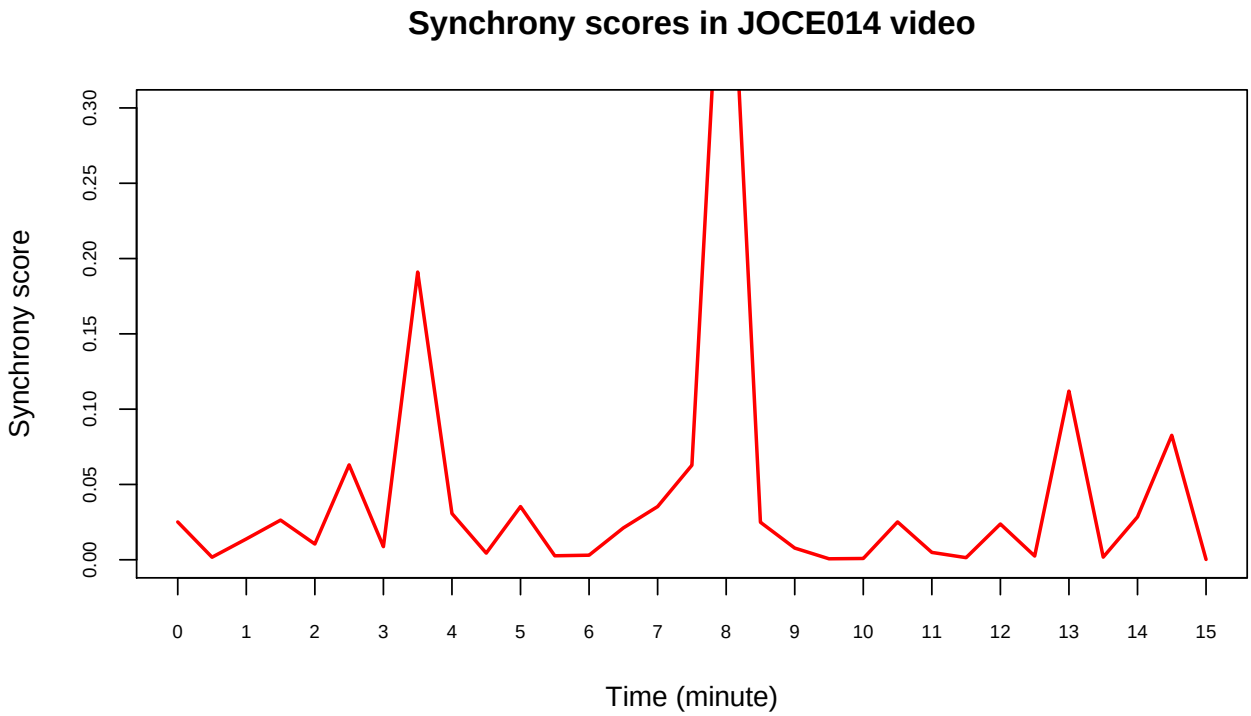

Synchrony scores in GROP039 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)

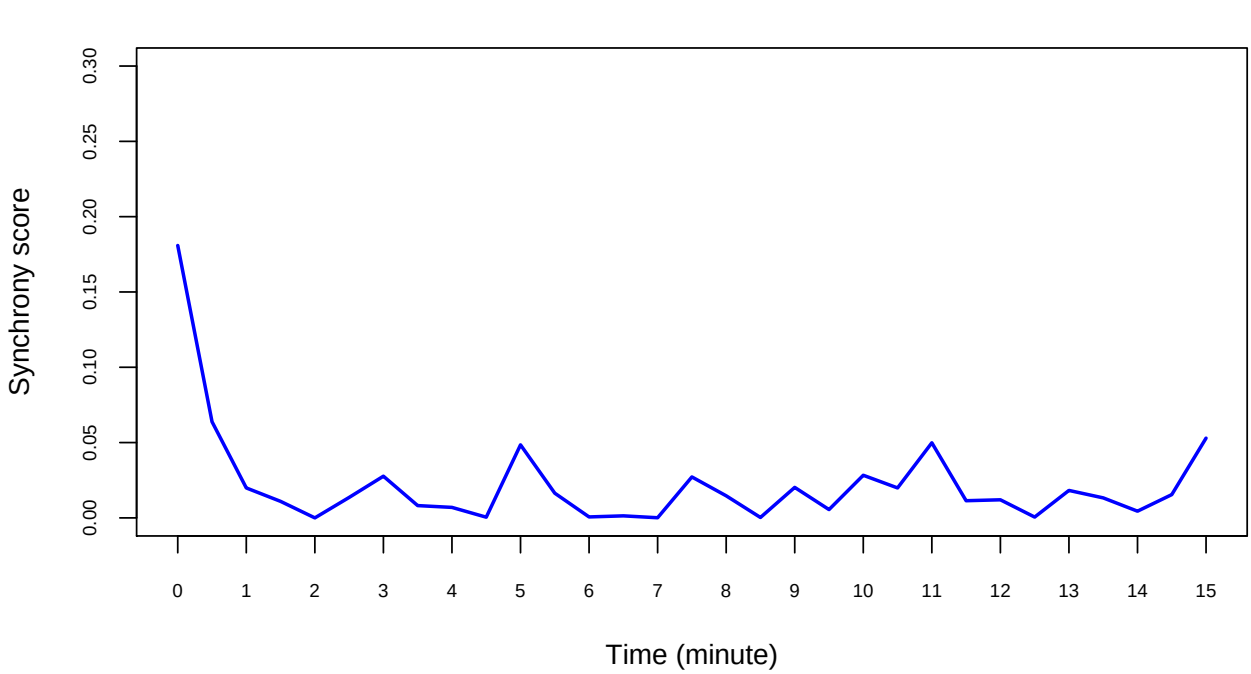

Synchrony scores in HAJA052 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)

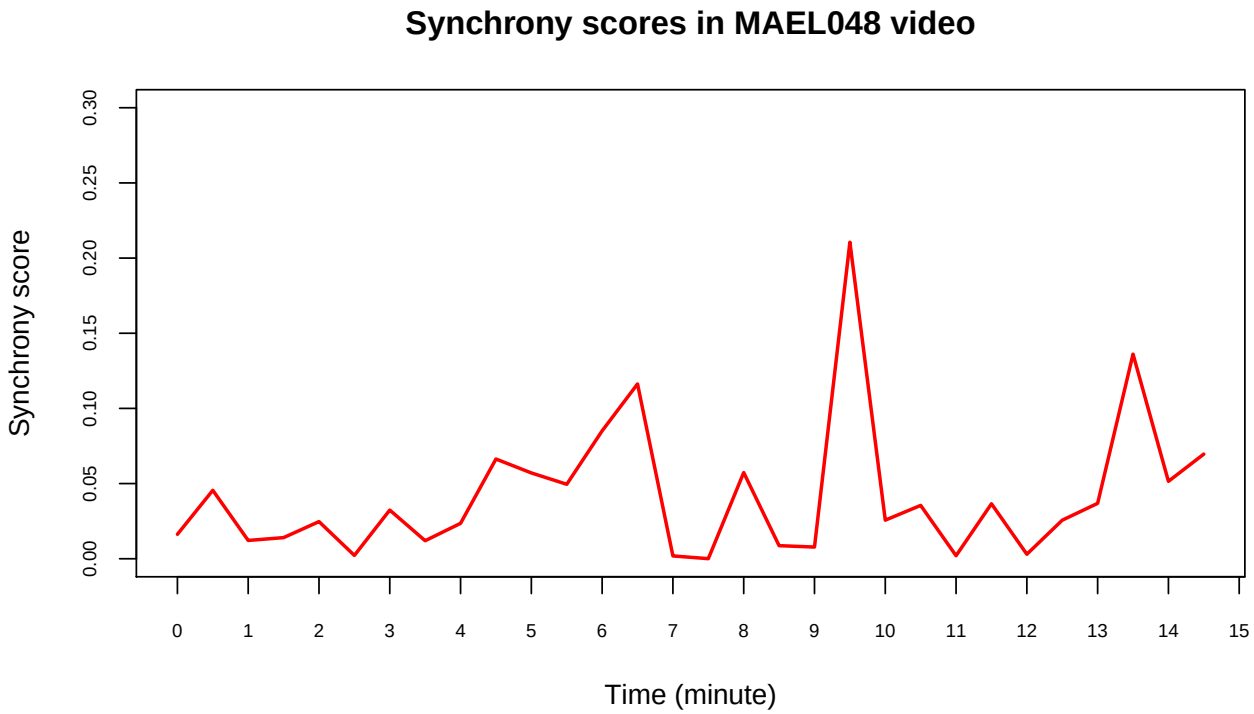

Synchrony scores in HUMA058 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
Time (minute)

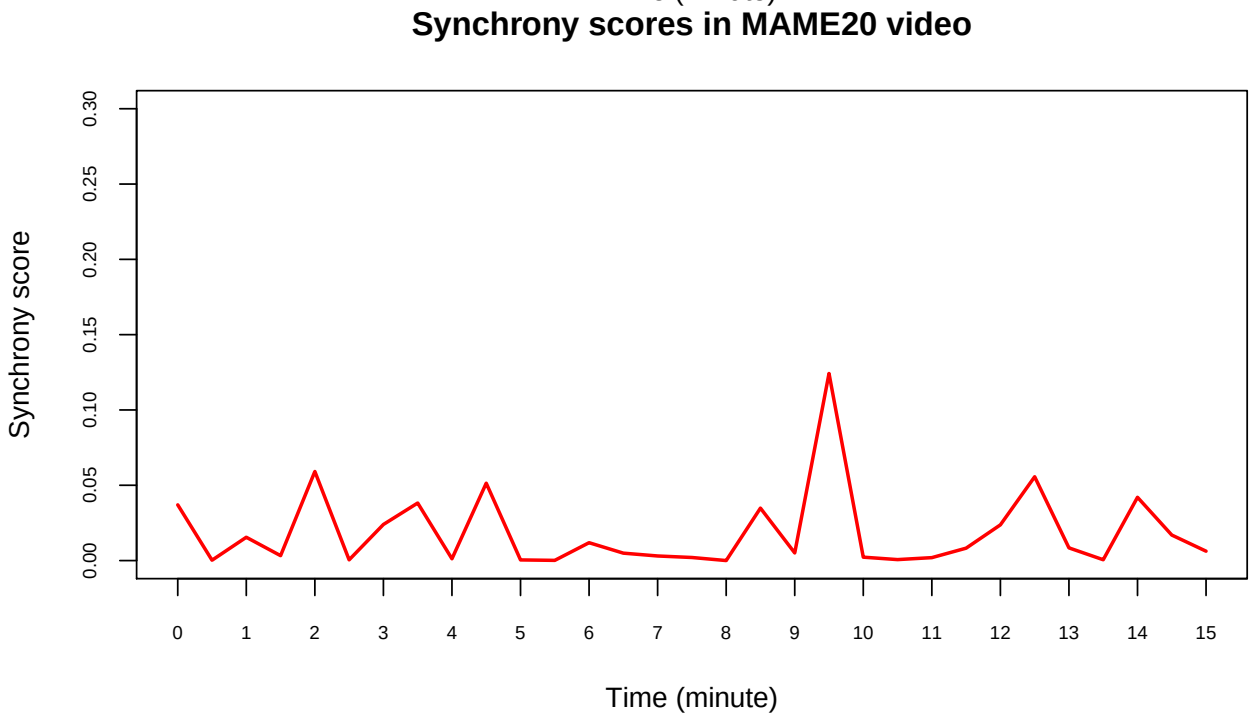

Synchrony scores in JAEM046 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)

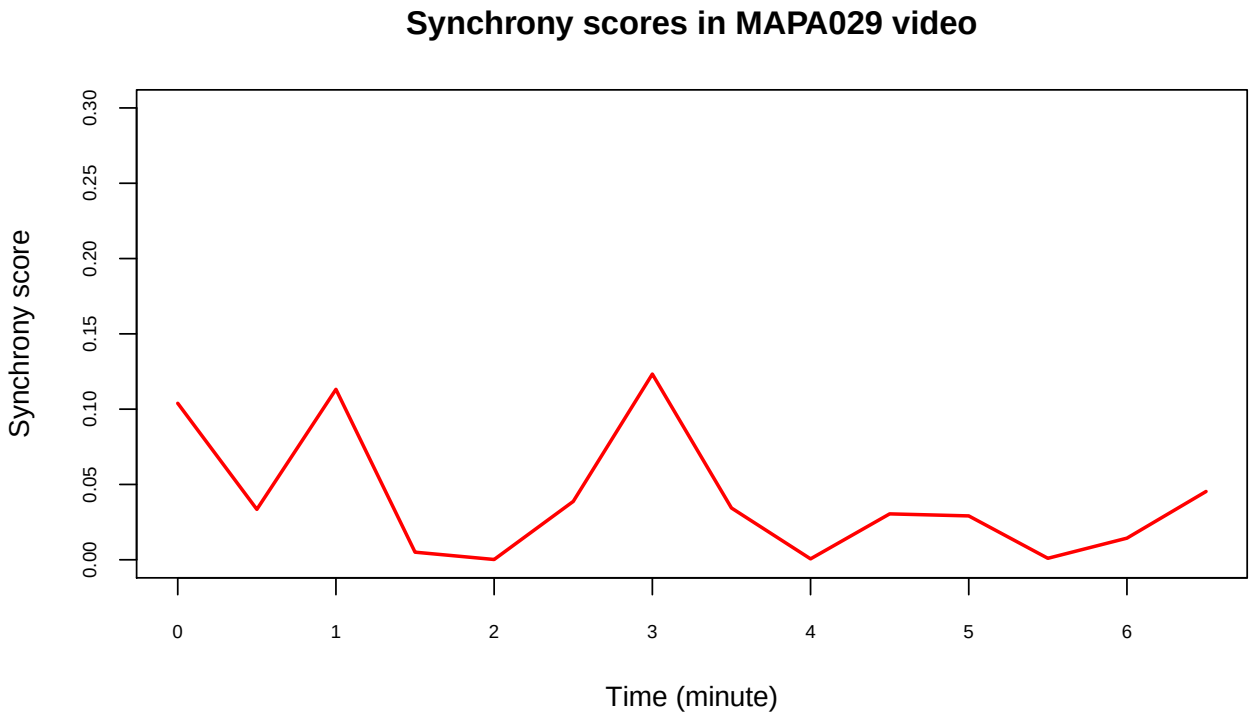

Synchrony scores in JEEO040 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
Time (minute)

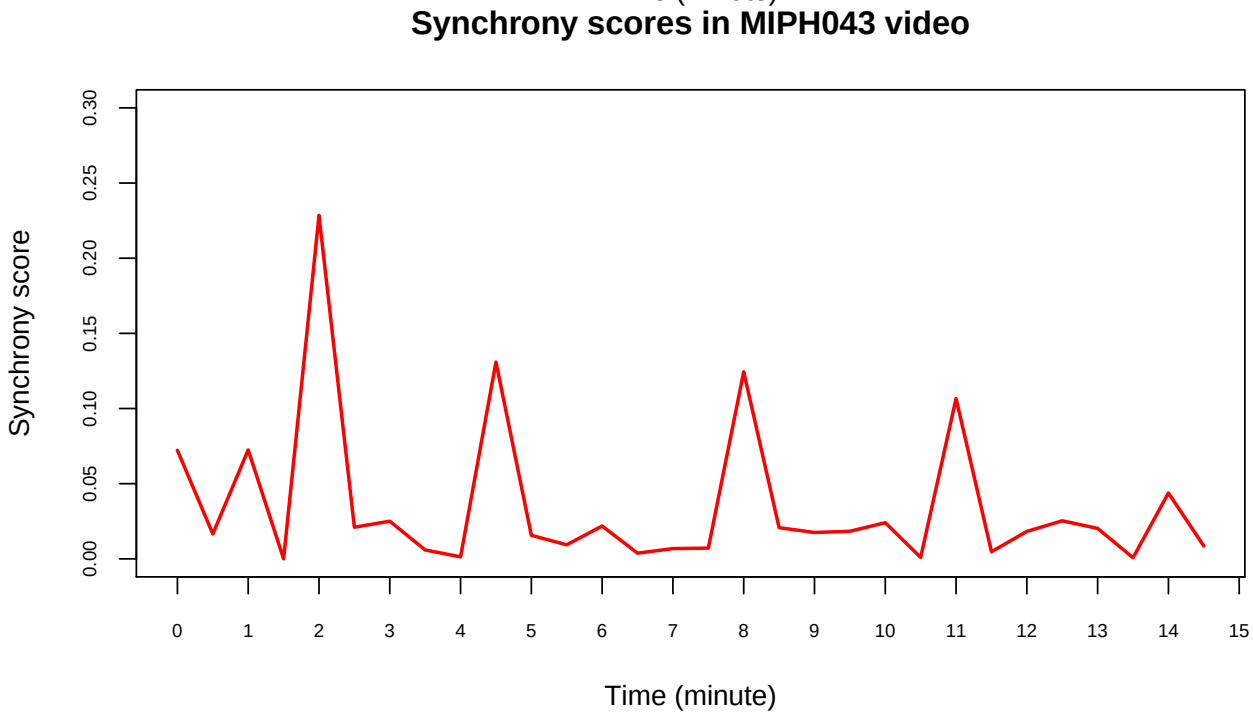

Synchrony scores in JOCE014 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)

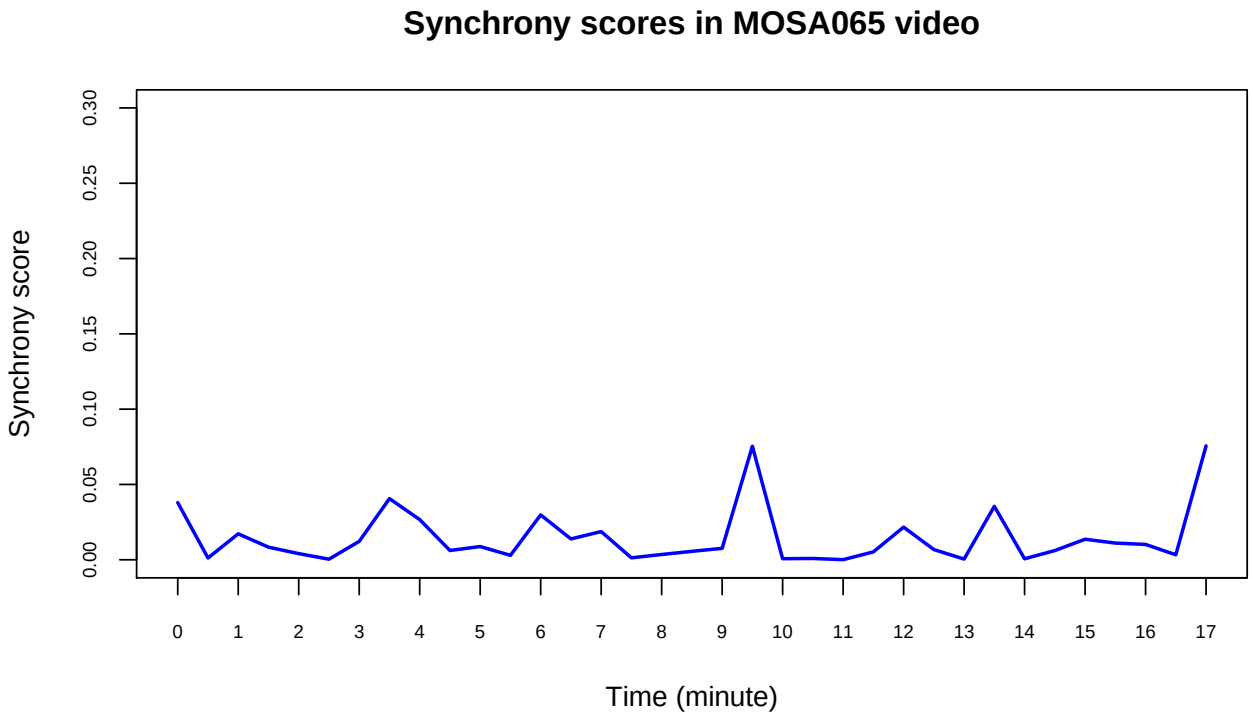

Synchrony scores in LACL video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)

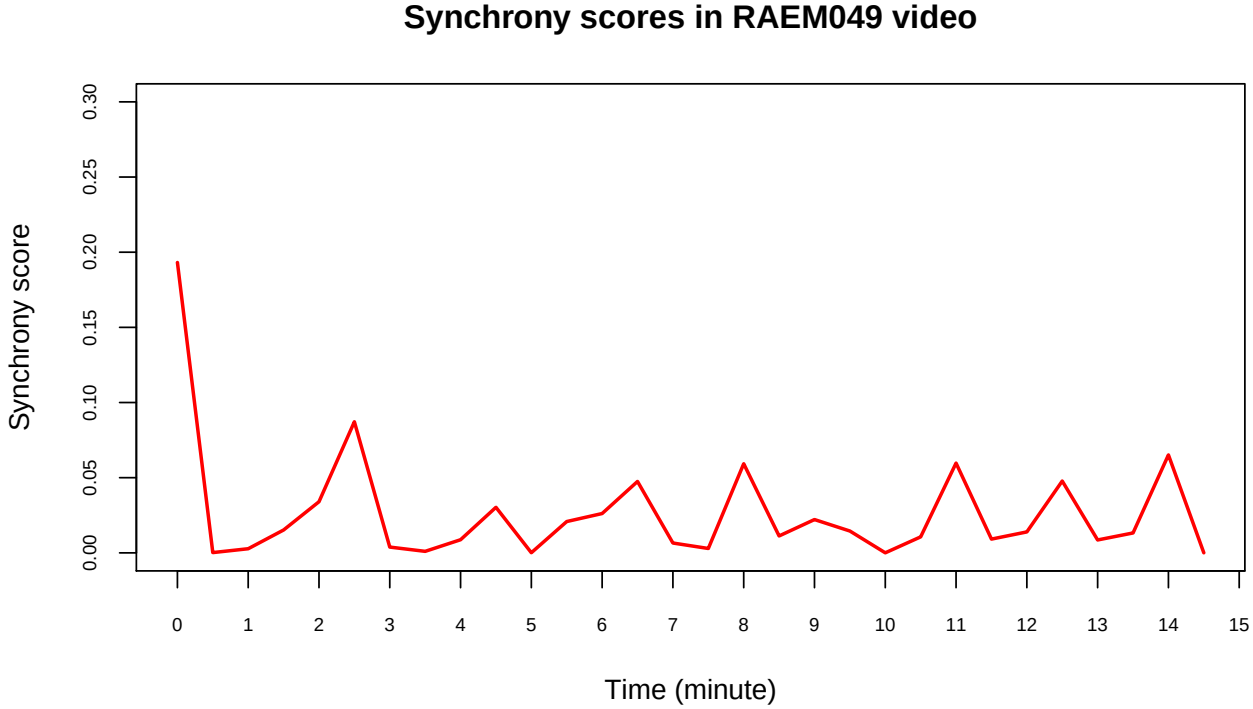

Synchrony scores in MAEL048 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)

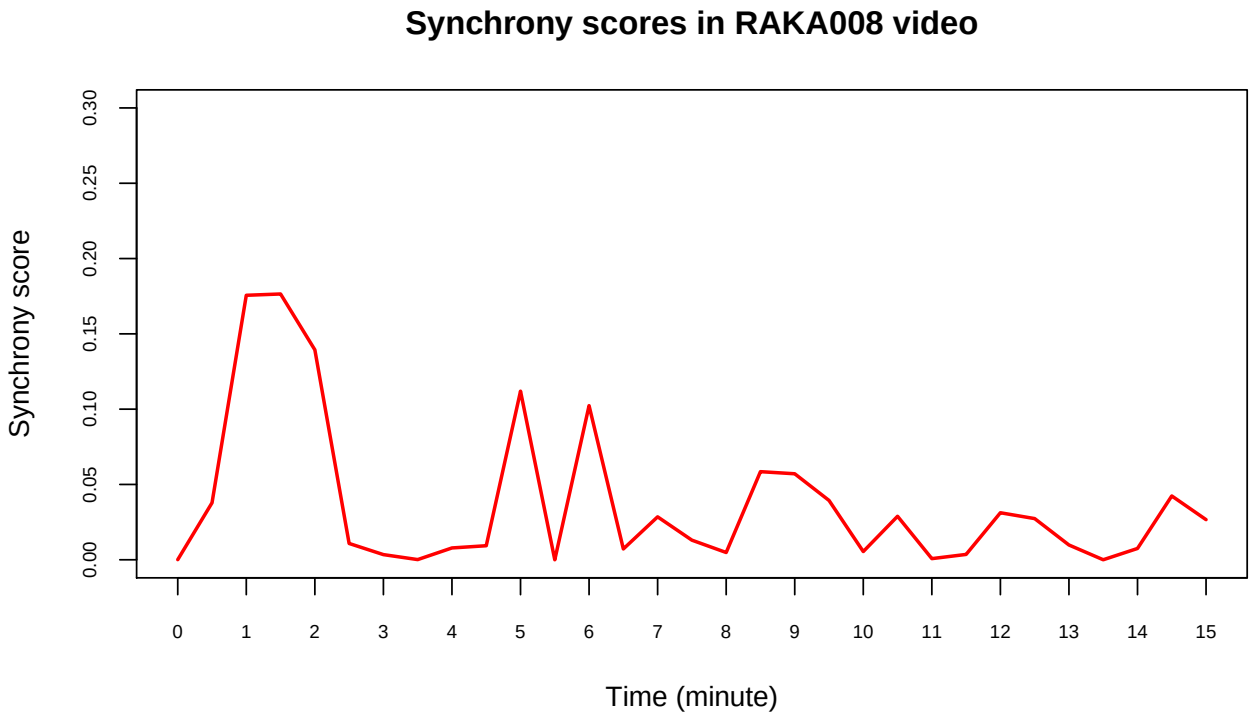

Synchrony scores in MAME20 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)

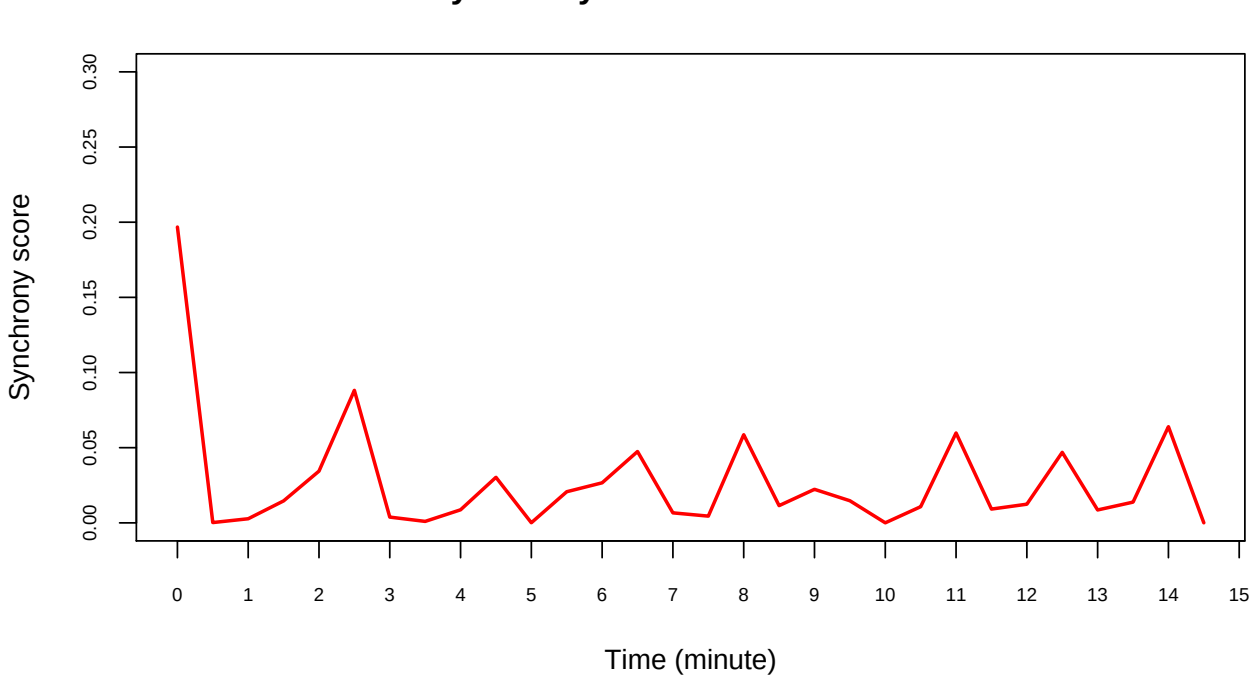

Synchrony scores in MAPA029 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
Time (minute)

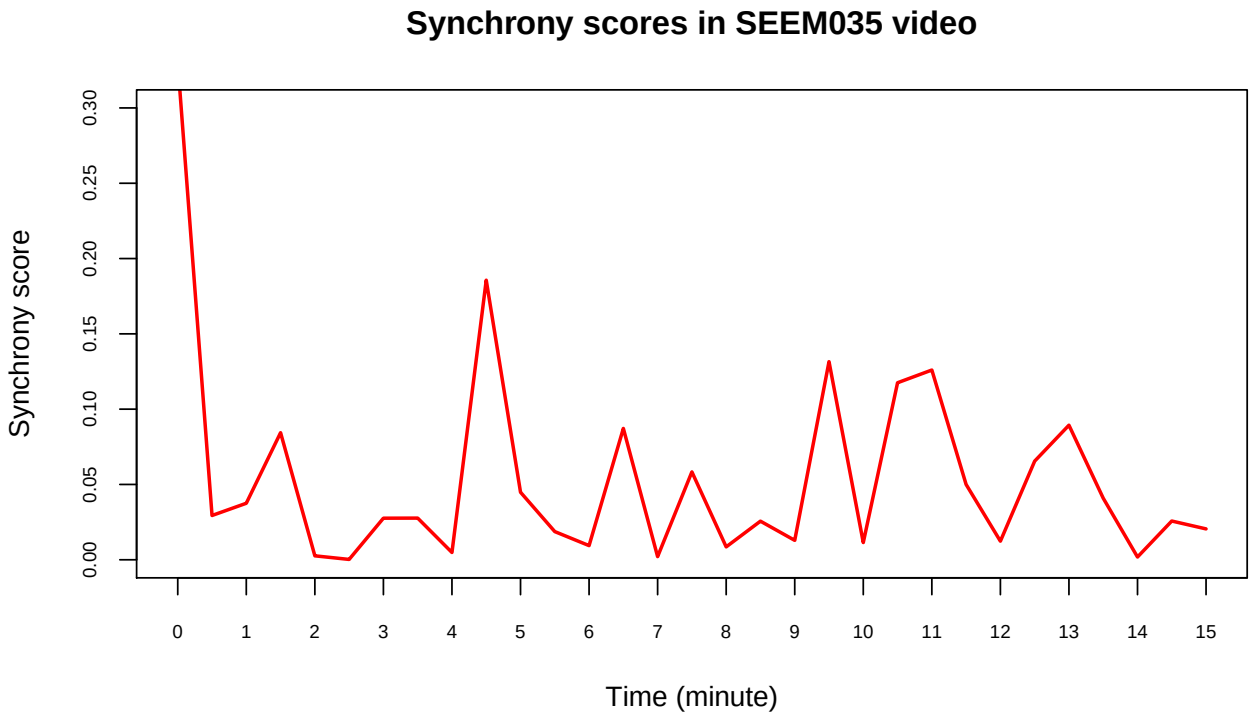

Synchrony scores in MIPH043 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)

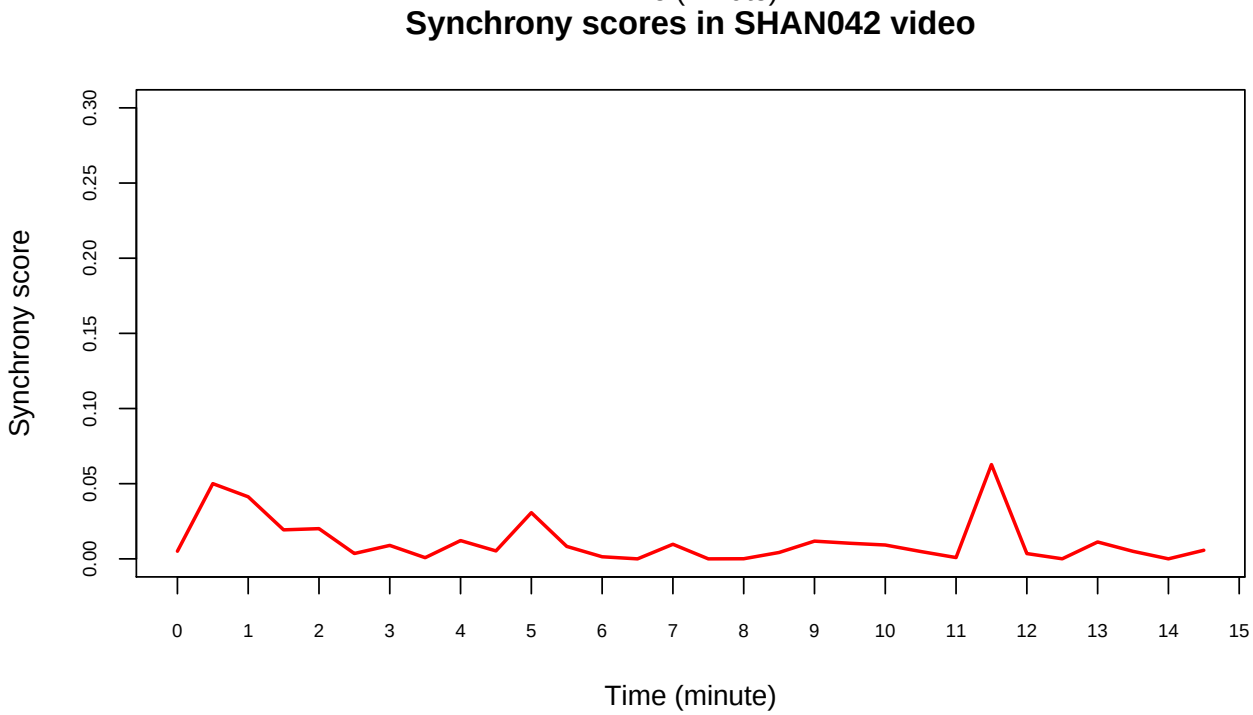

Synchrony scores in MOSA065 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
Time (minute)

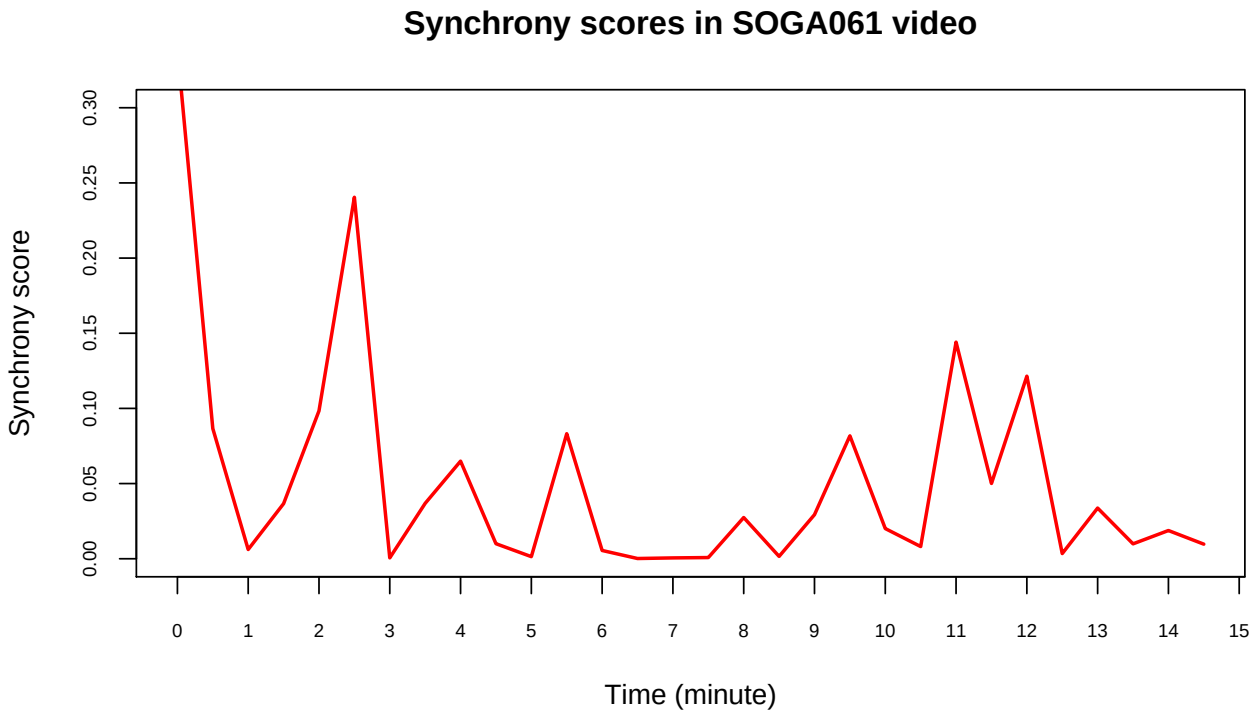

Synchrony scores in RAEM049 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)

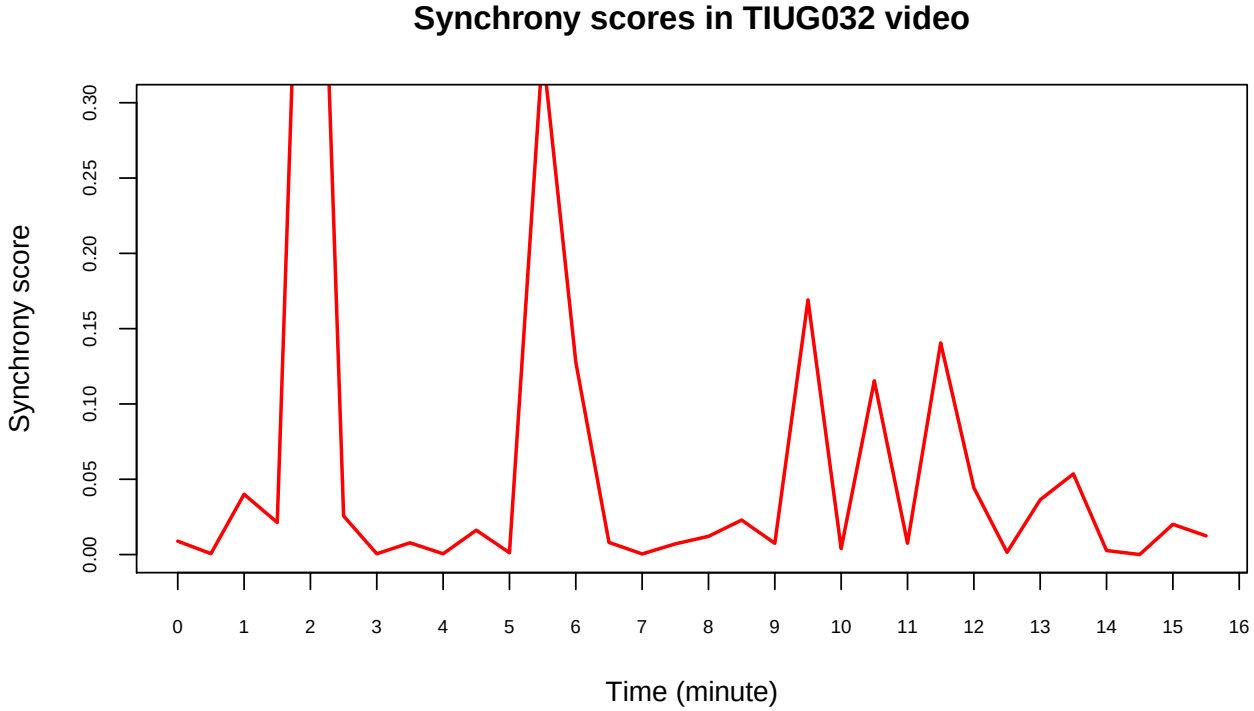

Synchrony scores in RAKA008 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)

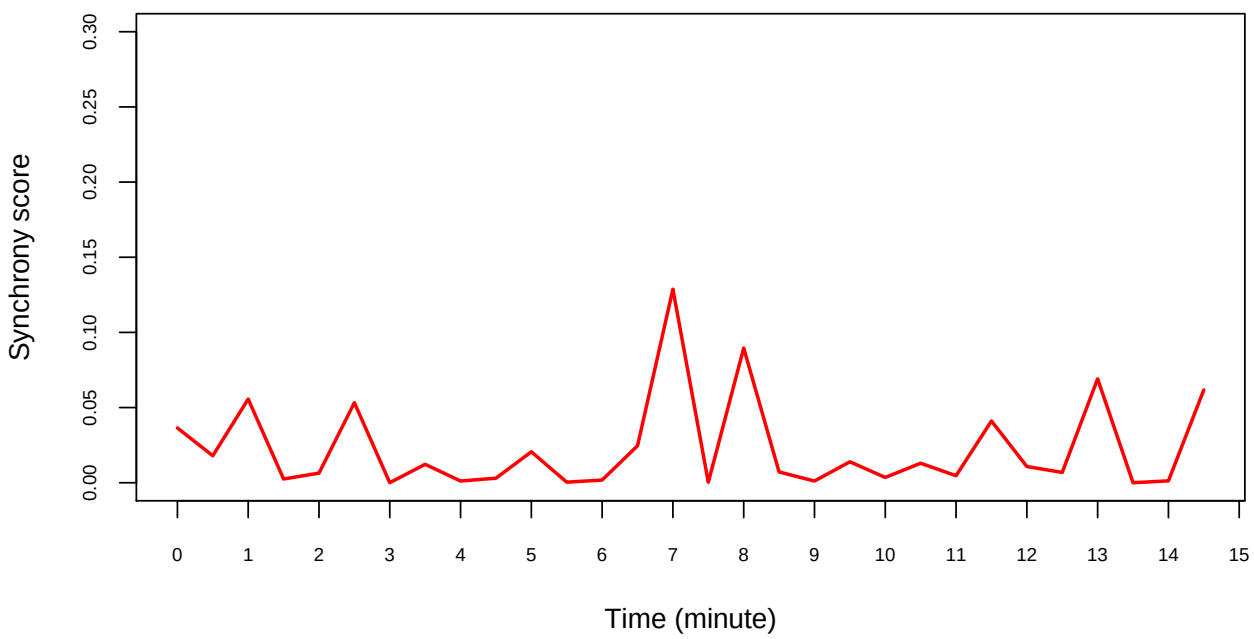

Synchrony scores in RIEM0 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)


Synchrony scores in SEEM035 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)


Synchrony scores in SHAN042 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Time (minute)


Synchrony scores in SOGA061 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
Time (minute)


Synchrony scores in TIUG032 video
Synchrony score
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
Time (minute)




```
#par(mar=c(4,4,4,3), mfrow=c(1,1))
#for (i in unique(SSInoLog$video))
#      {plot(SSInoLog[which(SSInoLog$video==i),]$Time_min,
#SSInoLog[which(SSInoLog$video==i),]$SSI_mo_ch, type="l", ylim=c(0, 0.3), #col=rainbow(4)[1], main=past
#lines(SSInoLog[which(SSInoLog$video==i),]$SSI_fa_mo_ch, col=rainbow(4)[2], lwd=2)
#lines(SSInoLog[which(SSInoLog$video==i),]$SSI_fa_ch, col=rainbow(4)[3], lwd=2)
#str(SSInoLog[which(SSInoLog$video==i),]$Time_min)
#str(SSInoLog[which(SSInoLog$video==i),]$SSI_mo_ch)
#str(SSInoLog[which(SSInoLog$video==i),]$SSI_fa_mo_ch)
#str(SSInoLog[which(SSInoLog$video==i),]$SSI_fa_ch)}

#legend("topleft", inset=.05, c("fa_mo", "fa_mo_ch", "fa_ch", "mo_ch"), col=rainbow(4), cex=0.6, lwd=2)

#legend ("topright", inset=.05, c(
#      paste ("Mean fa_mo :", round(mean(SSInoLog$SSI_fa_mo, na.rm=TRUE),3)),
#      paste ("Mean fa_mo_ch :", round(mean(SSInoLog$SSI_fa_mo_ch,na.rm=TRUE),3)),
#      paste ("Mean fa_ch :", round(mean(SSInoLog$SSI_fa_ch, na.rm=TRUE),3)),
#      paste ("Mean mo_ch :", round(mean(SSInoLog$SSI_mo_ch,na.rm=TRUE),3))),
#col=rainbow(4), cex=0.5, lty=2, lwd=1)}
```

```
cutFrames <- read.csv2("../Data/CSV/Cutframes.csv")
str(cutFrames)
```

```
## 'data.frame':    39 obs. of  7 variables:
##  $ indexList : Factor w/ 39 levels "1106","1606",..: 4 5 6 7 3 1 2 8 9 10 ...
##  $ CutBefore : Factor w/ 9 levels "00:06","00:07",..: NA NA NA NA NA NA NA 4 8 3 ...
##  $ CutMiddle1: Factor w/ 21 levels "04:26","04:43",..: NA NA NA NA NA NA NA 13 1 14 ...
##  $ CutMiddle2: Factor w/ 25 levels "05:02","05:08",..: NA NA NA NA NA NA NA 18 8 22 ...
##  $ CutFinal  : Factor w/ 24 levels "14:43","15:04",..: NA NA NA NA NA NA NA 23 13 18 ...
##  $ ChildSex  : Factor w/ 2 levels "Female","Male": NA NA NA NA NA NA NA 1 1 2 ...
##  $ ParentSex : Factor w/ 2 levels "Female","Male": NA NA NA NA NA NA NA 2 1 1 ...
```

```
cutFrames$CutBefore
```

```
##  [1] <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  00:09 00:20 00:08 00:10
## [12] 00:08 00:10 00:12 00:10 00:27 00:12 00:10 00:09 00:10 00:08 00:19
## [23] 00:10 00:10 00:08 00:10 00:06 00:19 <NA>  00:12 <NA>  <NA>  <NA>
## [34] 00:08 00:08 00:08 00:09 00:09 00:07
## Levels: 00:06 00:07 00:08 00:09 00:10 00:12 00:19 00:20 00:27
```

```
cutFramesCB <- strptime(cutFrames$CutBefore, format="%M:%S")
cutFrames$CutBefore <- as.character(cutFrames$CutBefore)
cutFrames$CutBefore
```

```
##  [1] NA      NA      NA      NA      NA      NA      NA      "00:09"
##  [9] "00:20" "00:08" "00:10" "00:08" "00:10" "00:12" "00:10" "00:27"
## [17] "00:12" "00:10" "00:09" "00:10" "00:08" "00:19" "00:10" "00:10"
## [25] "00:08" "00:10" "00:06" "00:19" NA      "00:12" NA      NA
## [33] NA      "00:08" "00:08" "00:08" "00:09" "00:09" "00:07"
```

```
cutFramesCB <- strsplit(cutFrames$CutBefore, split=":")

Cut <- c()
for (i in 1:nrow(cutFrames)){
  CutBeforeAlone <- (as.numeric(cutFramesCB[i][[1]][1]) + as.numeric(cutFramesCB[i][[1]][2])/60)
  Cut <- c(Cut, CutBeforeAlone)
  }
cutFrames$CutBeforeMin <- Cut
Cut <- c()
View(cutFrames)

Cut <- c()
cutFrames$CutMiddle1 <- as.character(cutFrames$CutMiddle1)
cutMiddleSplit <- strsplit(cutFrames$CutMiddle1, split=":")
for (i in 1:nrow(cutFrames)){
  CutAlone <- (as.numeric(cutMiddleSplit[i][[1]][1]) + as.numeric(cutMiddleSplit[i][[1]][2])/60)
  Cut <- c(Cut, CutAlone)
  print(CutAlone)
  }
```

```
## [1] NA
## [1] NA
## [1] NA
## [1] NA
## [1] NA
## [1] NA
## [1] NA
## [1] 5.316667
## [1] 4.433333
## [1] 5.333333
## [1] 5.333333
## [1] 4.933333
## [1] 5
## [1] 5.133333
## [1] 5.366667
## [1] 5.383333
## [1] 5.466667
```

```
## [1] 5.366667
## [1] 5.3
## [1] 5.133333
## [1] 5.266667
## [1] 5.566667
## [1] 5.05
## [1] 4.716667
## [1] 5.133333
## [1] 5.133333
## [1] 5.016667
## [1] 5.75
## [1] NA
## [1] 5.5
## [1] NA
## [1] NA
## [1] NA
## [1] 4.816667
## [1] 5.033333
## [1] 5.033333
## [1] 5.016667
## [1] 5.516667
## [1] 4.883333
```

```
cutFrames$CutMiddle1Min <- Cut
Cut <- c()
cutFrames$CutMiddle1
```

```
##  [1] NA      NA      NA      NA      NA      NA      NA      "05:19"
##  [9] "04:26" "05:20" "05:20" "04:56" "05:00" "05:08" "05:22" "05:23"
## [17] "05:28" "05:22" "05:18" "05:08" "05:16" "05:34" "05:03" "04:43"
## [25] "05:08" "05:08" "05:01" "05:45" NA      "05:30" NA      NA
## [33] NA      "04:49" "05:02" "05:02" "05:01" "05:31" "04:53"
```

```
cutFrames$CutMiddle1Min
```

```
##  [1]       NA       NA       NA       NA       NA       NA       NA
##  [8] 5.316667 4.433333 5.333333 5.333333 4.933333 5.000000 5.133333
## [15] 5.366667 5.383333 5.466667 5.366667 5.300000 5.133333 5.266667
## [22] 5.566667 5.050000 4.716667 5.133333 5.133333 5.016667 5.750000
## [29]       NA 5.500000       NA       NA       NA 4.816667 5.033333
## [36] 5.033333 5.016667 5.516667 4.883333
```

```
Cut <- c()
cutFrames$CutMiddle2 <- as.character(cutFrames$CutMiddle2)
cutMiddleSplit <- strsplit(cutFrames$CutMiddle2, split=":")
for (i in 1:nrow(cutFrames)){
  CutAlone <- (as.numeric(cutMiddleSplit[i][[1]][1]) + as.numeric(cutMiddleSplit[i][[1]][2])/60)
  Cut <- c(Cut, CutAlone)
  print(CutAlone)
  }
```

```
## [1] NA
## [1] NA
## [1] NA
## [1] NA
## [1] NA
```

```
## [1] NA
## [1] NA
## [1] 5.933333
## [1] 5.483333
## [1] 6.116667
## [1] 5.666667
## [1] 5.133333
## [1] 5.533333
## [1] 5.933333
## [1] 5.8
## [1] 6.15
## [1] 5.866667
## [1] 5.983333
## [1] 5.783333
## [1] 5.55
## [1] 6.033333
## [1] 6.1
## [1] 5.233333
## [1] 5.033333
## [1] 5.483333
## [1] 5.6
## [1] 5.733333
## [1] 6.566667
## [1] NA
## [1] 6.266667
## [1] NA
## [1] NA
## [1] NA
## [1] 5.316667
## [1] 5.283333
## [1] 5.183333
## [1] 5.583333
## [1] 5.983333
## [1] 5.333333
```

```
cutFrames$CutMiddle2Min <- Cut
Cut <- c()
cutFrames$CutMiddle2
```

```
##  [1] NA      NA      NA      NA      NA      NA      NA      "05:56"
##  [9] "05:29" "06:07" "05:40" "05:08" "05:32" "05:56" "05:48" "06:09"
## [17] "05:52" "05:59" "05:47" "05:33" "06:02" "06:06" "05:14" "05:02"
## [25] "05:29" "05:36" "05:44" "06:34" NA      "06:16" NA      NA
## [33] NA      "05:19" "05:17" "05:11" "05:35" "05:59" "05:20"
```

```
cutFrames$CutMiddle2Min
```

```
##  [1]       NA       NA       NA       NA       NA       NA       NA
##  [8] 5.933333 5.483333 6.116667 5.666667 5.133333 5.533333 5.933333
## [15] 5.800000 6.150000 5.866667 5.983333 5.783333 5.550000 6.033333
## [22] 6.100000 5.233333 5.033333 5.483333 5.600000 5.733333 6.566667
## [29]       NA 6.266667       NA       NA       NA 5.316667 5.283333
## [36] 5.183333 5.583333 5.983333 5.333333
```

```
Cut <- c()
cutFrames$CutFinal <- as.character(cutFrames$CutFinal)
```

```
cutSplit <- strsplit(cutFrames$CutFinal, split=":")
for (i in 1:nrow(cutFrames)){
  CutAlone <- (as.numeric(cutSplit[i][[1]][1]) + as.numeric(cutSplit[i][[1]][2])/60)
  Cut <- c(Cut, CutAlone)
  print(CutAlone)
  }
```

```
## [1] NA
## [1] NA
## [1] NA
## [1] NA
## [1] NA
## [1] NA
## [1] NA
## [1] 16.73333
## [1] 15.5
## [1] 16.08333
## [1] 15.46667
## [1] 15.13333
## [1] 15.21667
## [1] 15.3
## [1] 16.16667
## [1] 16.1
## [1] 16.41667
## [1] 16.06667
## [1] 15.5
## [1] 15.41667
## [1] 15.5
## [1] 16.83333
## [1] 15.11667
## [1] 14.71667
## [1] 15.35
## [1] 15.25
## [1] 15.06667
## [1] 15.9
## [1] NA
## [1] 16.43333
## [1] NA
## [1] NA
## [1] NA
## [1] 15.16667
## [1] 15.56667
## [1] 15.3
## [1] 15.21667
## [1] 15.88333
## [1] 15.1
```

```
cutFrames$CutFinalMin <- Cut
Cut <- c()
cutFrames$CutFinal
```

```
##  [1] NA      NA      NA      NA      NA      NA      NA      "16:44"
##  [9] "15:30" "16:05" "15:28" "15:08" "15:13" "15:18" "16:10" "16:06"
## [17] "16:25" "16:04" "15:30" "15:25" "15:30" "16:50" "15:07" "14:43"
## [25] "15:21" "15:15" "15:04" "15:54" NA      "16:26" NA      NA
```

```
## [33] NA      "15:10" "15:34" "15:18" "15:13" "15:53" "15:06"
```

```
cutFrames$CutFinalMin
```

```
##  [1]       NA       NA       NA       NA       NA       NA       NA
##  [8] 16.73333 15.50000 16.08333 15.46667 15.13333 15.21667 15.30000
## [15] 16.16667 16.10000 16.41667 16.06667 15.50000 15.41667 15.50000
## [22] 16.83333 15.11667 14.71667 15.35000 15.25000 15.06667 15.90000
## [29]       NA 16.43333       NA       NA       NA 15.16667 15.56667
## [36] 15.30000 15.21667 15.88333 15.10000
```

```
View(cutFrames)
```

```
str(SSInoLog)
```

```
## 'data.frame':    1200 obs. of  8 variables:
##  $ video       : Factor w/ 40 levels "1106","1606",..: 4 4 4 4 4 4 4 4 4 4 ...
##  $ X           : int  0 1 2 3 4 5 6 7 8 9 ...
##  $ Interval    : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Time_min    : num  0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 ...
##  $ SSI_mo_ch   : num  0.018627 0.024666 0.04506 0.001527 0.000817 ...
##  $ SSI_fa_ch   : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ SSI_fa_mo   : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ SSI_fa_mo_ch: num  NA NA NA NA NA NA NA NA NA NA ...
```

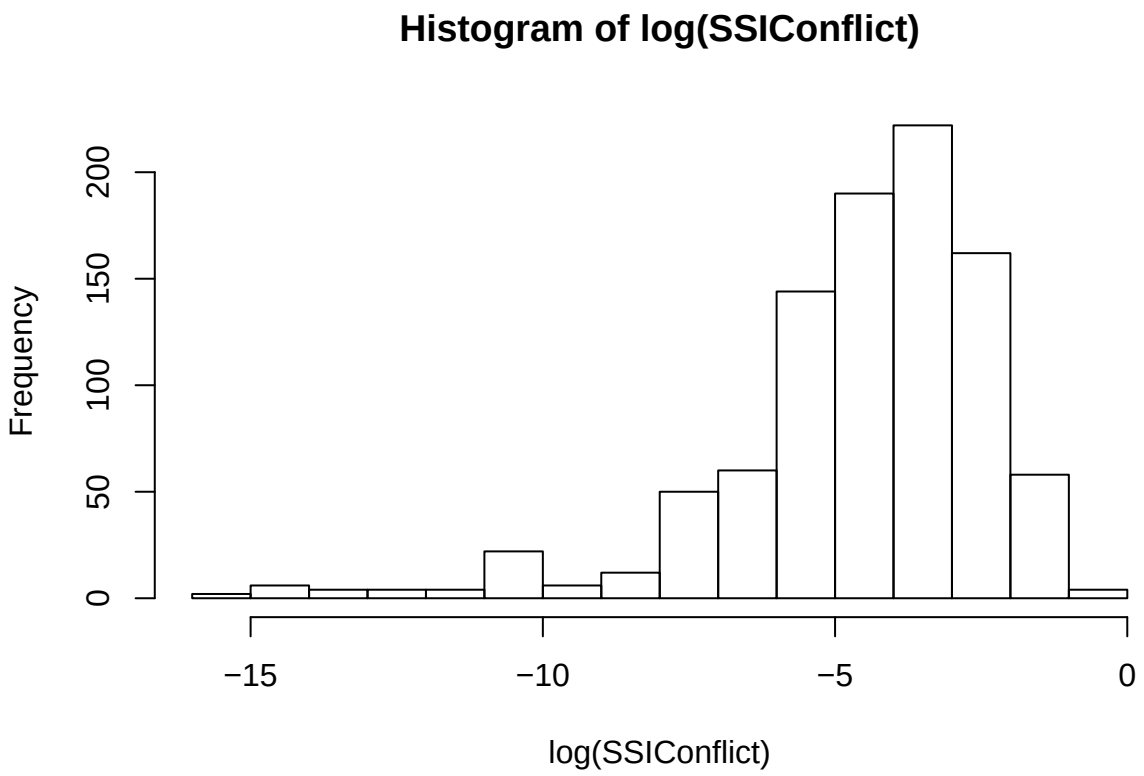
```
a <- merge(SSInoLog, cutFrames, by.x="video", by.y="indexList")
#View(a)
a$LabelVideo <- rep(NA, nrow(a))
a[which(a$Time_min < a$CutBeforeMin),]$LabelVideo <- "Cut"

a[which(a$Time_min > a$CutBeforeMin & a$Time_min < a$CutMiddle1Min),]$LabelVideo <- "No Conflict"

a[which(a$Time_min > a$CutMiddle1Min & a$Time_min < a$CutMiddle2Min),]$LabelVideo <- "Cut"

a[which(a$Time_min > a$CutMiddle2Min & a$Time_min < a$CutFinalMin),]$LabelVideo <- "Conflict"

#a[which(a$Time_min >= a$CutFinalMin),]$LabelVideo <- "Cut"

SSIConflict <- c(a[which(a$LabelVideo=="Conflict"),]$SSI_mo_ch, a[which(a$LabelVideo=="Conflict"),]$SSI
mean(SSIConflict, na.rm=TRUE)
```

```
## [1] 0.03921076
```

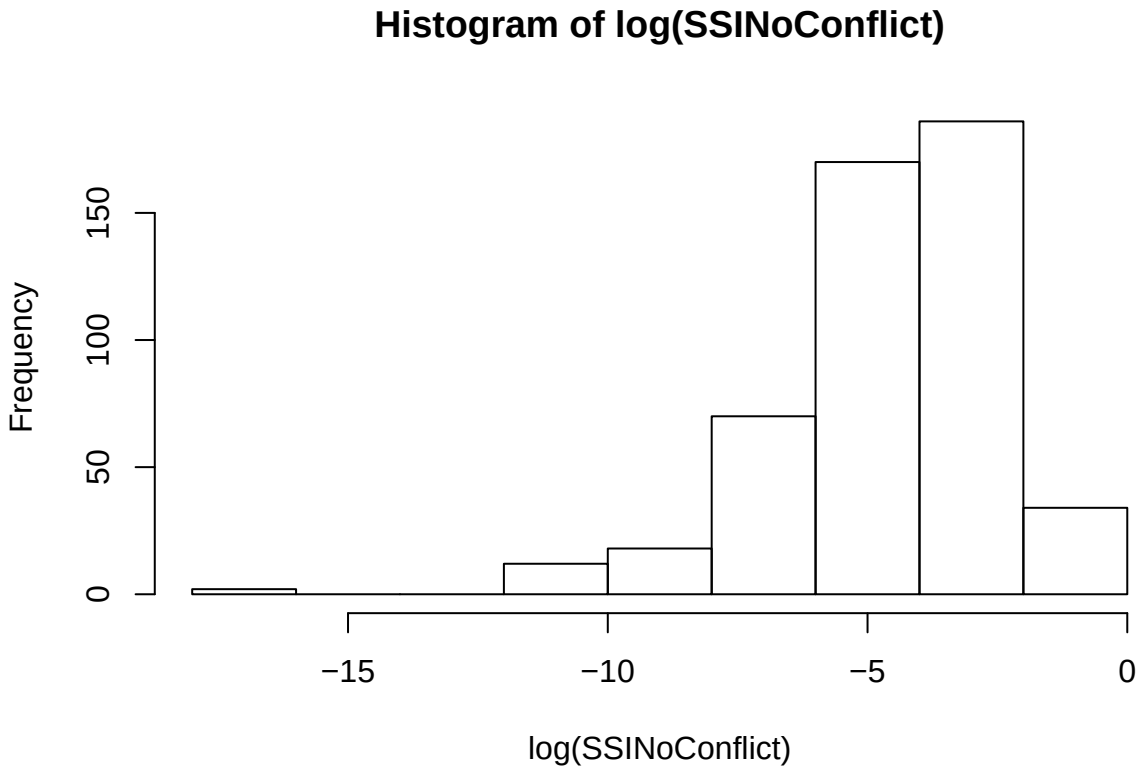
```
SSINoConflict <- c(a[which(a$LabelVideo=="No Conflict"),]$SSI_mo_ch, a[which(a$LabelVideo=="No Conflict
mean(SSINoConflict, na.rm=TRUE)
```

```
## [1] 0.04026161
```

```
hist(log(SSIConflict))
```



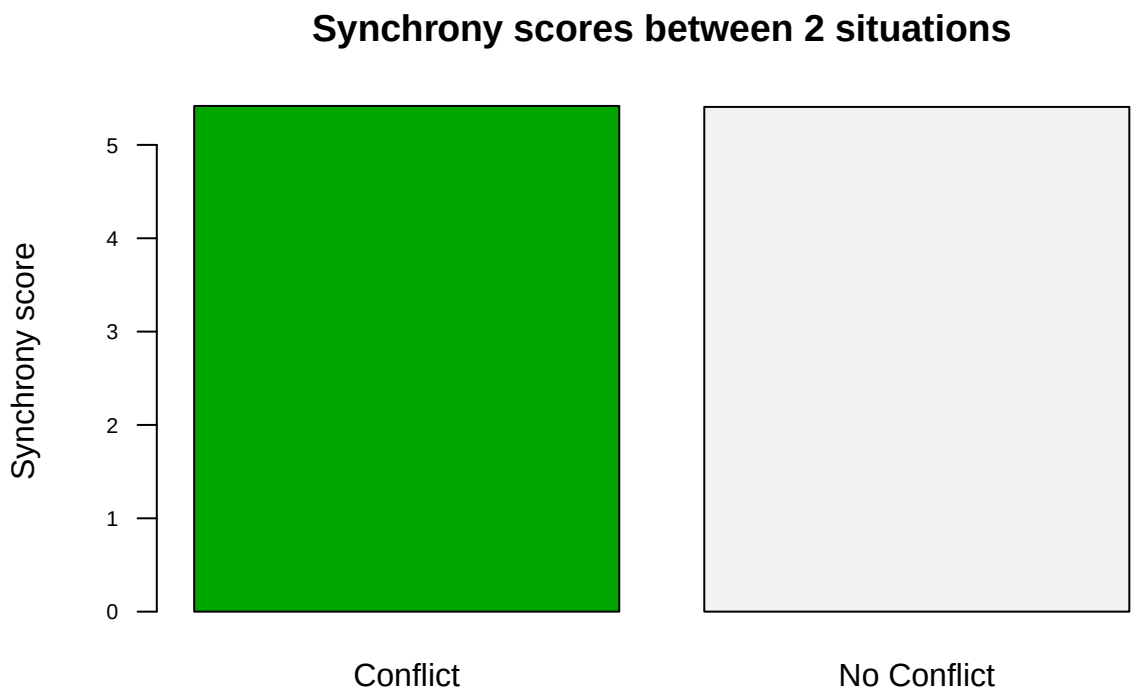

```
hist(log(SSINoConflict))
```



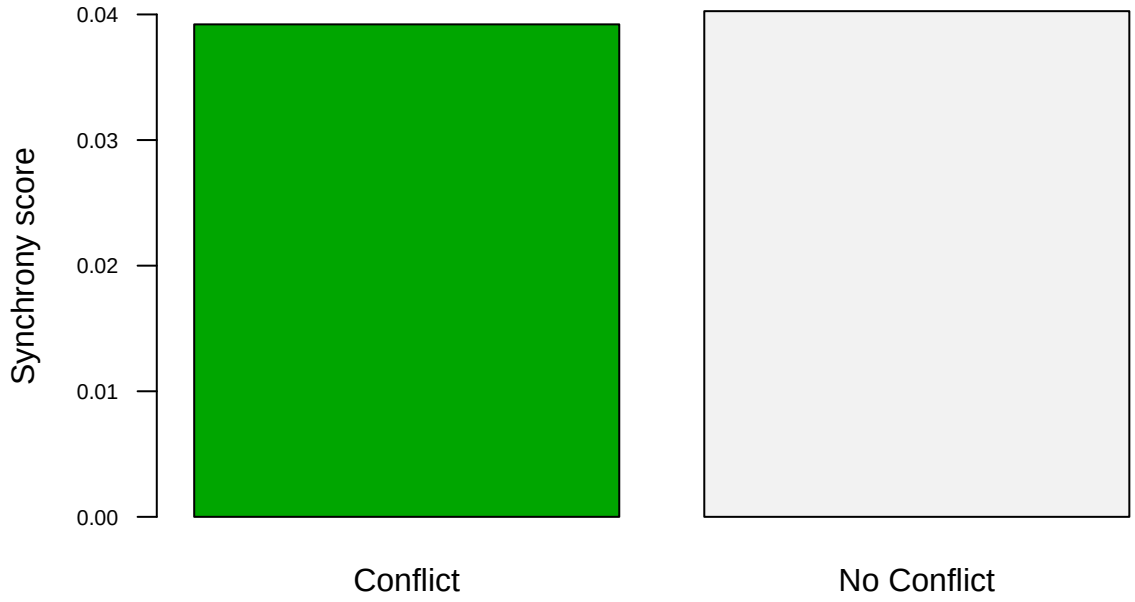

```
mean(log(SSIConflict), na.rm=TRUE)
```

```
## [1] -4.582897
```

```
barplot(c(mean(log(SSIConflict), na.rm=TRUE)+10, mean(log(SSINoConflict), na.rm=TRUE)+10), names=c("Con
```




```
barplot(c(mean(SSIConflict, na.rm=TRUE), mean(SSINoConflict, na.rm=TRUE)), names=c("Conflict", "No Confl
```




```
df <- data.frame(NA, NA)
for (i in unique(a$video)){
      print (i)
      b <- mean(a[which(a$LabelVideo=="No Conflict" & a$video==i),]$SSI_mo_ch)
      print(b)
      e <- mean(a[which(a$LabelVideo=="Conflict" & a$video==i),]$SSI_mo_ch)
```

```
      print(c)
      d <- c(b, e)
      df <- cbind(df,d)
      print(df)}
```

```
## [1] "1106"
## [1] NaN
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d
## 1  NA    NA NaN
## 2  NA    NA NaN
## [1] "1606"
## [1] NaN
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d
## 1  NA    NA NaN NaN
## 2  NA    NA NaN NaN
## [1] "206"
## [1] NaN
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d
## 1  NA    NA NaN NaN NaN
## 2  NA    NA NaN NaN NaN
## [1] "34"
## [1] NaN
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d
## 1  NA    NA NaN NaN NaN NaN
## 2  NA    NA NaN NaN NaN NaN
## [1] "37"
## [1] NaN
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d
## 1  NA    NA NaN NaN NaN NaN NaN
## 2  NA    NA NaN NaN NaN NaN NaN
## [1] "41"
## [1] NaN
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d
## 1  NA    NA NaN NaN NaN NaN NaN NaN
## 2  NA    NA NaN NaN NaN NaN NaN NaN
## [1] "48"
## [1] NaN
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN
## [1] "BAJE059"
## [1] NA
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA
## [1] "BALE050"
```

```
## [1] 0.04432081
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307
## [1] "BALU062"
## [1] 0.01886765
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
## [1] "BEAL036"
## [1] 0.05482111
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d
## 1 0.05482111
## 2 0.03528263
## [1] "BEAM031"
## [1] 0.07269493
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d
## 1 0.05482111 0.07269493
## 2 0.03528263 0.07676492
## [1] "BICA"
## [1] 0.04543333
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d
## 1 0.05482111 0.07269493 0.04543333
## 2 0.03528263 0.07676492 0.05129765
## [1] "BRLO041"
## [1] 0.0134373
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730
## 2 0.03528263 0.07676492 0.05129765 0.02978409
## [1] "COLO022"
## [1] 0.05563912
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d          d          d
```

```
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557
## [1] "DIPE004"
## [1] 0.0535899
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
## [1] "DOMA"
## [1] 0.0195961
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##            d
## 1 0.01959610
## 2 0.02287883
## [1] "DRNE"
## [1] 0.04255482
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##            d          d
## 1 0.01959610 0.04255482
## 2 0.02287883 0.02599562
## [1] "FOMA057"
## [1] NA
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##            d          d  d
## 1 0.01959610 0.04255482 NA
## 2 0.02287883 0.02599562 NA
## [1] "GROP039"
## [1] 0.03022675
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d          d          d          d
```

```
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##           d          d  d          d
## 1 0.01959610 0.04255482 NA 0.03022675
## 2 0.02287883 0.02599562 NA 0.03701956
## [1] "HAJA052"
## [1] 0.0690037
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##           d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##           d          d  d          d          d
## 1 0.01959610 0.04255482 NA 0.03022675 0.06900370
## 2 0.02287883 0.02599562 NA 0.03701956 0.04339679
## [1] "HUMA058"
## [1] 0.02300349
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##           d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##           d          d  d          d          d          d
## 1 0.01959610 0.04255482 NA 0.03022675 0.06900370 0.02300349
## 2 0.02287883 0.02599562 NA 0.03701956 0.04339679 0.02265999
## [1] "JAEM046"
## [1] 0.03943233
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##           d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##           d          d  d          d          d          d          d
## 1 0.01959610 0.04255482 NA 0.03022675 0.06900370 0.02300349 0.03943233
## 2 0.02287883 0.02599562 NA 0.03701956 0.04339679 0.02265999 0.04697950
## [1] "JEEO040"
## [1] 0.02113752
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##           d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##           d          d  d          d          d          d          d
## 1 0.01959610 0.04255482 NA 0.03022675 0.06900370 0.02300349 0.03943233
## 2 0.02287883 0.02599562 NA 0.03701956 0.04339679 0.02265999 0.04697950
##           d
```

```
## 1 0.02113752
## 2 0.09009637
## [1] "JOCE014"
## [1] 0.03854909
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##            d          d  d          d          d          d          d
## 1 0.01959610 0.04255482 NA 0.03022675 0.06900370 0.02300349 0.03943233
## 2 0.02287883 0.02599562 NA 0.03701956 0.04339679 0.02265999 0.04697950
##            d          d
## 1 0.02113752 0.03854909
## 2 0.09009637 0.04637298
## [1] "LACL"
## [1] NA
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##            d          d  d          d          d          d          d
## 1 0.01959610 0.04255482 NA 0.03022675 0.06900370 0.02300349 0.03943233
## 2 0.02287883 0.02599562 NA 0.03701956 0.04339679 0.02265999 0.04697950
##            d          d  d
## 1 0.02113752 0.03854909 NA
## 2 0.09009637 0.04637298 NA
## [1] "MAEL048"
## [1] 0.02900099
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##            d          d  d          d          d          d          d
## 1 0.01959610 0.04255482 NA 0.03022675 0.06900370 0.02300349 0.03943233
## 2 0.02287883 0.02599562 NA 0.03701956 0.04339679 0.02265999 0.04697950
##            d          d  d          d
## 1 0.02113752 0.03854909 NA 0.02900099
## 2 0.09009637 0.04637298 NA 0.05055375
## [1] "MAME20"
## [1] 0.01762869
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d          d          d          d
```

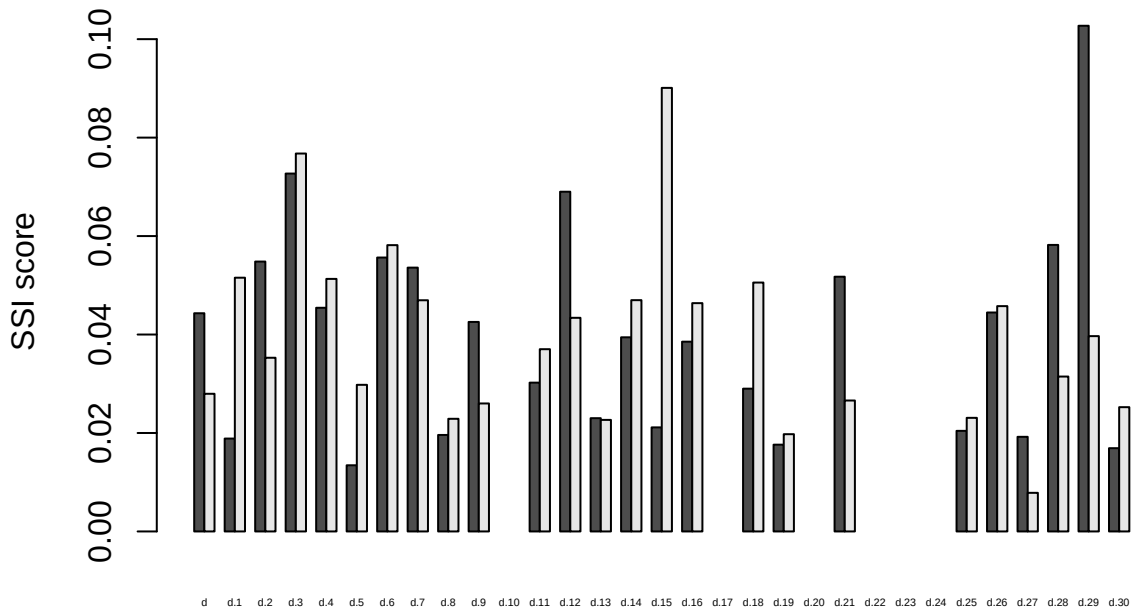
```
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##           d          d  d          d          d          d          d
## 1 0.01959610 0.04255482 NA 0.03022675 0.06900370 0.02300349 0.03943233
## 2 0.02287883 0.02599562 NA 0.03701956 0.04339679 0.02265999 0.04697950
##           d          d  d          d          d
## 1 0.02113752 0.03854909 NA 0.02900099 0.01762869
## 2 0.09009637 0.04637298 NA 0.05055375 0.01975602
## [1] "MAPA029"
## [1] NaN
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##           d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##           d          d  d          d          d          d          d
## 1 0.01959610 0.04255482 NA 0.03022675 0.06900370 0.02300349 0.03943233
## 2 0.02287883 0.02599562 NA 0.03701956 0.04339679 0.02265999 0.04697950
##           d          d  d          d          d   d
## 1 0.02113752 0.03854909 NA 0.02900099 0.01762869 NaN
## 2 0.09009637 0.04637298 NA 0.05055375 0.01975602 NaN
## [1] "MIPH043"
## [1] 0.05173998
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##           d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##           d          d  d          d          d          d          d
## 1 0.01959610 0.04255482 NA 0.03022675 0.06900370 0.02300349 0.03943233
## 2 0.02287883 0.02599562 NA 0.03701956 0.04339679 0.02265999 0.04697950
##           d          d  d          d          d   d          d
## 1 0.02113752 0.03854909 NA 0.02900099 0.01762869 NaN 0.05173998
## 2 0.09009637 0.04637298 NA 0.05055375 0.01975602 NaN 0.02659465
## [1] "MOSA065"
## [1] NaN
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##           d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##           d          d  d          d          d          d          d
## 1 0.01959610 0.04255482 NA 0.03022675 0.06900370 0.02300349 0.03943233
## 2 0.02287883 0.02599562 NA 0.03701956 0.04339679 0.02265999 0.04697950
##           d          d  d          d          d   d          d   d
## 1 0.02113752 0.03854909 NA 0.02900099 0.01762869 NaN 0.05173998 NaN
## 2 0.09009637 0.04637298 NA 0.05055375 0.01975602 NaN 0.02659465 NaN
## [1] "RAEM049"
```

```
## [1] NaN
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##            d          d  d          d          d          d          d
## 1 0.01959610 0.04255482 NA 0.03022675 0.06900370 0.02300349 0.03943233
## 2 0.02287883 0.02599562 NA 0.03701956 0.04339679 0.02265999 0.04697950
##            d          d  d          d          d   d          d   d   d
## 1 0.02113752 0.03854909 NA 0.02900099 0.01762869 NaN 0.05173998 NaN NaN
## 2 0.09009637 0.04637298 NA 0.05055375 0.01975602 NaN 0.02659465 NaN NaN
## [1] "RAKA008"
## [1] NaN
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##            d          d  d          d          d          d          d
## 1 0.01959610 0.04255482 NA 0.03022675 0.06900370 0.02300349 0.03943233
## 2 0.02287883 0.02599562 NA 0.03701956 0.04339679 0.02265999 0.04697950
##            d          d  d          d          d   d          d   d   d
## 1 0.02113752 0.03854909 NA 0.02900099 0.01762869 NaN 0.05173998 NaN NaN
## 2 0.09009637 0.04637298 NA 0.05055375 0.01975602 NaN 0.02659465 NaN NaN
##     d
## 1 NaN
## 2 NaN
## [1] "RIEM0"
## [1] 0.02043336
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##            d          d  d          d          d          d          d
## 1 0.01959610 0.04255482 NA 0.03022675 0.06900370 0.02300349 0.03943233
## 2 0.02287883 0.02599562 NA 0.03701956 0.04339679 0.02265999 0.04697950
##            d          d  d          d          d   d          d   d   d
## 1 0.02113752 0.03854909 NA 0.02900099 0.01762869 NaN 0.05173998 NaN NaN
## 2 0.09009637 0.04637298 NA 0.05055375 0.01975602 NaN 0.02659465 NaN NaN
##     d          d
## 1 NaN 0.02043336
## 2 NaN 0.02307545
## [1] "SEEM035"
## [1] 0.04447027
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
```

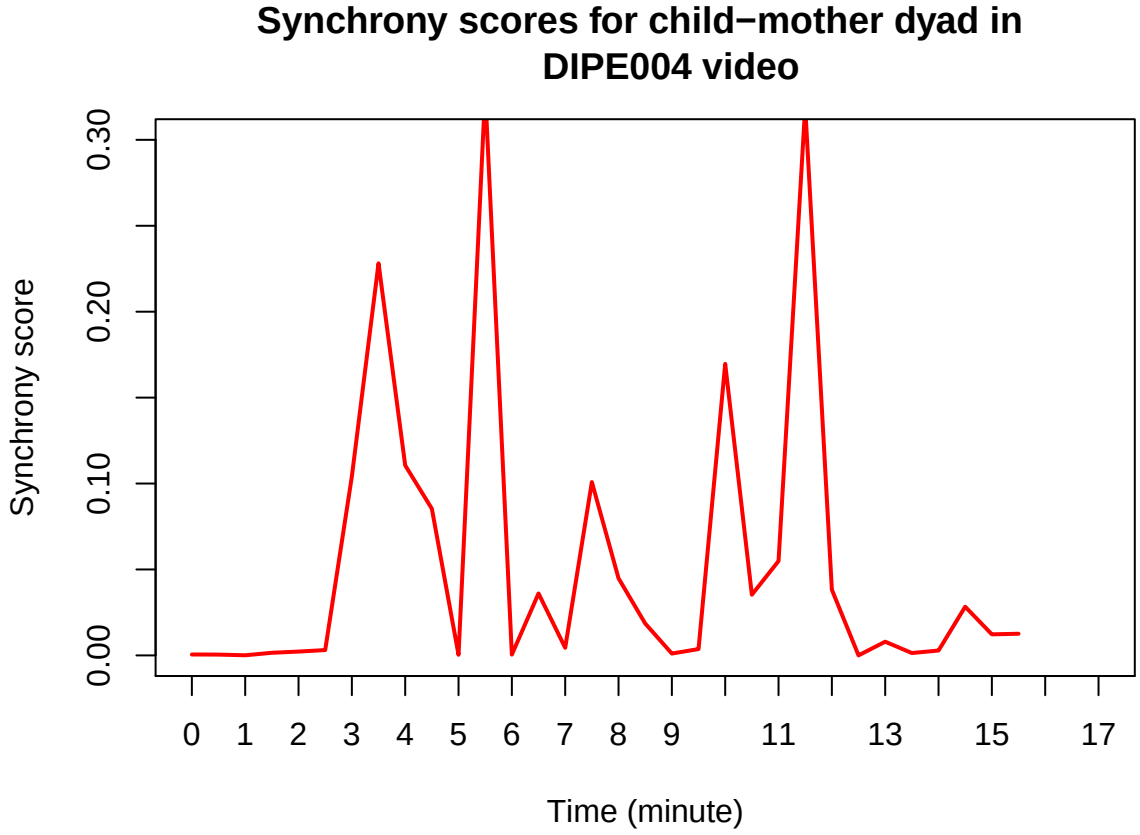
```
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##            d          d  d          d          d          d          d
## 1 0.01959610 0.04255482 NA 0.03022675 0.06900370 0.02300349 0.03943233
## 2 0.02287883 0.02599562 NA 0.03701956 0.04339679 0.02265999 0.04697950
##            d          d  d          d          d   d          d   d   d
## 1 0.02113752 0.03854909 NA 0.02900099 0.01762869 NaN 0.05173998 NaN NaN
## 2 0.09009637 0.04637298 NA 0.05055375 0.01975602 NaN 0.02659465 NaN NaN
##     d          d          d
## 1 NaN 0.02043336 0.04447027
## 2 NaN 0.02307545 0.04576938
## [1] "SHAN042"
## [1] 0.01921827
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##            d          d  d          d          d          d          d
## 1 0.01959610 0.04255482 NA 0.03022675 0.06900370 0.02300349 0.03943233
## 2 0.02287883 0.02599562 NA 0.03701956 0.04339679 0.02265999 0.04697950
##            d          d  d          d          d   d          d   d   d
## 1 0.02113752 0.03854909 NA 0.02900099 0.01762869 NaN 0.05173998 NaN NaN
## 2 0.09009637 0.04637298 NA 0.05055375 0.01975602 NaN 0.02659465 NaN NaN
##     d          d          d          d
## 1 NaN 0.02043336 0.04447027 0.019218266
## 2 NaN 0.02307545 0.04576938 0.007846251
## [1] "SOGA061"
## [1] 0.05819861
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##            d          d  d          d          d          d          d
## 1 0.01959610 0.04255482 NA 0.03022675 0.06900370 0.02300349 0.03943233
## 2 0.02287883 0.02599562 NA 0.03701956 0.04339679 0.02265999 0.04697950
##            d          d  d          d          d   d          d   d   d
## 1 0.02113752 0.03854909 NA 0.02900099 0.01762869 NaN 0.05173998 NaN NaN
## 2 0.09009637 0.04637298 NA 0.05055375 0.01975602 NaN 0.02659465 NaN NaN
##     d          d          d          d          d
## 1 NaN 0.02043336 0.04447027 0.019218266 0.05819861
## 2 NaN 0.02307545 0.04576938 0.007846251 0.03146273
## [1] "TIUG032"
## [1] 0.1027168
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
```

```
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##            d          d  d          d          d          d          d
## 1 0.01959610 0.04255482 NA 0.03022675 0.06900370 0.02300349 0.03943233
## 2 0.02287883 0.02599562 NA 0.03701956 0.04339679 0.02265999 0.04697950
##            d          d  d          d          d   d          d   d   d
## 1 0.02113752 0.03854909 NA 0.02900099 0.01762869 NaN 0.05173998 NaN NaN
## 2 0.09009637 0.04637298 NA 0.05055375 0.01975602 NaN 0.02659465 NaN NaN
##     d          d          d          d          d          d
## 1 NaN 0.02043336 0.04447027 0.019218266 0.05819861 0.10271680
## 2 NaN 0.02307545 0.04576938 0.007846251 0.03146273 0.03967581
## [1] "VINO"
## [1] 0.01691105
## function (..., recursive = FALSE)  .Primitive("c")
##   NA. NA..1   d   d   d   d   d   d   d  d          d          d
## 1  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.04432081 0.01886765
## 2  NA    NA NaN NaN NaN NaN NaN NaN NaN NA 0.02796307 0.05154120
##            d          d          d          d          d          d
## 1 0.05482111 0.07269493 0.04543333 0.01343730 0.05563912 0.05358990
## 2 0.03528263 0.07676492 0.05129765 0.02978409 0.05815557 0.04695914
##            d          d  d          d          d          d          d
## 1 0.01959610 0.04255482 NA 0.03022675 0.06900370 0.02300349 0.03943233
## 2 0.02287883 0.02599562 NA 0.03701956 0.04339679 0.02265999 0.04697950
##            d          d  d          d          d   d          d   d   d
## 1 0.02113752 0.03854909 NA 0.02900099 0.01762869 NaN 0.05173998 NaN NaN
## 2 0.09009637 0.04637298 NA 0.05055375 0.01975602 NaN 0.02659465 NaN NaN
##     d          d          d          d          d          d          d
## 1 NaN 0.02043336 0.04447027 0.019218266 0.05819861 0.10271680 0.01691105
## 2 NaN 0.02307545 0.04576938 0.007846251 0.03146273 0.03967581 0.02524400
```

```r
df <- df[,-(1:10)]

barplot(as.matrix(df), beside=TRUE, xlab="Video", ylab="SSI score", cex.names=0.3)
```




```
par(mar=c(4,4,4,3), mfrow=c(1,1))
plot(SSInoLog[which(SSInoLog$video=="DIPE004"),]$Time_min, SSInoLog[which(SSInoLog$video=="DIPE004"),]$S
     DIPE004 video", xlab = "Time (minute)", ylab="Synchrony score", lwd=2, xaxp=c(0,length(SSInoLog$Tim
```



Evolution of synchrony through time, raw each second

```
par(mar=c(4,4,4,4))
    col <- 1
for (i in 5:length(SSI)){
    plot(1:length(SSI[,i]), SSI[,i], type="l",
    col=rainbow(4)[col], main = names(SSI)[i])
    col <- col+1}
```

Evolution of synchrony through time, mean by minute

```
par(mar=c(4,4,4,4))
    col = 1
for (indexSSI in 5:length(SSI)){
    IntervalNumbersVideo <- ceiling(length(SSI[,indexSSI])/6)
    SSIColumn <- SSI[,indexSSI]
    SSIMinute <- c()
    for (i in 1:IntervalNumbersVideo){
          borneInf <- 1+(i-1)*6
          borneSup <- i * 6
          SSIVectorInterval <- SSIColumn[borneInf:borneSup]
          mean <- mean(SSIVectorInterval, na.rm=TRUE)
          SSIMinute <- c(SSIMinute, mean)}
    plot(1:length(SSIMinute), SSIMinute, type="l", col=rainbow(11)[col], main = names(SSI)[indexSSI])
    col <- col+1}
```

Evolution of synchrony through time, mean by 10 minutes

```
par(mar=c(4,4,4,4))
    col = 1
for (indexSSI in 5:length(SSI)){
    IntervalNumbersVideo <- ceiling(length(SSI[,indexSSI])/60)
    SSIColumn <- SSI[,indexSSI]
    SSITenMinute <- c()
    for (i in 1:IntervalNumbersVideo){
          borneInf <- 1+(i-1)*60
          borneSup <- i * 60
          SSIVectorInterval <- SSIColumn[borneInf:borneSup]
          mean <- mean(SSIVectorInterval, na.rm=TRUE)
          SSITenMinute <- c(SSITenMinute, mean)}
    plot(1:length(SSITenMinute), SSITenMinute, type="l", col=rainbow(4)[col], main = names(SSI)[indexSS
    col <- col+1}
```

Psychometric database

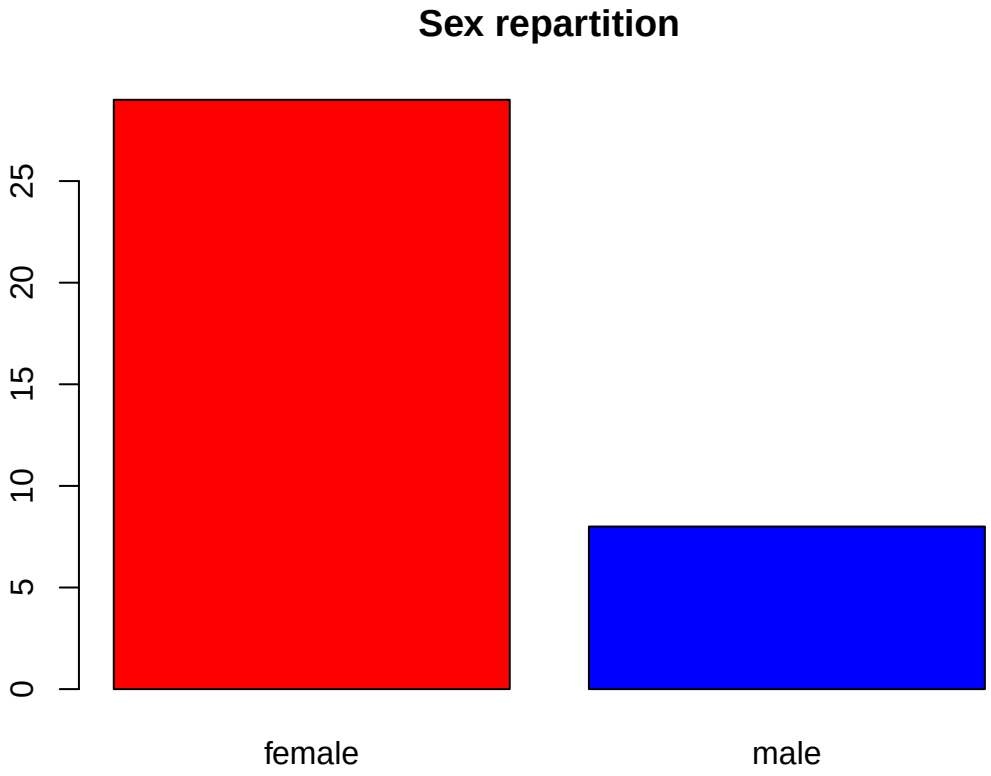
```
psycho <- read.csv2("/Users/Ofix/Documents/Fac/internat/Recherche/projets/synchro/synchroData/Monrado/Da
psycho <- psycho[-(38:40),]
```

```
str(psycho)
#View(psycho)
```

Demographic description

Sex

```
psycho$Sex[which(psycho$Sex == 1 )] <- "male"
psycho$Sex[which(psycho$Sex == 2 )] <- "female"

par(mar=c(3,4,4,4))
barplot(table(psycho$Sex), col=c("red", "blue"), main ="Sex repartition")
```



Age

```
psycho$Birthday <- as.Date(psycho$Birthday, format="%d/%m/%y")
psycho$interview_date <- as.Date(psycho$interview_date, format="%d/%m/%y")
str(psycho$Birthday )
```

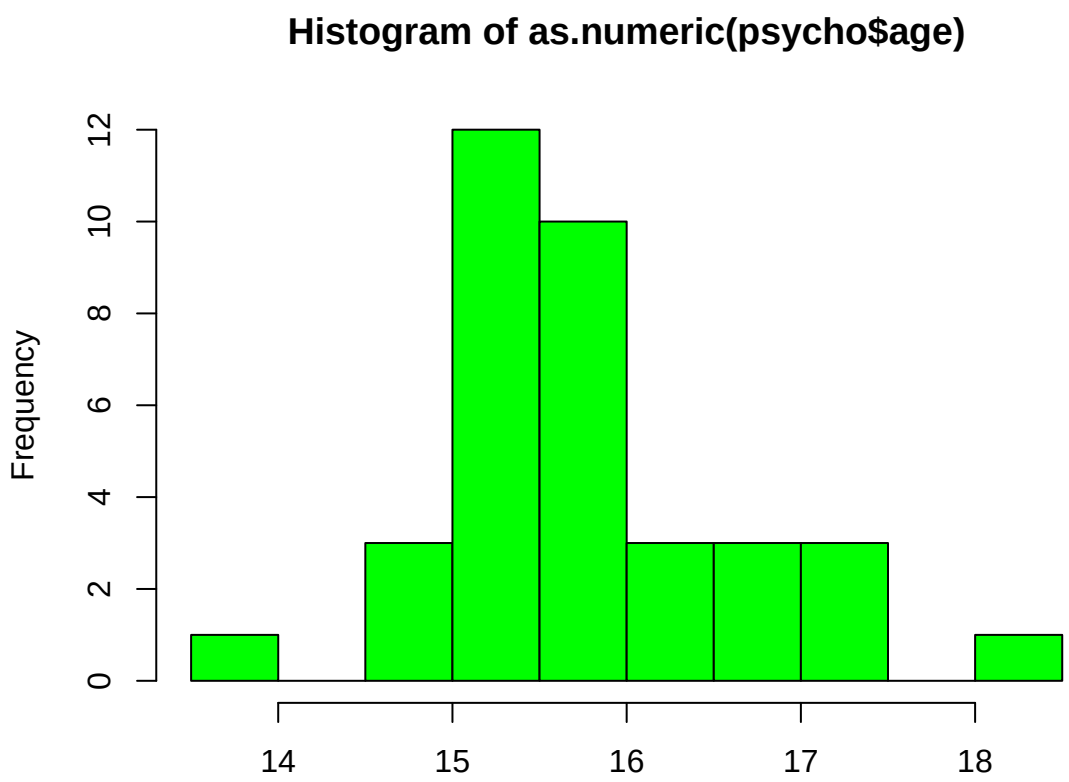
```
##  Date[1:37], format: "1997-06-18" "1997-12-03" "1997-04-07" "1999-09-08" ...
```

```
str(psycho$interview_date)
```

```
##  Date[1:37], format: "2014-01-09" "2014-01-16" "2014-04-17" "2014-04-29" ...
```

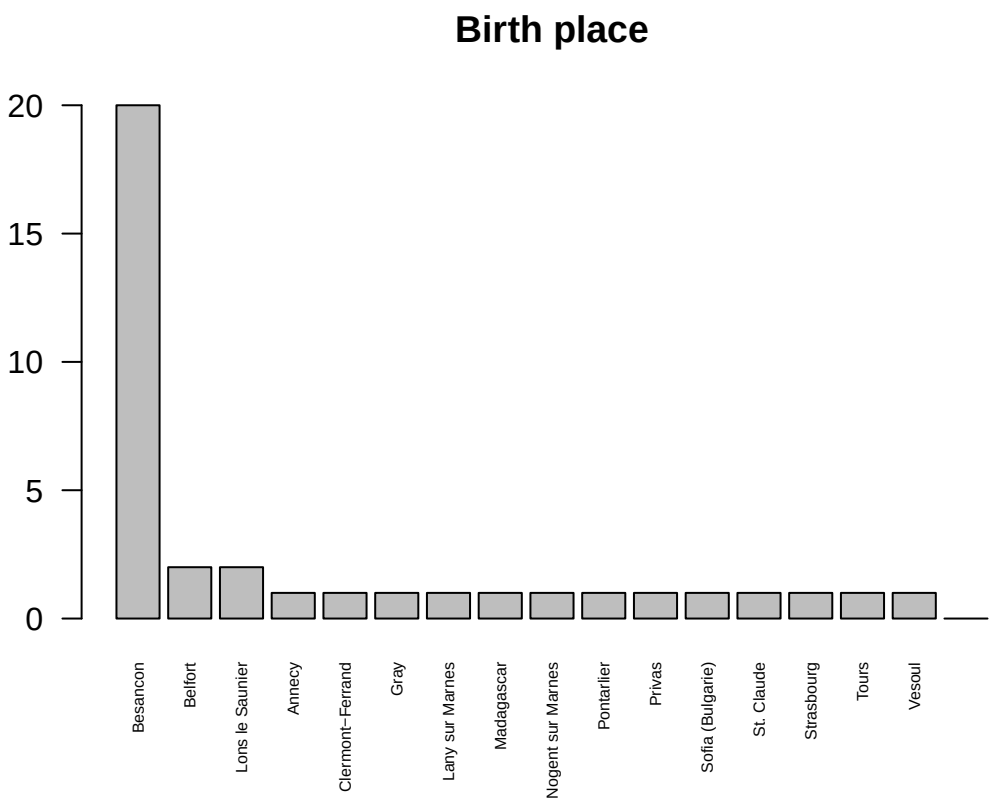
```
psycho$age <- (psycho$interview_date-psycho$Birthday)/365.25
```

```
par(mar=c(3,4,4,4))
hist(as.numeric(psycho$age), col="green")
```



Birth places

```
par(mar=c(5,4,4,4))
barplot(sort(table(psycho$Birth_place), decreasing = TRUE ), las=2, cex.names=0.5, main="Birth place")
```

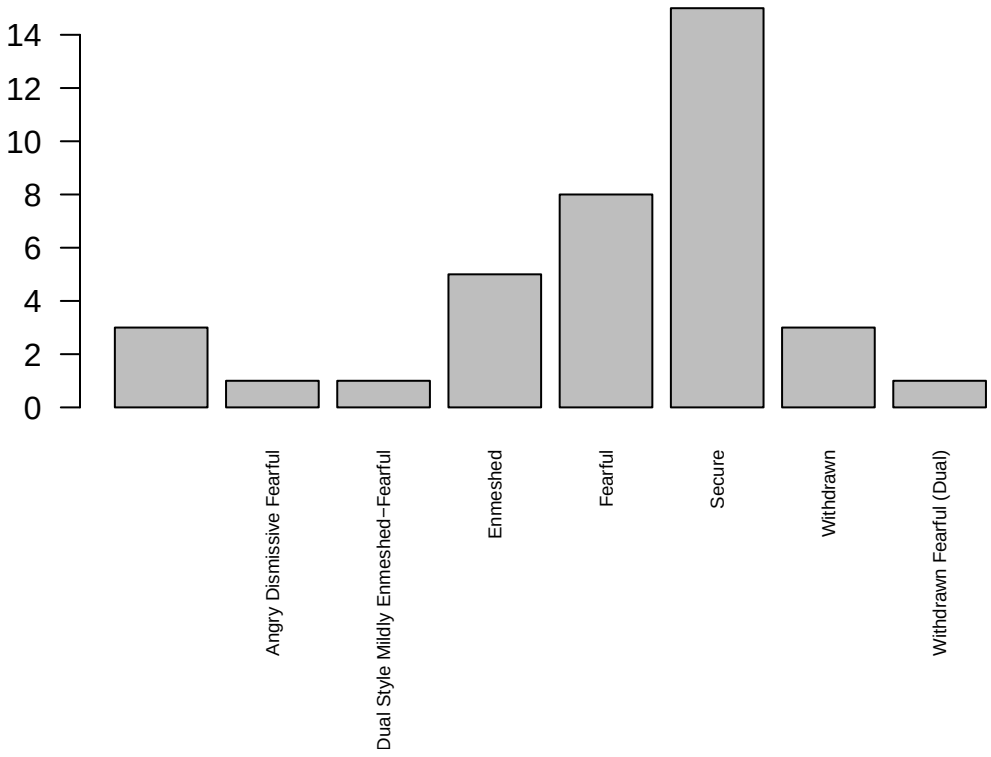



```
psycho$Birth_place
```

```
##  [1] Gray              Madagascar        Besancon
##  [4] Lany sur Marnes   Besancon          Besancon
##  [7] Sofia (Bulgarie)  Besancon          Besancon
## [10] Lons le Saunier   Besancon          Besancon
## [13] Besancon          Besancon          Pontarlier
## [16] Lons le Saunier   Besancon          Besancon
## [19] Besancon          Nogent sur Marnes Belfort
## [22] Strasbourg        Besancon          Besancon
## [25] Besancon          Besancon          Annecy
## [28] Belfort           Besancon          Besancon
## [31] St. Claude        Privas            Clermont-Ferrand
## [34] Tours             Besancon          Besancon
## [37] Vesoul
## 17 Levels:  Annecy Belfort Besancon Clermont-Ferrand ... Vesoul
```

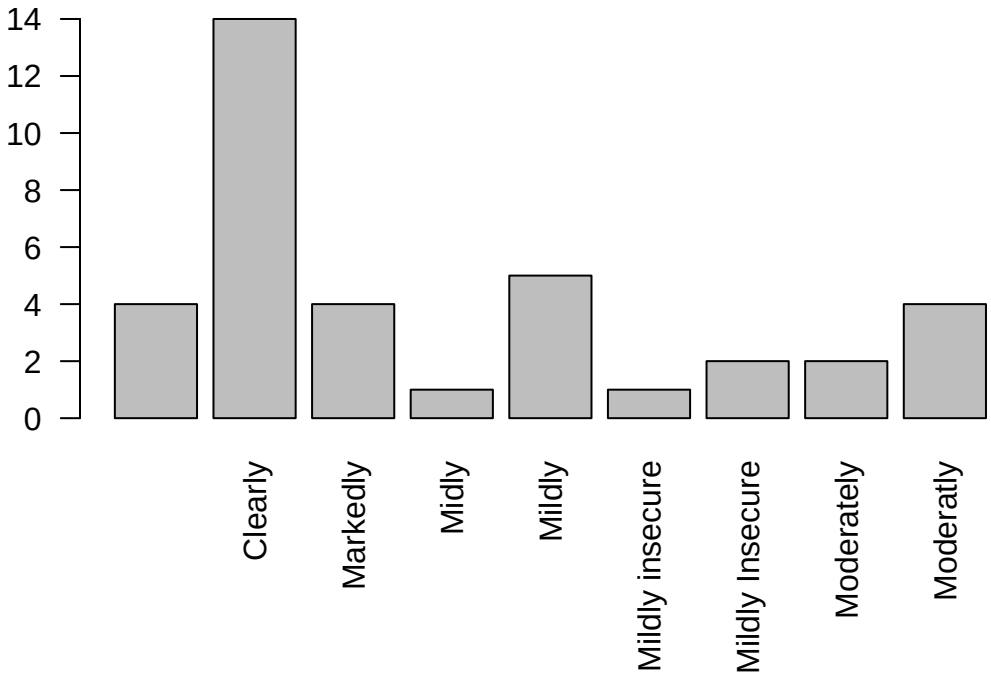
Attachement styles

```
par(mar=c(9,5,3,3))
barplot(table(psycho$attachement_style), las=2, cex.names=0.6)
```



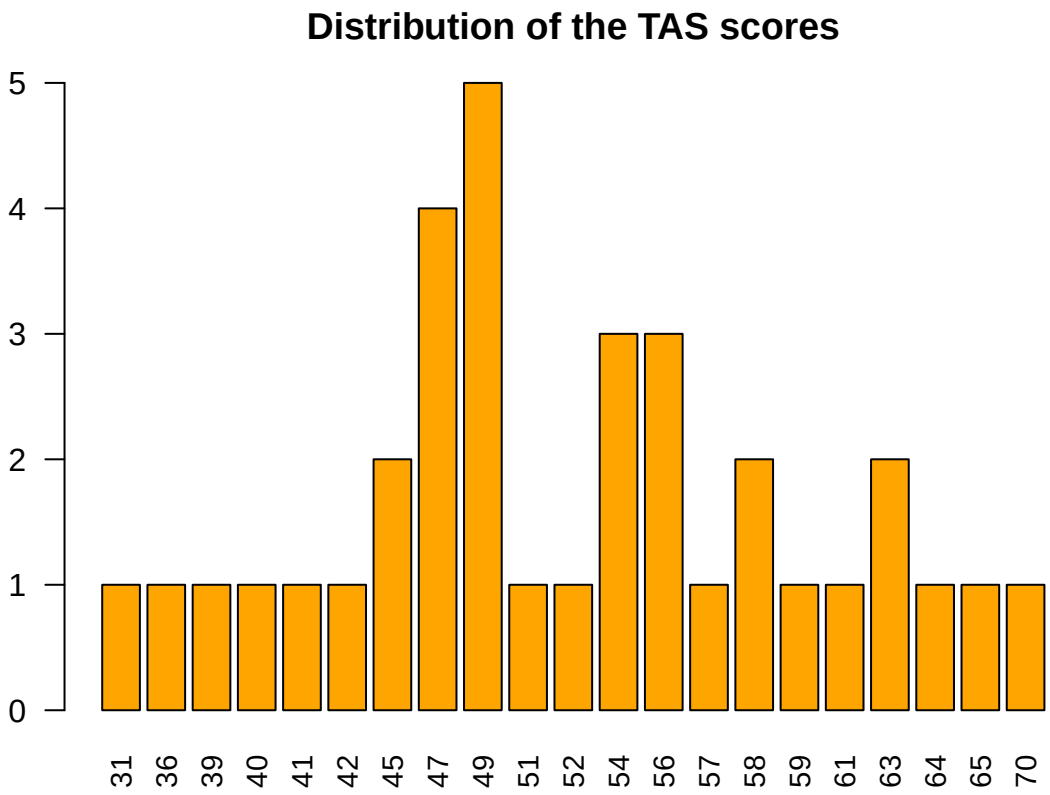
Insecurity level

```
par(mar=c(9,5,3,3))
barplot(table(psycho$insecurite_level), las=2)
```



TAS

```
par(mar=c(3,3,3,3))
barplot(table(psycho$TAS_total), las=2, col="orange", main="Distribution of the TAS scores", cex.name=0
```



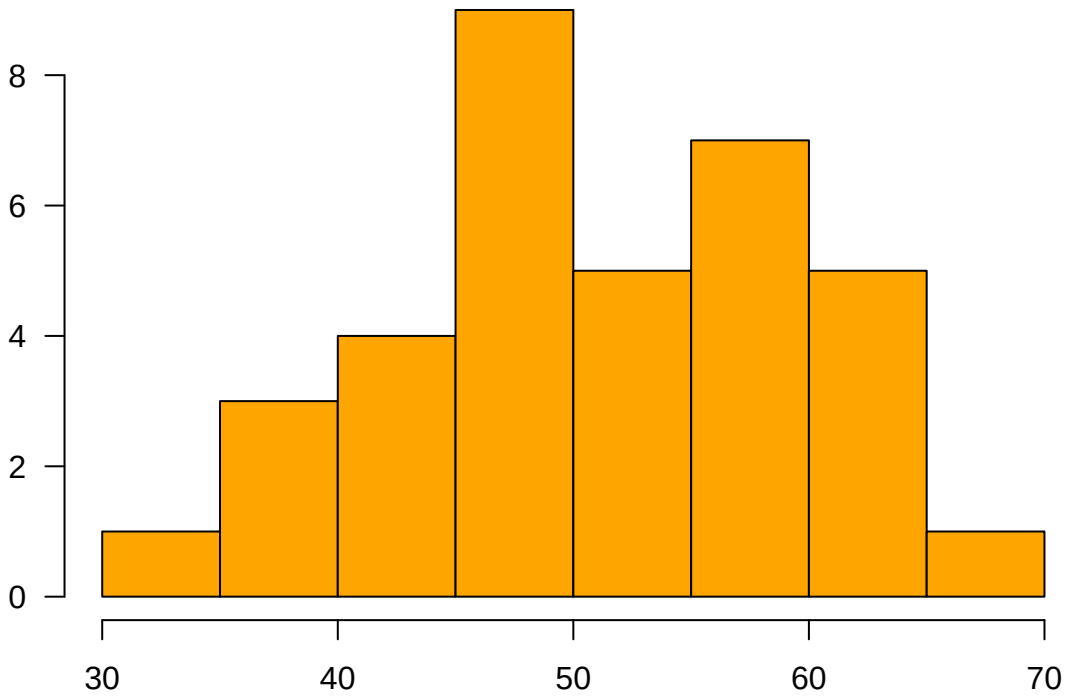

```
hist(psycho$TAS_total, las=1, col="orange", main="Distribution of the TAS scores", cex.name=0.9)
```

```
## Warning in plot.window(xlim, ylim, "", ...): "cex.name" is not a graphical
## parameter
```

```
## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## "cex.name" is not a graphical parameter
```

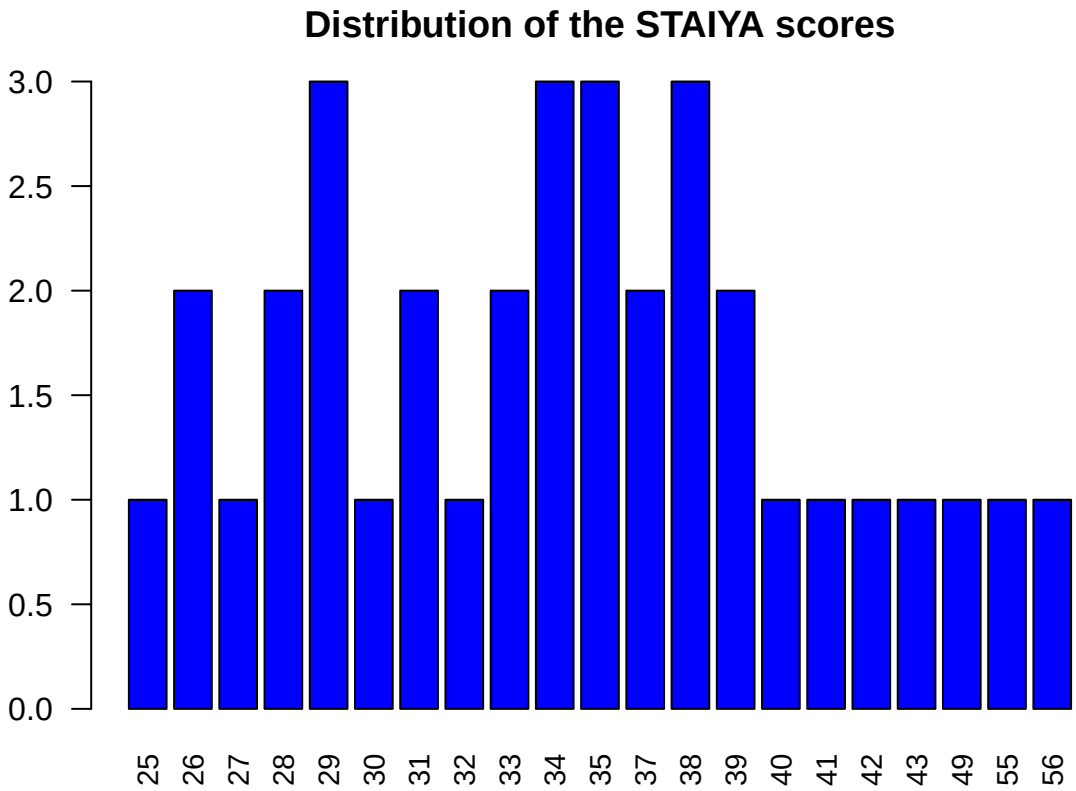
```
## Warning in axis(1, ...): "cex.name" is not a graphical parameter
```

```
## Warning in axis(2, ...): "cex.name" is not a graphical parameter
```



STAIYA

```
par(mar=c(3,3,3,3))
barplot(table(psycho$STAIYA_total), las=2, col="blue", main="Distribution of the STAIYA scores", cex.nar
```



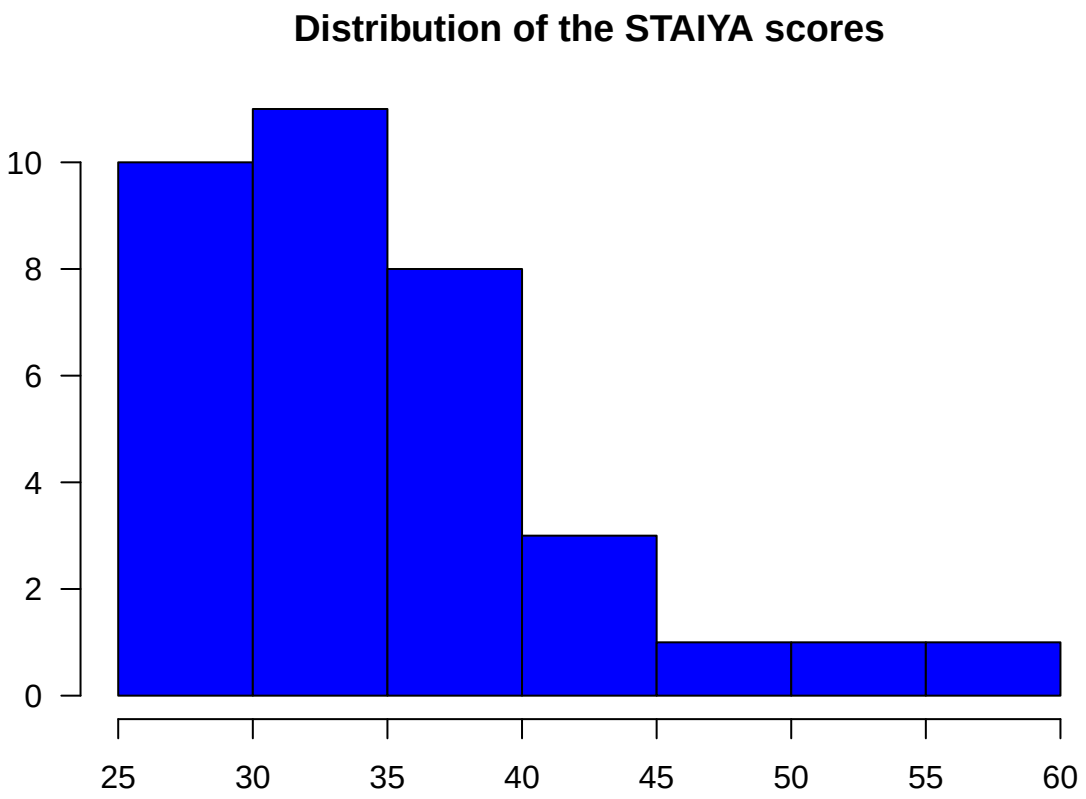

```
hist(psycho$STAIYA_total, las=1, col="blue", main="Distribution of the STAIYA scores", cex.name=0.9)
```

```
## Warning in plot.window(xlim, ylim, "", ...): "cex.name" is not a graphical
## parameter
```

```
## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## "cex.name" is not a graphical parameter
```

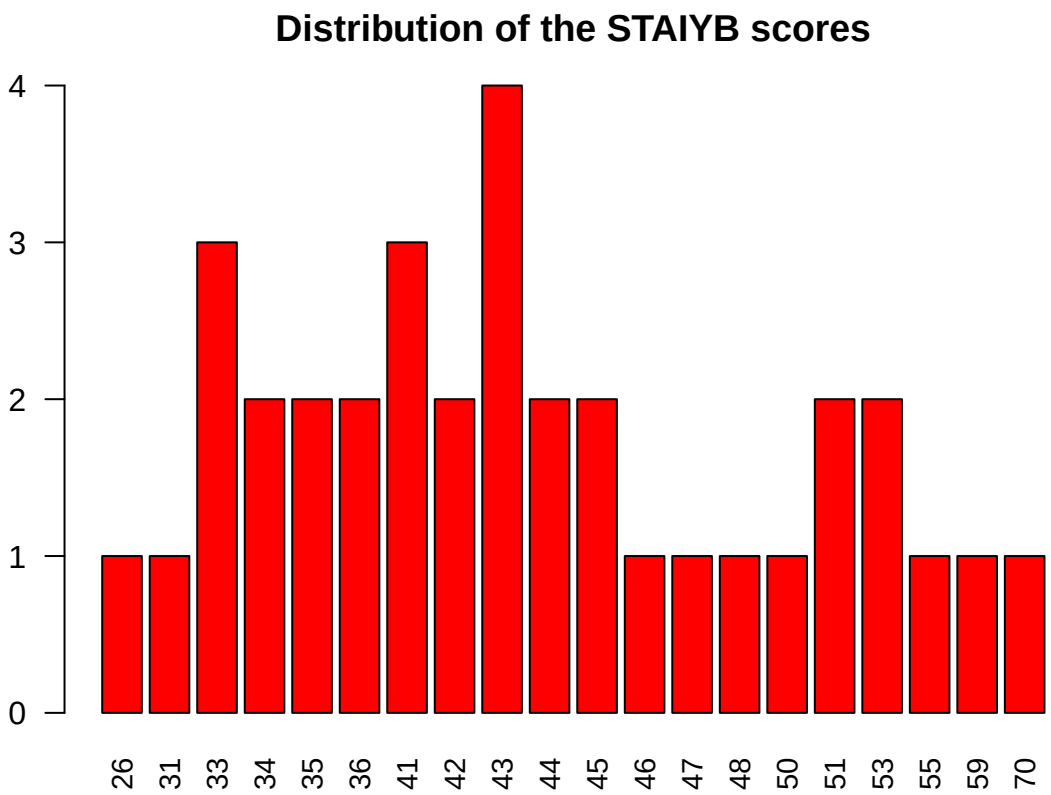
```
## Warning in axis(1, ...): "cex.name" is not a graphical parameter
```

```
## Warning in axis(2, ...): "cex.name" is not a graphical parameter
```



STAIYB

```
par(mar=c(3,3,3,3))
barplot(table(psycho$STAIYB_total), las=2, col="red", main="Distribution of the STAIYB scores", cex.nam
```



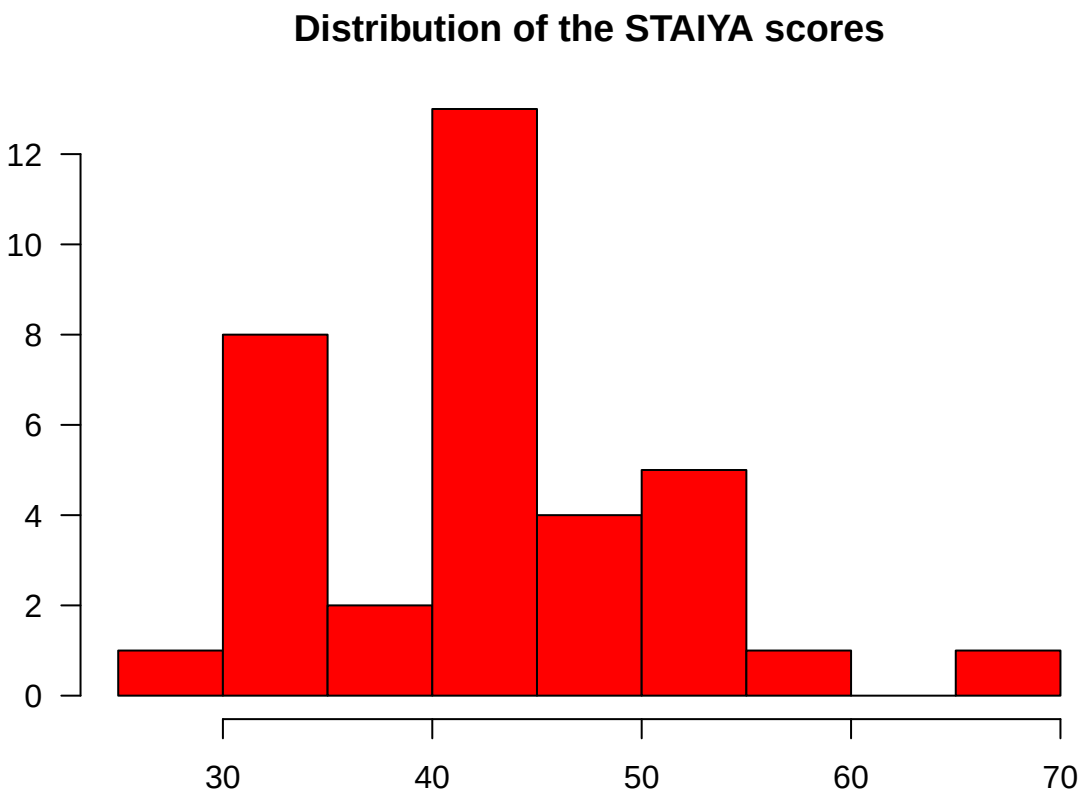

```
hist(psycho$STAIYB_total, las=1, col="red", main="Distribution of the STAIYA scores", cex.name=0.9)
```

```
## Warning in plot.window(xlim, ylim, "", ...): "cex.name" is not a graphical
## parameter
```

```
## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## "cex.name" is not a graphical parameter
```

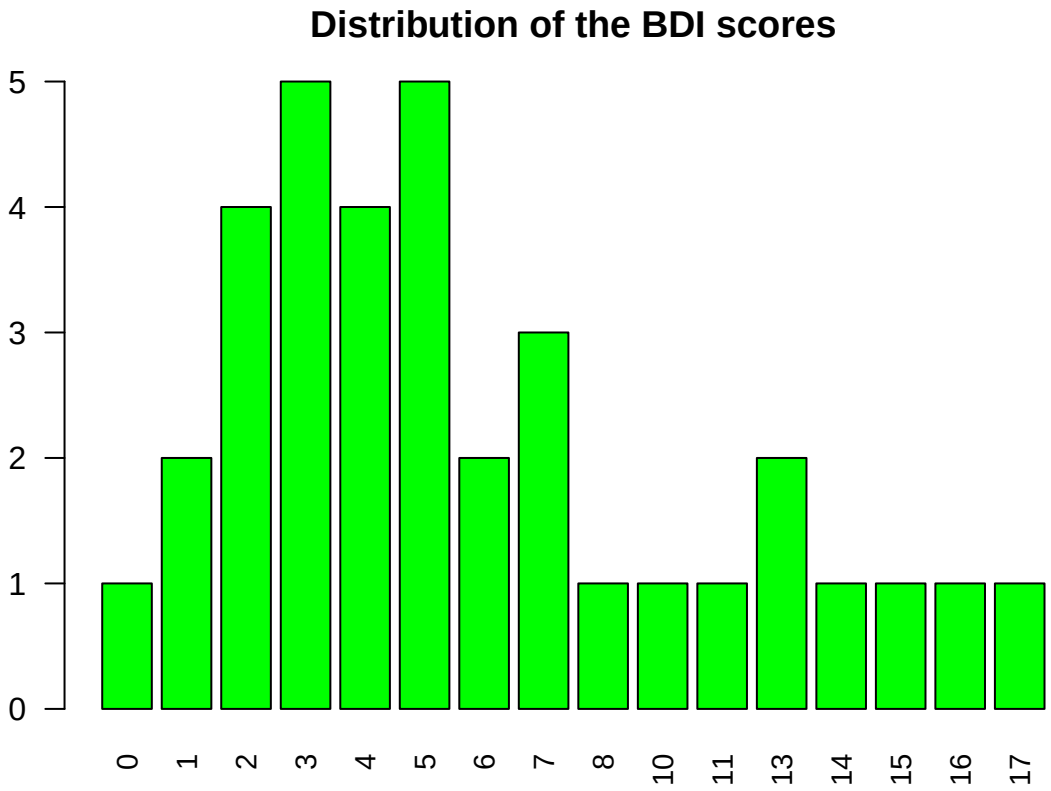
```
## Warning in axis(1, ...): "cex.name" is not a graphical parameter
```

```
## Warning in axis(2, ...): "cex.name" is not a graphical parameter
```



BDI total

```
par(mar=c(3,3,3,3))
barplot(table(psycho$BDI_total), las=2, col="green", main="Distribution of the BDI scores", cex.name=0.
```



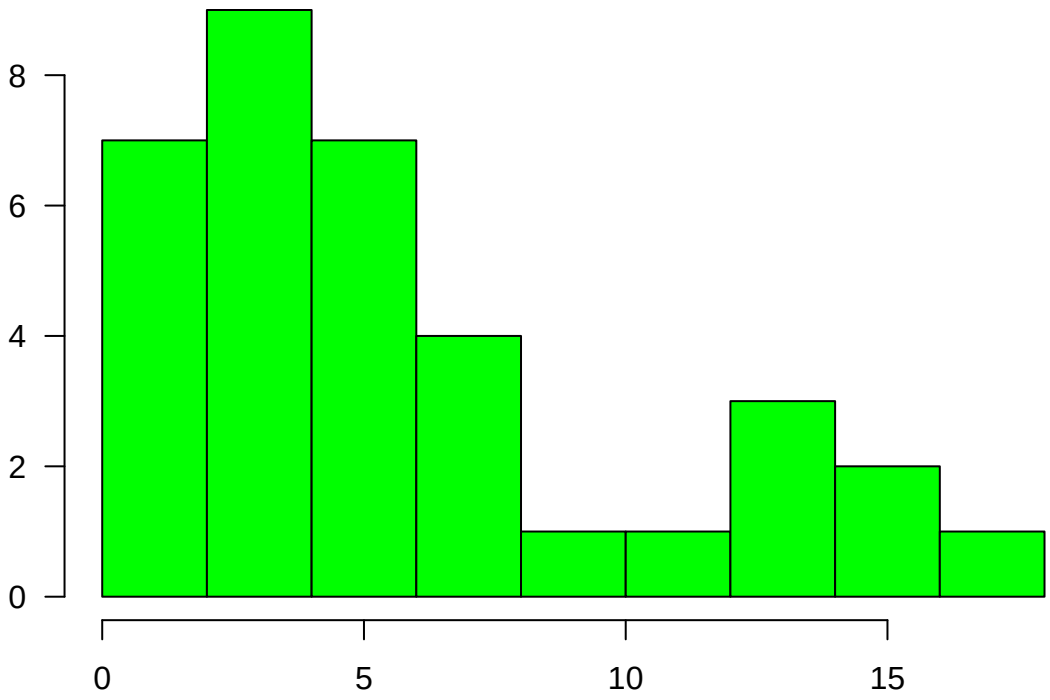

```
hist(psycho$BDI_total, las=1, col="green", main="Distribution of the BDI scores", cex.name=0.9)
```

```
## Warning in plot.window(xlim, ylim, "", ...): "cex.name" is not a graphical
## parameter
```

```
## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## "cex.name" is not a graphical parameter
```

```
## Warning in axis(1, ...): "cex.name" is not a graphical parameter
```

```
## Warning in axis(2, ...): "cex.name" is not a graphical parameter
```



Models of synchrony

```
SSI_fa_th_lme <- lmer(SSI_fa_th ~ Time_min + (1|video), data=SSI)
summary(SSI_fa_th_lme)
#plot(SSI_fa_th_lme)
res <- residuals(SSI_fa_th_lme)
hist(SSI$SSI_fa_th)
qqnorm(res)
SSI_fa_th_List <- c()
for (i in families){
  SSI_fa_th_List <- c(SSI_fa_th_List, mean(SSI[which(SSI$video==i),]$SSI_fa_th, na.rm=TRUE))
}
print(SSI_fa_th_List)
#plot(SSI_fa_th_List, type="b")
```

```
# log of the data
log_SSI_fa_th <- hist(log(SSI$SSI_fa_th))
SSI_fa_th_log_lme <- lmer(log(SSI_fa_th) ~ Time_min + (1|video), data=SSI)
res_log <- residuals(SSI_fa_th_log_lme)
qqnorm(res_log)
summary(SSI_fa_th_log_lme)
```

```
# root square of the data
sq_SSI_fa_th <- hist(sqrt(SSI$SSI_fa_th))
SSI_fa_th_sq_lme <- lmer(sqrt(SSI_fa_th) ~ Time_min + (1|video), data=SSI)
res_sq <- residuals(SSI_fa_th_sq_lme)
qqnorm(res_sq)
summary(SSI_fa_th_sq_lme)
```