

Synchrony in relationship, example with MONRADO Data

Thomas Gargot

May, 31st 2016

Contents

Preparation of the data	2
Fixed variables	2
Functions list	2
Before importing data	4
Import data	6
Clean dataframe	6
Clean time annotations data frame (cutFrames)	7
Psychometric database	9
Merge the data and the annotation data frames	11
Presentation of the data	11
Length of the videos in minutes	18
Length of the videos in number of frames	19
Configurations of the videos	19
Number of Available (True) and Not Available (False) data for each participant	19
Global Motion history	21
Mean Motion history by video by participant	21
Global motion history by situation : “conflict” vs “no conflict”	21
Log of Global motion history by situation : “conflict” vs “no conflict”	22
Motion history histogram by frame (raw data), all videos	30
Sliding interval	34
Non overlapping interval	35
Focus on the motion history of the first 10 seconds of the first video 00034	35
Sliding interval function on a 5 frames interval	35
Motion history of the father during 10-20 seconds of the first video 00034	37
Mean motion history by 10 sec plots	39
Export no log data in text files	56
Export log data in text files	57
SyncPy utilisation for creating synchrony dataframes	58
Description of SSIlog data frame	59
Description of noLogSSI data frame	59
Synchrony scores log for each dyad, triad and for the whole group	60
Synchrony scores noLog for each dyad, triad and for the whole group	72

Demographic and Psychometric data	84
Demographic description	84
Attachement styles	86
Insecurity level	87
TAS	87
STAIYA	88
STAIYB	89
BDI total	90

Preparation of the data

Fixed variables

```

FileExtension <- ".MTS.avi_res.csv"

# working directory
# where this report is
setwd("/Users/0fix/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 computes 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

Mean motion history (non overlapping 5 frames intervals) 00034 video, 2nd second

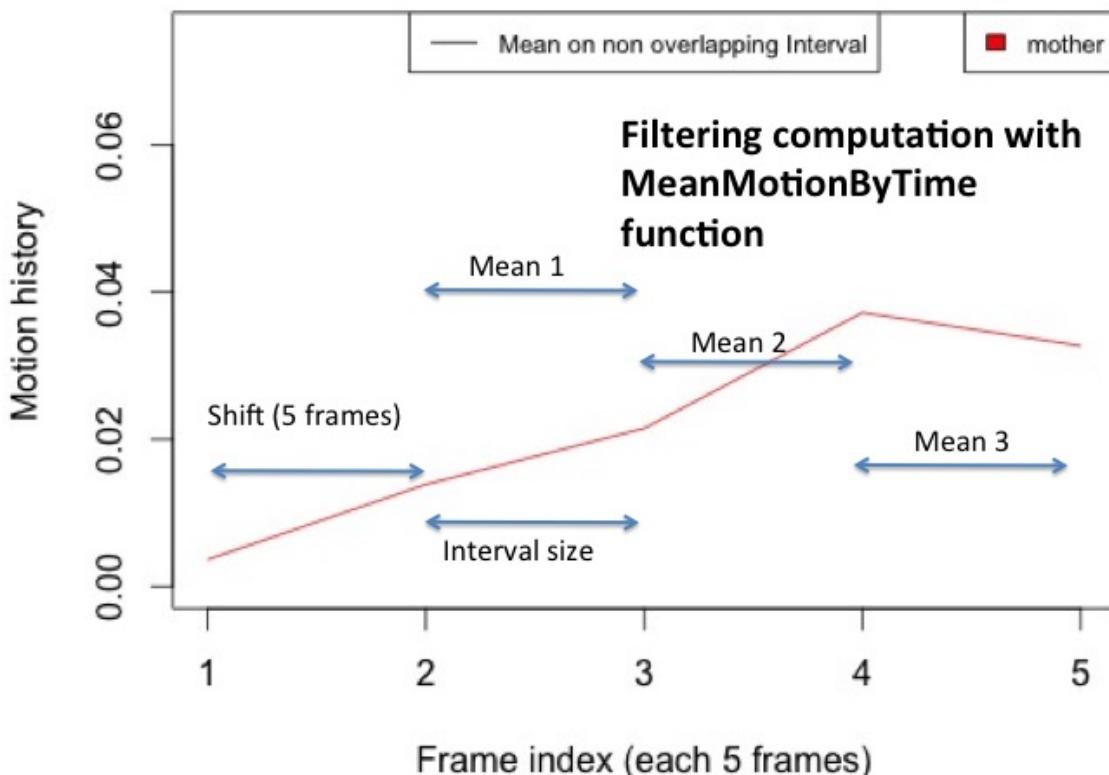


Figure 1:

```

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 computes the mean on a given interval. The interval overlap, so the frequency of the data don't change. It stays at 25 frames/s (fig.2.).

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

```

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)}

```

Before importing data

We prepared videos in AVI format (converted by Converting video script) and frames extracted by Frames extractor script and non relevant part of the images masked by Paintbrush in green.

With these videos and frames, we used the motionHistoryExtractor.cpp script in C++ which extracted motion history in CSV files for each video. This script use the opencv module. There is a filter in this script to avoid too much noise.

Data dictionary of raw files

They are in the form videoname.MTS.avi_res.csv

- **frame** : index of the frame, with frame rate of 25/s
- **father** : motion history for father from 0 (no pixel change) to 1 (all pixels are changed)
- **mother** : idem for mother

Mean motion history (Sliding 5 frames interval) on 00034 video, 2nd second

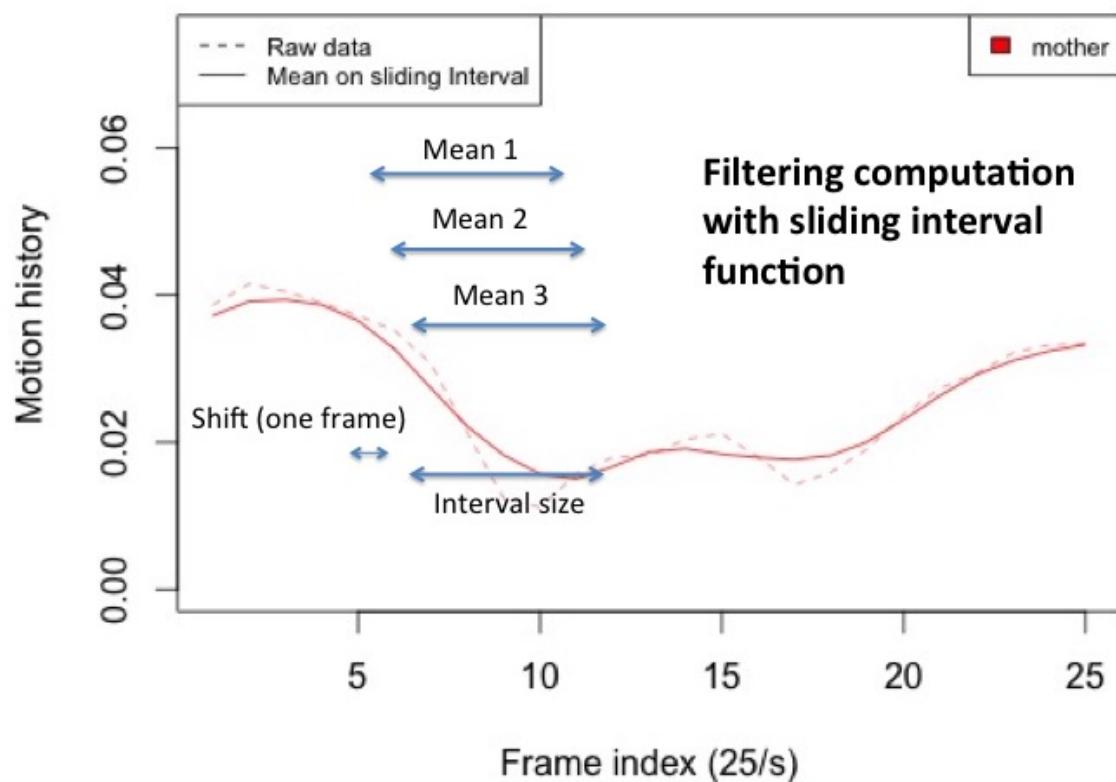


Figure 2:

- **child** : idem for child
- **therapist** : idem for therapist (non relevant here but used in INCANT study)
- **file** : name of the file in the form : videoname.MTS.avi

Import data

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

Clean dataframe

Add new columns: compute minutes and log on data frame The timeMin is calculated with a frame rate of 25/sec.

```
# Delete 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 fileList){
  name <- sub(FileExtension, "", i)
  families <- c(families, name)
}
families

## [1] "1606"    "BAJE059"  "BALU062"  "BEAL036"  "BEAM031"  "BRL0041"  "COL0022"
## [8] "DIPE004"  "DOMA"     "DRNE"     "FOMA057"  "GROP039"  "HAJA052"  "HUMA058"
## [15] "JAEM046"  "JEE0040"  "JOCE014"  "LACL"     "MAEL048"  "MAME20"   "MIPH043"
## [22] "MOSA065"  "NAMA045"  "NUMA027"  "OGGA034"  "PAMA029"  "PELI020"  "RAEM049"
## [29] "RAMA054"  "SEEM035"  "SHANO42"  "SOGA061"  "TIUG032"  "VINO"

NumberOfvideos <- length(families)
NumberOfvideos

## [1] 34

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

# Compute log
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", "fatherShifted", "mother", "motherShifted", "logFather", "logMother")]
```

Data dictionary of clean data data dataframe

- **family** : code of the family
- **frame** : index of the frame, with frame rate of 25/s
- **timeMin** : time in minute for each video ie frame/(25*60)
- **child** : motion history for child from 0 (no pixel change) to 1 (all pixels are changed)
- **childShifted** : motion history of child + mininimum of datachild[which(datachild >0)]/2 to avoid 0 values which log can't be computed
- **logChild** : natural logarithm of childShifted
- **father** : motion history of father from 0 (no pixel change) to 1 (all pixels are changed)
- **fatherShifted** : motion history of father + mininimum of datachild[which(datachild >0)]/2 to avoid 0 values which log can't be computed
- **logFather** : natural logarithm of childShifted
- **mother** : motion history of mother from 0 (no pixel change) to 1 (all pixels are changed)
- **motherShifted** : motion history of mother + mininimum of datachild[which(datachild >0)]/2 to avoid 0 values which log can't be computed
- **logMother** : natural logarithm of motherShifted

Clean time annotations data frame (cutFrames)

Preparing cutFrames

The cutFrames data frame was done manually by looking manually all videos and definining:

- when the experimenter leaves the room and the interaction begin,

Between is the non conflictual discussion

- when the experimenter comes back to ask participants to have a conflictual discussion
- when the experimenter leaves and the conflictual discussion begins Between is the conflictual discussion
- when the experimenter comes back to shut down the camera
- Sex of the child
- Sex of the parent

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

## 'data.frame': 34 obs. of 7 variables:
##   $ family    : Factor w/ 34 levels "1606","BAJE059",...: 1 2 3 4 5 6 7 8 9 10 ...
##   $ CutBefore : Factor w/ 11 levels "00:06","00:07",...: 6 4 3 5 3 7 5 11 7 5 ...
##   $ CutMiddle1: Factor w/ 26 levels "04:43","04:53",...: 21 15 16 16 4 10 17 18 20 17 ...
##   $ CutMiddle2: Factor w/ 31 levels "05:02","05:08",...: 27 20 25 14 2 20 17 26 19 21 ...
##   $ CutFinal  : Factor w/ 28 levels "11:40","14:43",...: 25 26 21 14 6 11 23 22 24 20 ...
##   $ ChildSex  : Factor w/ 2 levels "Female","Male": 1 1 2 1 1 1 1 1 1 2 ...
##   $ ParentSex : Factor w/ 2 levels "Female","Male": 2 2 1 1 1 1 1 1 1 1 ...

# Change the vector in character and cut the sting in two parts minutes and second for the 4 time labels
cutFrames$CutBefore <- as.character(cutFrames$CutBefore)
cutFramesCB <- strsplit(cutFrames$CutBefore, split=":")

# Compute the time in minutes from time in minutes and seconds for each video for Cut Before
```

```

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

# Compute the time in minutes from time in minutes and seconds for each video for Cut CutMiddle1
Cut <- c()
cutFrames$CutMiddle1 <- as.character(cutFrames$CutMiddle1)
cutMiddleSplit <- strsplit(cutFrames$CutMiddle1, split=":")
for (i in 1:nrow(cutFrames)){
  CutAlone <- (as.numeric(cutMiddleSplit[i][[1]][1]) + as.numeric(cutMiddleSplit[i][[1]][2])/60)
  Cut <- c(Cut, CutAlone)
}
cutFrames$CutMiddle1Min <- Cut

# Compute the time in minutes from time in minutes and seconds for each video for Cut CutMiddle2
Cut <- c()
cutFrames$CutMiddle2 <- as.character(cutFrames$CutMiddle2)
cutMiddleSplit <- strsplit(cutFrames$CutMiddle2, split=":")
for (i in 1:nrow(cutFrames)){
  CutAlone <- (as.numeric(cutMiddleSplit[i][[1]][1]) + as.numeric(cutMiddleSplit[i][[1]][2])/60)
  Cut <- c(Cut, CutAlone)
}
cutFrames$CutMiddle2Min <- Cut

# Compute the time in minutes from time in minutes and seconds for each video for Cut CutFinal
Cut <- c()
cutFrames$CutFinal <- as.character(cutFrames$CutFinal)
cutSplit <- strsplit(cutFrames$CutFinal, split=":")
for (i in 1:nrow(cutFrames)){
  CutAlone <- (as.numeric(cutSplit[i][[1]][1]) + as.numeric(cutSplit[i][[1]][2])/60)
  Cut <- c(Cut, CutAlone)
}
cutFrames$CutFinalMin <- Cut
Cut <- c()

```

Data dictionary of cutFrames dataframe

- **family** : code of the family
- **CutBefore** : when the experimenter leave the room and the interaction begin character string in the form min:sec
- **CutMiddle1** : when the experimenter come back to explain that the participants are asked to have a conflictual discussion, character string in the form min:sec
- **CutMiddle2** : when the experimenter leave and the conflictual discussion begin Between is the conflictual discussion, character string in the form min:sec
- **CutFinal** : when the experimenter come back to shut down the camera, character string in the form min:sec
- **ChildSex** : Factor variable : Male or Female
- **ParentSex** : Factor variable : Male or Female
- **CutBeforeMin** : when the experimenter leave the room and the interaction begin numeric variable in minutes

- **CutMiddle1Min** : when the experimenter come back to explain that the participants are asked to have a conflictual discussion, numeric variable in minutes
- **CutMiddle2Min** : when the experimenter leave and the conflictual discussion begin Between is the conflictual discussion, numeric variable in minutes
- **CutFinalMin** : when the experimenter come back to shut down the camera, numeric variable in minutes

Psychometric database

Preparing psycho dataframe

This demographic and psychometric data were collected by the MONRADO team. The data file was cleaned following this flowchart (fig.3).

```
psycho <- read.csv2("/Users/0fix/Documents/Fac/internat/Recherche/projets/synchro/synchroData/Monrado/D...  
  
# replace 1 code by male and 2 by female  
psycho$Sex[which(psycho$Sex == 1)] <- "male"  
psycho$Sex[which(psycho$Sex == 2)] <- "female"  
  
psycho$Birth_place <- as.character(psycho$Birth_place)  
# Clean Besancon town name with a special character  
psycho$Birth_place[which(psycho$Birth_place == "Besan\x8don")] <- "Besancon"  
  
psycho$Num_identification <- NULL  
psycho$Sex <- NULL  
psycho$TASDF <- NULL  
psycho$TASIF <- NULL  
psycho$TASEOT<- NULL  
psycho <- rename (psycho, c("Num._ident_videos" = "family"))  
#View(psycho)  
#str(psycho)
```

Data dictionary of cutFrames dataframe

- **family** : code of the family
- **interview_date** : interview_date in a character format
- **Birthday** : Birthday of the child
- **Birth_place** : Birth place of the child
- **attachement_style** : attachement style in 7 factors
- **attachement_cluster** : attachement style in 5 factors, DUALS will be excluded
- **Insecurity_level** : level of insecurity not used
- **global score** : global score of attachment not used
- **TAS1** : score for 1st question of TAS Questionnaire : Toronto Alexithymia Score idem until TAS20 (Alexithymia questionnaire : difficulty to express emotions)
- **TAS_total** : total score of TAS
- **STAIYA1** : score for the 1st question of State-Trait Anxiety Inventory State score (at this specific moment), for the 1st question of State-Trait Anxiety Inventory State idem until 20
- **STAIYA_total** : total score of STAIYA Anxiety State score (at this specific moment)
- **STAIYB1** : score for the 1st question of State-Trait Anxiety Inventory Trait score (not at this specific moment), idem until 20
- **STAIYB_total** : total score of STAIYB Anxiety Trait score (not at this specific moment)

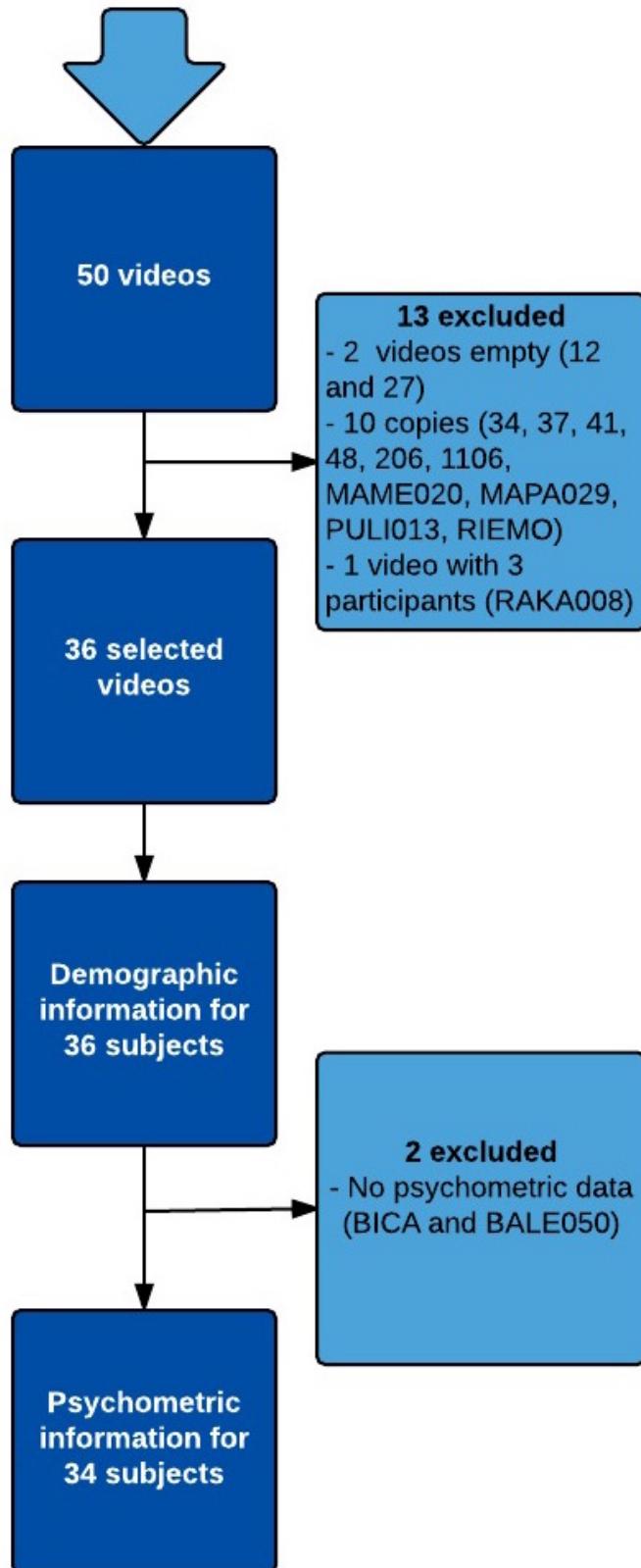


Figure 3:
10

- **BDI1** : score for the 1st question of the BDI (Beck Depression Inventory questionnaire) (Depression questionnaire) idem until BDI13
- **BDI_total** : total score of BDI (Beck Depression Inventory questionnaire)

Merge the data and the annotation data frames

```
# merge the two data frames
data <- merge(data, cutFrames, by.x="family", by.y="family")
data <- merge(data, psycho, by.x="family", by.y="family")

# reorder by family, then by frame order
data <- data[order(data$family, data$frame),]

# Create a column corresponding to the phase of the video for each time limit
data$LabelVideo <- rep(NA, nrow(data))
data[which(data$timeMin <= data$CutBeforeMin),]$LabelVideo <- "Cut"

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

data[which(data$timeMin >= data$CutMiddle1Min & data$timeMin <= data$CutMiddle2Min),]$LabelVideo <- "Cut"

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

data[which(data$timeMin >= data$CutFinalMin),]$LabelVideo <- "Cut"

#View(data)
```

- **LabelVideo** : description of the phase of the video : cut, No-Conflict, Cut, or Conflict

Presentation of the data

```
str(data)
```

```
## 'data.frame': 802970 obs. of 107 variables:
## $ family      : Factor w/ 34 levels "1606","BAJE059",...
## $ 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 0.005132 0.006495 0.000241 0.000829 0.000149 ...
## $ childShifted : num 0.005132 0.006495 0.000241 0.00083 0.00015 ...
## $ logChild    : num -5.27 -5.04 -8.33 -7.09 -8.81 ...
## $ father       : num 0.002321 0.005813 0.000678 0.001026 0.00058 ...
## $ fatherShifted: num 0.002322 0.005814 0.000679 0.001027 0.00058 ...
## $ logFather   : num -6.07 -5.15 -7.3 -6.88 -7.45 ...
## $ mother      : num NA NA NA NA NA NA NA NA NA ...
## $ motherShifted: num NA NA NA NA NA NA NA NA NA ...
## $ logMother   : num NA NA NA NA NA NA NA NA NA ...
## $ CutBefore   : chr "00:11" "00:11" "00:11" "00:11" ...
## $ CutMiddle1  : chr "05:30" "05:30" "05:30" "05:30" ...
## $ CutMiddle2  : chr "06:16" "06:16" "06:16" "06:16" ...
## $ CutFinal    : chr "16:27" "16:27" "16:27" "16:27" ...
## $ ChildSex    : Factor w/ 2 levels "Female","Male": 1 1 1 1 1 1 1 1 1 1 ...
```

```

## $ ParentSex : Factor w/ 2 levels "Female","Male": 2 2 2 2 2 2 2 2 2 ...
## $ CutBeforeMin : num 0.183 0.183 0.183 0.183 0.183 ...
## $ CutMiddle1Min : num 5.5 5.5 5.5 5.5 5.5 5.5 5.5 5.5 5.5 ...
## $ CutMiddle2Min : num 6.27 6.27 6.27 6.27 6.27 6.27 ...
## $ CutFinalMin : num 16.4 16.4 16.4 16.4 16.4 16.4 ...
## $ interview_date : Factor w/ 30 levels "02/12/14","03/12/14",...: 27 27 27 27 27 27 27 27 27 ...
## $ Birthday : Factor w/ 33 levels "01/09/99","01/11/99",...: 28 28 28 28 28 28 28 28 28 ...
## $ Birth_place : chr "Clermont-Ferrand" "Clermont-Ferrand" "Clermont-Ferrand" "Clermont-Ferrand" ...
## $ attachement_style : Factor w/ 7 levels "Angry Dismissive Fearful",...: 4 4 4 4 4 4 4 ...
## $ attachement_cluster: Factor w/ 5 levels "DU","DUAL","FE",...: 3 3 3 3 3 3 3 3 3 ...
## $ insecurite_level : Factor w/ 9 levels "","Clearly","Markedly",...: 3 3 3 3 3 3 3 3 3 ...
## $ global_score : Factor w/ 14 levels "","10 Midly Fearful",...: 6 6 6 6 6 6 6 6 6 ...
## $ TAS1 : int 4 4 4 4 4 4 4 4 4 ...
## $ TAS2 : int 5 5 5 5 5 5 5 5 5 ...
## $ TAS3 : int 5 5 5 5 5 5 5 5 5 ...
## $ TAS4 : int 2 2 2 2 2 2 2 2 2 ...
## $ TAS5 : int 5 5 5 5 5 5 5 5 5 ...
## $ TAS6 : int 1 1 1 1 1 1 1 1 1 ...
## $ TAS7 : int 3 3 3 3 3 3 3 3 3 ...
## $ TAS8 : int 1 1 1 1 1 1 1 1 1 ...
## $ TAS9 : int 5 5 5 5 5 5 5 5 5 ...
## $ TAS10 : int 4 4 4 4 4 4 4 4 4 ...
## $ TAS11 : int 5 5 5 5 5 5 5 5 5 ...
## $ TAS12 : int 1 1 1 1 1 1 1 1 1 ...
## $ TAS13 : int 5 5 5 5 5 5 5 5 5 ...
## $ TAS14 : int 4 4 4 4 4 4 4 4 4 ...
## $ TAS15 : int 1 1 1 1 1 1 1 1 1 ...
## $ TAS16 : int 1 1 1 1 1 1 1 1 1 ...
## $ TAS17 : int 4 4 4 4 4 4 4 4 4 ...
## $ TAS18 : int 5 5 5 5 5 5 5 5 5 ...
## $ TAS19 : int 5 5 5 5 5 5 5 5 5 ...
## $ TAS20 : int 3 3 3 3 3 3 3 3 3 ...
## $ TAS_total : int 56 56 56 56 56 56 56 56 56 ...
## $ STAIYA1 : int 2 2 2 2 2 2 2 2 2 ...
## $ STAIYA2 : int 2 2 2 2 2 2 2 2 2 ...
## $ STAIYA3 : int 1 1 1 1 1 1 1 1 1 ...
## $ STAIYA4 : int 1 1 1 1 1 1 1 1 1 ...
## $ STAIYA5 : int 4 4 4 4 4 4 4 4 4 ...
## $ STAIYA6 : int 1 1 1 1 1 1 1 1 1 ...
## $ STAIYA7 : int 1 1 1 1 1 1 1 1 1 ...
## $ STAIYA8 : int 4 4 4 4 4 4 4 4 4 ...
## $ STAIYA9 : int 2 2 2 2 2 2 2 2 2 ...
## $ STAIYA10 : int 2 2 2 2 2 2 2 2 2 ...
## $ STAIYA11 : int 3 3 3 3 3 3 3 3 3 ...
## $ STAIYA12 : int 1 1 1 1 1 1 1 1 1 ...
## $ STAIYA13 : int 1 1 1 1 1 1 1 1 1 ...
## $ STAIYA14 : int 1 1 1 1 1 1 1 1 1 ...
## $ STAIYA15 : int 3 3 3 3 3 3 3 3 3 ...
## $ STAIYA16 : int 1 1 1 1 1 1 1 1 1 ...
## $ STAIYA17 : int 1 1 1 1 1 1 1 1 1 ...
## $ STAIYA18 : int 3 3 3 3 3 3 3 3 3 ...
## $ STAIYA19 : int 3 3 3 3 3 3 3 3 3 ...
## $ STAIYA20 : int 2 2 2 2 2 2 2 2 2 ...
## $ STAIYA_total : int 39 39 39 39 39 39 39 39 39 ...

```

```

## $ STAIYB1 : int 2 2 2 2 2 2 2 2 2 2 ...
## $ STAIYB2 : int 3 3 3 3 3 3 3 3 3 3 ...
## $ STAIYB3 : int 4 4 4 4 4 4 4 4 4 4 ...
## $ STAIYB4 : int 4 4 4 4 4 4 4 4 4 4 ...
## $ STAIYB5 : int 2 2 2 2 2 2 2 2 2 2 ...
## $ STAIYB6 : int 4 4 4 4 4 4 4 4 4 4 ...
## $ STAIYB7 : int 4 4 4 4 4 4 4 4 4 4 ...
## $ STAIYB8 : int 3 3 3 3 3 3 3 3 3 3 ...
## $ STAIYB9 : int 4 4 4 4 4 4 4 4 4 4 ...
## $ STAIYB10 : int 3 3 3 3 3 3 3 3 3 3 ...
## $ STAIYB11 : int 4 4 4 4 4 4 4 4 4 4 ...
## $ STAIYB12 : int 3 3 3 3 3 3 3 3 3 3 ...
## $ STAIYB13 : int 4 4 4 4 4 4 4 4 4 4 ...
## $ STAIYB14 : int 4 4 4 4 4 4 4 4 4 4 ...
## $ STAIYB15 : int 4 4 4 4 4 4 4 4 4 4 ...
## $ STAIYB16 : int 3 3 3 3 3 3 3 3 3 3 ...
## $ STAIYB17 : int 4 4 4 4 4 4 4 4 4 4 ...
## $ STAIYB18 : int 4 4 4 4 4 4 4 4 4 4 ...
## $ STAIYB19 : int 4 4 4 4 4 4 4 4 4 4 ...
## $ STAIYB20 : int 3 3 3 3 3 3 3 3 3 3 ...
## $ STAIYB_total : int 70 70 70 70 70 70 70 70 70 70 ...
## $ BDI1 : int 1 1 1 1 1 1 1 1 1 1 ...
## $ BDI2 : int 3 3 3 3 3 3 3 3 3 3 ...
## $ BDI3 : int 1 1 1 1 1 1 1 1 1 1 ...
## $ BDI4 : int 1 1 1 1 1 1 1 1 1 1 ...
## $ BDI5 : int 0 0 0 0 0 0 0 0 0 0 ...
## $ BDI6 : int 1 1 1 1 1 1 1 1 1 1 ...
## $ BDI7 : int 1 1 1 1 1 1 1 1 1 1 ...
## [list output truncated]
summary(data)

```

```

##      family      frame     timeMin       child
## MOSA065: 26975  Min.   : 1   Min.   : 0.000667  Min.   :0.0000000
## HUMA058: 25631  1st Qu.: 5905  1st Qu.: 3.936667  1st Qu.:0.0006849
## BAJE059: 25295  Median :11809  Median : 7.872667  Median :0.0034616
## 1606   : 24947  Mean    :11838  Mean    : 7.891817  Mean   :0.0094846
## DOMA   : 24863  3rd Qu.:17713  3rd Qu.:11.808667 3rd Qu.:0.0117680
## COL0022: 24443  Max.   :26975  Max.   :17.983333  Max.   :0.9270200
## (Other):650816
##      childShifted      logChild       father      fatherShifted
## Min.   :0.0000004  Min.   :-14.66620  Min.   :0.0  Min.   :0.0
## 1st Qu.:0.0006853  1st Qu.: -7.28564  1st Qu.:0.0  1st Qu.:0.0
## Median :0.0034621  Median : -5.66589  Median :0.0  Median :0.0
## Mean   :0.0094850  Mean   : -6.14384  Mean   :0.0  Mean   :0.0
## 3rd Qu.:0.0117684  3rd Qu.: -4.44233  3rd Qu.:0.0  3rd Qu.:0.0
## Max.   :0.9270204  Max.   : -0.07578  Max.   :0.1  Max.   :0.1
##                               NA's   :655136  NA's   :655136
##      logFather      mother      motherShifted      logMother
## Min.   :-14.6   Min.   :0.00   Min.   :0.00   Min.   :-14.78
## 1st Qu.: -9.3  1st Qu.:0.00   1st Qu.:0.00   1st Qu.: -8.28
## Median : -7.1  Median :0.00   Median :0.00   Median : -6.37
## Mean   : -7.5  Mean   :0.01   Mean   :0.01   Mean   : -6.96
## 3rd Qu.: -5.3  3rd Qu.:0.01   3rd Qu.:0.01   3rd Qu.: -5.07
## Max.   : -2.0  Max.   :0.96   Max.   :0.96   Max.   : -0.04

```

```

##  NA's    :655136   NA's    :147834   NA's    :147834   NA's    :147834
##  CutBefore      CutMiddle1      CutMiddle2
##  Length:802970  Length:802970  Length:802970
##  Class :character  Class :character  Class :character
##  Mode  :character  Mode  :character  Mode  :character
##
##
##
##
##      CutFinal      ChildSex      ParentSex      CutBeforeMin
##  Length:802970  Female:659648  Female:655136  Min.   :0.1000
##  Class :character  Male  :143322  Male  :147834  1st Qu.:0.1333
##  Mode  :character                           Median  :0.1667
##                                         Mean    :0.1878
##                                         3rd Qu.:0.1833
##                                         Max.   :0.4500
##
##      CutMiddle1Min  CutMiddle2Min  CutFinalMin interview_date
##  Min.   :4.717    Min.   :5.033    Min.   :11.67  26/03/15: 47962
##  1st Qu.:5.033    1st Qu.:5.433    1st Qu.:15.22 13/05/15: 47122
##  Median  :5.217    Median  :5.800    Median  :15.47  04/12/14: 46558
##  Mean    :5.281    Mean    :5.830    Mean    :15.59  18/12/14: 45814
##  3rd Qu.:5.383    3rd Qu.:6.033    3rd Qu.:16.08 21/04/15: 26975
##  Max.   :7.083    Max.   :7.833    Max.   :17.80  16/04/15: 25631
##                                         (Other)  :562908
##
##      Birthday      Birth_place
##  06/09/99: 51418  Length:802970
##  13/12/96: 25631  Class :character
##  01/12/99: 25295  Mode  :character
##  27/05/99: 24947
##  28/11/98: 24863
##  18/06/97: 24431
##  (Other)  :626385
##
##      attachement_style attachement_cluster
##  Angry Dismissive Fearful      : 23447  DU      : 22847
##  Dual Style Mildly Enmeshed-Fearful: 22847  DUAL    : 46618
##  Enmeshed                           :145266  FE      :313175
##  Fearful                            :167909  Secure   :350025
##  Secure                             :350025  Withdrawn: 70305
##  Withdrawn                         : 70305
##  Withdrawn Fearful (Dual)        : 23171
##
##      insecurite_level          global_score      TAS1
##  Clearly    :327022  13 Clearly secure :256273  Min.   :1.000
##  Mildly     :121063                           :139230  1st Qu.:3.000
##  Markedly   : 95768  9 Mildly Enmeshed  : 73077  Median  :4.000
##  Moderately : 92324  3 Markedly Fearful : 72597  Mean    :3.317
##  Moderately: 47110  4 Moderately Fearful: 47110  3rd Qu.:4.000
##  : 47050    9 Midly Enmshec   : 25295  Max.   :5.000
##  (Other)    : 72633  (Other)           :189388
##
##      TAS2          TAS3          TAS4          TAS5
##  Min.   :1.000  Min.   :1.000  Min.   :1.000  Min.   :2.000
##  1st Qu.:3.000  1st Qu.:1.000  1st Qu.:2.000  1st Qu.:4.000
##  Median  :4.000  Median  :1.000  Median  :2.000  Median  :4.000
##  Mean    :3.419  Mean    :1.831  Mean    :2.567  Mean    :4.059

```

```

## 3rd Qu.:4.000 3rd Qu.:3.000 3rd Qu.:4.000 3rd Qu.:5.000
## Max. :5.000  Max. :5.000  Max. :5.000  Max. :5.000
##
##      TAS6          TAS7          TAS8          TAS9
## Min.  :1.000  Min.  :1.000  Min.  :1.000  Min.  :1.000
## 1st Qu.:2.000 1st Qu.:1.000 1st Qu.:2.000 1st Qu.:2.000
## Median :3.000  Median :2.000  Median :2.000  Median :3.000
## Mean   :2.803  Mean   :2.224  Mean   :2.321  Mean   :3.237
## 3rd Qu.:4.000 3rd Qu.:3.000 3rd Qu.:3.000 3rd Qu.:4.000
## Max.  :5.000  Max.  :5.000  Max.  :5.000  Max.  :5.000
## NA's   :23447  NA's   :22967                NA's   :23507
##      TAS10         TAS11         TAS12         TAS13
## Min.  :1.000  Min.  :1.000  Min.  :1.000  Min.  :1.000
## 1st Qu.:4.000 1st Qu.:2.000 1st Qu.:2.000 1st Qu.:1.000
## Median :4.000  Median :4.000  Median :3.000  Median :3.000
## Mean   :3.961  Mean   :3.401  Mean   :2.829  Mean   :2.591
## 3rd Qu.:5.000 3rd Qu.:4.000 3rd Qu.:4.000 3rd Qu.:3.000
## Max.  :5.000  Max.  :5.000  Max.  :5.000  Max.  :5.000
##
##      TAS14         TAS15         TAS16         TAS17
## Min.  :1.00  Min.  :1.00  Min.  :1.000  Min.  :1.000
## 1st Qu.:1.00 1st Qu.:2.00 1st Qu.:1.000 1st Qu.:1.000
## Median :3.00  Median :3.00  Median :3.000  Median :2.000
## Mean   :2.68  Mean   :2.87  Mean   :2.562  Mean   :2.442
## 3rd Qu.:4.00 3rd Qu.:4.00 3rd Qu.:3.000 3rd Qu.:4.000
## Max.  :5.00  Max.  :5.00  Max.  :5.000  Max.  :5.000
##
##      TAS18         TAS19         TAS20        TAS_total
## Min.  :1.000  Min.  :2.000  Min.  :1.000  Min.  :31.0
## 1st Qu.:3.000 1st Qu.:4.000 1st Qu.:1.000 1st Qu.:47.0
## Median :4.000  Median :4.000  Median :2.000  Median :51.0
## Mean   :4.049  Mean   :4.094  Mean   :2.365  Mean   :51.3
## 3rd Qu.:5.000 3rd Qu.:5.000 3rd Qu.:3.000 3rd Qu.:57.0
## Max.  :5.000  Max.  :5.000  Max.  :5.000  Max.  :70.0
##
##      STAIYA1       STAIYA2       STAIYA3       STAIYA4
## Min.  :1.00  Min.  :1.000  Min.  :1.000  Min.  :1.000
## 1st Qu.:1.00 1st Qu.:1.000 1st Qu.:1.000 1st Qu.:1.000
## Median :2.00  Median :2.000  Median :1.000  Median :2.000
## Mean   :2.06  Mean   :1.909  Mean   :1.287  Mean   :1.747
## 3rd Qu.:3.00 3rd Qu.:2.000 3rd Qu.:1.000 3rd Qu.:2.000
## Max.  :4.00  Max.  :4.000  Max.  :3.000  Max.  :4.000
##
##      STAIYA5       STAIYA6       STAIYA7       STAIYA8
## Min.  :1.000  Min.  :1.000  Min.  :1.000  Min.  :1.000
## 1st Qu.:2.000 1st Qu.:1.000 1st Qu.:1.000 1st Qu.:2.000
## Median :2.000  Median :1.000  Median :1.000  Median :2.000
## Mean   :2.412  Mean   :1.584  Mean   :1.289  Mean   :2.469
## 3rd Qu.:3.000 3rd Qu.:2.000 3rd Qu.:2.000 3rd Qu.:3.000
## Max.  :4.000  Max.  :4.000  Max.  :3.000  Max.  :4.000
##
##      STAIYA9       STAIYA10      STAIYA11      STAIYA12
## Min.  :1.000  Min.  :1.000  Min.  :1.000  Min.  :1.000
## 1st Qu.:1.000 1st Qu.:1.000 1st Qu.:1.000 1st Qu.:1.000

```

```

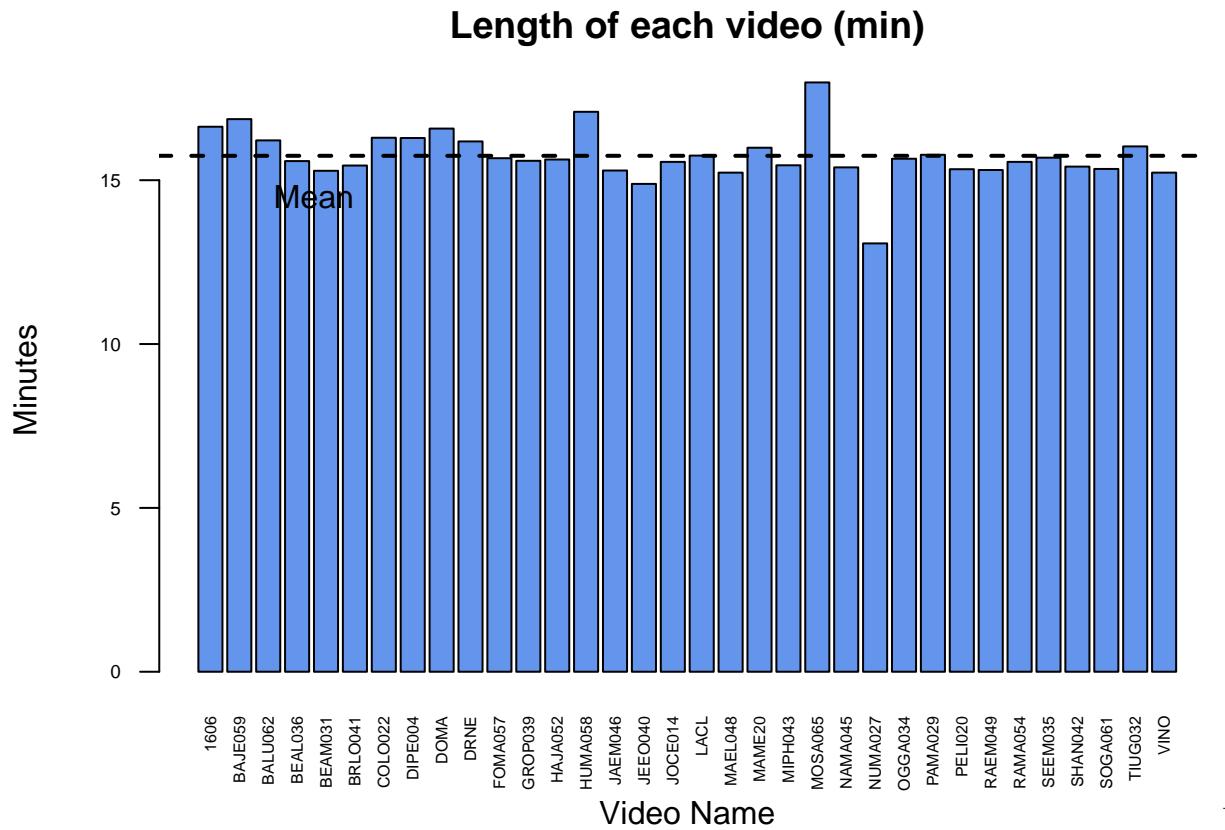
## Median :2.000 Median :2.000 Median :1.000 Median :1.000
## Mean   :1.764 Mean   :1.701 Mean   :1.498 Mean   :1.554
## 3rd Qu.:2.000 3rd Qu.:2.000 3rd Qu.:2.000 3rd Qu.:2.000
## Max.   :3.000 Max.   :3.000 Max.   :4.000 Max.   :3.000
##
##      STAIYA13      STAIYA14      STAIYA15      STAIYA16
## Min.   :1.000 Min.   :1.000 Min.   :1.000 Min.   :1.000
## 1st Qu.:1.000 1st Qu.:1.000 1st Qu.:1.000 1st Qu.:1.000
## Median :1.000 Median :2.000 Median :2.000 Median :1.000
## Mean   :1.403 Mean   :1.879 Mean   :1.824 Mean   :1.703
## 3rd Qu.:2.000 3rd Qu.:2.000 3rd Qu.:2.000 3rd Qu.:2.000
## Max.   :3.000 Max.   :4.000 Max.   :4.000 Max.   :4.000
##
##      STAIYA17      STAIYA18      STAIYA19      STAIYA20
## Min.   :1.000 Min.   :1.000 Min.   :1.000 Min.   :1.000
## 1st Qu.:1.000 1st Qu.:1.000 1st Qu.:2.000 1st Qu.:1.000
## Median :2.000 Median :1.000 Median :2.000 Median :1.000
## Mean   :1.967 Mean   :1.728 Mean   :2.178 Mean   :1.498
## 3rd Qu.:3.000 3rd Qu.:2.000 3rd Qu.:3.000 3rd Qu.:2.000
## Max.   :4.000 Max.   :4.000 Max.   :3.000 Max.   :3.000
##
##      STAIYA_total    STAIYB1      STAIYB2      STAIYB3
## Min.   :25.00  Min.   :1.0  Min.   :1.000 Min.   :2.00
## 1st Qu.:30.00 1st Qu.:1.0  1st Qu.:2.000 1st Qu.:2.00
## Median :34.00  Median :2.0  Median :2.000 Median :3.00
## Mean   :35.46  Mean   :1.7  Mean   :1.942 Mean   :2.65
## 3rd Qu.:39.00 3rd Qu.:2.0  3rd Qu.:2.000 3rd Qu.:3.00
## Max.   :56.00  Max.   :3.0  Max.   :3.000 Max.   :4.00
##
##      STAIYB4      STAIYB5      STAIYB6      STAIYB7
## Min.   :1.000 Min.   :1.000 Min.   :1.000 Min.   :1.000
## 1st Qu.:1.000 1st Qu.:1.000 1st Qu.:2.000 1st Qu.:2.000
## Median :2.000 Median :2.000 Median :3.000 Median :2.000
## Mean   :2.056 Mean   :1.914 Mean   :2.621 Mean   :2.302
## 3rd Qu.:3.000 3rd Qu.:2.000 3rd Qu.:3.000 3rd Qu.:3.000
## Max.   :4.000 Max.   :4.000 Max.   :4.000 Max.   :4.000
## NA's   :22331          NA's   :22967
##
##      STAIYB8      STAIYB9      STAIYB10     STAIYB11
## Min.   :1.000 Min.   :1.000 Min.   :1.000 Min.   :1.000
## 1st Qu.:1.000 1st Qu.:2.000 1st Qu.:1.000 1st Qu.:2.000
## Median :2.000 Median :2.000 Median :1.000 Median :2.000
## Mean   :1.711 Mean   :2.319 Mean   :1.497 Mean   :2.293
## 3rd Qu.:2.000 3rd Qu.:3.000 3rd Qu.:2.000 3rd Qu.:3.000
## Max.   :3.000 Max.   :4.000 Max.   :3.000 Max.   :4.000
##
##      STAIYB12     STAIYB13     STAIYB14     STAIYB15
## Min.   :1.000 Min.   :1.000 Min.   :1.000 Min.   :1.000
## 1st Qu.:2.000 1st Qu.:1.000 1st Qu.:2.000 1st Qu.:1.000
## Median :3.000 Median :2.000 Median :3.000 Median :2.000
## Mean   :2.593 Mean   :1.864 Mean   :2.752 Mean   :1.917
## 3rd Qu.:4.000 3rd Qu.:2.000 3rd Qu.:3.000 3rd Qu.:2.000
## Max.   :4.000 Max.   :4.000 Max.   :4.000 Max.   :4.000
##
##      STAIYB16     STAIYB17     STAIYB18     STAIYB19

```

```

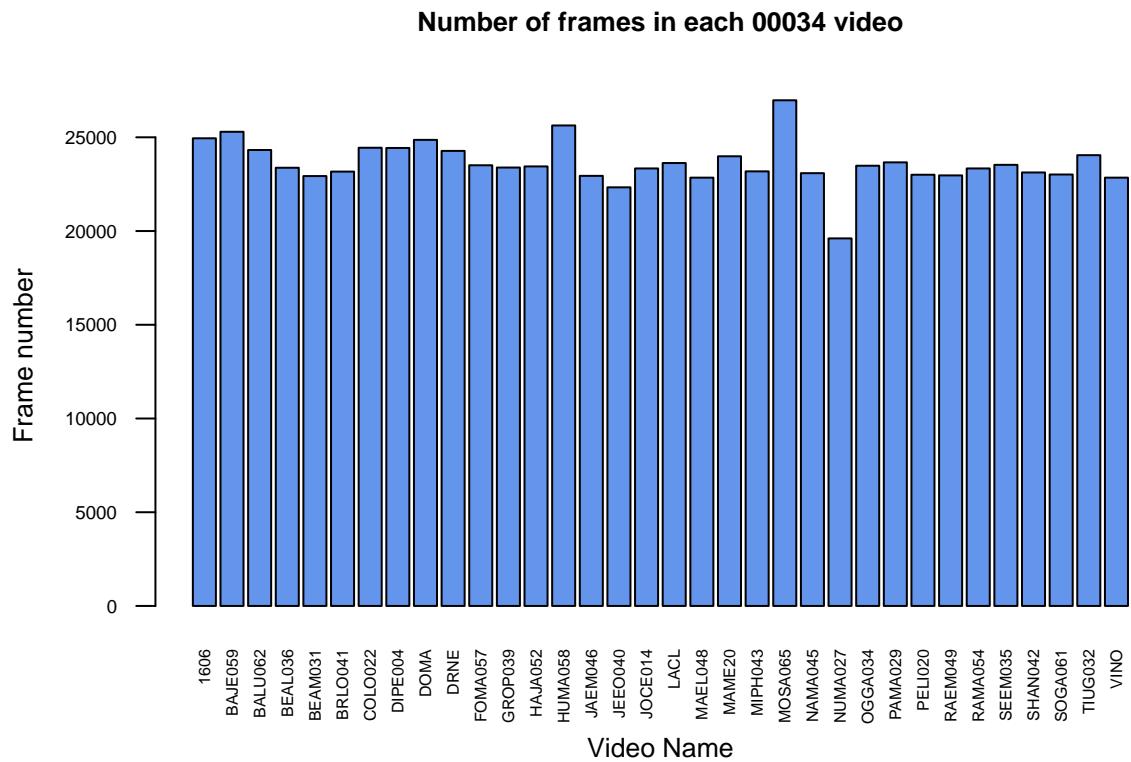
## Min. :1.00  Min. :1.000  Min. :1.000  Min. :1.000
## 1st Qu.:2.00 1st Qu.:2.000 1st Qu.:1.000 1st Qu.:2.000
## Median :2.00 Median :2.000 Median :1.000 Median :2.000
## Mean :2.23 Mean :2.101 Mean :1.879 Mean :2.183
## 3rd Qu.:3.00 3rd Qu.:2.000 3rd Qu.:3.000 3rd Qu.:2.000
## Max. :4.00 Max. :4.000 Max. :4.000 Max. :4.000
##
##      STAIYB20      STAIYB_total      BDI1      BDI2
## Min. :1.000  Min. :26.00  Min. :0.000  Min. :0.0000
## 1st Qu.:2.000 1st Qu.:35.00 1st Qu.:0.000 1st Qu.:0.0000
## Median :2.000 Median :43.00  Median :0.000  Median :0.0000
## Mean :2.323 Mean :42.72  Mean :0.349  Mean :0.5063
## 3rd Qu.:3.000 3rd Qu.:47.00 3rd Qu.:1.000 3rd Qu.:1.0000
## Max. :4.000 Max. :70.00  Max. :2.000  Max. :3.0000
##
##      BDI3      BDI4      BDI5      BDI6
## Min. :0.0000  Min. :0.0000  Min. :0.0000  Min. :0.0000
## 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:0.0000
## Median :0.0000 Median :0.0000 Median :0.0000 Median :0.0000
## Mean :0.2349 Mean :0.2043 Mean :0.6359 Mean :0.4427
## 3rd Qu.:0.0000 3rd Qu.:0.0000 3rd Qu.:1.0000 3rd Qu.:1.0000
## Max. :1.0000 Max. :1.0000 Max. :3.0000 Max. :1.0000
##
##      BDI7      BDI8      BDI9      BDI10
## Min. :0.000  Min. :0.0000  Min. :0.0000  Min. :0.0000
## 1st Qu.:0.000 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:0.0000
## Median :0.000 Median :0.0000 Median :1.0000 Median :0.0000
## Mean :0.118 Mean :0.1426 Mean :0.8647 Mean :0.4389
## 3rd Qu.:0.000 3rd Qu.:0.0000 3rd Qu.:1.0000 3rd Qu.:1.0000
## Max. :1.000 Max. :1.0000 Max. :2.0000 Max. :3.0000
##
##      BDI11      BDI12      BDI13      BDI_total
## Min. :0.0000  Min. :0.0000  Min. :0.0000  Min. : 0.000
## 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.: 3.000
## Median :1.0000 Median :1.0000 Median :0.0000 Median : 5.000
## Mean :0.7998 Mean :0.8224 Mean :0.3697 Mean : 5.929
## 3rd Qu.:1.0000 3rd Qu.:1.0000 3rd Qu.:1.0000 3rd Qu.: 7.000
## Max. :3.0000 Max. :2.0000 Max. :2.0000 Max. :17.000
##
##      LabelVideo
## Length:802970
## Class :character
## Mode :character
##
##
```

Length of the videos in minutes



We can see that the video are very comparable. THere is a shorter video in which the dyad doesn't really understand the request. The mother stand up at the end and go to see the experimenter.

Length of the videos in number of frames

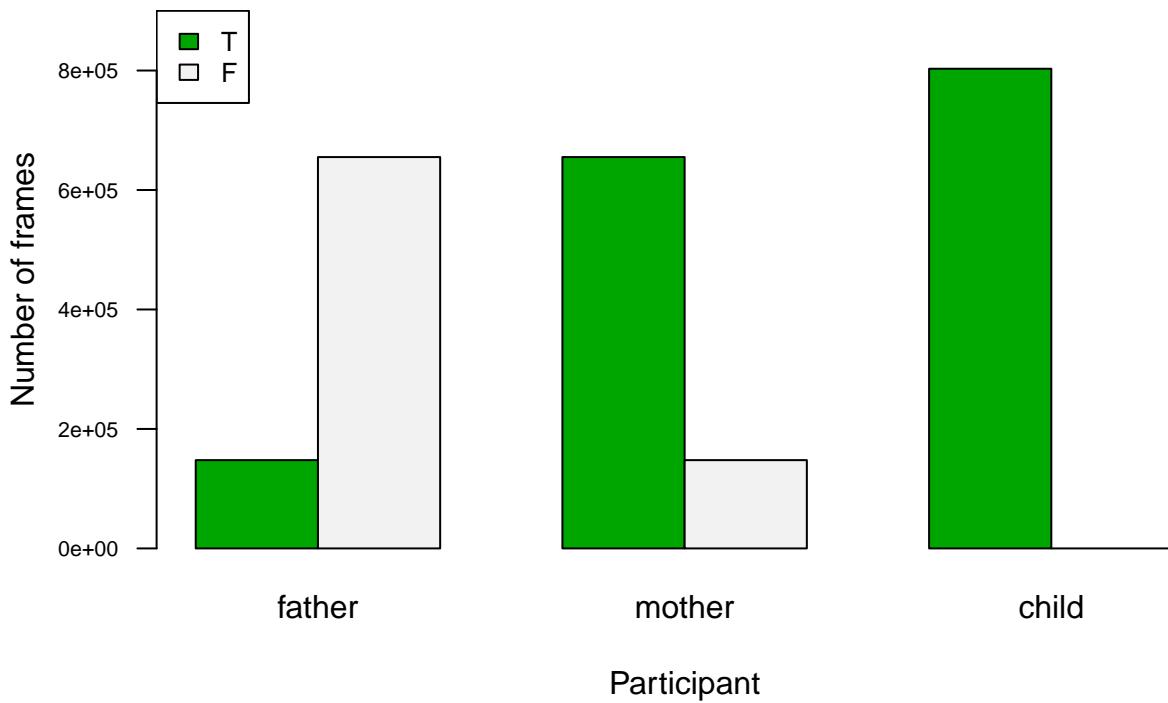


Configurations of the videos

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

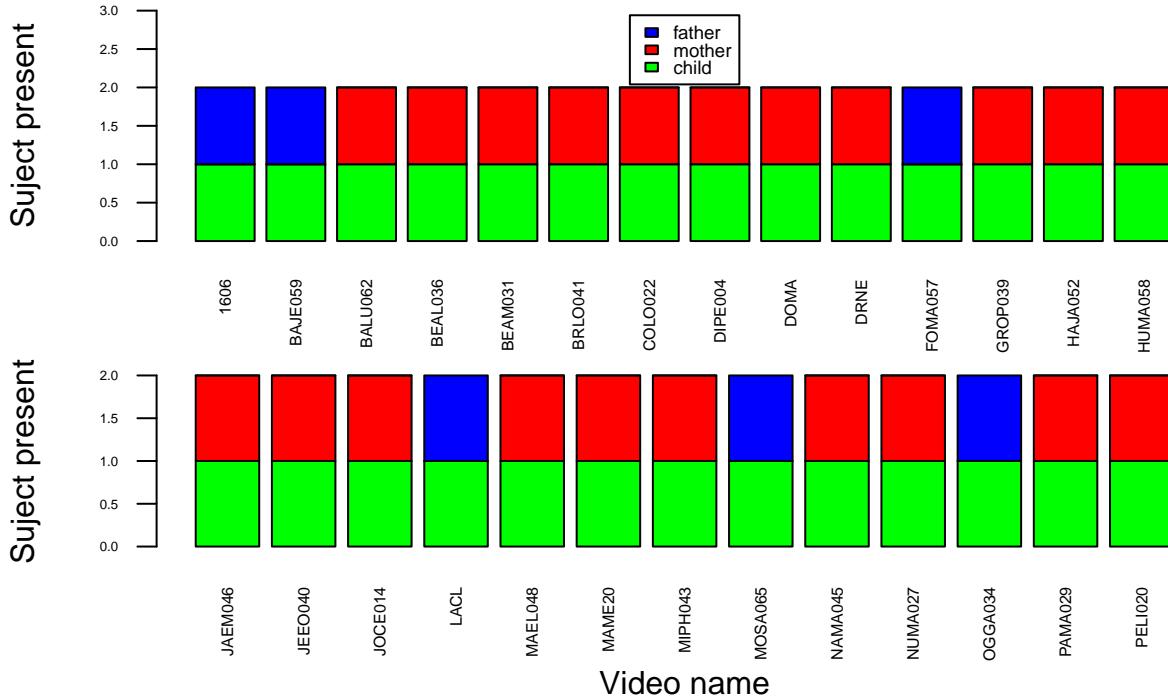
```
##      [,1]  [,2]  [,3]
## FALSE 147834 655136 802970
## TRUE   655136 147834      0
```

Number of available data by participant



- All the participants involved are filmed.
- All the children are filmed and we have all the data for each.
- More often there is the mother with him/her but sometimes, it is the father

Configuration of participant by video

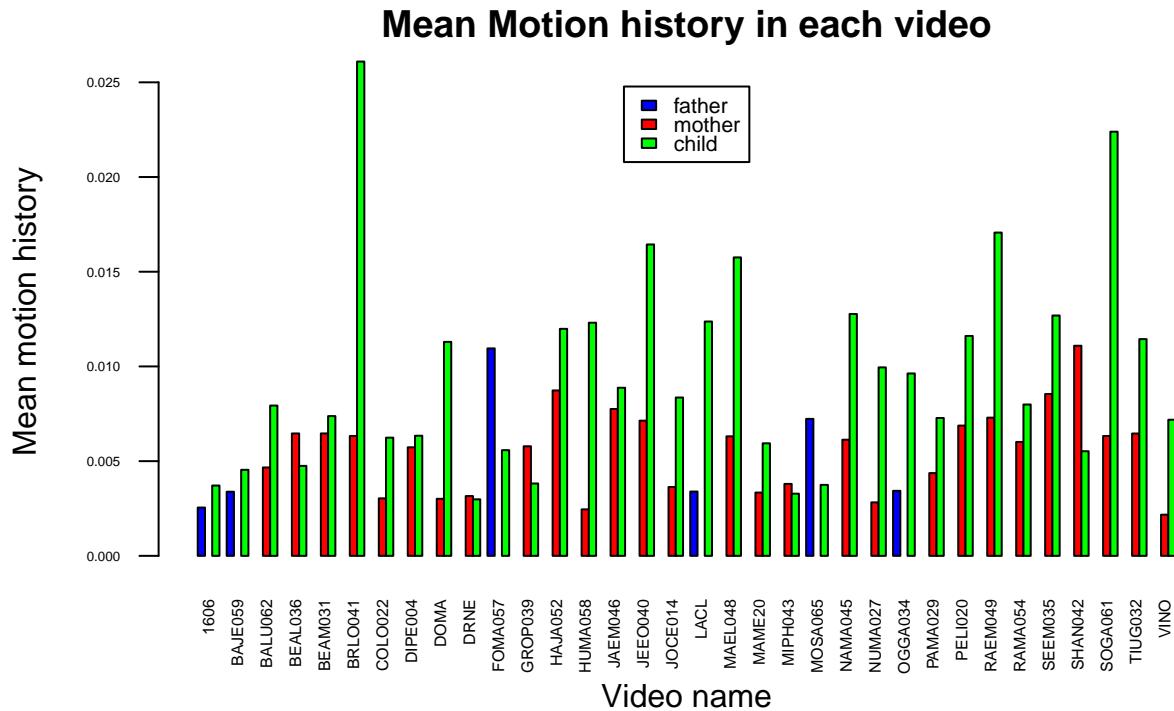


We can see that configurations of subjects are very similar (with always 2 subjects, we excluded RAKA008 with 3 subjects). More often the child is with his mother. Consequently, it makes the comparisons of the

videos easier than in the INCANT study.

Global Motion history

Mean Motion history by video by participant



Global motion history by situation : “conflict” vs “no conflict”

```

MeanMotionNoConflict <- c(
  mean(data[which(data$LabelVideo=="No Conflict"),]$father, na.rm=TRUE),
  mean(data[which(data$LabelVideo=="No Conflict"),]$mother, na.rm=TRUE),
  mean(data[which(data$LabelVideo=="No Conflict"),]$child, na.rm=TRUE))

MeanMotionConflict <- c(
  mean(data[which(data$LabelVideo=="Conflict")],$father, na.rm=TRUE),
  mean(data[which(data$LabelVideo=="Conflict")],$mother, na.rm=TRUE),
  mean(data[which(data$LabelVideo=="Conflict")],$child, na.rm=TRUE))

MeanMotion<- data.frame(MeanMotionNoConflict, MeanMotionConflict, names=c("father", "mother", "child"))

##   MeanMotionNoConflict MeanMotionConflict  names
## 1      0.004555560     0.005024248 father
## 2      0.005280418     0.005301225 mother
## 3      0.009970003     0.008760330  child

par(mar=c(5,4,4,1))
barplot (as.matrix(MeanMotion[,1:2]), beside=TRUE, ylab= "Mean Motion history", col=colOrderList, names

```

```

    without and with conflict")
par(mar=c(1,0.5,0.5,1))
legend("top", inset=.05, ParticipantsList,
      fill=colOrderList, cex=0.7)

```

](SyncPsychoMonrado_files/figure-latex/raw Global motion history by situation : “conflict” vs “no conflict”-1.pdf)

We can see that it seems to be a decrease of movement in conflict among child but not much change among parents.

Log of Global motion history by situation : “conflict” vs “no conflict”

```

MeanMotionNoConflict <- c(
  mean(data[which(data$LabelVideo=="No Conflict")],logFather, na.rm=TRUE),
  mean(data[which(data$LabelVideo=="No Conflict")],logMother, na.rm=TRUE),
  mean(data[which(data$LabelVideo=="No Conflict")],logChild, na.rm=TRUE))

MeanMotionConflict <- c(
  mean(data[which(data$LabelVideo=="Conflict")],logFather, na.rm=TRUE),
  mean(data[which(data$LabelVideo=="Conflict")],logMother, na.rm=TRUE),
  mean(data[which(data$LabelVideo=="Conflict")],logChild, na.rm=TRUE))

MeanMotionlog <- data.frame(MeanMotionNoConflict, MeanMotionConflict, names=c("father", "mother", "child"))

##   MeanMotionNoConflict MeanMotionConflict   names
## 1      0.004555560     0.005024248   father
## 2      0.005280418     0.005301225   mother
## 3      0.009970003     0.008760330   child

par(mar=c(5,4,4,1))
barplot (as.matrix(MeanMotionlog[,1:2]), beside=TRUE, ylab= "Log of Mean Motion history", col=colOrderList,
          without and with conflict")
legend("topleft", inset=.02, ParticipantsList,
      fill=colOrderList, cex=0.6)

```

](SyncPsychoMonrado_files/figure-latex/raw log Global motion history by situation : “conflict” vs “no conflict”-1.pdf)

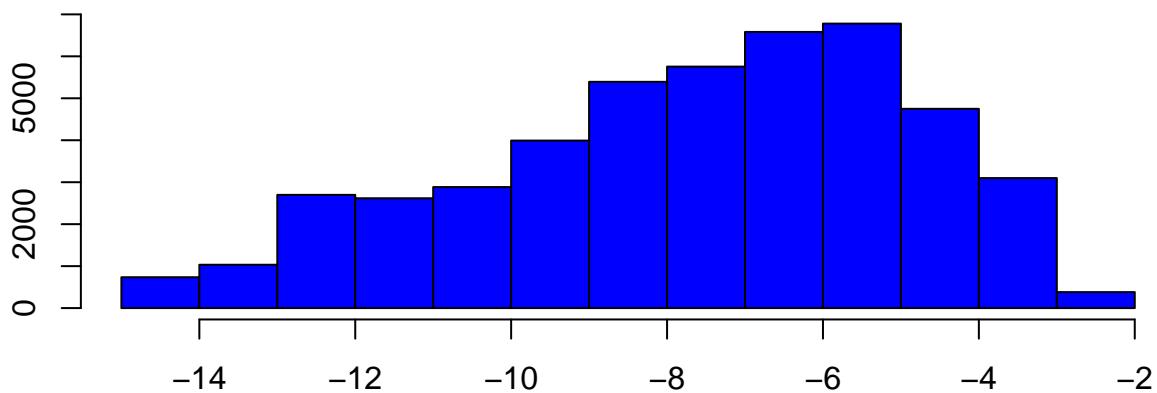
With a log distribution of motion history, we can't see any more this difference.

```

par(mar=c(3,3,4,1))
hist(data[which(data$LabelVideo=="No Conflict")],logFather, col="blue", main= "Motion History histogram")

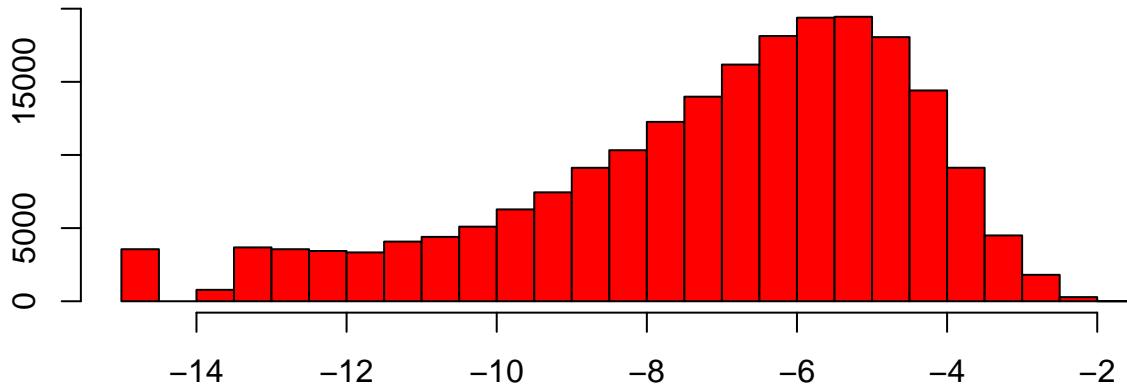
```

Motion History histogram for father, no conflict



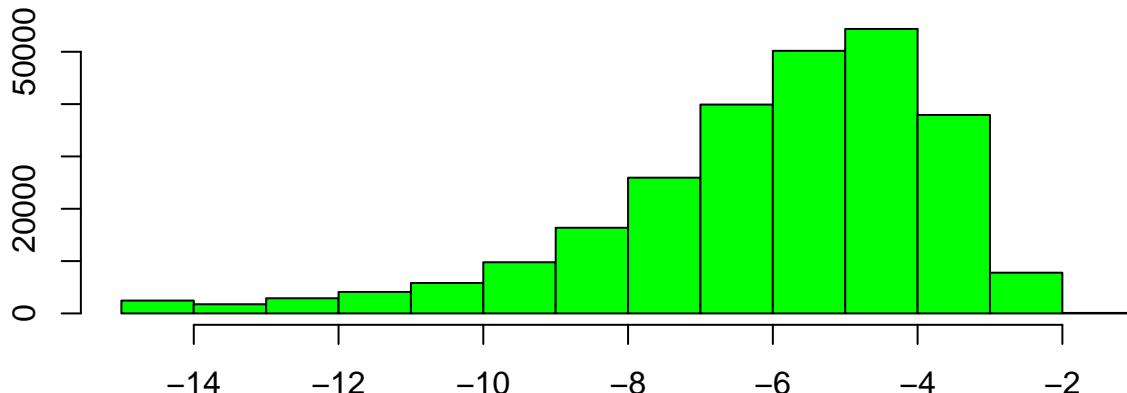
```
hist(data[which(data$LabelVideo=="No Conflict"),]$logMother, col="red", main= "Motion History histogram for father, no conflict")
```

Motion History histogram for mother, no conflict



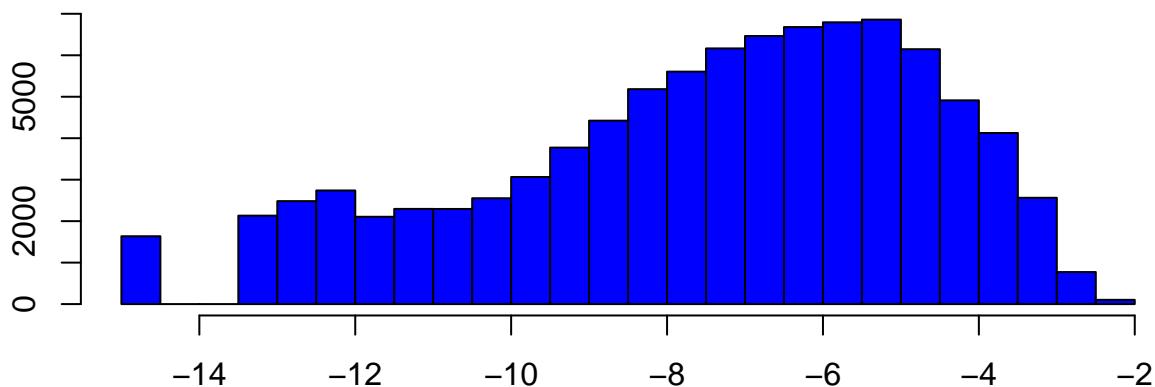
```
hist(data[which(data$LabelVideo=="No Conflict"),]$logChild, col="green", main= "Motion History histogram for mother, no conflict")
```

Motion History histogram for child, no conflict



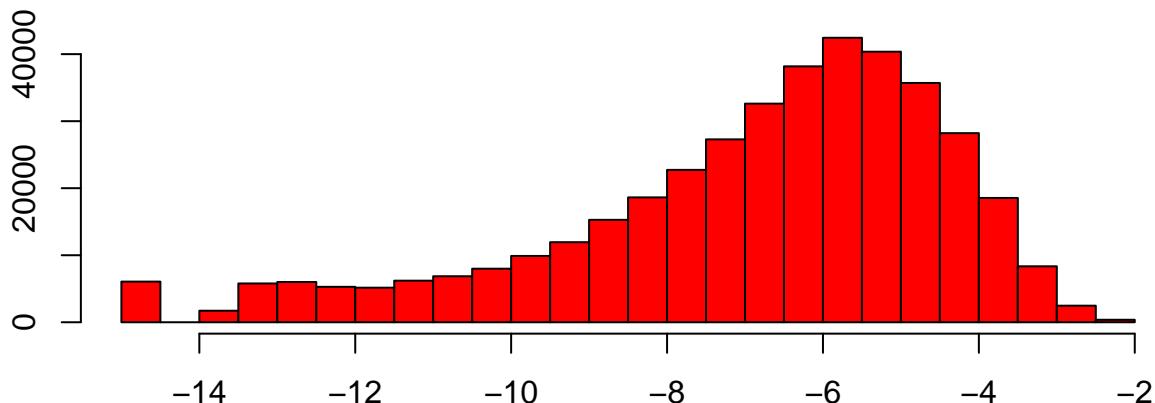
```
hist(data[which(data$LabelVideo=="Conflict"),]$logFather, col="blue", main= "Motion History histogram for father, conflict")
```

Motion History histogram for father, in conflict



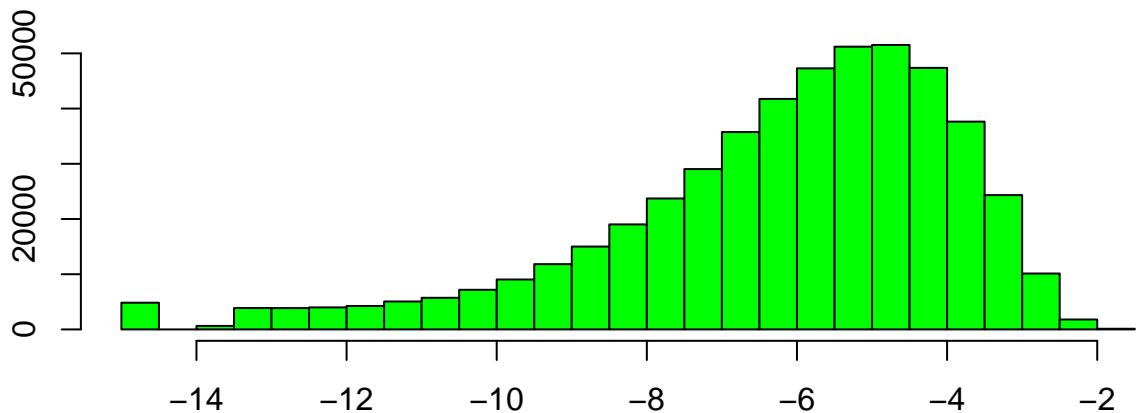
```
hist(data[which(data$LabelVideo=="Conflict")],]$logMother, col="red", main= "Motion History histogram for father in conflict")
```

Motion History histogram for mother, in conflict



```
hist(data[which(data$LabelVideo=="Conflict")],]$logChild, col="green", main= "Motion History histogram for mother in conflict")
```

Motion History histogram for child, in conflict



```
table(psycho$attachement_cluster)
```

```
##  
##          DU      DUAL        FE      Secure Withdrawn  
##          1           2         13        15          3
```

If we exclude dual attachement styles which are more complicated, heterogenous and not very numerous : BRLO041 (Withdrawn Fearful), HAJA052 (Angry Dismissive Fearful). We got 4 attachement styles that we can compare.

```

attachementStyles <- c("FE", "Secure", "Withdrawn", "DU")
for (i in attachementStyles){
  MeanMotionNoConflict <- c(
    mean(data[which(data$LabelVideo=="No Conflict" & data$attachement_cluster==i),]$father, na.rm=TRUE),
    mean(data[which(data$LabelVideo=="No Conflict" & data$attachement_cluster==i),]$mother, na.rm=TRUE),
    mean(data[which(data$LabelVideo=="No Conflict" & data$attachement_cluster==i),]$child, na.rm=TRUE))

  MeanMotionConflict <- c(
    mean(data[which(data$LabelVideo=="Conflict"& data$attachement_cluster==i),]$father , na.rm=TRUE),
    mean(data[which(data$LabelVideo=="Conflict"& data$attachement_cluster==i),]$mother , na.rm=TRUE),
    mean(data[which(data$LabelVideo=="Conflict" & data$attachement_cluster==i),]$child, na.rm=TRUE))

  #print(i)
  #print (MeanMotionNoConflict)
  #print (MeanMotionConflict)

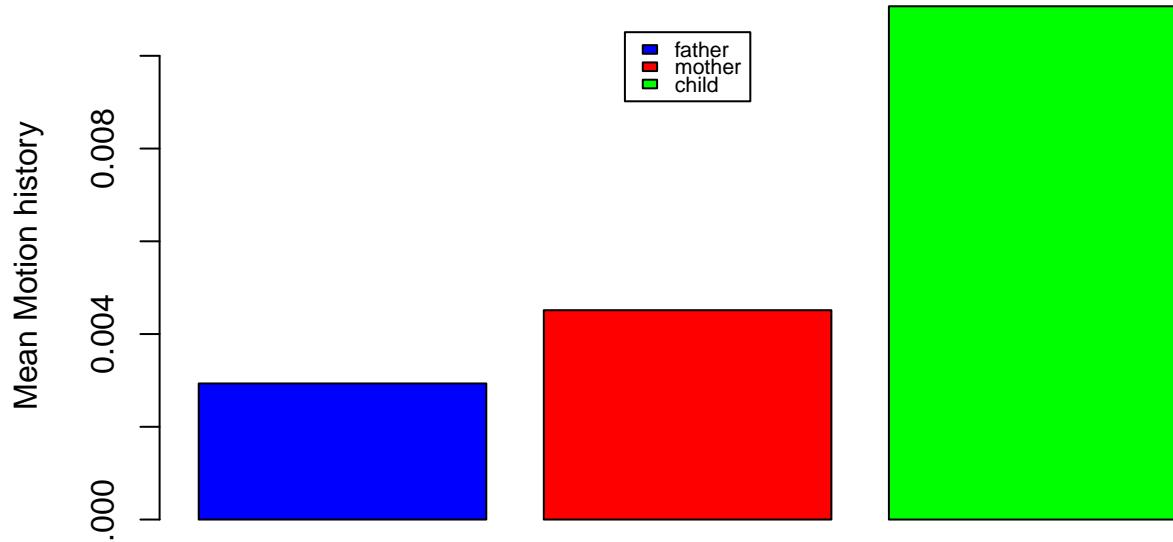
  par(mar=c(5,4,4,1))
  barplot (MeanMotionNoConflict, beside=FALSE, ylab = "Mean Motion history", col=colOrderList, main="without conflict, attachement cluster", i))
  par(mar=c(1,0.5,0.5,1))
  legend("top", inset=.05, ParticipantsList,
         fill=colOrderList, cex=0.7)

  par(mar=c(5,4,4,1))
  barplot (MeanMotionConflict, beside=TRUE, ylab = "Mean Motion history", col=colOrderList, main="with conflict, attachement cluster", i))
  par(mar=c(1,0.5,0.5,1))
  legend("top", inset=.05, ParticipantsList,
         fill=colOrderList, cex=0.7)
}

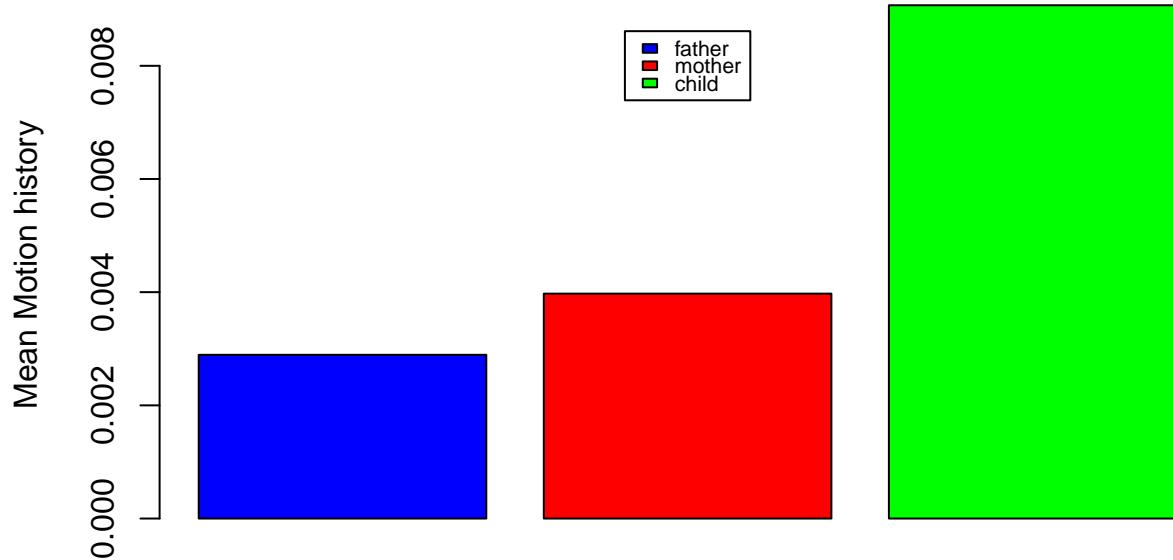
}

```

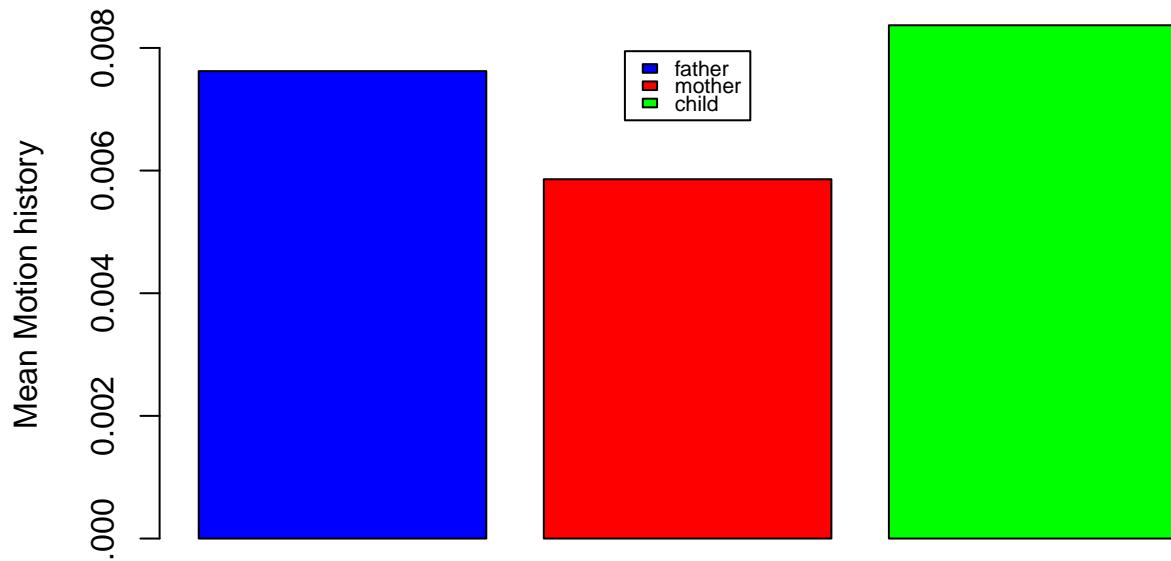
**Mean Motion history for each participant
without conflict, attachment cluster FE**



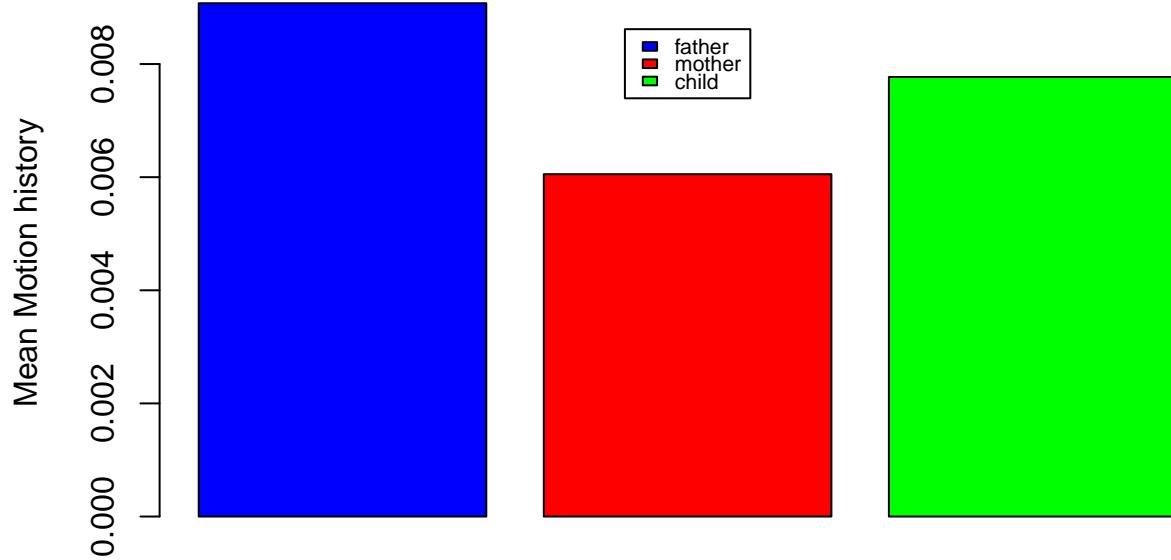
**Mean Motion history for each participant
with conflict, attachment cluster FE**



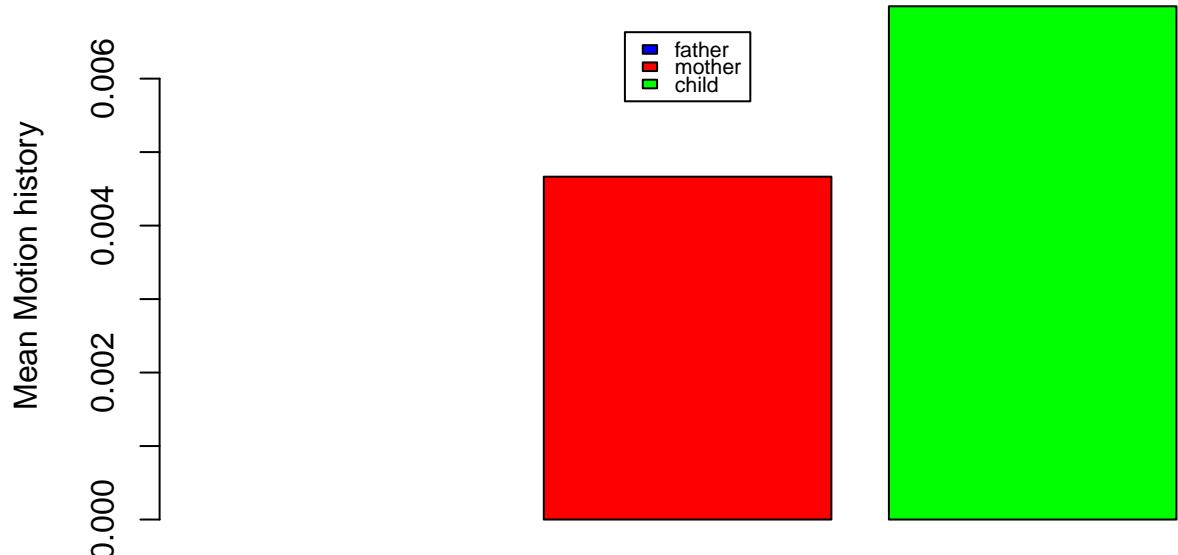
**Mean Motion history for each participant
without conflict, attachment cluster Secure**



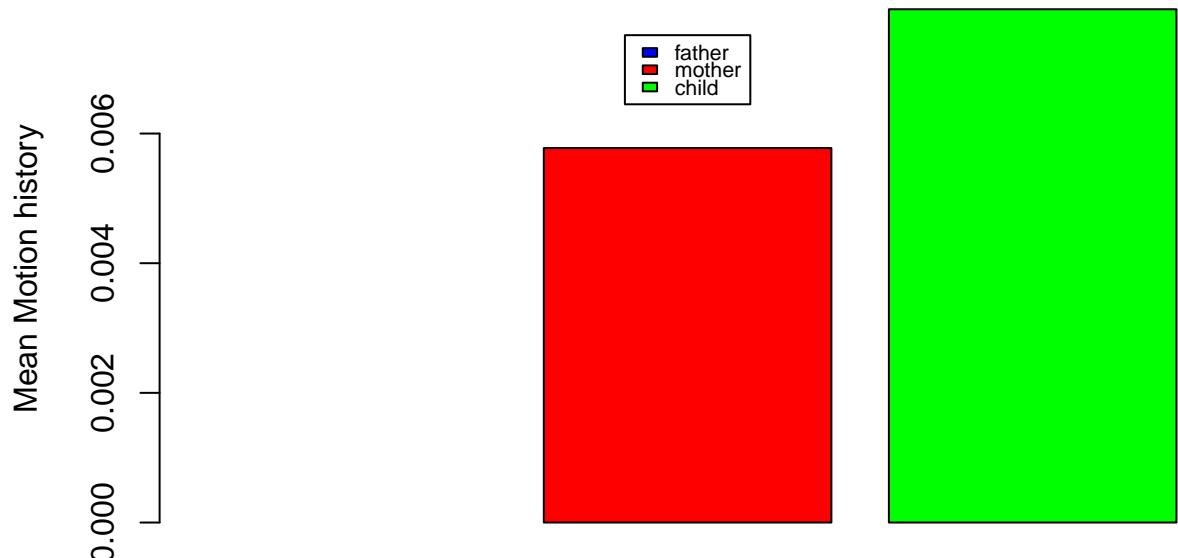
**Mean Motion history for each participant
with conflict, attachment cluster Secure**



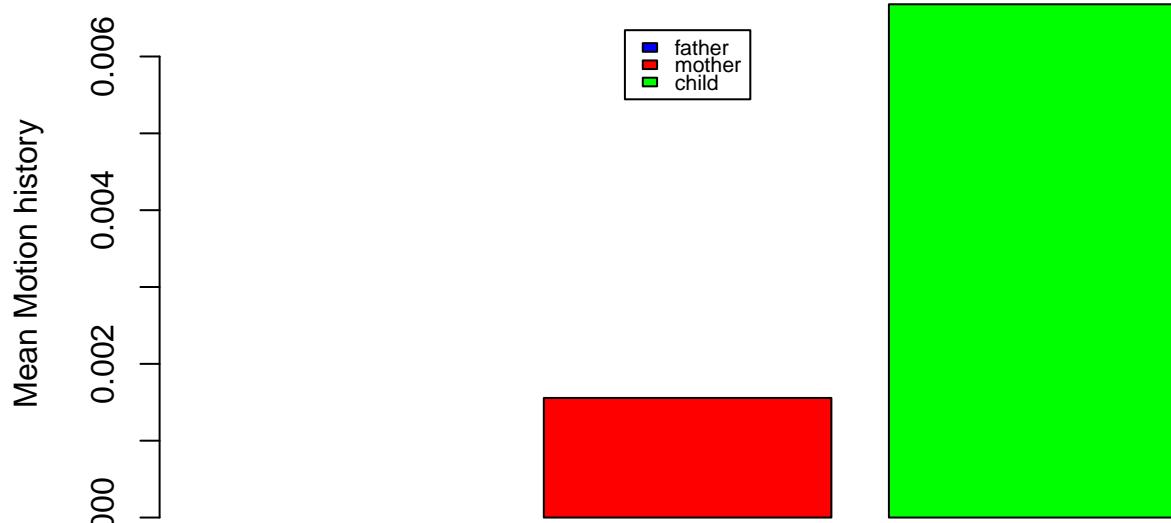
**Mean Motion history for each participant
without conflict, attachement cluster Withdrawn**



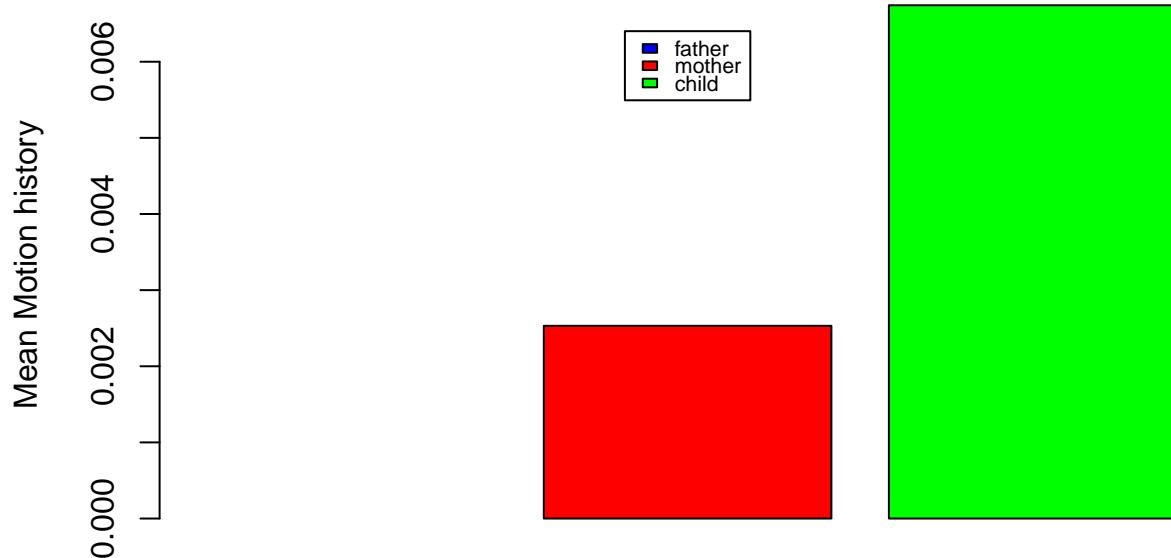
**Mean Motion history for each participant
with conflict, attachement cluster Withdrawn**



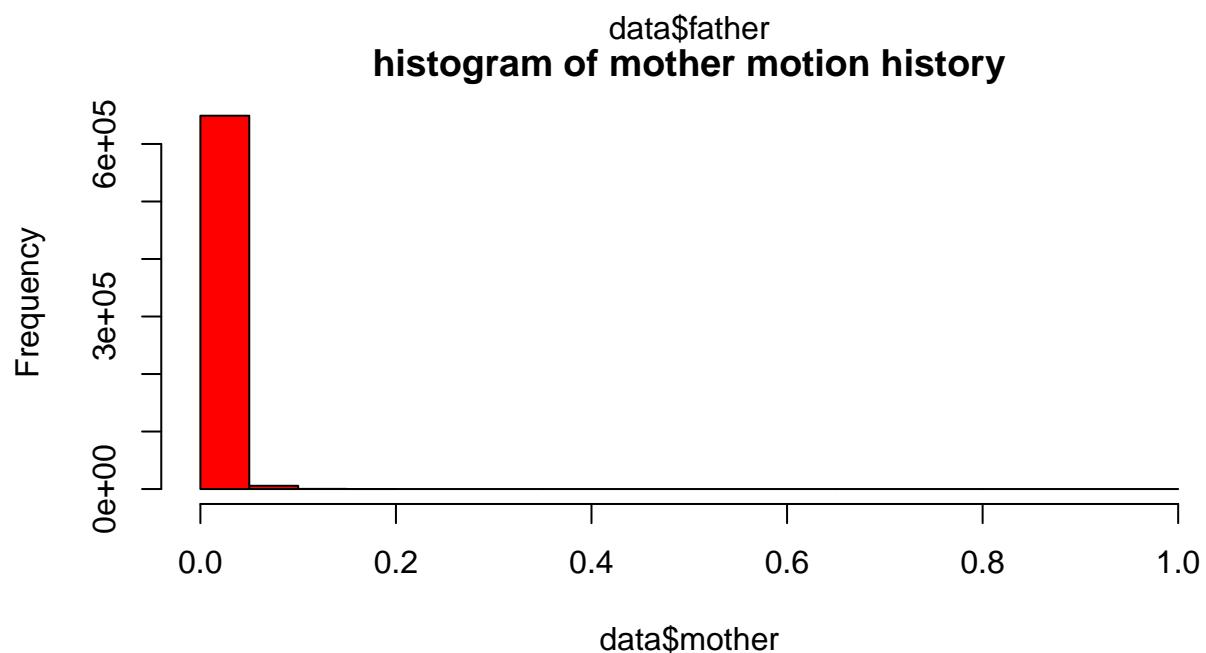
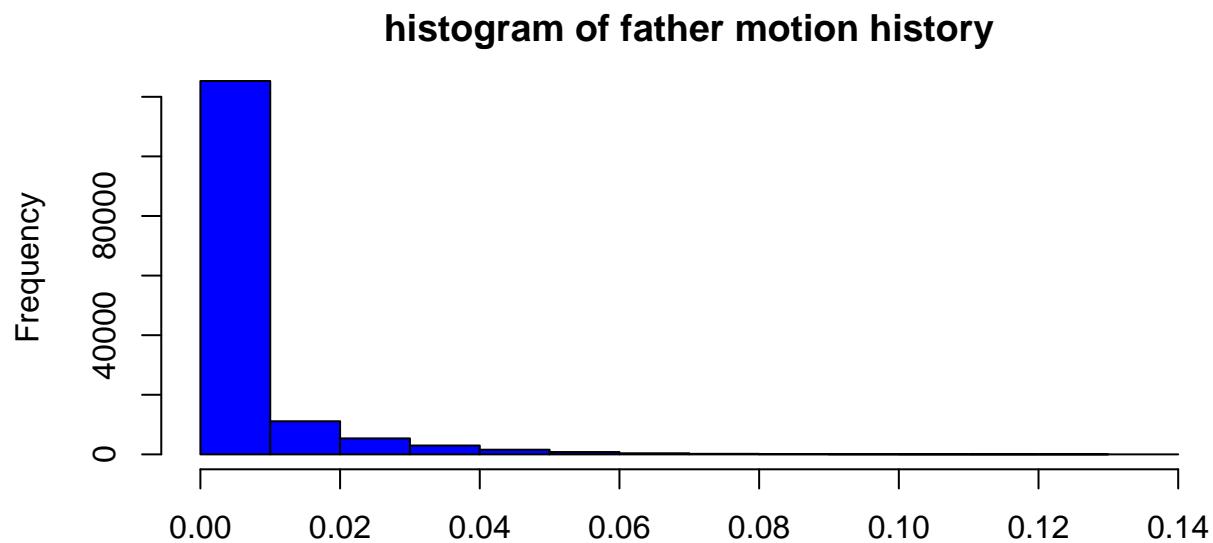
**Mean Motion history for each participant
without conflict, attachement cluster DU**

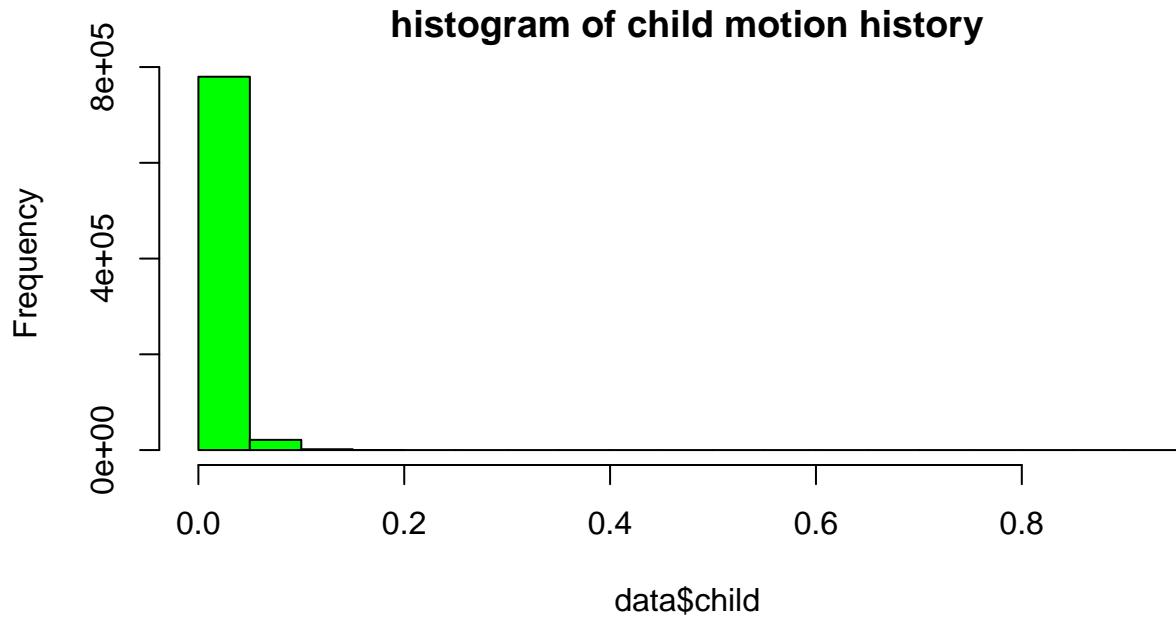


**Mean Motion history for each participant
with conflict, attachement cluster DU**



Motion history histogram by frame (raw data), all videos



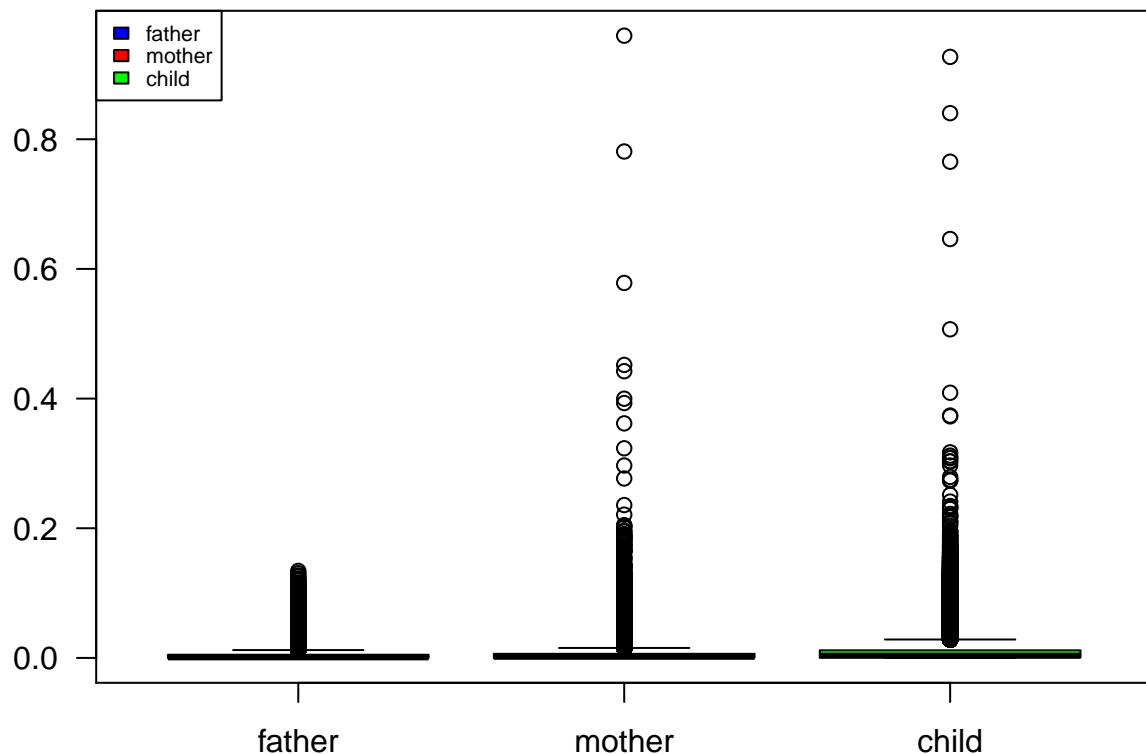


The motion history is not normalized at all. Most of motions are very small but some of them are much big (long tail). This is very usual with this algorithm extraction motion history.

The subjects data are very similar.

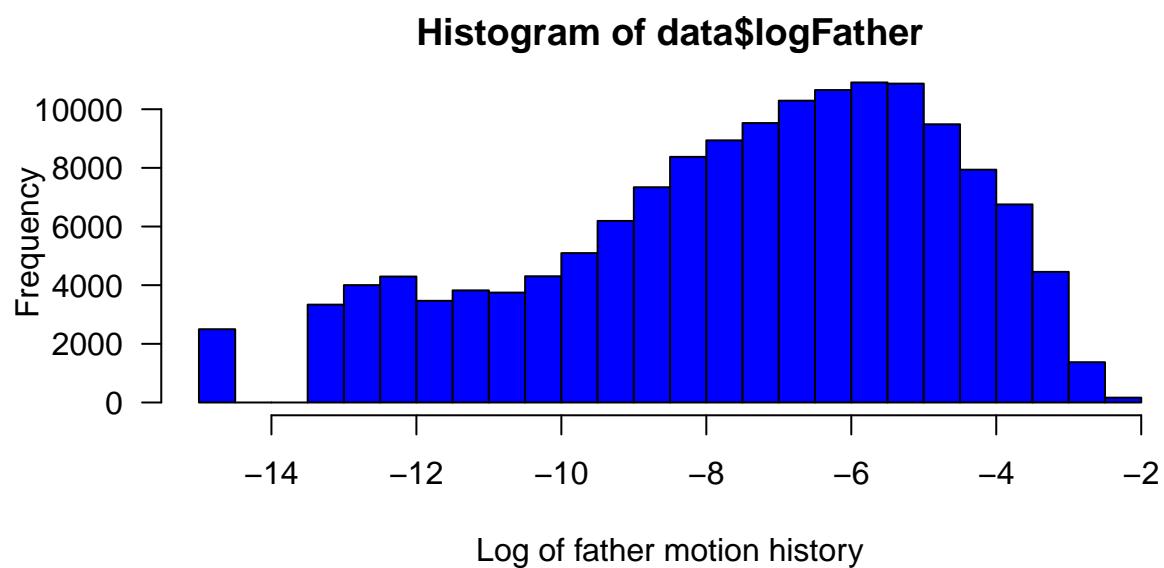
```
par(mar=c(3,3,2,2))
boxplot(data$father, data$mother, data$child,
        col=colOrderList,
        names=ParticipantsList,
        main= "Motion history by frame box plots (raw data), all videos", las=1)
par(mar=c(1,0.5,0.5,1))
legend("topleft", ParticipantsList, fill=colOrderList, cex=0.7)
```

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



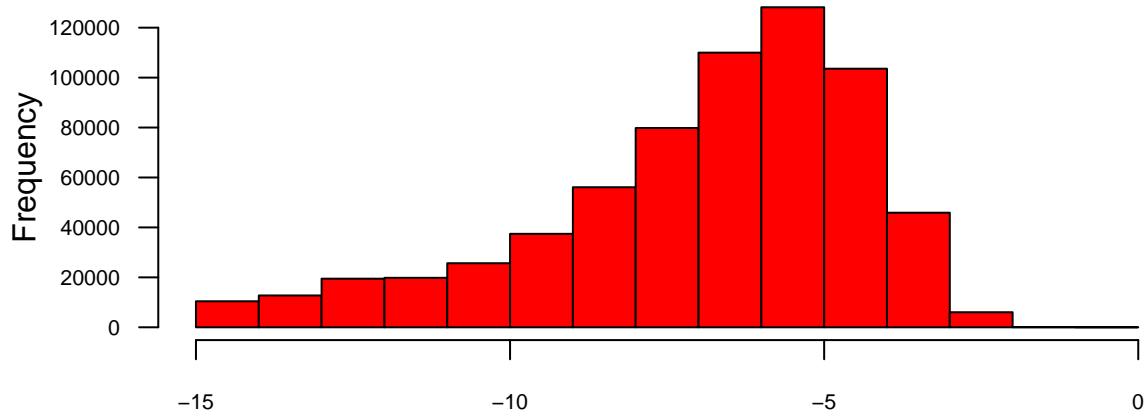
The subjects data distribution are very similar.

```
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)
```

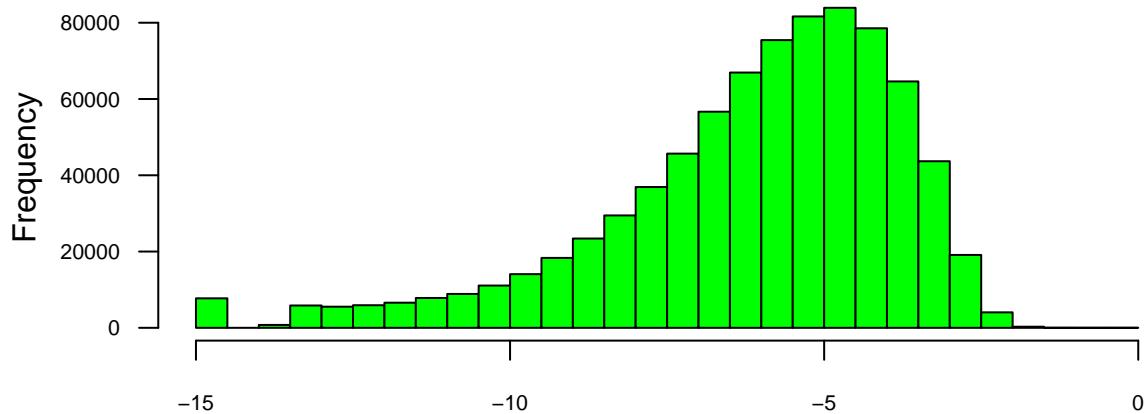
Histogram of data\$logMother



Log of mother motion history

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

Histogram of data\$logChild

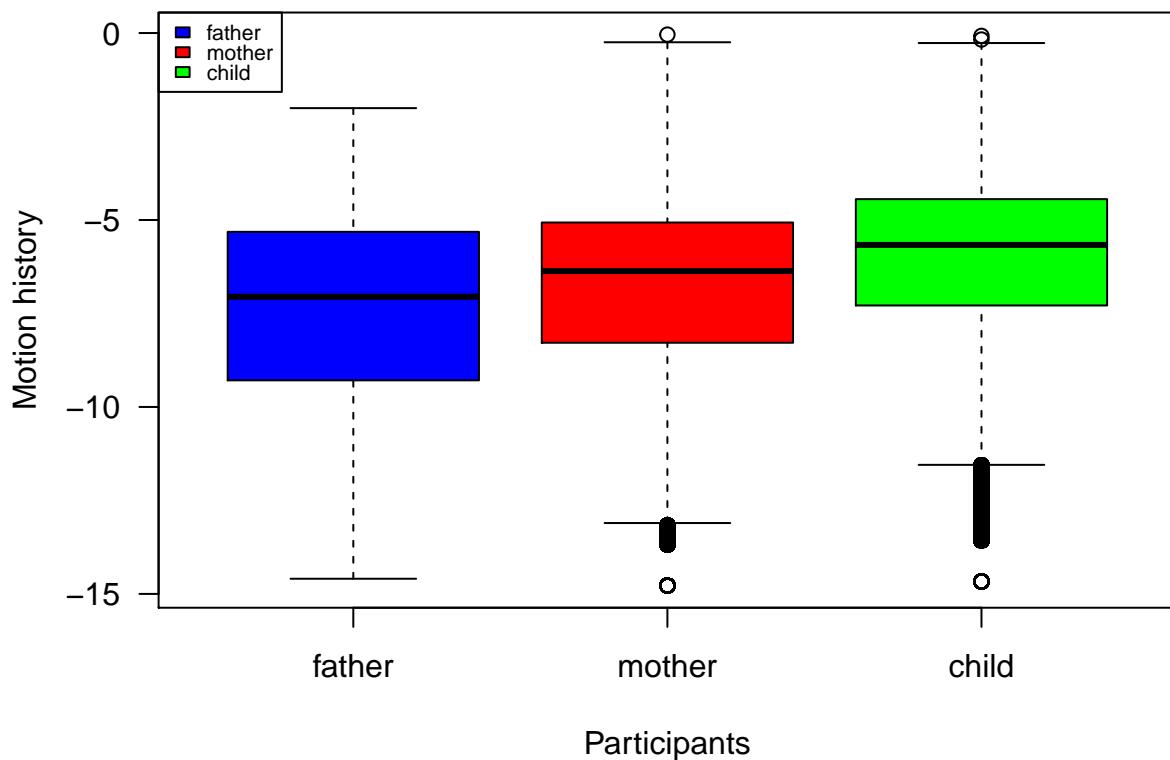


Log of child motion history

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

```
data$childShifted <- data$child + min(data$child[which(data$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)
```

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



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
slidedFather <- SlidingInterval("father", 1 , 5, data)
slidedChild <- SlidingInterval("child", 1 , 5, data)

summary(slidedFather)

##      Min.    1st Qu.     Median      Mean    3rd Qu.      Max.
## 2.600e-07 1.551e-05 1.333e-04 2.550e-03 8.358e-04 9.641e-02

summary(slidedChild)

##      Min.    1st Qu.     Median      Mean    3rd Qu.      Max.
## 5.500e-07 2.057e-04 1.127e-03 3.713e-03 3.886e-03 7.121e-02
```

Non overlapping interval

```
fatherFive <- MeanMotionByTime("father", index0fvideos=1, interval=5, data)
childFive <- MeanMotionByTime("child", index0fvideos=1, interval=5, data)

summary(childFive)

##      Min.    1st Qu.     Median      Mean    3rd Qu.      Max.
## 1.110e-06 2.049e-04 1.118e-03 3.713e-03 3.900e-03 7.121e-02

summary(fatherFive)

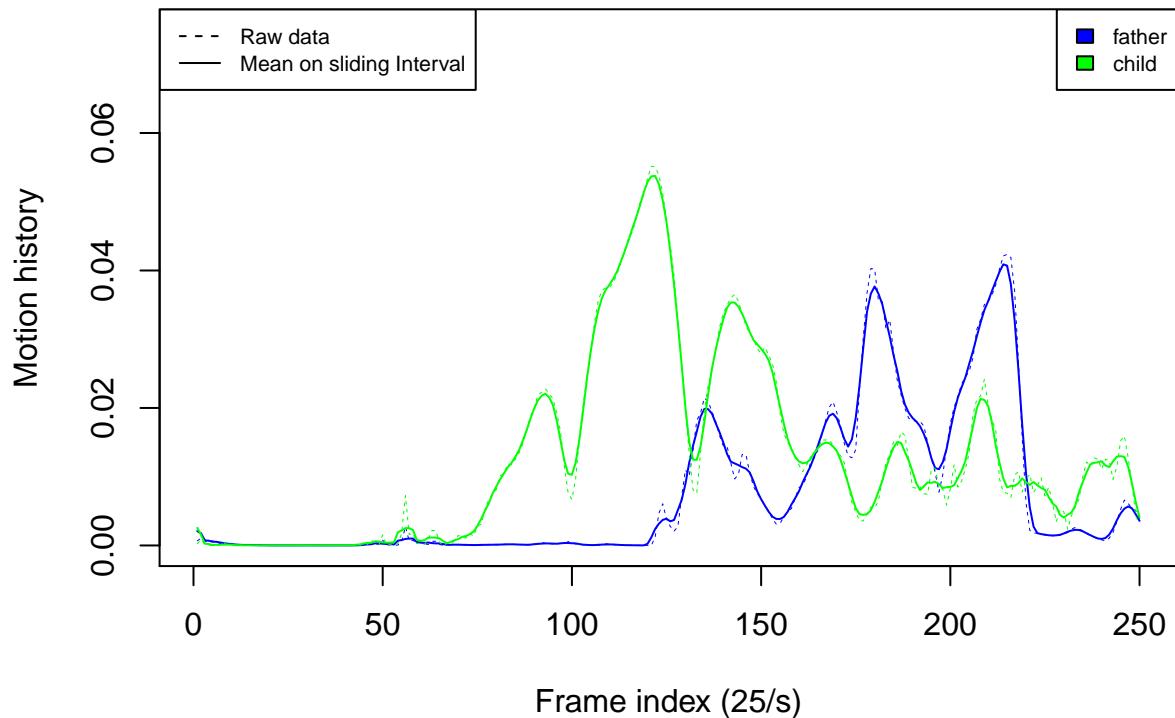
##      Min.    1st Qu.     Median      Mean    3rd Qu.      Max.
## 2.600e-07 1.551e-05 1.330e-04 2.550e-03 8.329e-04 9.137e-02
```

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$father[3:252], main="Mean motion history (Sliding 5 frames interval)
on 00034 video, first 10 seconds ", xlab="Frame index (25/s)",
ylab="Motion history",
col="blue", type="l", lty=2, lwd=0.5, ylim=c(0, 0.075))
lines(slidedFather[1:250], col="blue", lty=1)
lines(slidedChild[1:250], col="green", lty=1)
lines(data$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(1,3)], fill=colOrderList[c(1,3)], cex=0.7)
```

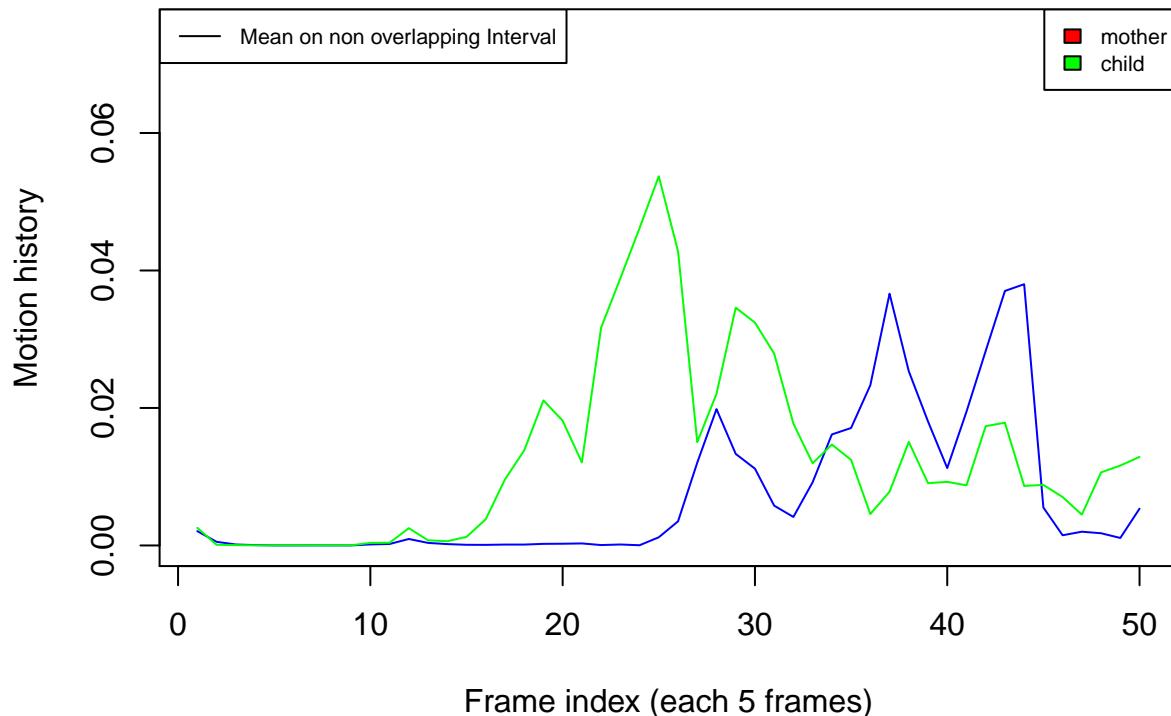
Mean motion history (Sliding 5 frames interval) on 00034 video, first 10 seconds



Non overlapping interval function on a 5 frames interval

```
par(mar=c(4,4,4,2))
plot (1:50, fatherFive[1:50], type="l", col="blue",
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)
```

Mean Motion history (non overlapping 5 frames intervals) for father on 00034 video, first 10 seconds

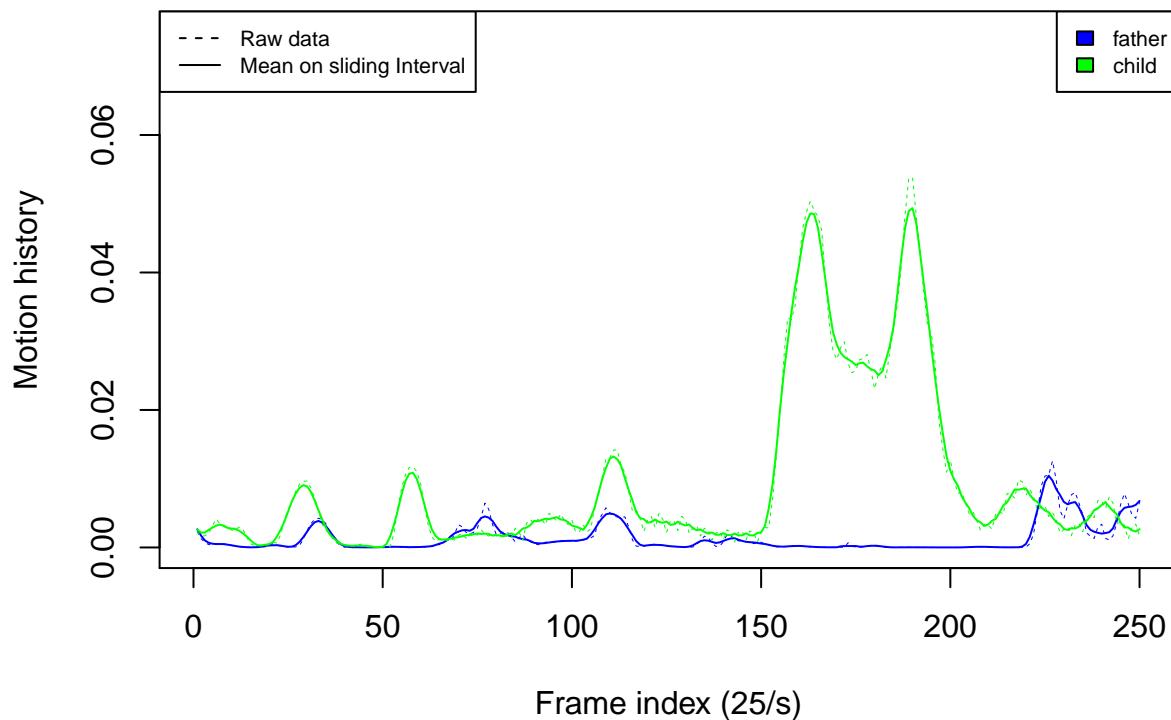


Motion history of the father during 10-20 seconds of the first video 00034

Non overlapping interval function on a 5 frames interval

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

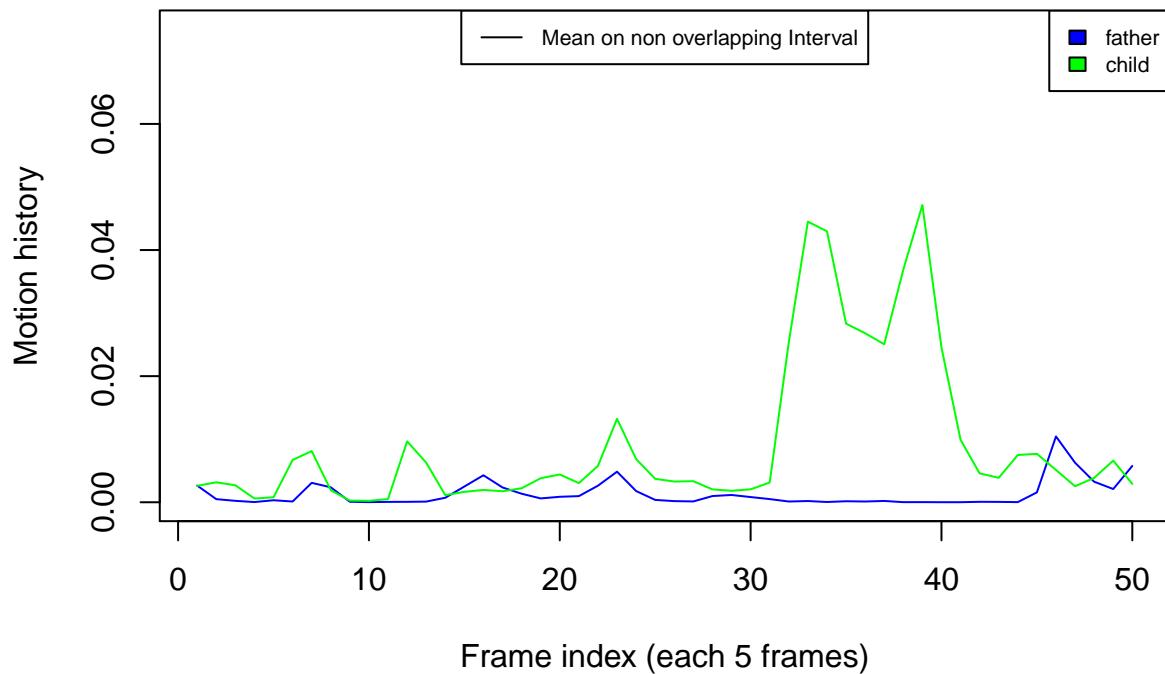
Mean motion history (Sliding 5 frames interval) for father on 00034 video, 10–20 seconds



Non overlapping interval function on a 5 frames interval

```
plot (1:50, fatherFive[51:100], type="l", col="blue",
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(1,3)], fill=colOrderList[c(1,3)], cex=0.7)
```

Mean motion history (non overlapping 5 frames intervals) on 00034 video, between 10–20 seconds



Mean motion history by 10 sec plots

```

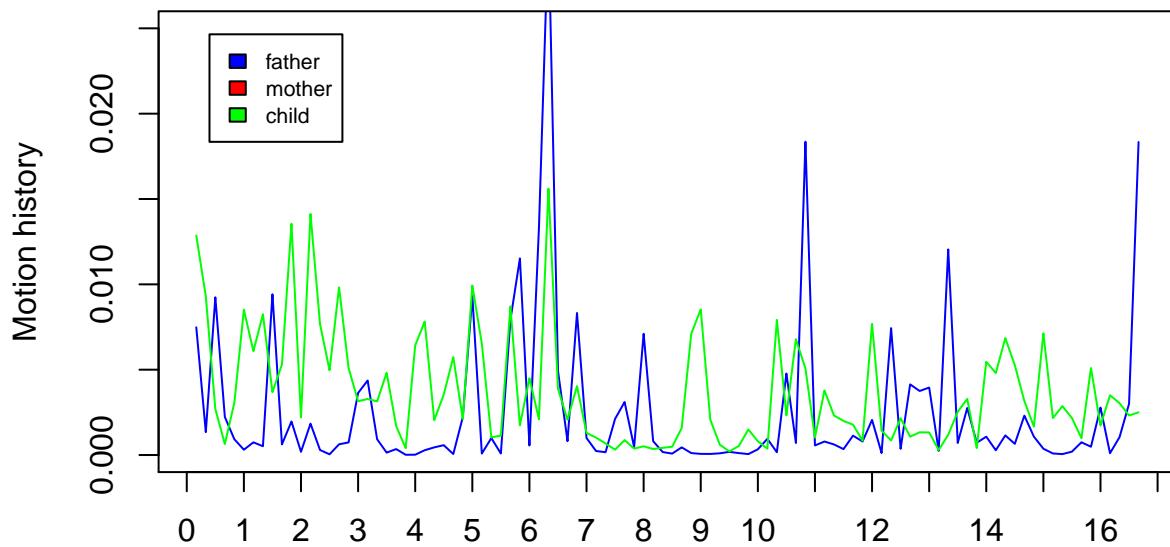
for (i in 1:Number0fvideos){
  fatherMinute<- MeanMotionByTime("father", index0fvideos=i, interval=250, data)

  motherMinute<- MeanMotionByTime("mother", index0fvideos=i, interval=250, data)

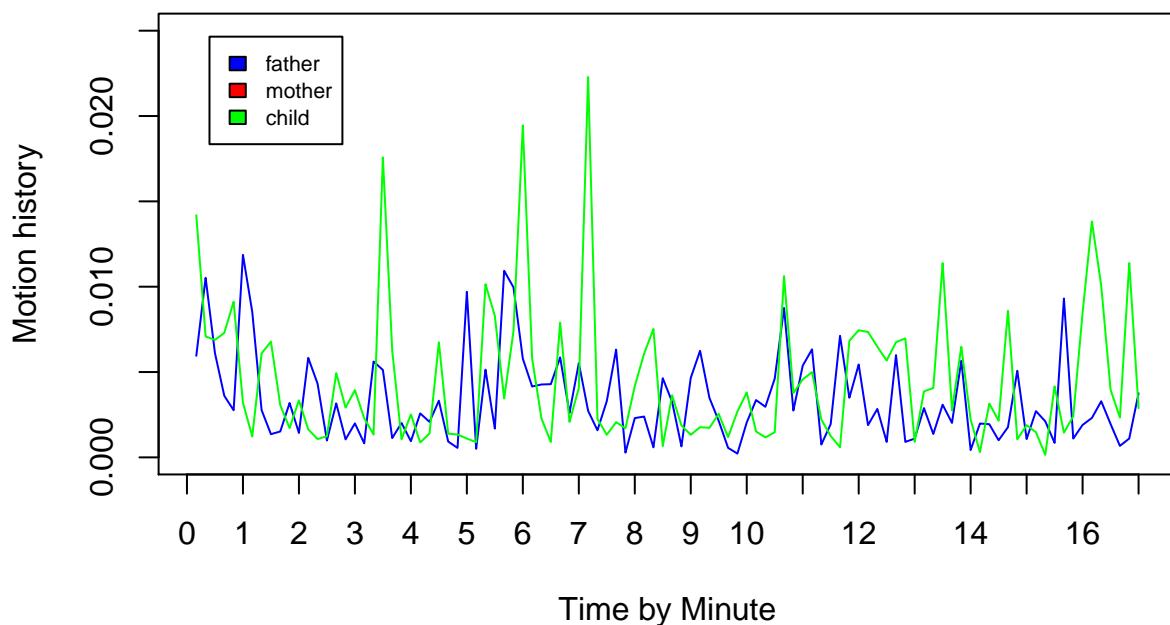
  childMinute<- MeanMotionByTime("child", index0fvideos=i, interval=250, data)

  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, 25E-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)}
  
```

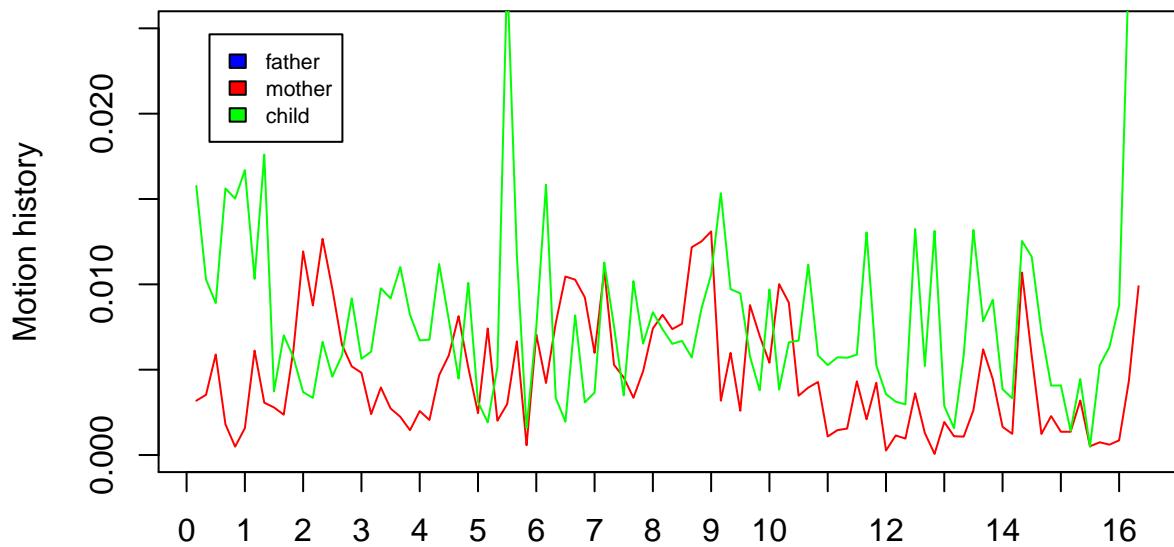
**Mean motion history (non overlapping 10 sec intervals)
on 1606 video**



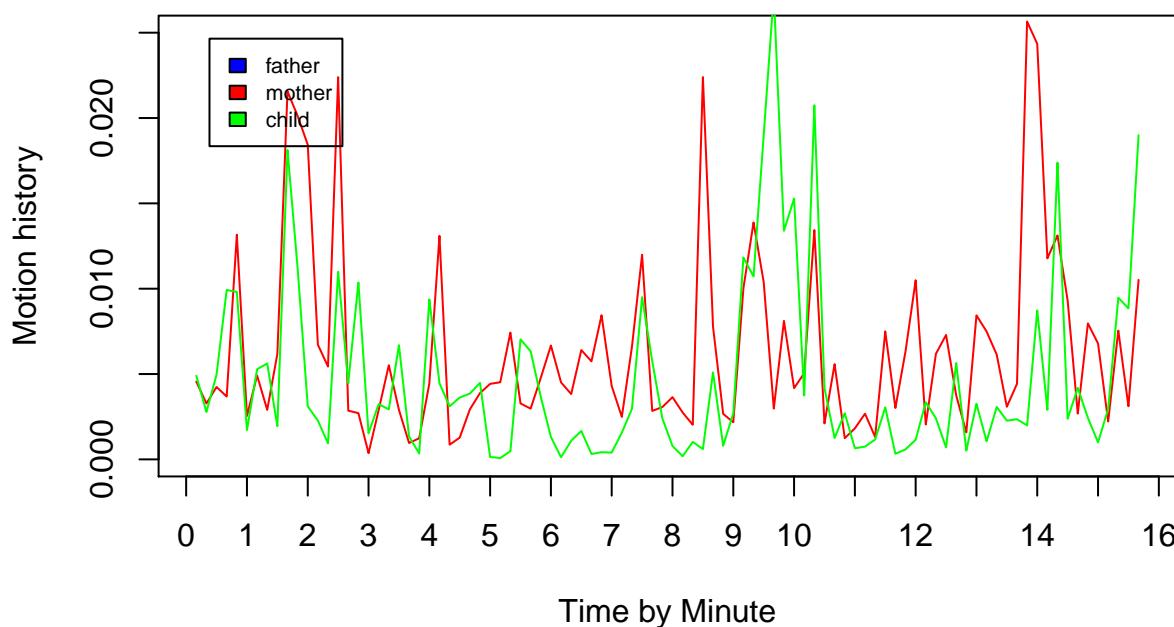
**Time by Minute
Mean motion history (non overlapping 10 sec intervals)
on BAJE059 video**



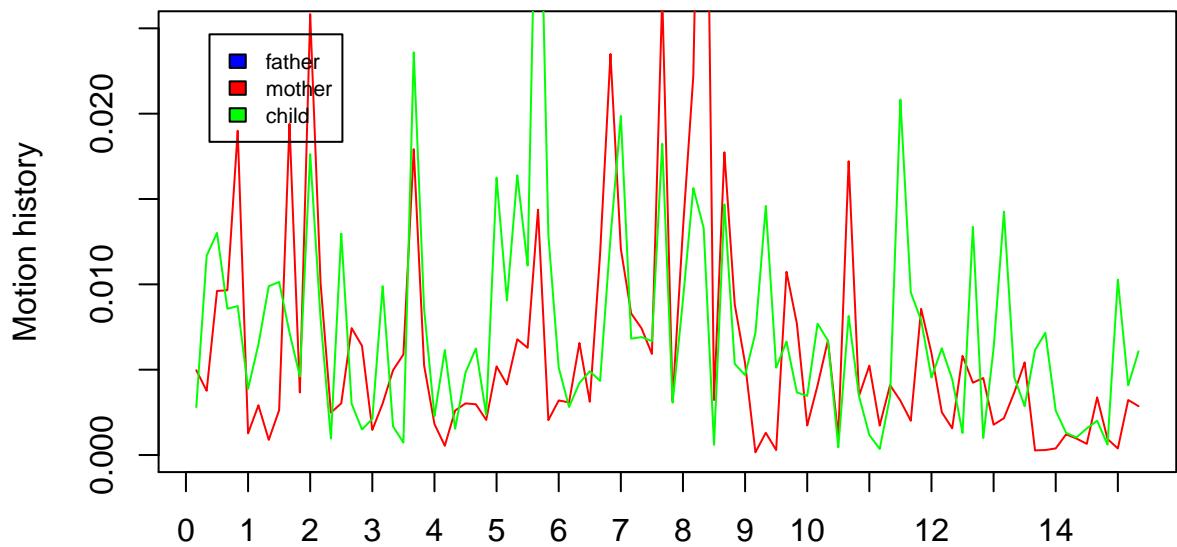
**Mean motion history (non overlapping 10 sec intervals)
on BALU062 video**



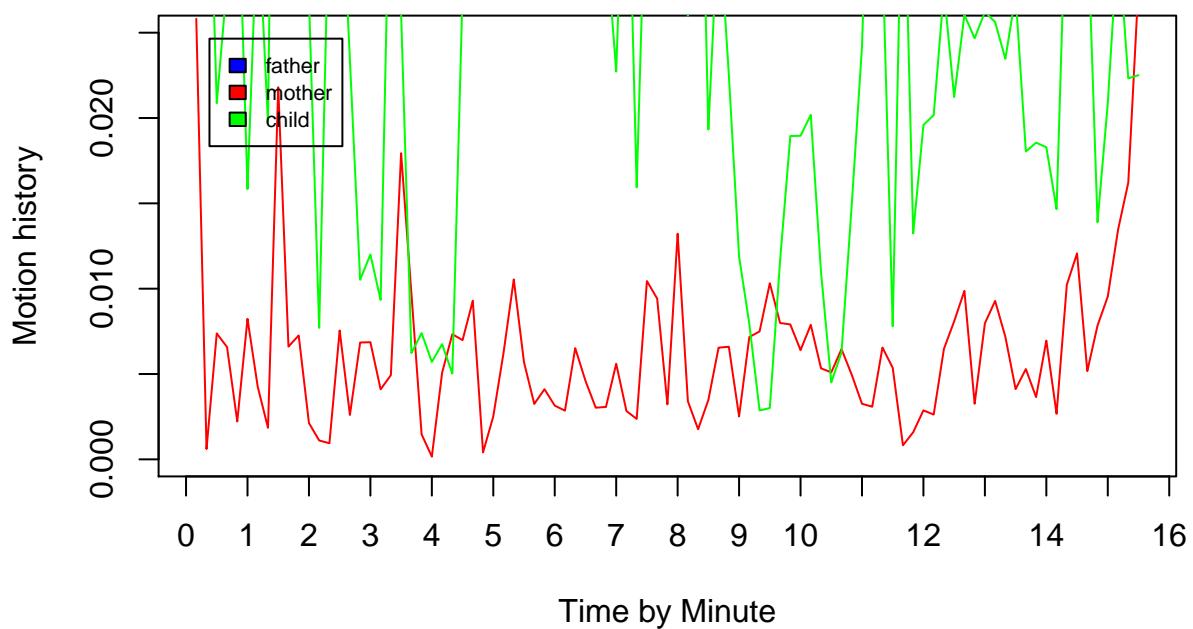
**Time by Minute
Mean motion history (non overlapping 10 sec intervals)
on BEAL036 video**



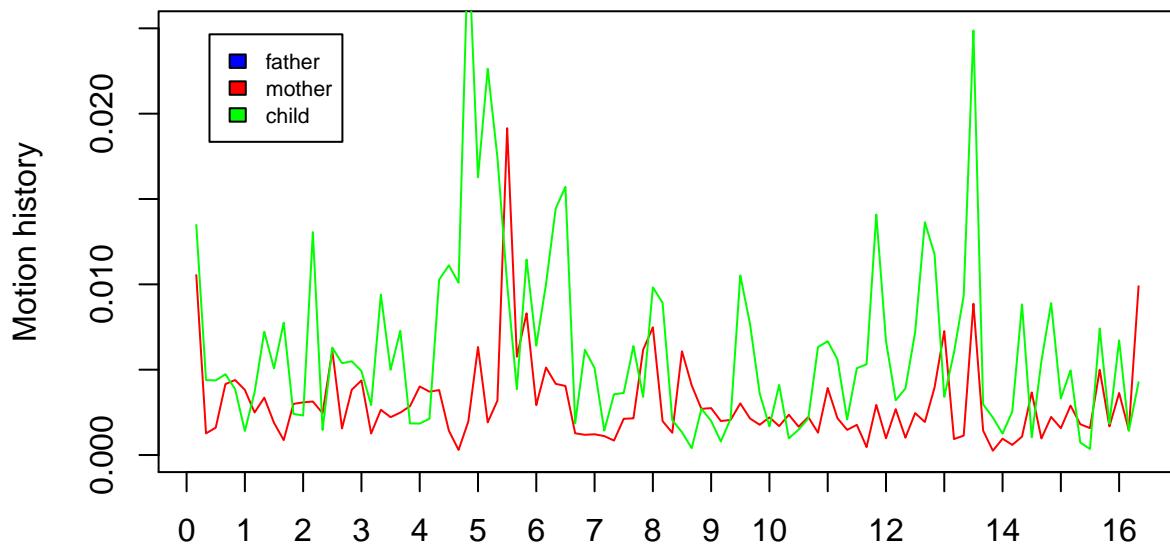
**Mean motion history (non overlapping 10 sec intervals)
on BEAM031 video**



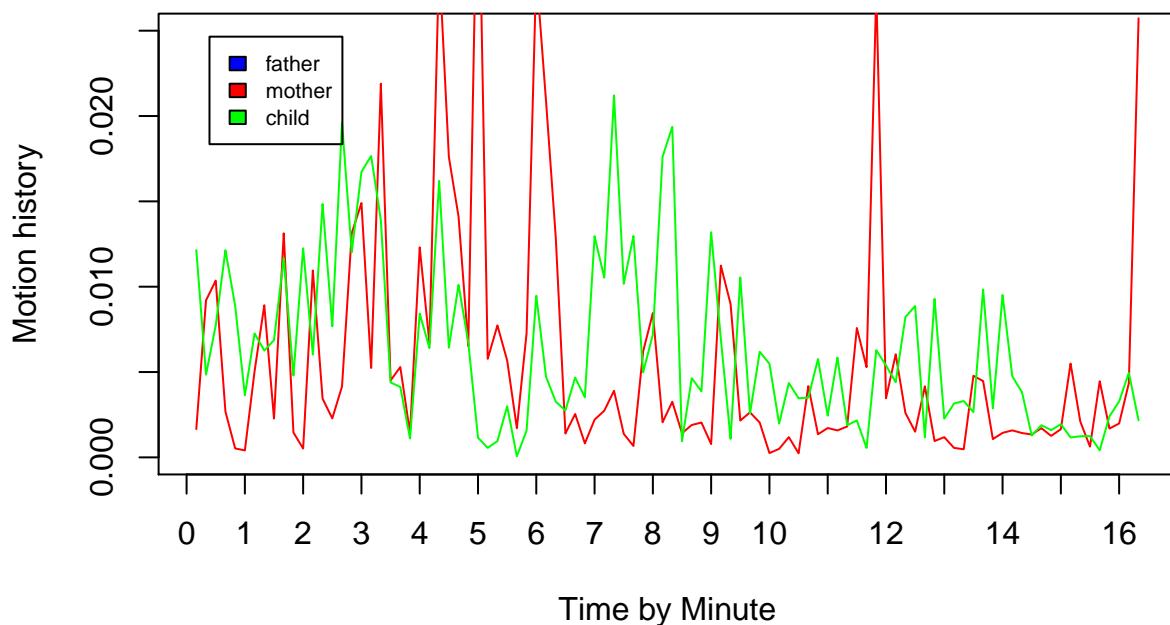
**Mean motion history (non overlapping 10 sec intervals)
on BRLO041 video**



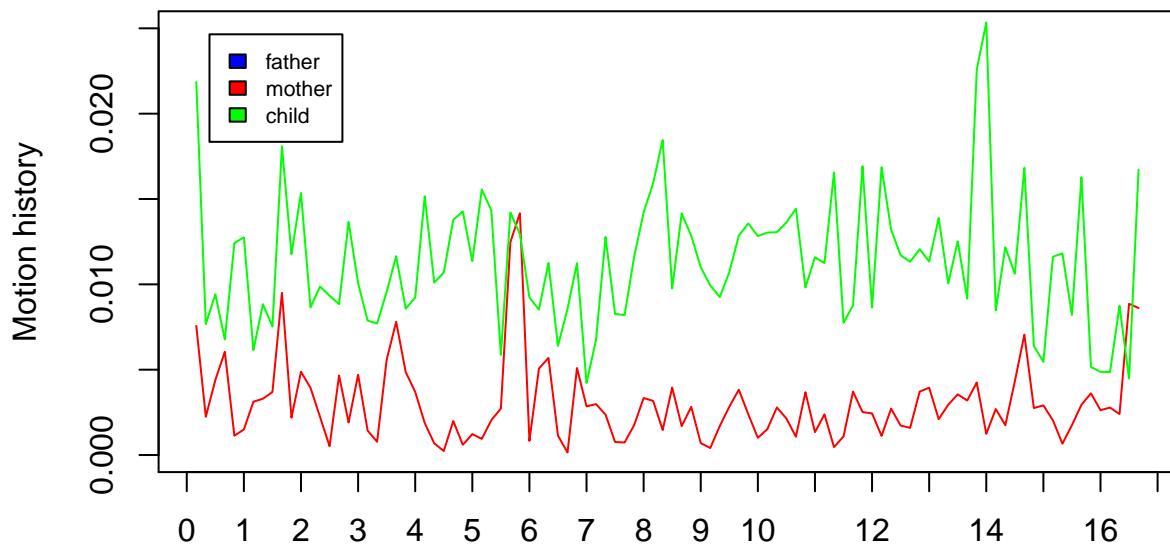
**Mean motion history (non overlapping 10 sec intervals)
on COLO022 video**



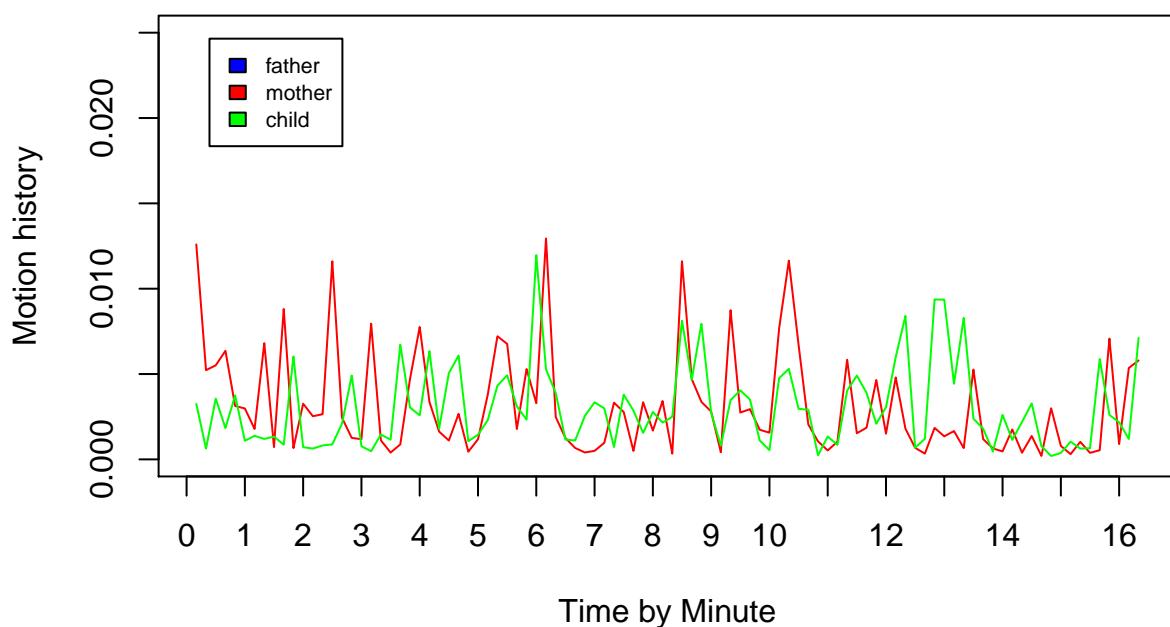
**Time by Minute
Mean motion history (non overlapping 10 sec intervals)
on Dipe004 video**



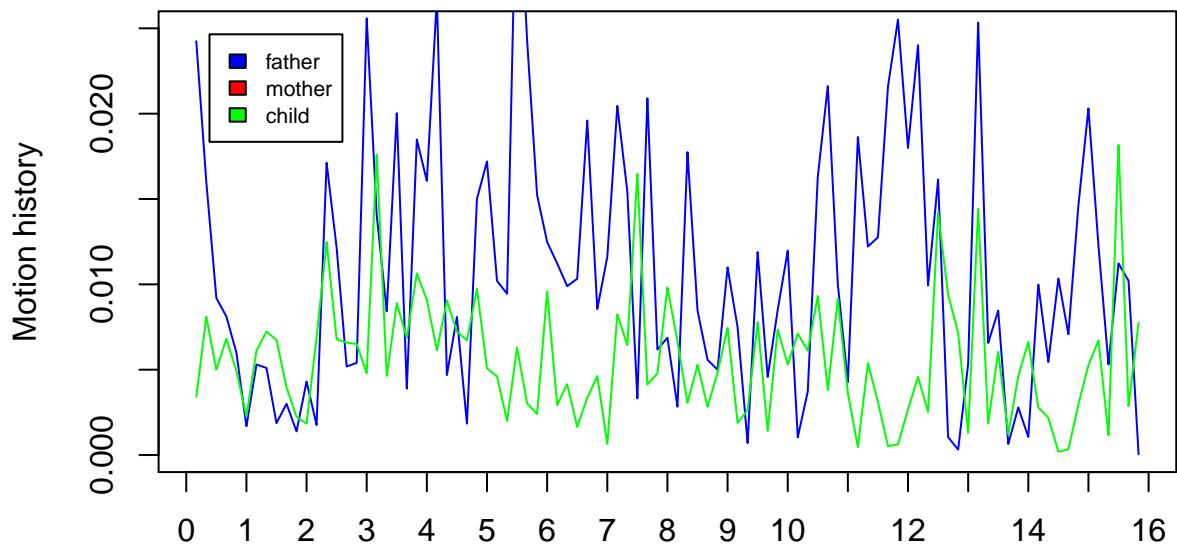
**Mean motion history (non overlapping 10 sec intervals)
on DOMA video**



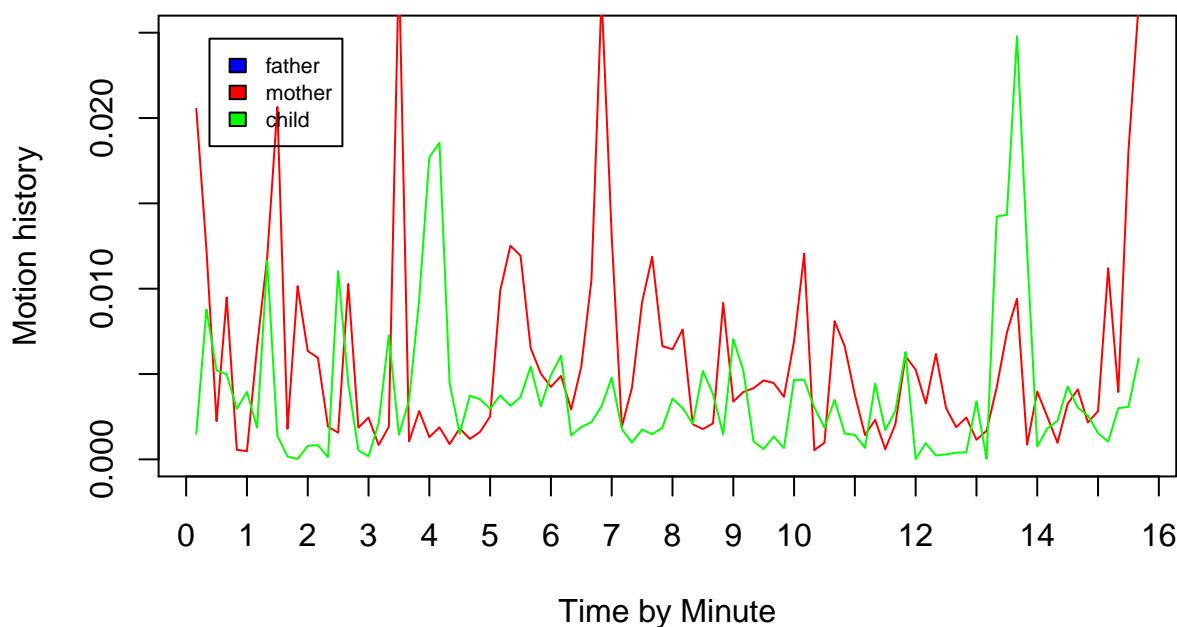
**Time by Minute
Mean motion history (non overlapping 10 sec intervals)
on DRNE video**



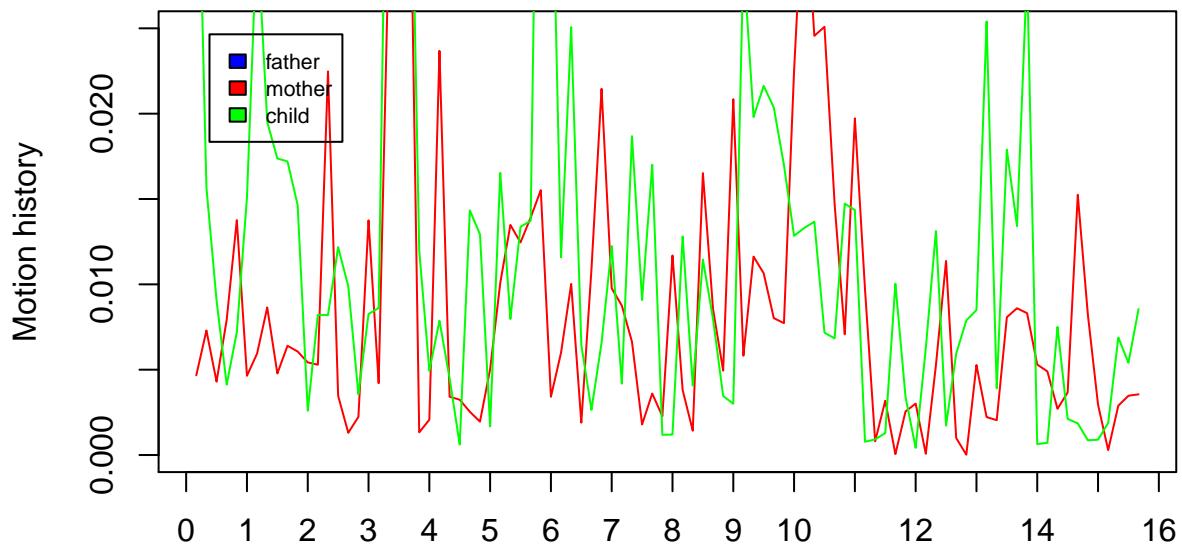
**Mean motion history (non overlapping 10 sec intervals)
on FOMA057 video**



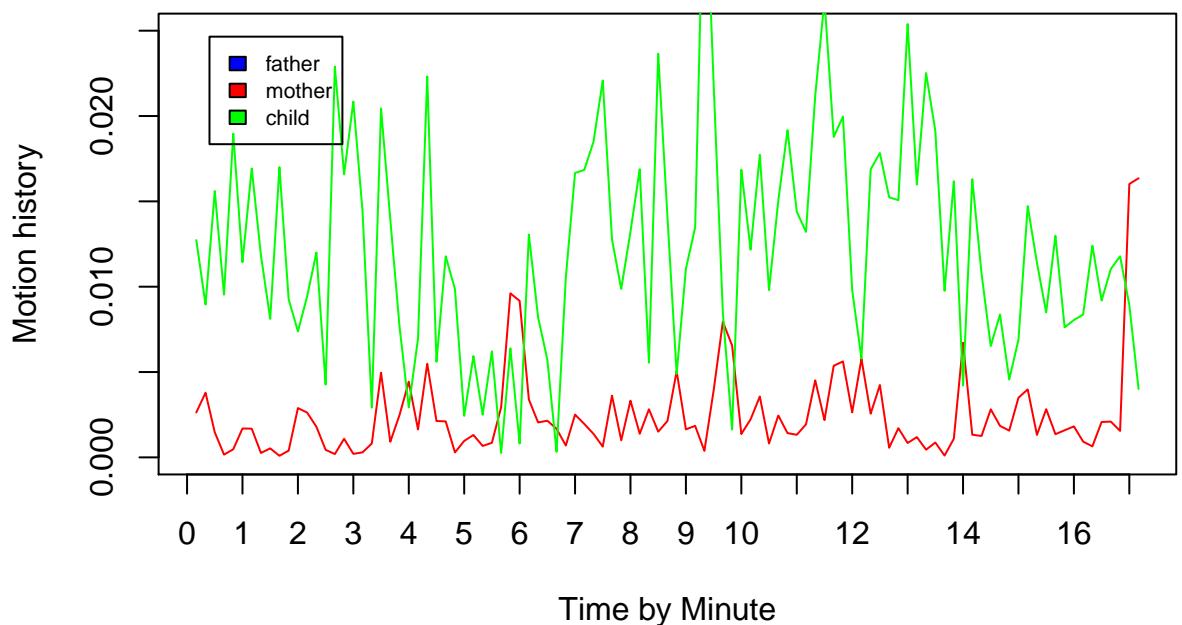
**Time by Minute
Mean motion history (non overlapping 10 sec intervals)
on GROP039 video**



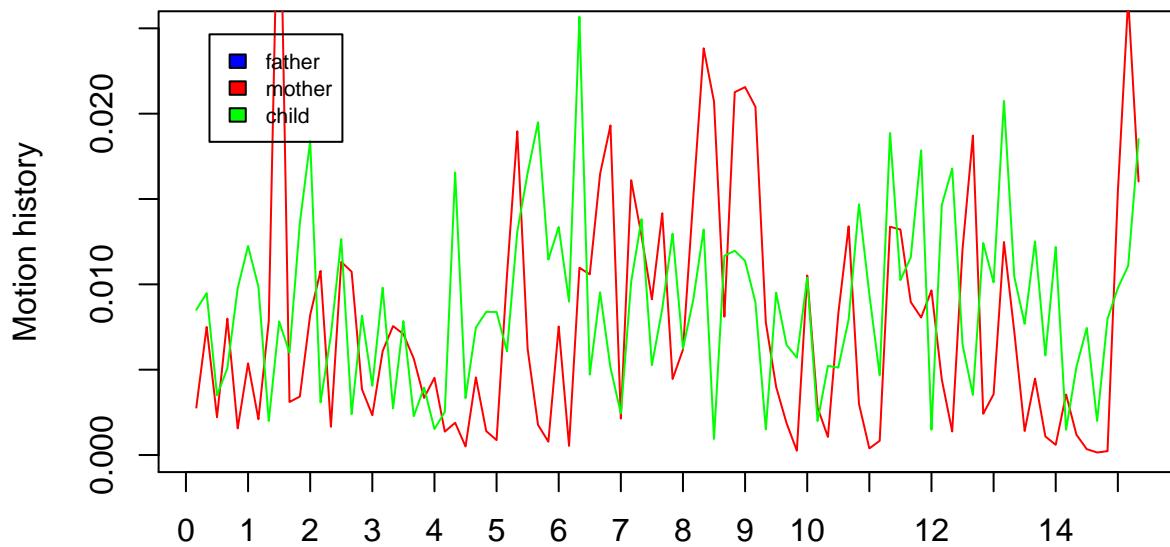
**Mean motion history (non overlapping 10 sec intervals)
on HAJA052 video**



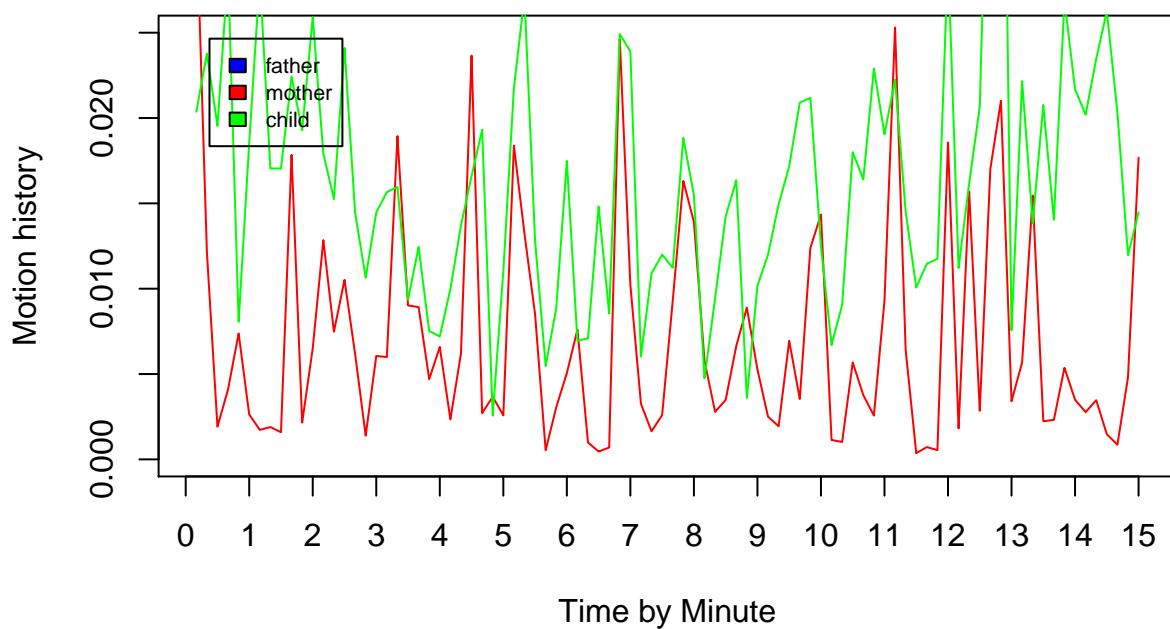
**Time by Minute
Mean motion history (non overlapping 10 sec intervals)
on HUMA058 video**



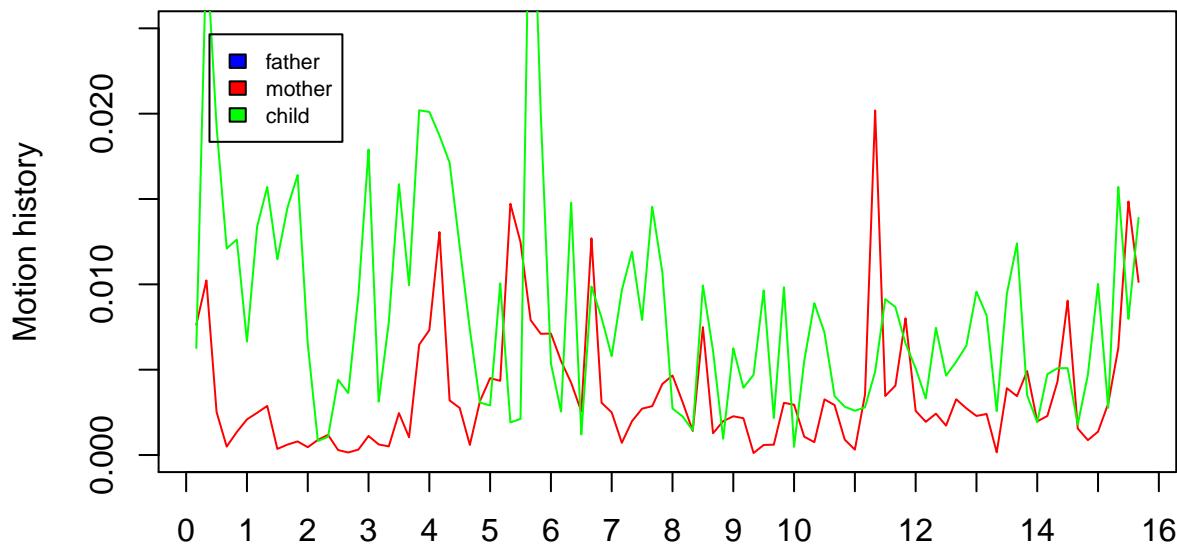
**Mean motion history (non overlapping 10 sec intervals)
on JAEM046 video**



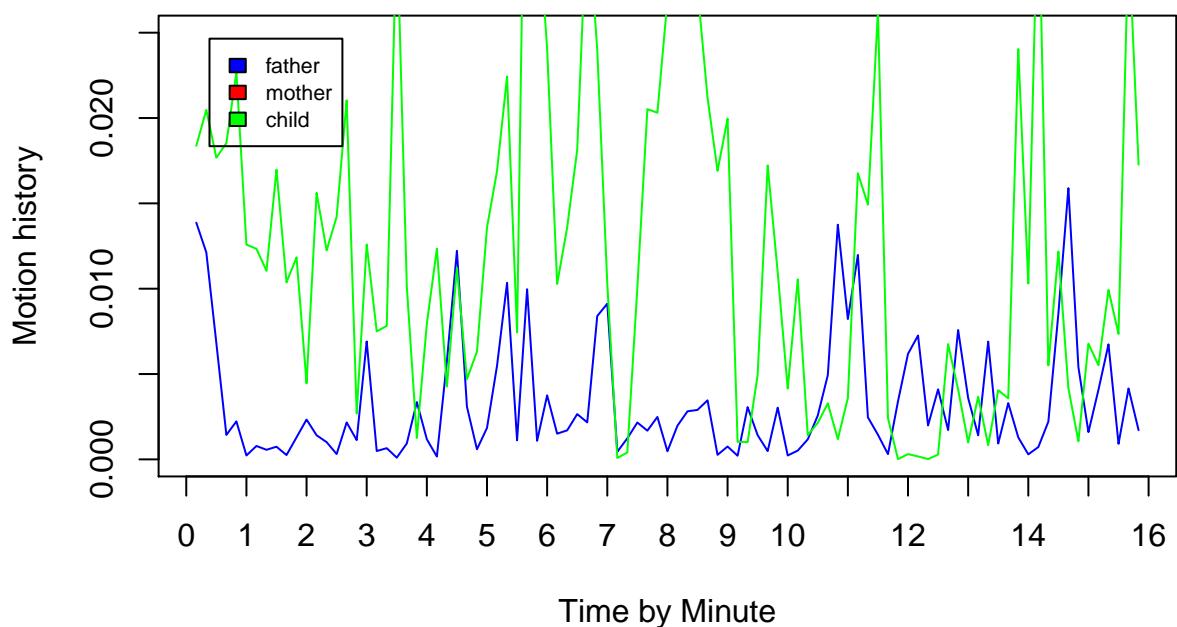
**Mean motion history (non overlapping 10 sec intervals)
on JEEO040 video**



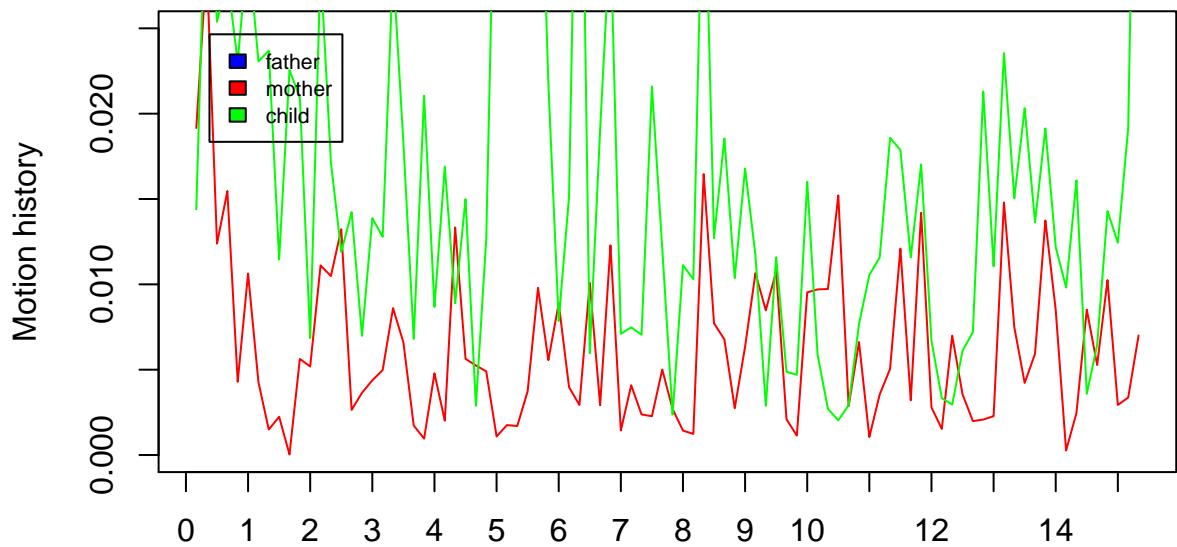
**Mean motion history (non overlapping 10 sec intervals)
on JOCE014 video**



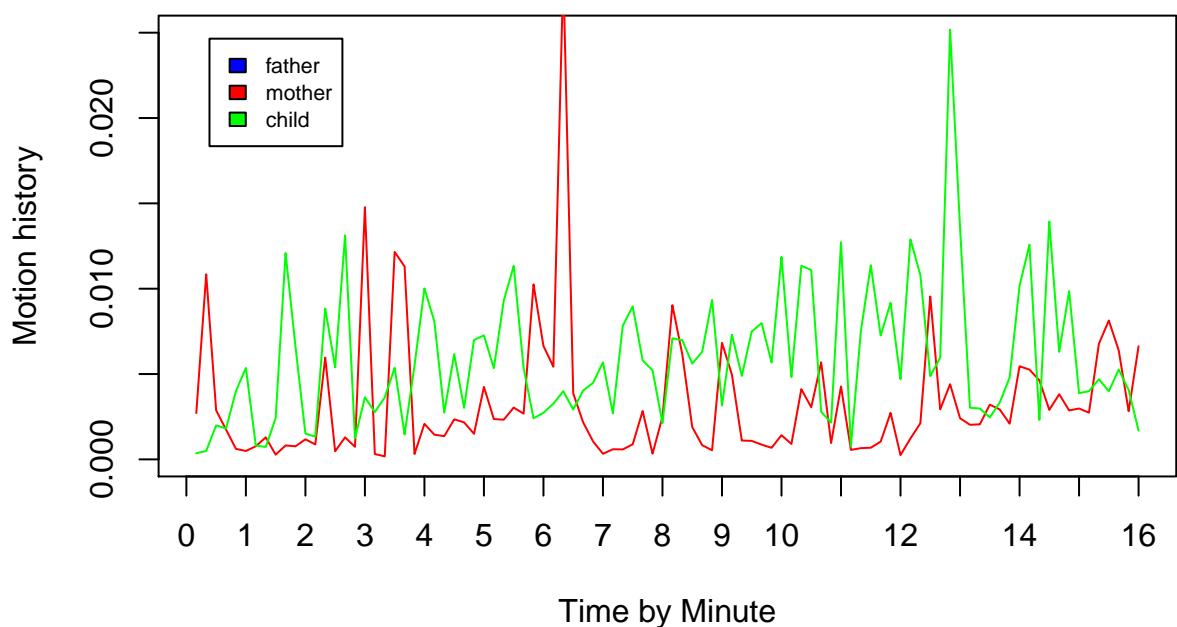
**Time by Minute
Mean motion history (non overlapping 10 sec intervals)
on LACL video**



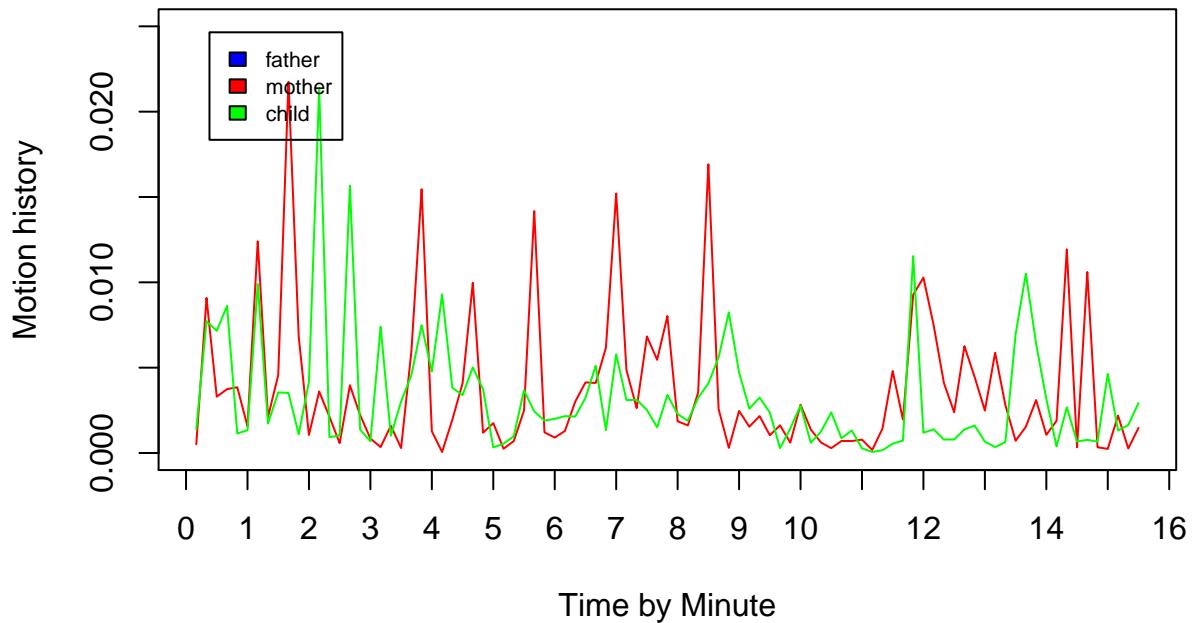
**Mean motion history (non overlapping 10 sec intervals)
on MAEL048 video**



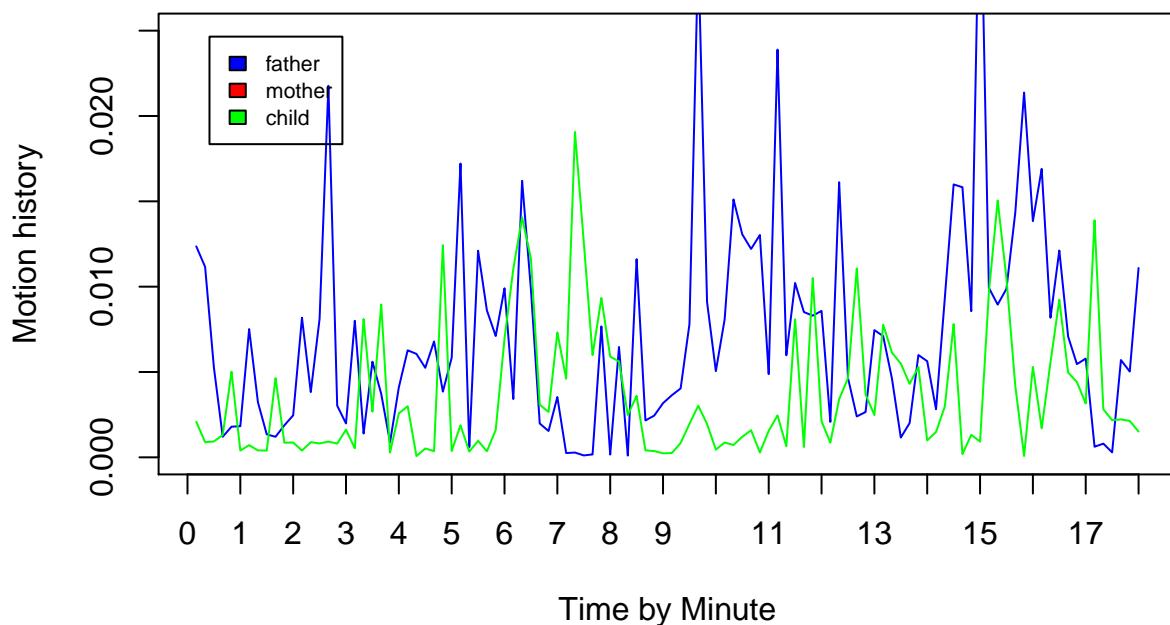
**Mean motion history (non overlapping 10 sec intervals)
on MAME20 video**



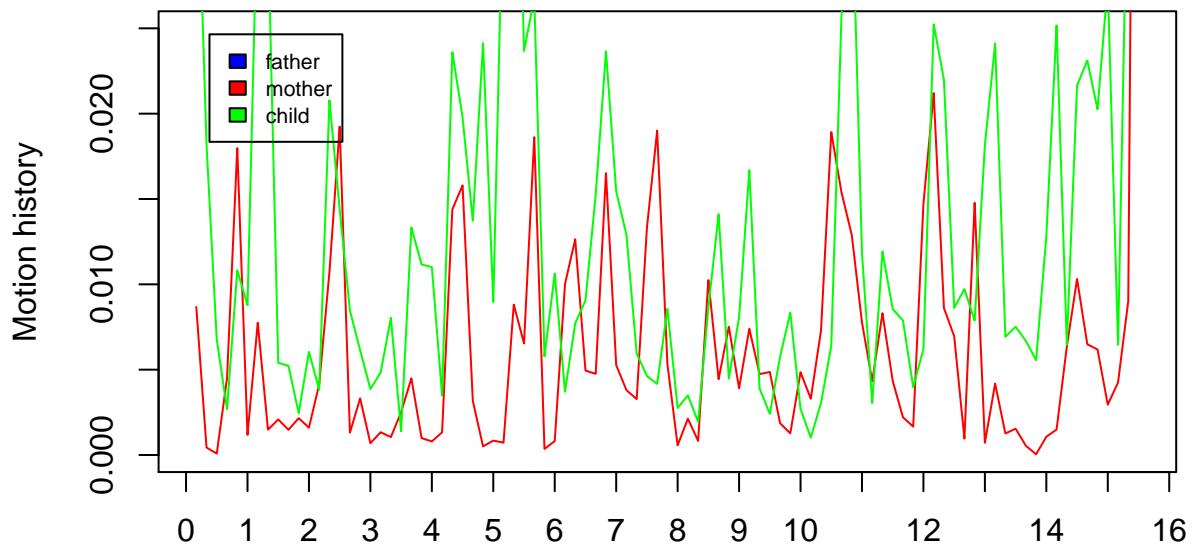
**Mean motion history (non overlapping 10 sec intervals)
on MIPH043 video**



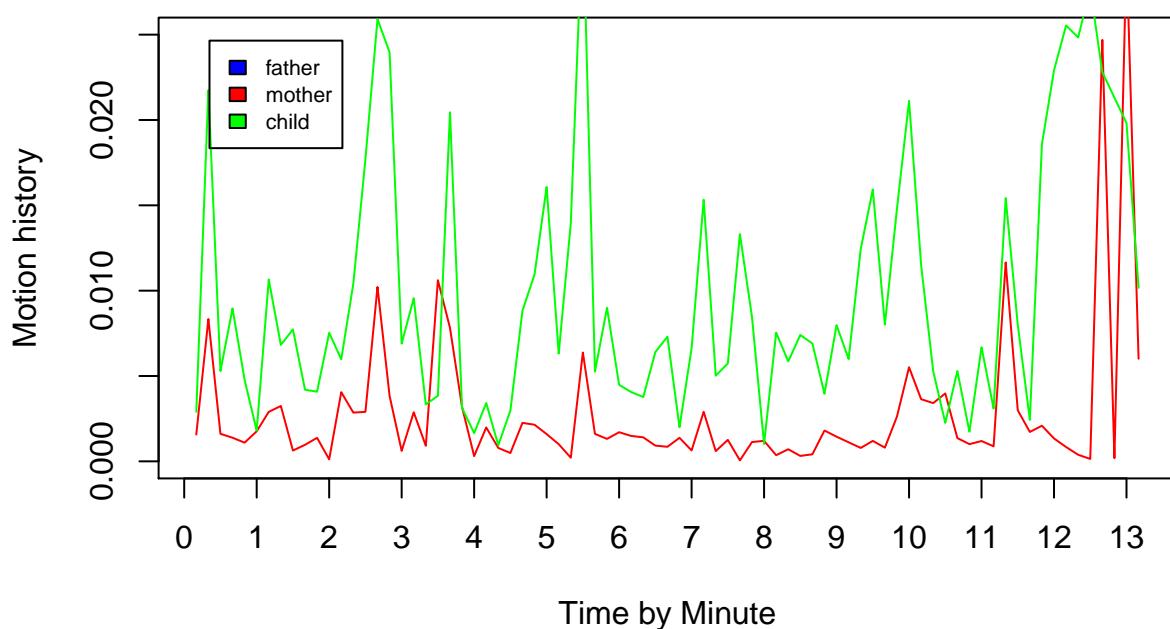
**Mean motion history (non overlapping 10 sec intervals)
on MOSA065 video**



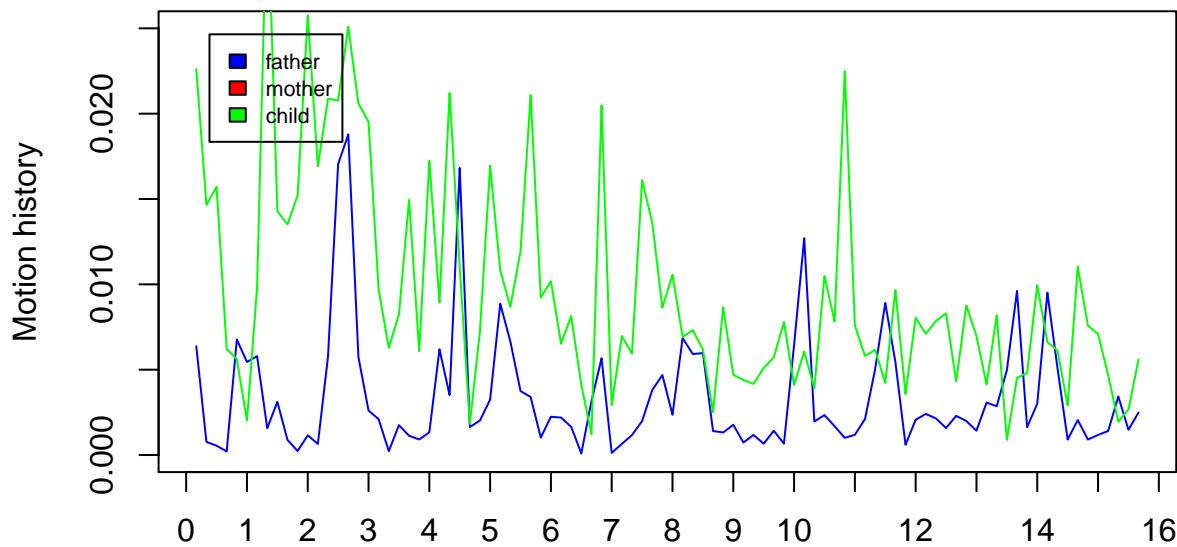
**Mean motion history (non overlapping 10 sec intervals)
on NAMA045 video**



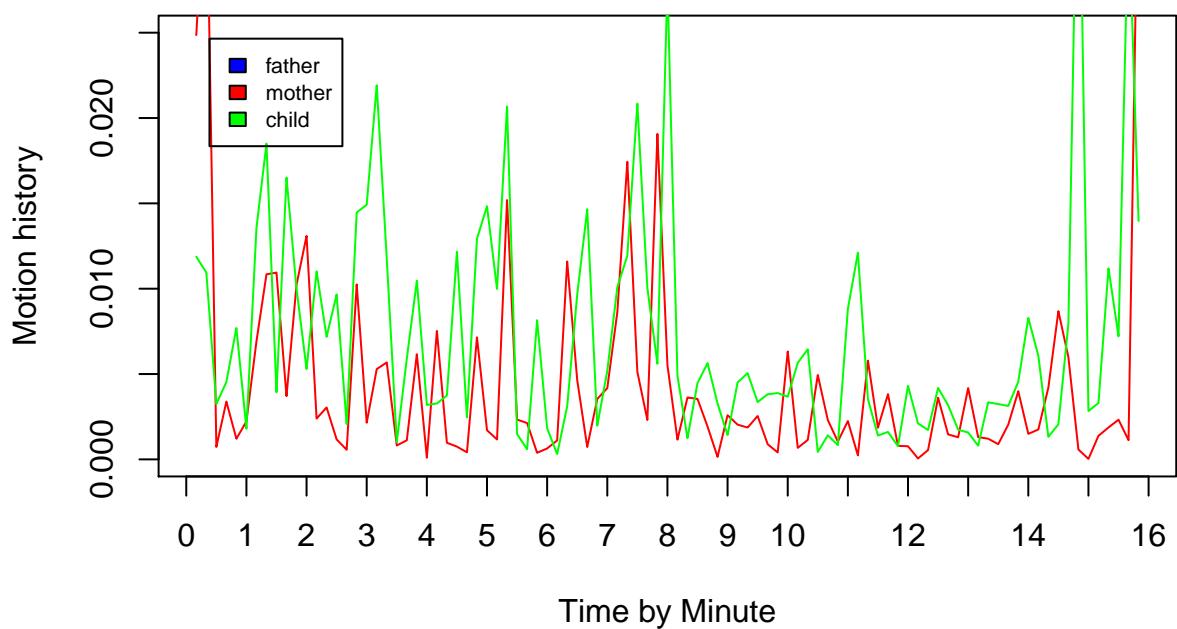
**Time by Minute
Mean motion history (non overlapping 10 sec intervals)
on NUMA027 video**



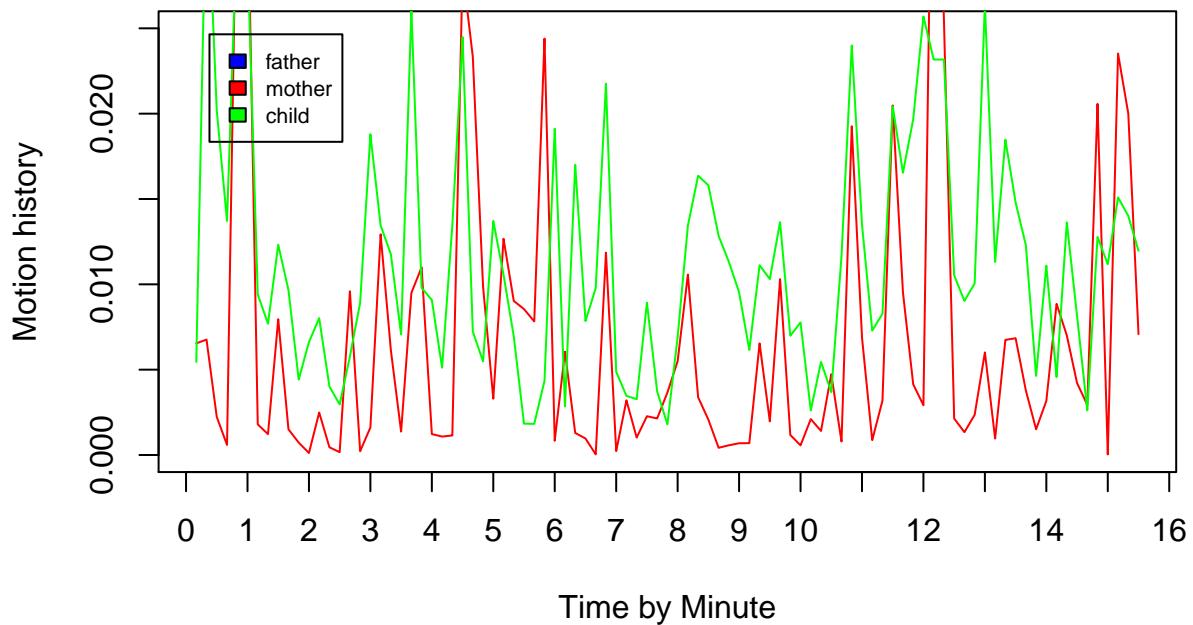
**Mean motion history (non overlapping 10 sec intervals)
on OGGA034 video**



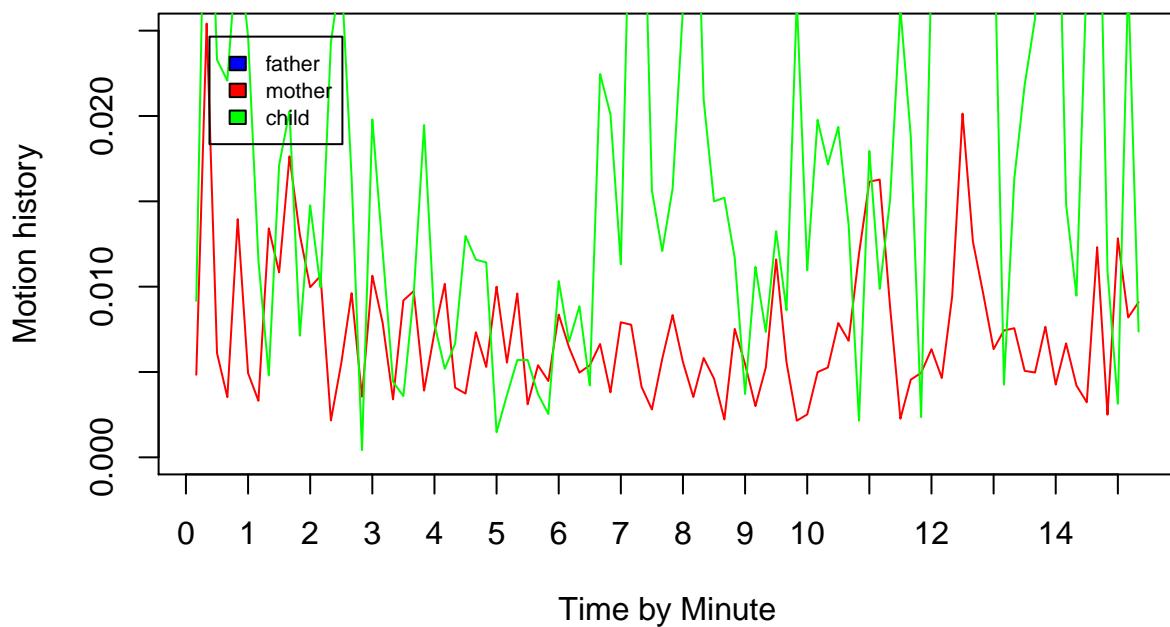
**Time by Minute
Mean motion history (non overlapping 10 sec intervals)
on PAMA029 video**



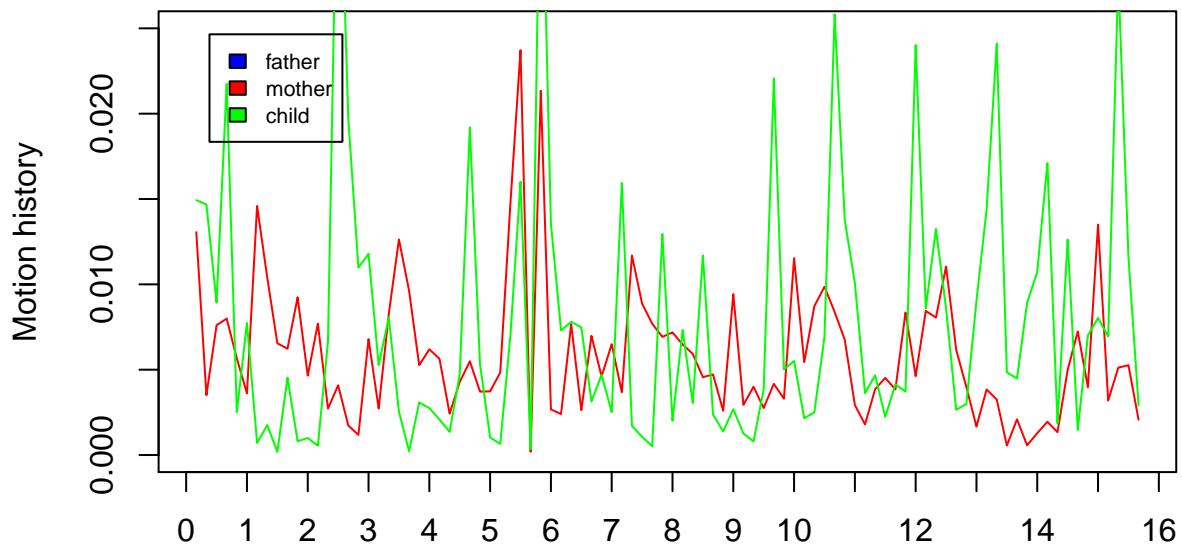
**Mean motion history (non overlapping 10 sec intervals)
on PELI020 video**



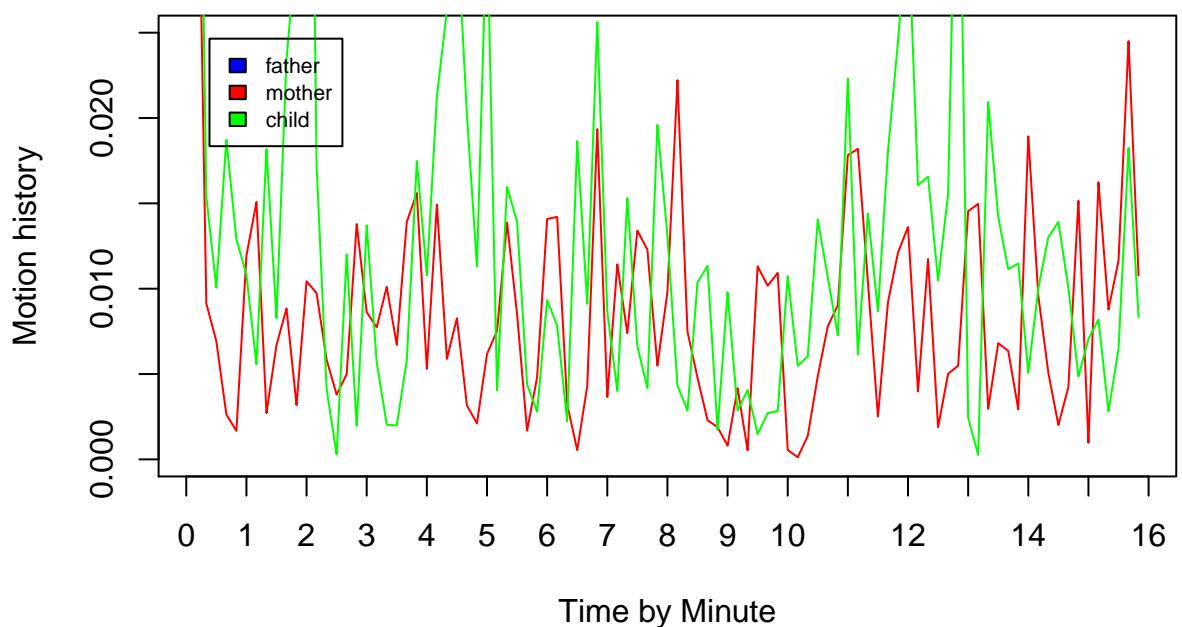
**Mean motion history (non overlapping 10 sec intervals)
on RAEM049 video**



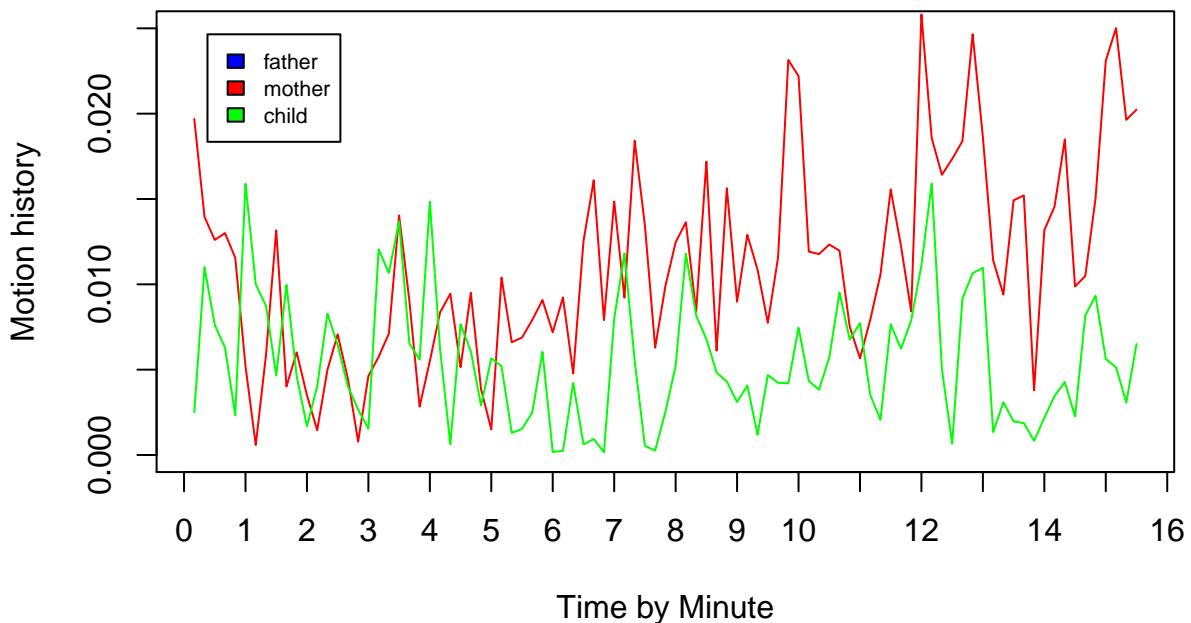
**Mean motion history (non overlapping 10 sec intervals)
on RAMA054 video**



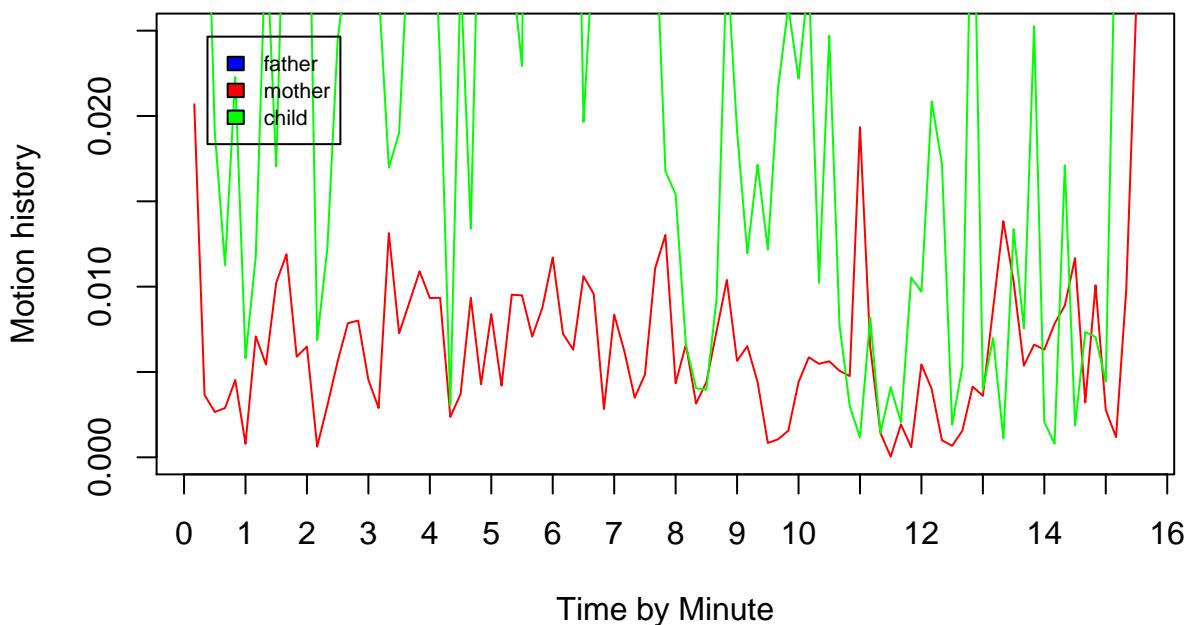
**Time by Minute
Mean motion history (non overlapping 10 sec intervals)
on SEEM035 video**



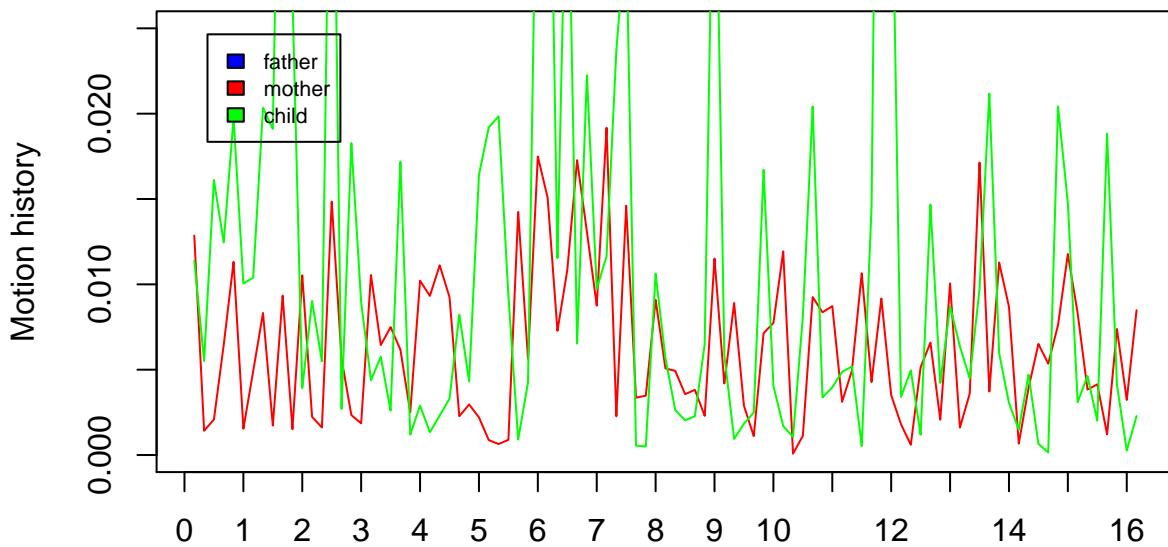
**Mean motion history (non overlapping 10 sec intervals)
on SHAN042 video**



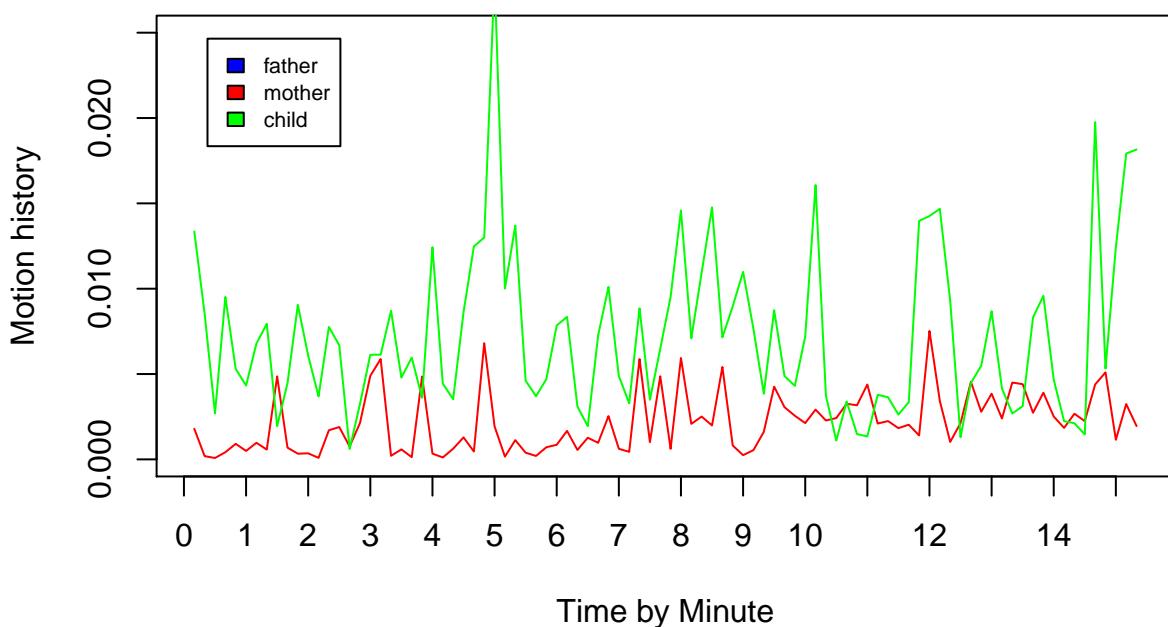
**Mean motion history (non overlapping 10 sec intervals)
on SOGA061 video**



Mean motion history (non overlapping 10 sec intervals) on TIUG032 video



Time by Minute Mean motion history (non overlapping 10 sec intervals) on VINO video



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滑动父亲", videoName))
slidedFather <- SlidingInterval("father", videoIndex, 5, data)

print(paste("Computing滑动母亲", videoName))
slidedMother <- SlidingInterval("mother", videoIndex, 5, data)

print(paste("Computing滑动孩子", videoName))
slidedChild <- SlidingInterval("child", videoIndex, 5, data)

slidedVideo <- data.frame(
  slidedFather, slidedMother, slidedChild,
  "video"=rep(families[videoIndex], length(slidedFather)),
  frame_index = 1:length(slidedFather))
# write the file
  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滑动父亲", videoName))
  slidedFather <- SlidingInterval("logFather", videoIndex, 5, data)
  print(paste("Computing滑动母亲", videoName))
  slidedMother <- SlidingInterval("logMother", videoIndex, 5, data)
  print(paste("Computing滑动孩子", 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 the file
  write.csv(slidedVideo, paste("../Data/CSV/filtered/log/", videoName, ".log.slideddata.csv", sep=""))
  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 putting 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"
SSIInoLogFilesList <- list.files("../Data/CSV/Synchrony/noLog/S_estimator", full.name=TRUE)
#SSIInoLogFilesList

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': 1055 obs. of 6 variables:
## $ video      : chr "Name" "1606" "1606" "1606" ...
## $ 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_fa_ch: num NA 0.06097 0.015151 0.008147 0.000207 ...
## $ SSI_mo_ch: num NA NA NA NA NA NA NA NA NA ... 

SSIlog$video <- as.factor(ssilog$video)
SSIlog <- SSIlog[-which(ssilog$video=="Name"),]
SSIlog$Interval <- NULL
SSIlog <- rename (SSIlog, c("video" = "family"))
SSIlog <- rename (SSIlog, c("X" = "SSI-interval"))

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

```

# print(str(SSIalone))
SSInoLog<- rbind.fill(SSInoLog, SSIalone)}
SSInoLog$video <- as.factor(SSInoLog$video)
SSInoLog <- SSInoLog[-which(SSInoLog$video=="Name"),]
SSInoLog$Interval <- NULL
SSInoLog <- rename (SSIlog, c("video" = "family"))

## The following `from` values were not present in `x`: video
SSInoLog <- rename (SSIlog, c("X" = "SSI-interval"))

## The following `from` values were not present in `x`: X
#SSInoLog$action <- rep(NA, nrow(SSInoLog))
#SSInoLog[which(video == & timeMin==)]$action

```

Description of SSIlog data frame

```

str(SIlog)

## 'data.frame': 1054 obs. of 5 variables:
## $ family      : Factor w/ 35 levels "1606","BAJE059",...
## $ SSI-interval: int 0 1 2 3 4 5 6 7 8 9 ...
## $ Time_min    : num 0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 ...
## $ SSI_fa_ch   : num 0.06097 0.015151 0.008147 0.000207 0.023456 ...
## $ SSI_mo_ch   : num NA ...
#View(SI)

```

Data dictionary of SSIlog data frame

- **family** : code of the family
- **SSI-interval**** : interval of SSI
- **Time_min** : Time in minutes
- **SSI_fa_ch** : SSI index of Synchrony between father and child
- **SSI_mo_ch** : SSI index of Synchrony between mother and child

Description of noLogSSI data frame

```

str(SSInoLog)

## 'data.frame': 1054 obs. of 5 variables:
## $ family      : Factor w/ 35 levels "1606","BAJE059",...
## $ SSI-interval: int 0 1 2 3 4 5 6 7 8 9 ...
## $ Time_min    : num 0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 ...
## $ SSI_fa_ch   : num 0.06097 0.015151 0.008147 0.000207 0.023456 ...
## $ SSI_mo_ch   : num NA NA NA NA NA NA NA NA NA ...
#View(SSInoLog)

```

Data dictionary of SSIlog data frame

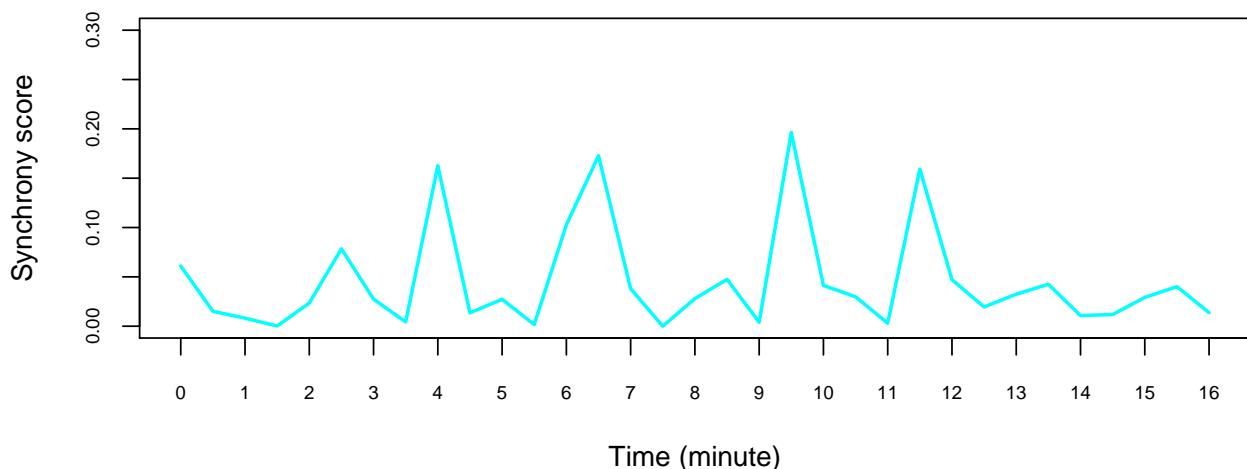
- family : code of the family
- SSI-interval : interval of SSI
- Time_min : Time in minutes
- SSI_fa_ch : SSI index of Synchrony between father and child
- SSI_mo_ch : SSI index of Synchrony between mother and child

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

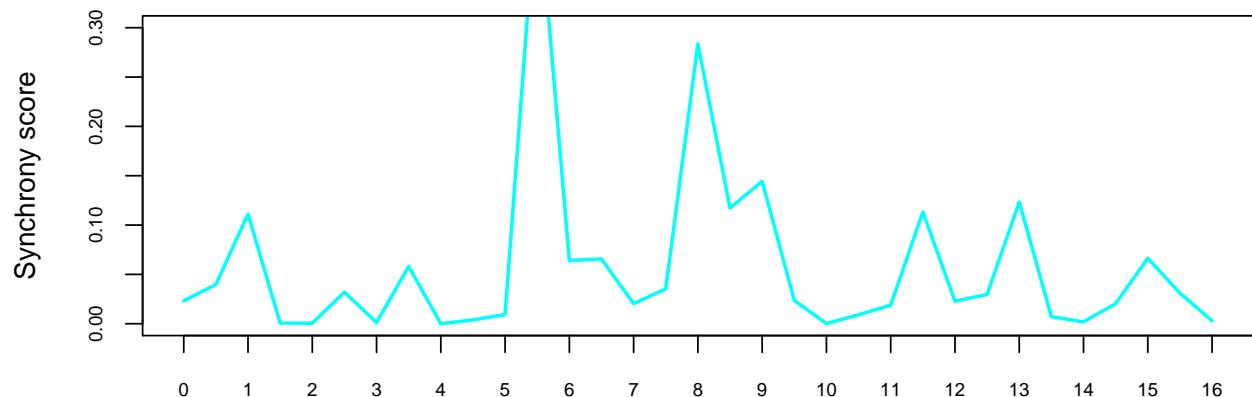
```
par(mar=c(4,4,4,3), mfrow=c(1,1))
for (i in unique(SSIlog$family)){
  if (all(!is.na(SSIlog[which(SSIlog$family==i),]$SSI_mo_ch)==TRUE)){
    #print(SSIlog[which(SSIlog$family==i),]$Time_min)
    #print(str(SSIlog[which(SSIlog$family==i),]$SSI_mo_ch))
    plot(SSIlog[which(SSIlog$family==i),]$Time_min,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

    else if(all(!is.na(SSIlog[which(SSIlog$family==i),]$SSI_fa_ch)==TRUE)){
      plot(SSIlog[which(SSIlog$family==i),]$Time_min, SSIlog[which(SSIlog$family==i),]$SSI_fa_ch,
            ylim=c(0, 0.3), main=paste("Synchrony scores in", i, "family"), xlab = "Time (minute)", yl
    else{print("error")}}
```

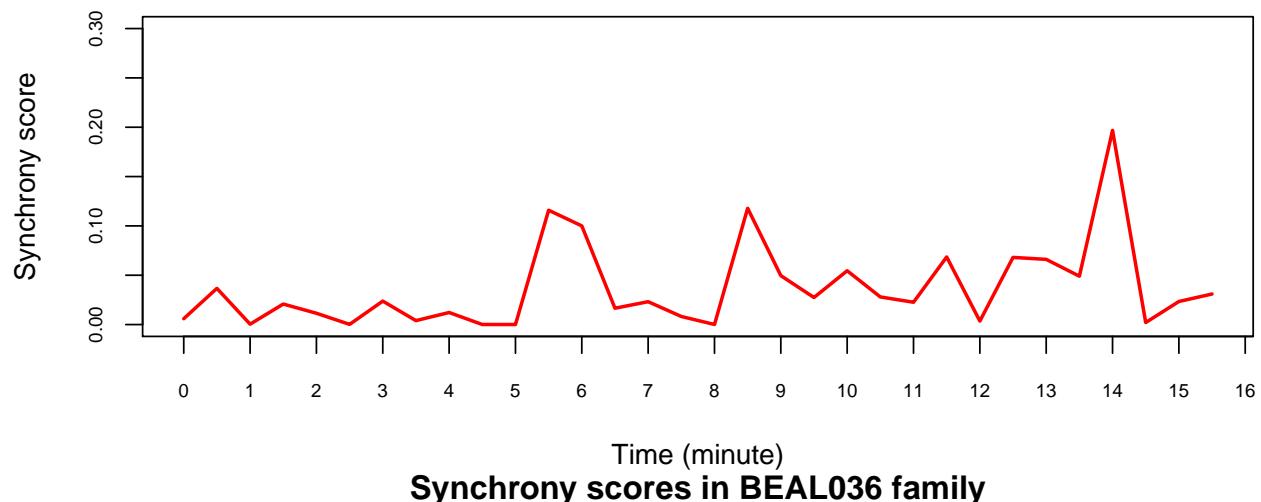
Synchrony scores in 1606 family



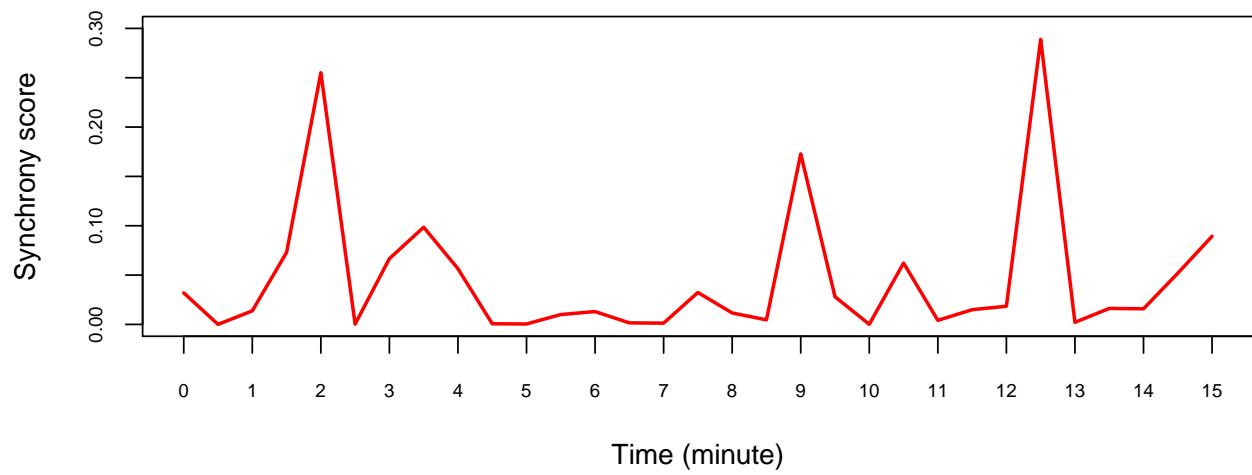
Synchrony scores in BAJE059 family



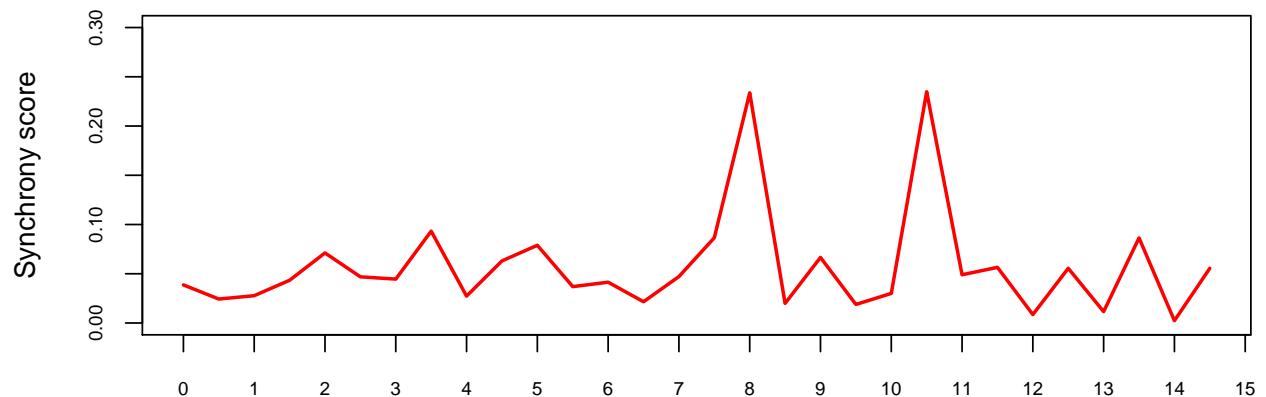
Synchrony scores in BALU062 family



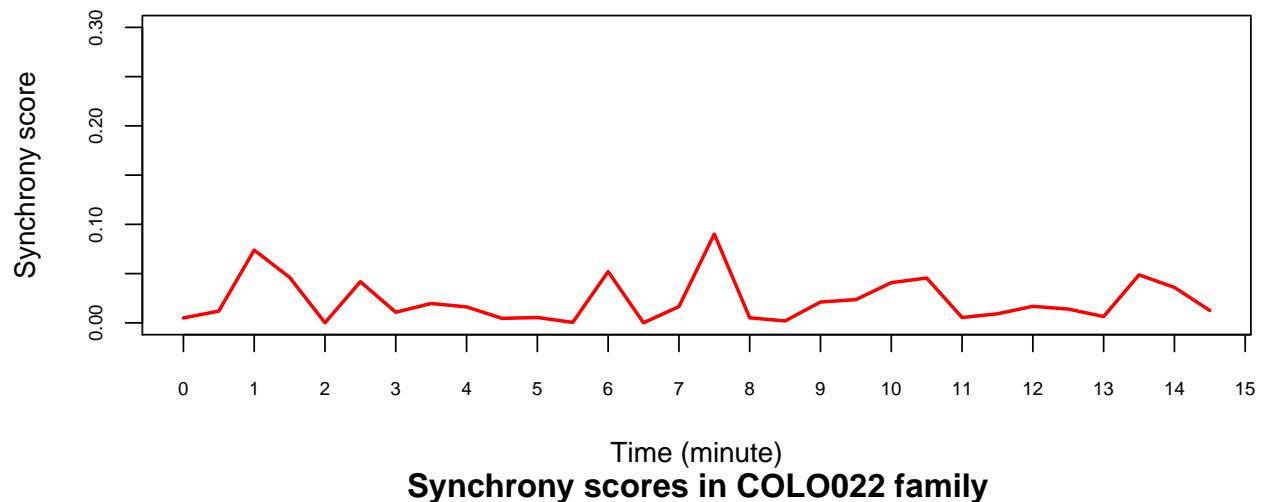
Synchrony scores in BEAL036 family



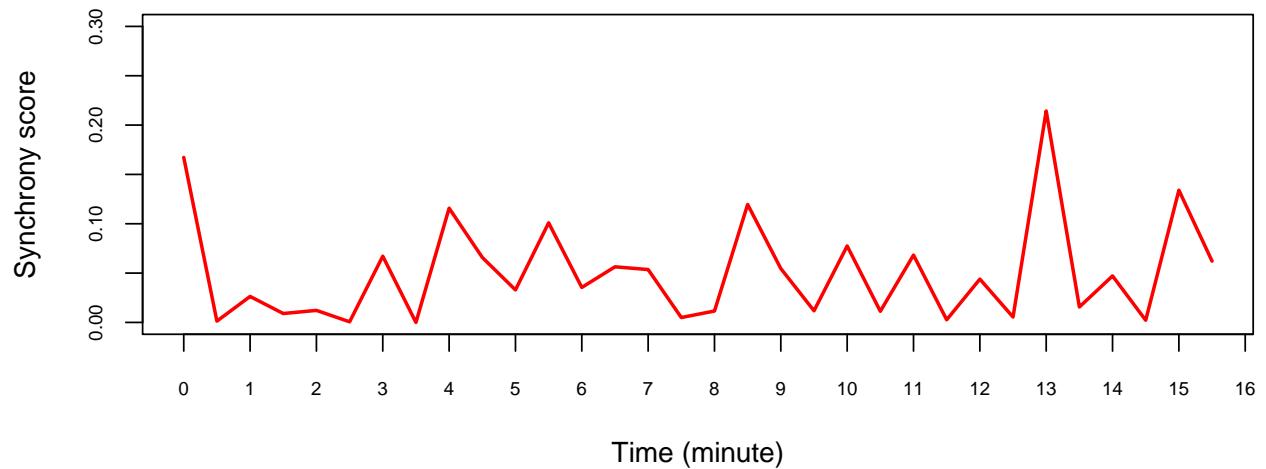
Synchrony scores in BEAM031 family



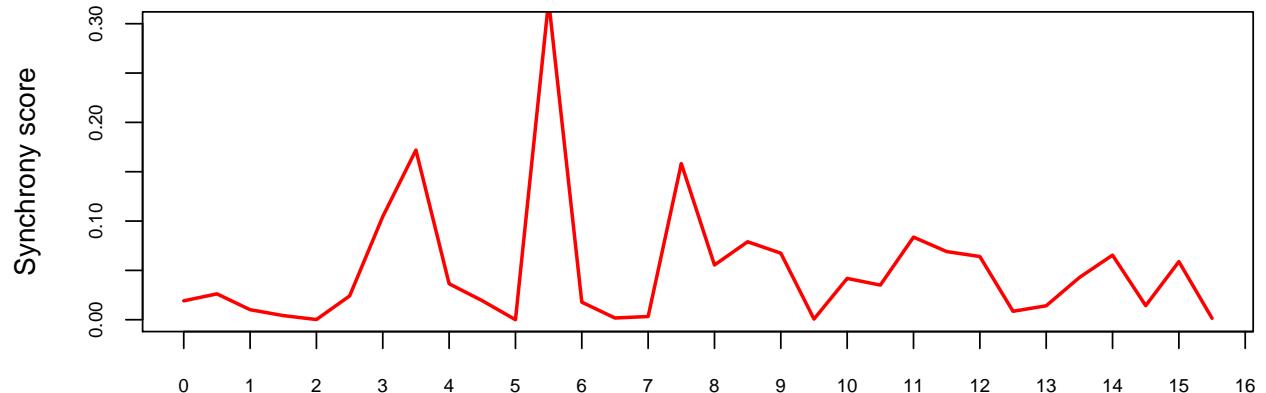
Synchrony scores in BRLO041 family



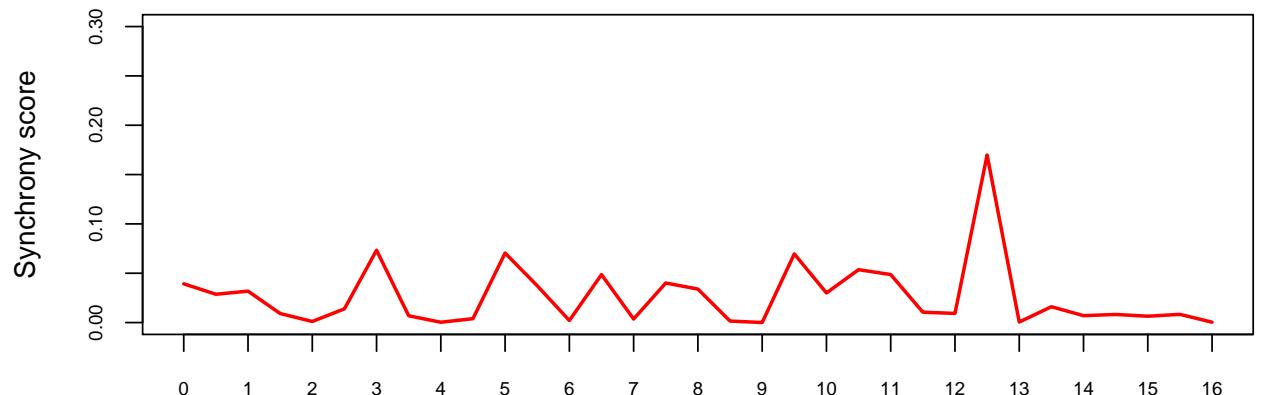
Synchrony scores in COLO022 family



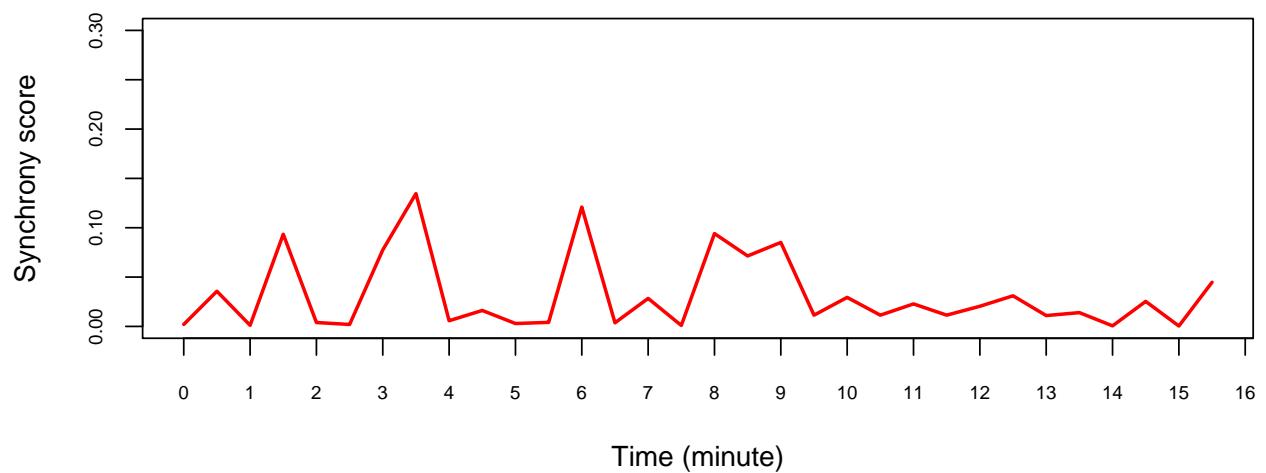
Synchrony scores in DIPE004 family



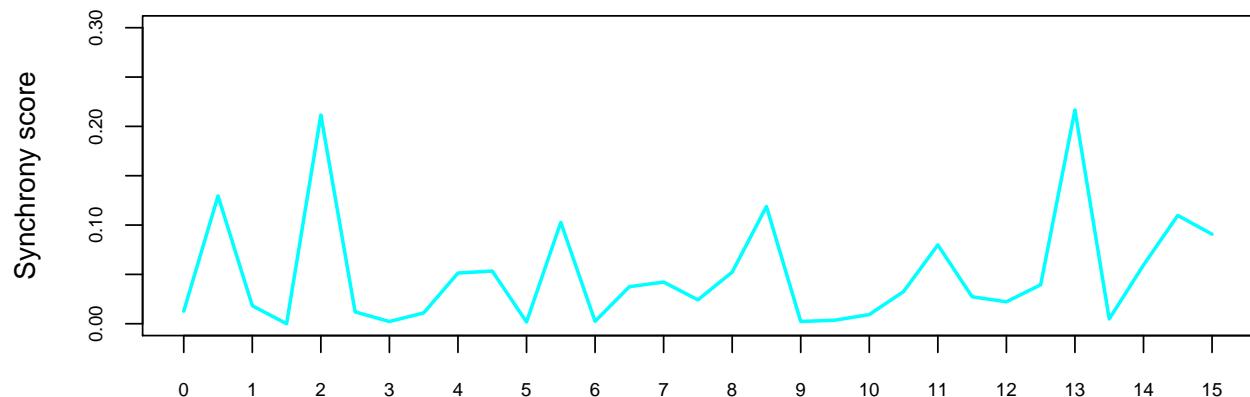
Synchrony scores in DOMA family



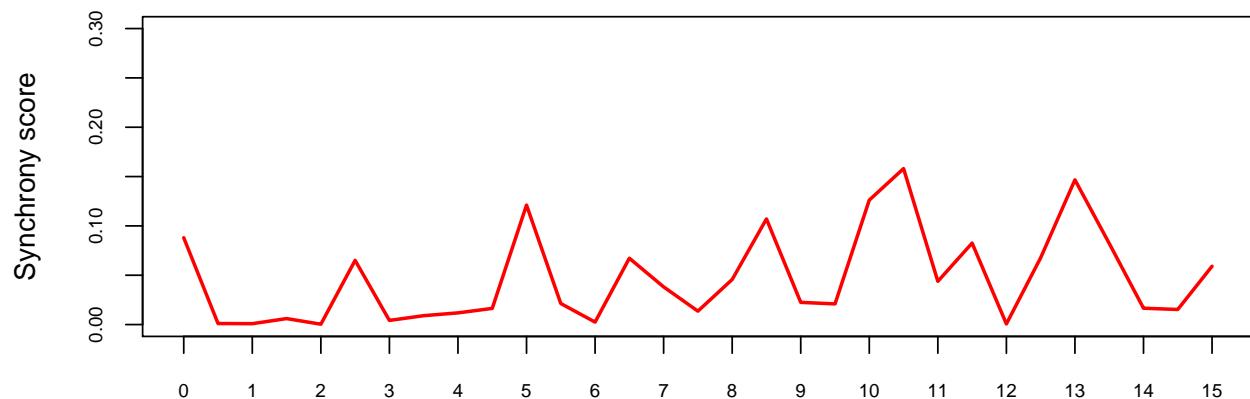
Synchrony scores in DRNE family



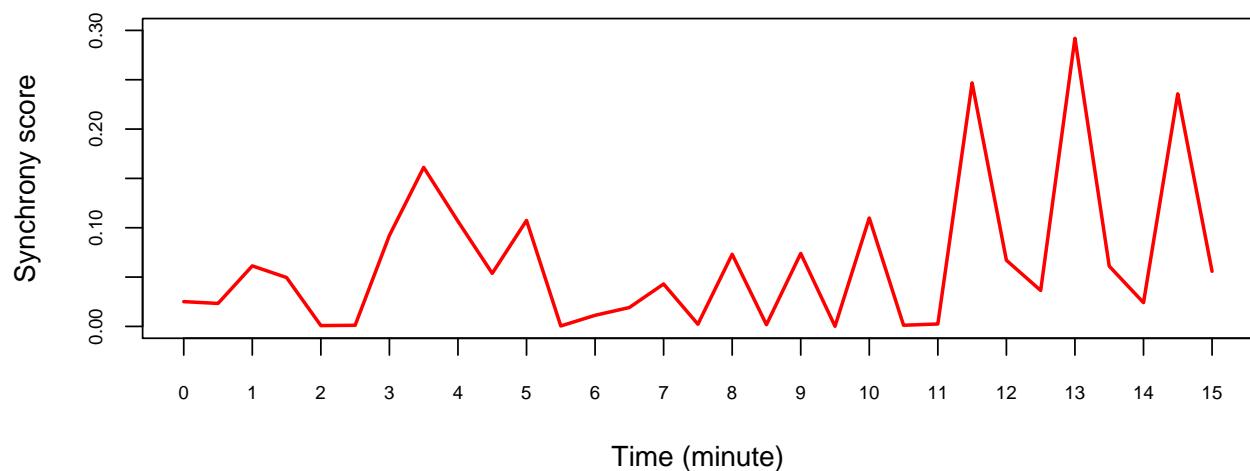
Synchrony scores in FOMA057 family



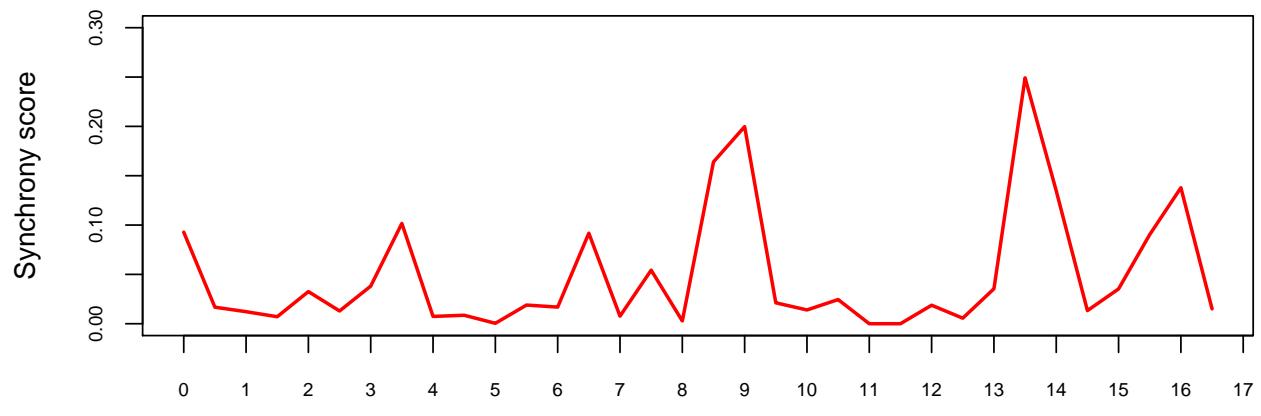
Synchrony scores in GROP039 family



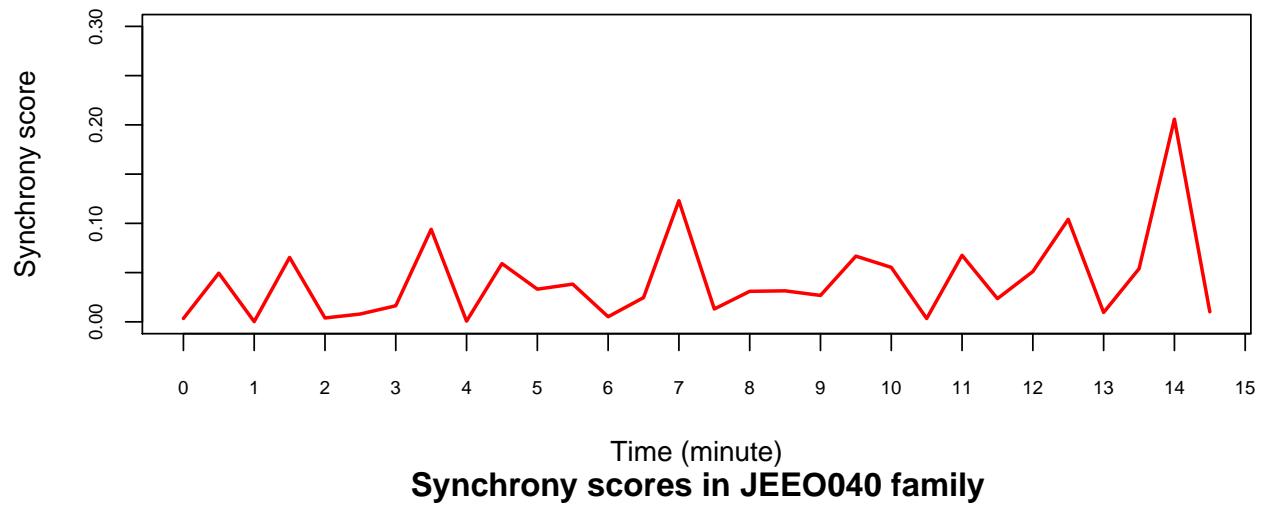
Synchrony scores in HAJA052 family



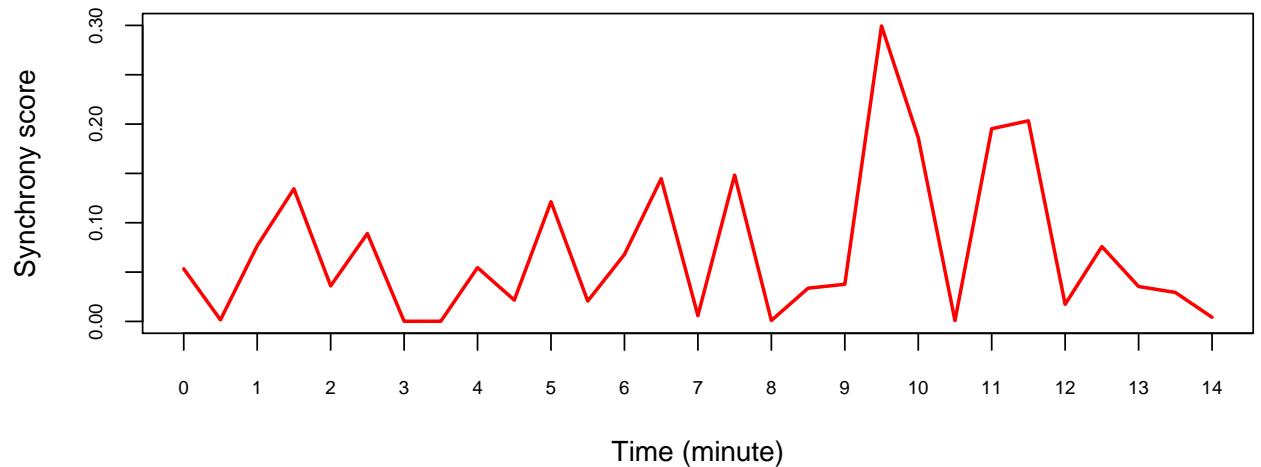
Synchrony scores in HUMA058 family



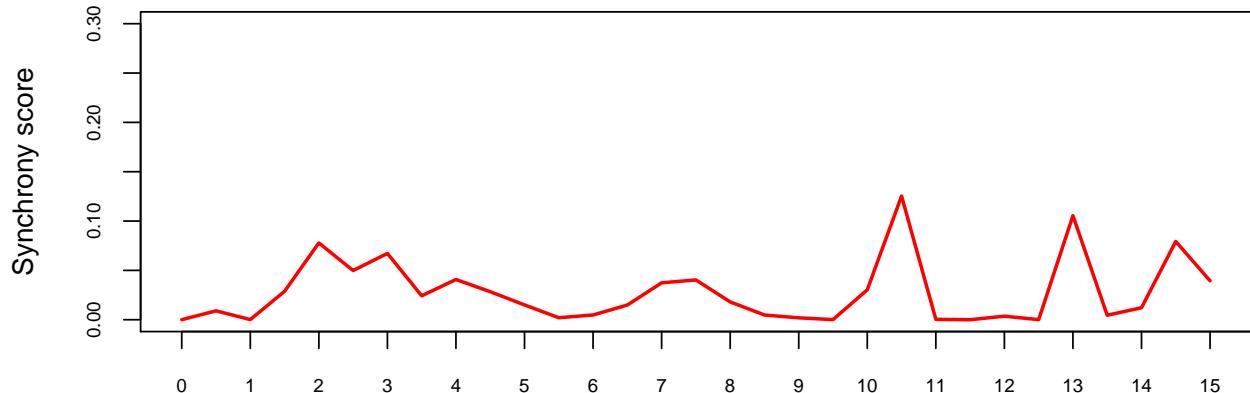
Synchrony scores in JAEM046 family



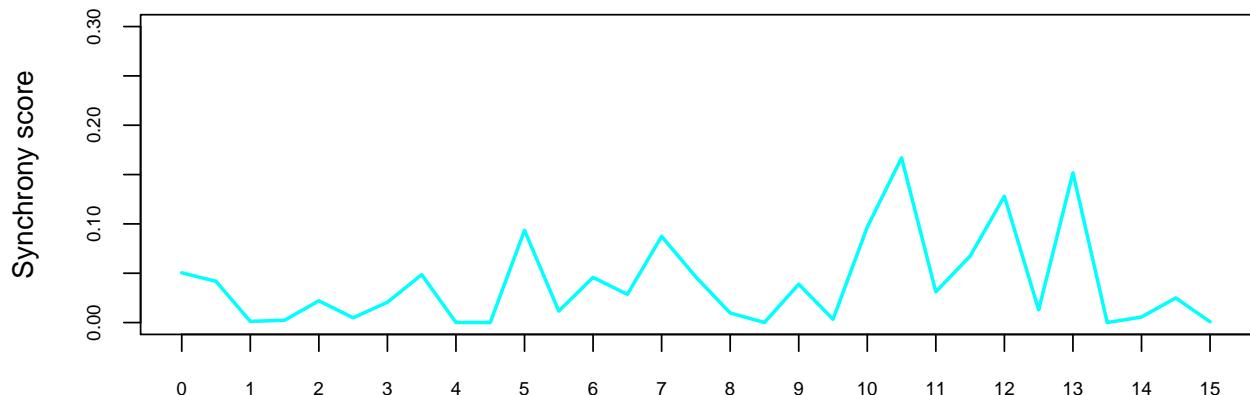
Synchrony scores in JEEO040 family



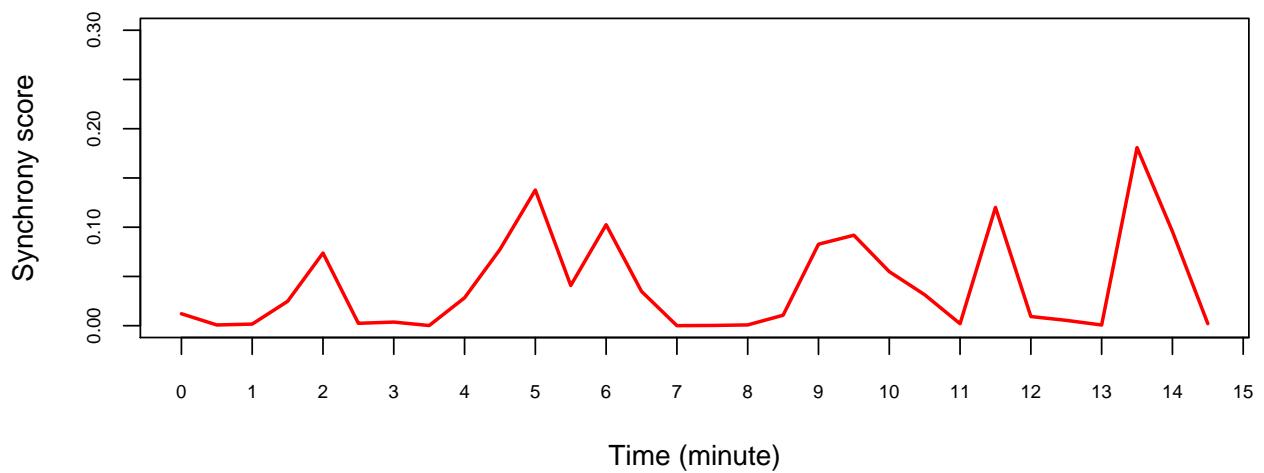
Synchrony scores in JOCE014 family



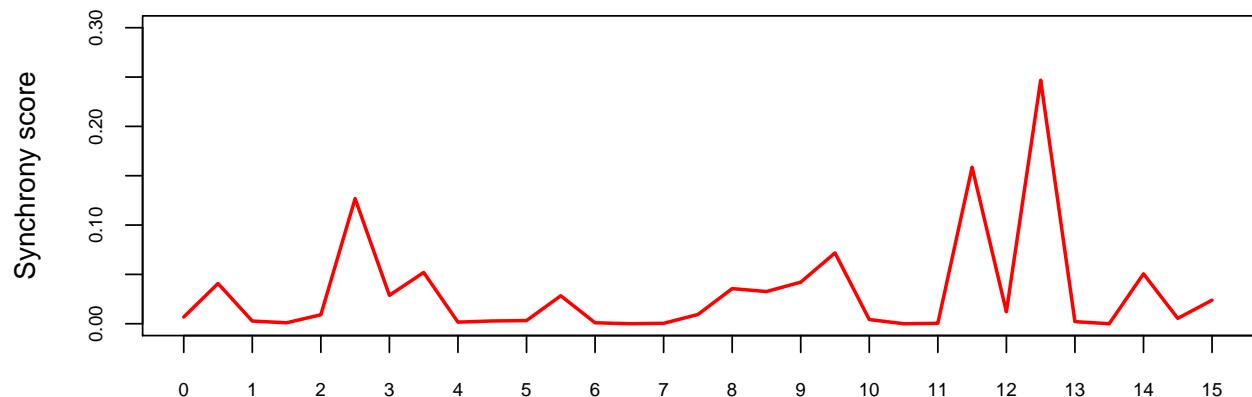
Synchrony scores in LACL family



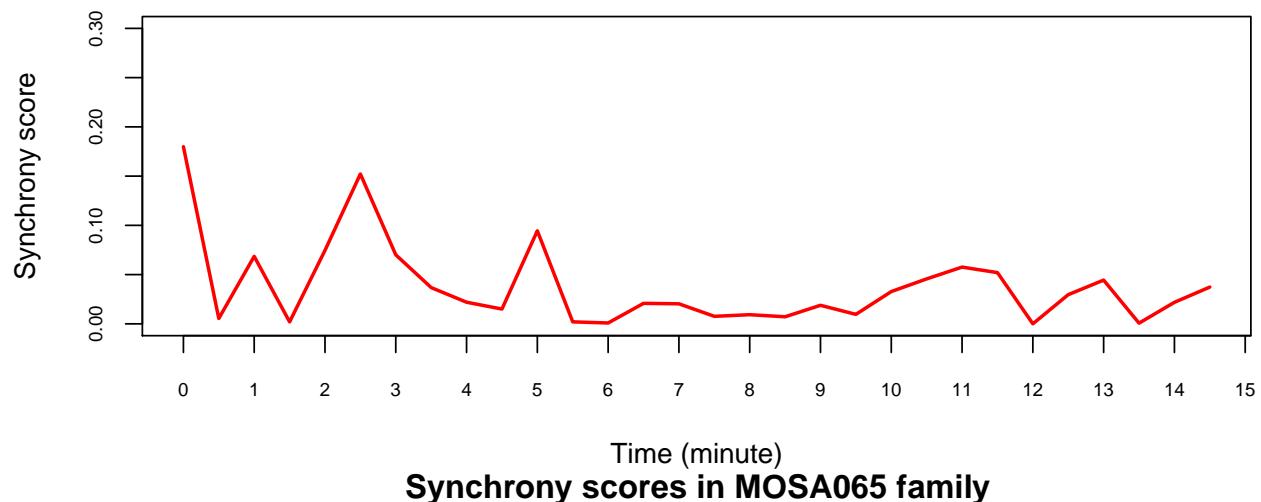
Synchrony scores in MAEL048 family



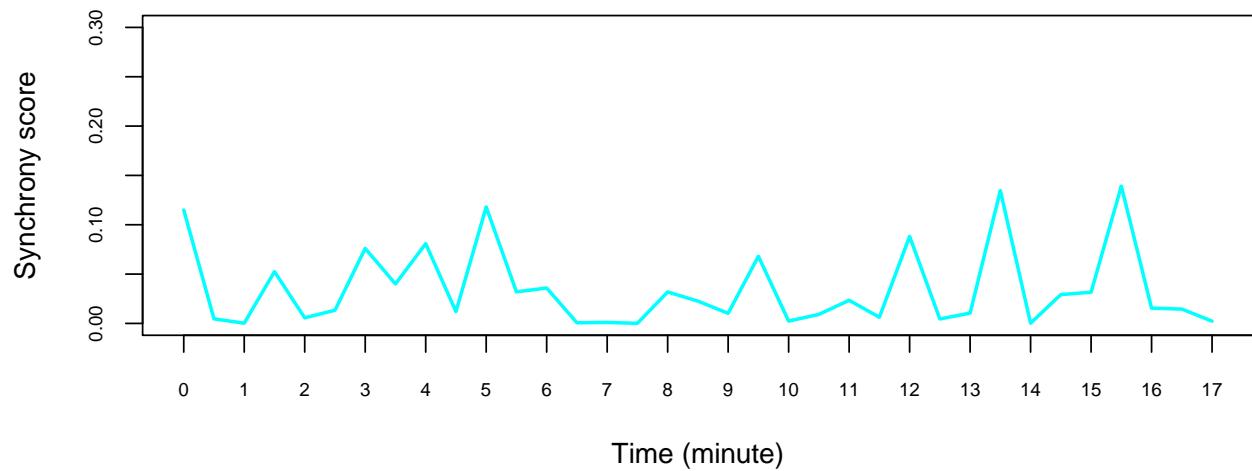
Synchrony scores in MAME20 family



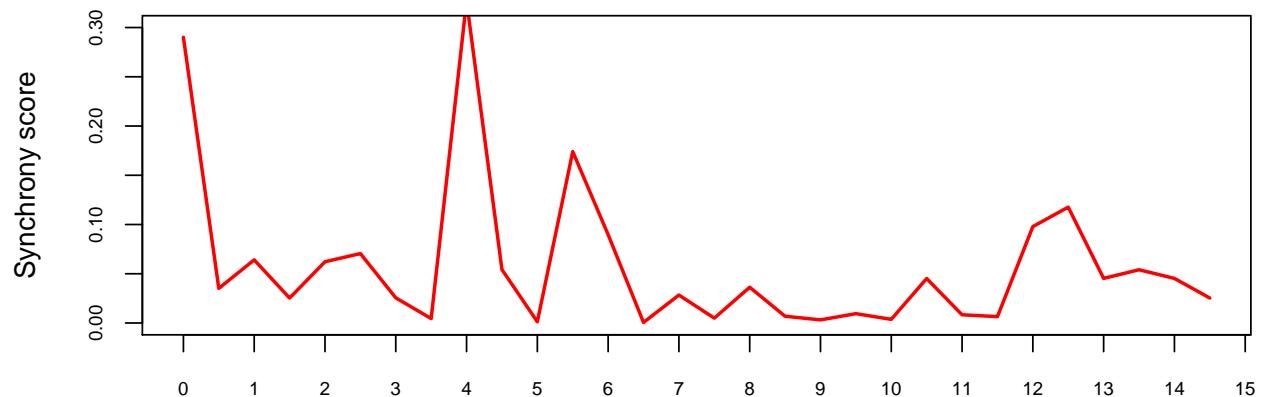
Synchrony scores in MIPH043 family



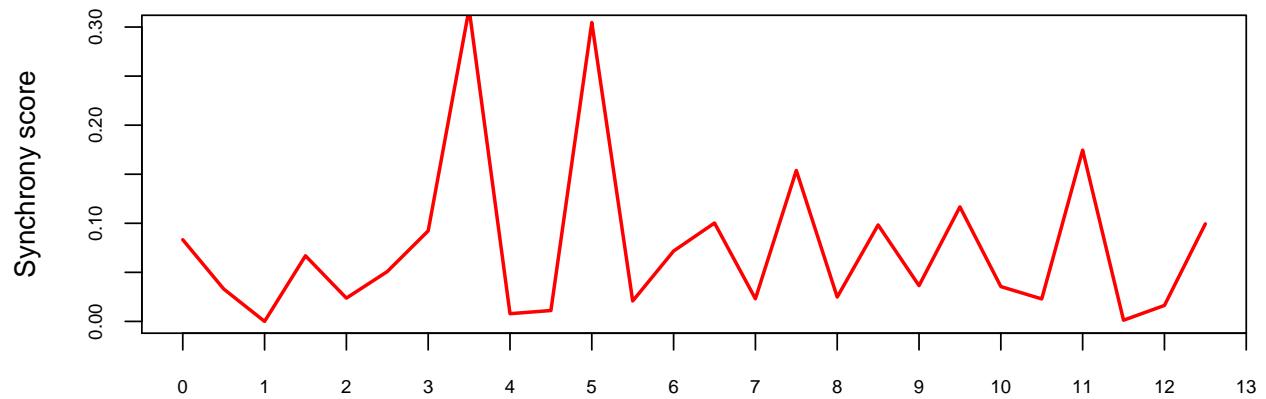
Synchrony scores in MOSA065 family



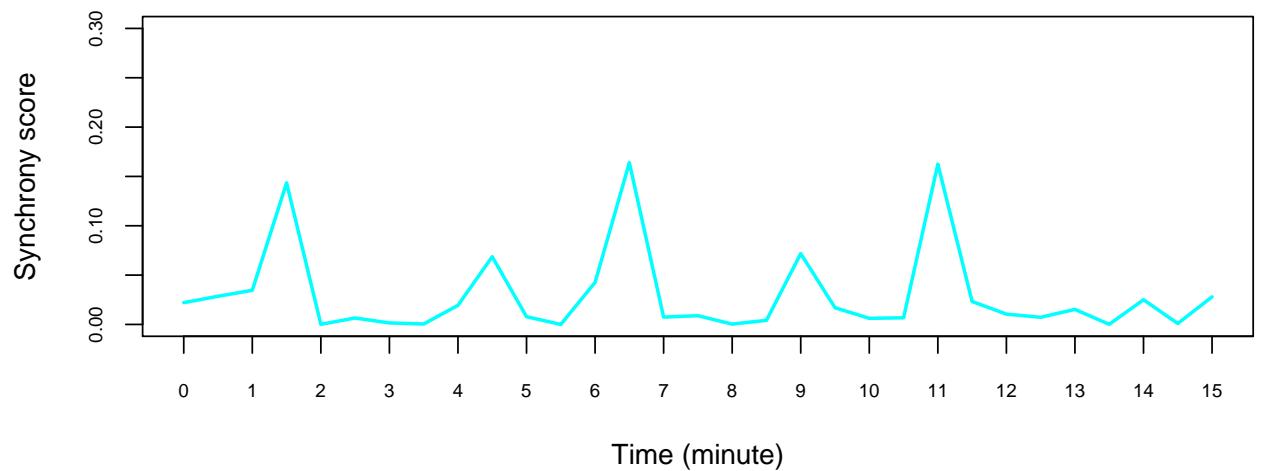
Synchrony scores in NAMA045 family



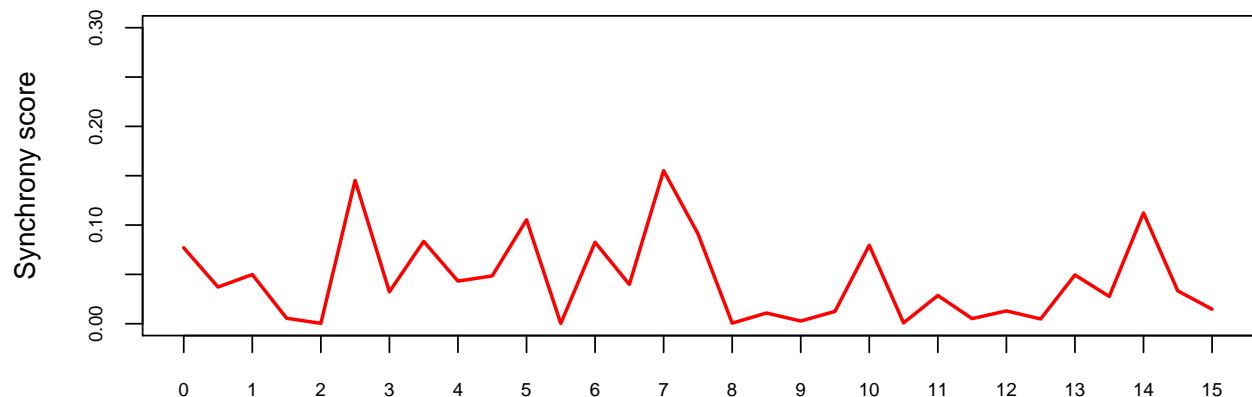
Synchrony scores in NUMA027 family



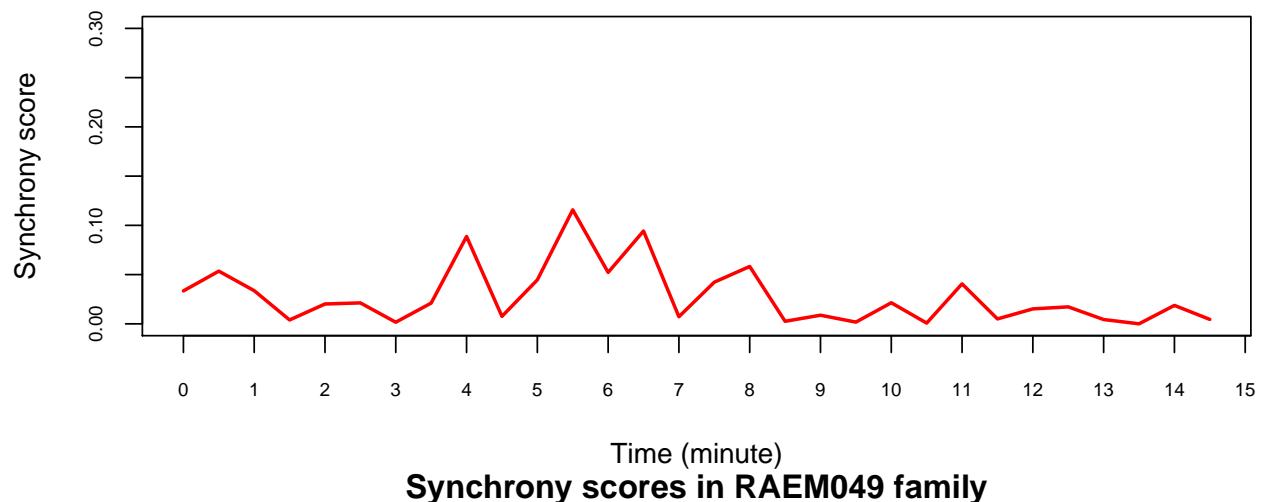
Synchrony scores in OGGA034 family



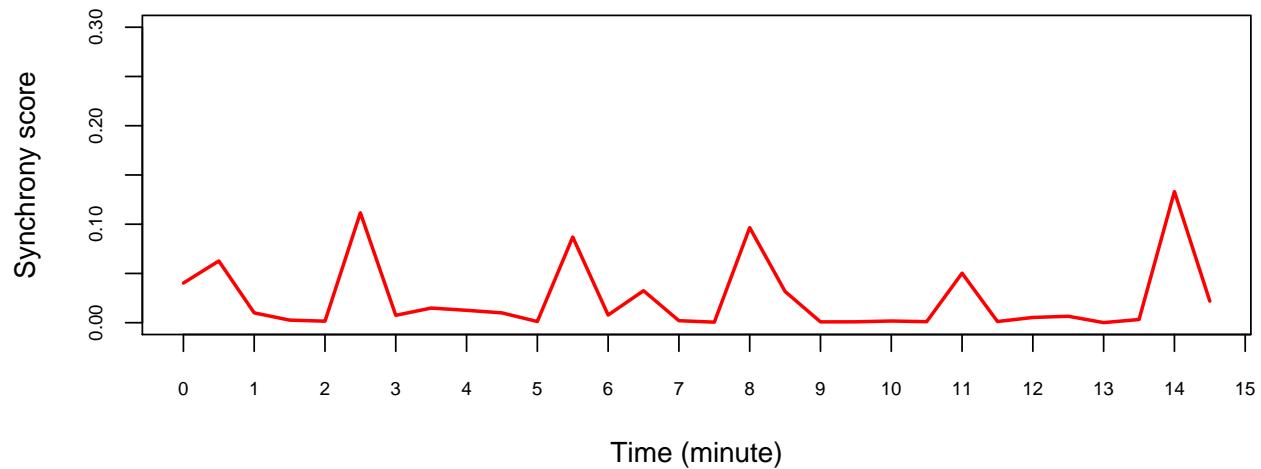
Synchrony scores in PAMA029 family



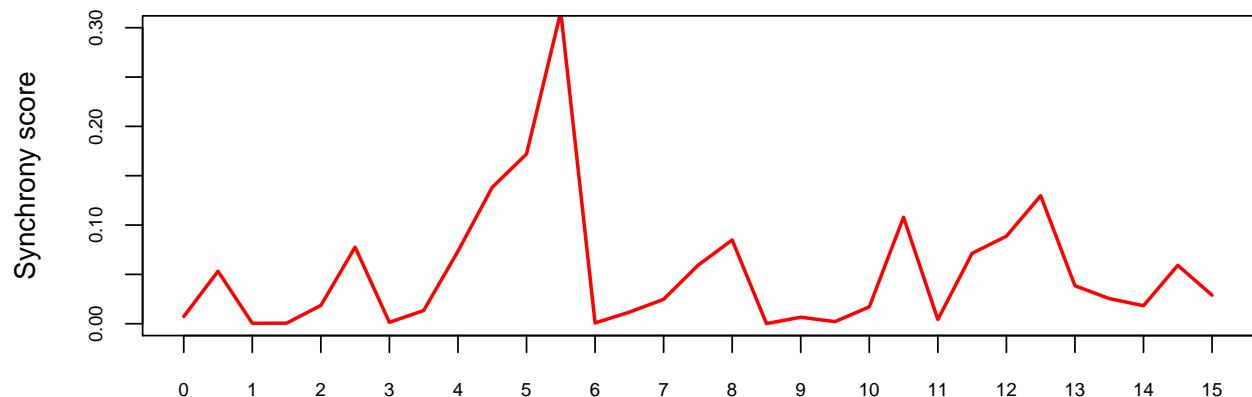
Synchrony scores in PELI020 family



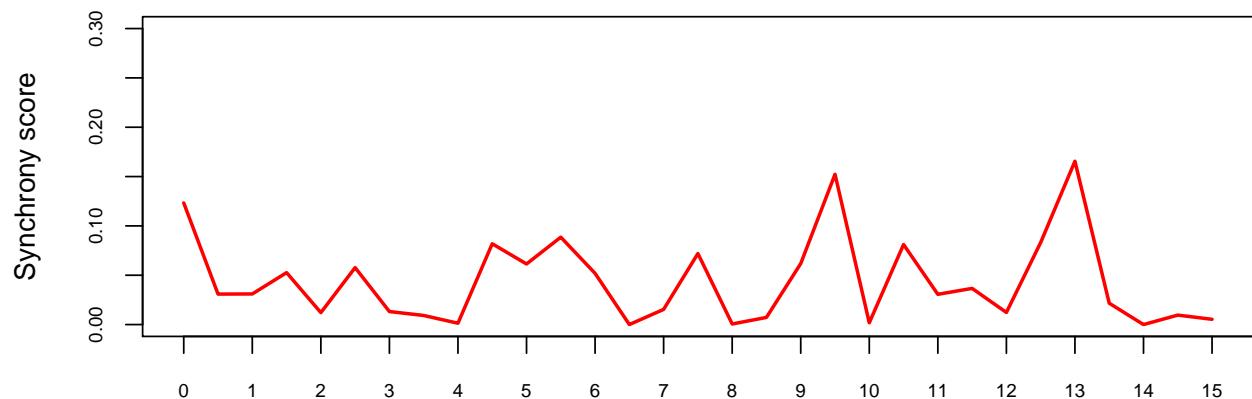
Synchrony scores in RAEM049 family



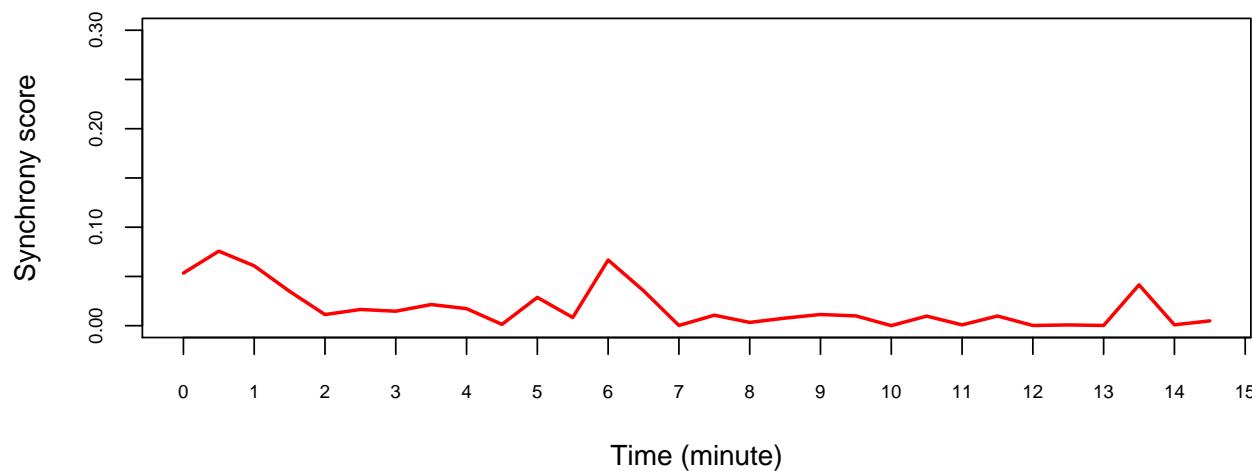
Synchrony scores in RAMA054 family



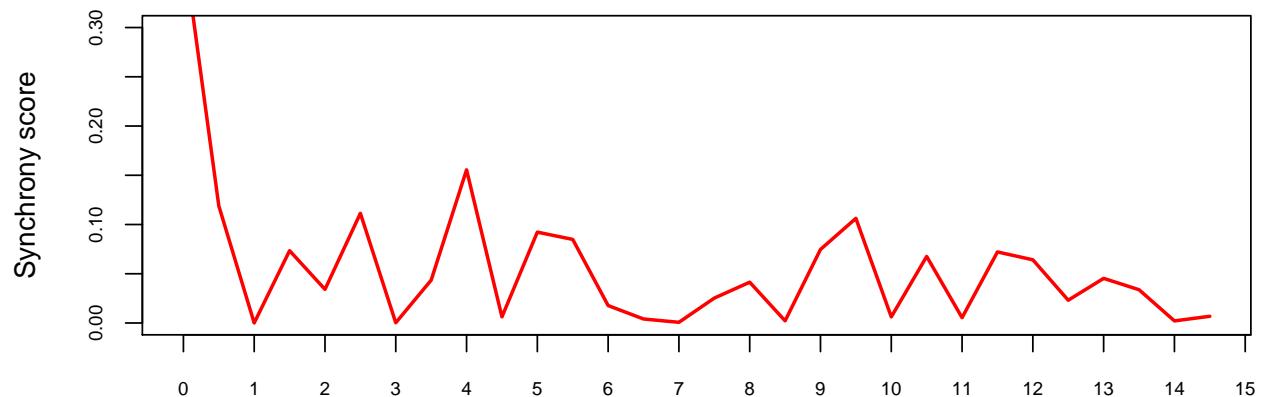
Synchrony scores in SEEM035 family



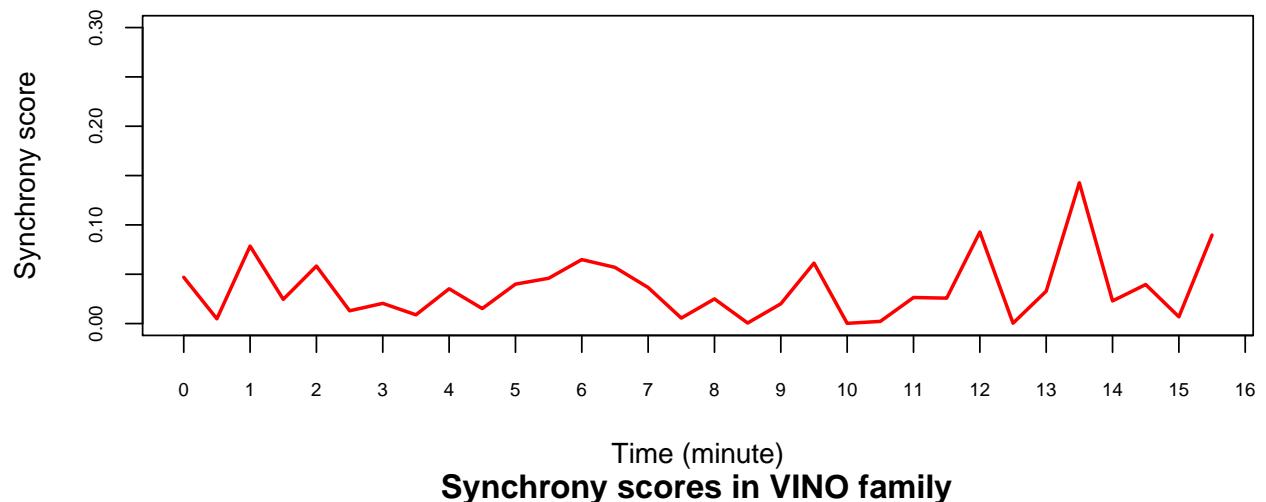
Synchrony scores in SHAN042 family



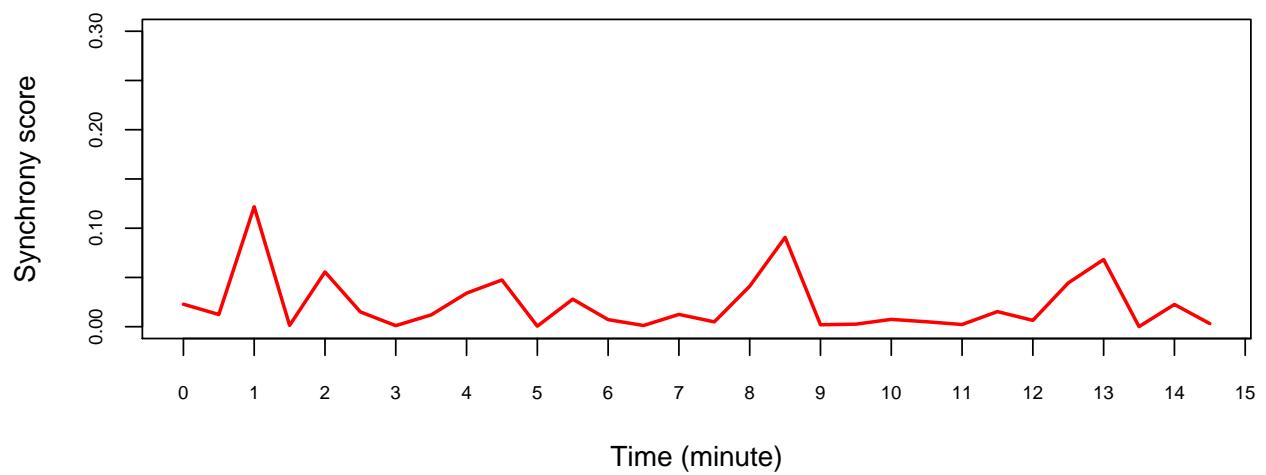
Synchrony scores in SOGA061 family



Synchrony scores in TIUG032 family



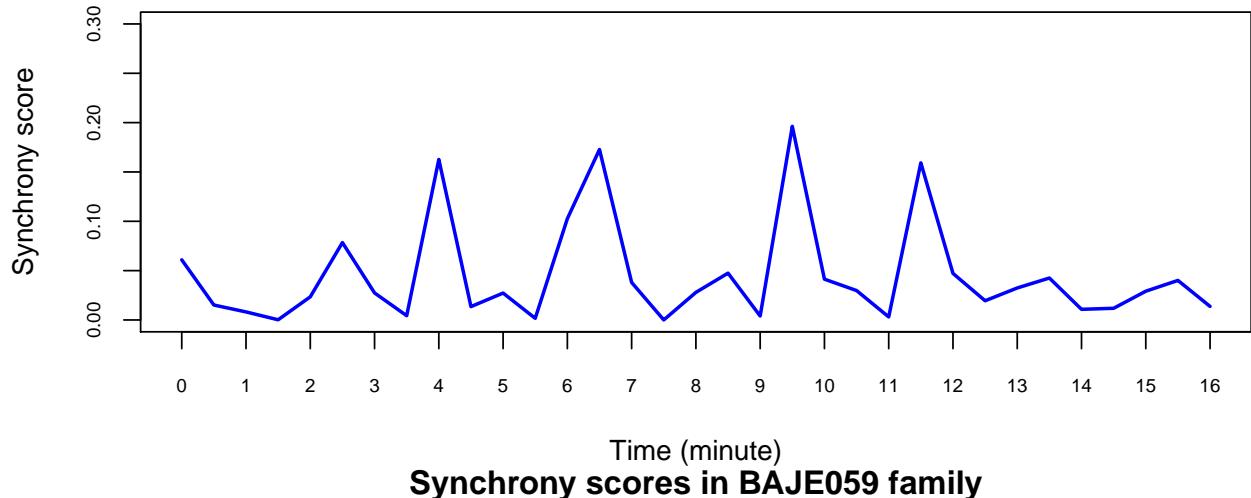
Synchrony scores in VINO family



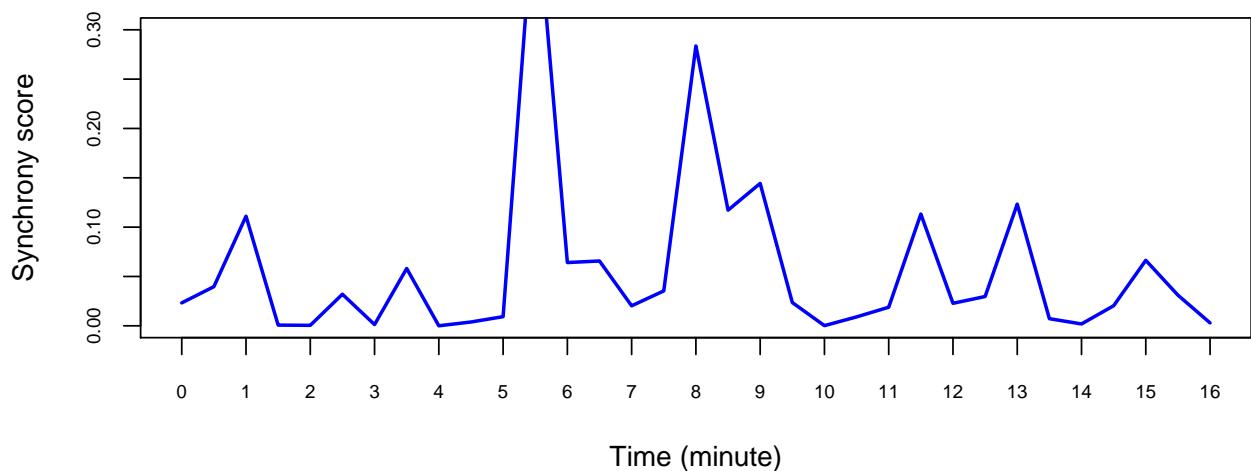
Synchrony scores noLog for each dyad, triad and for the whole group

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

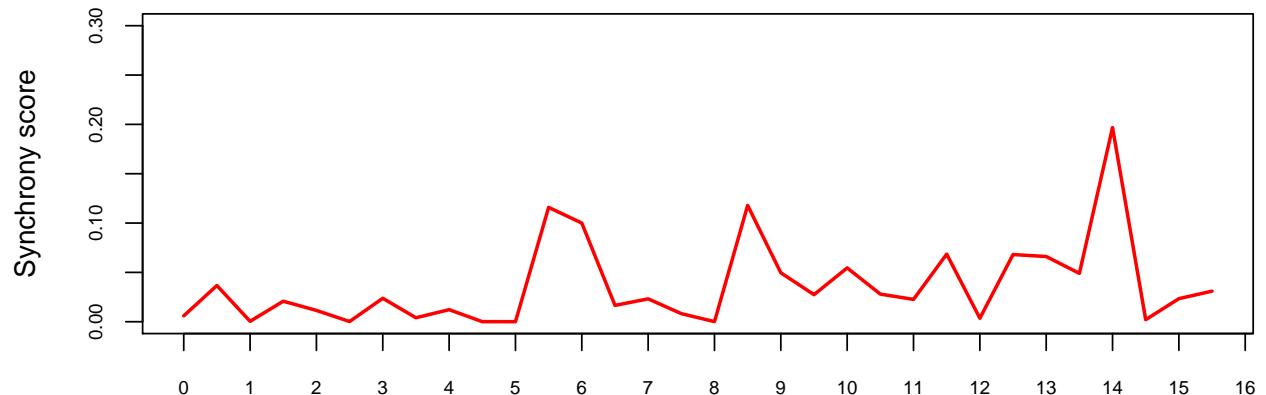
Synchrony scores in 1606 family



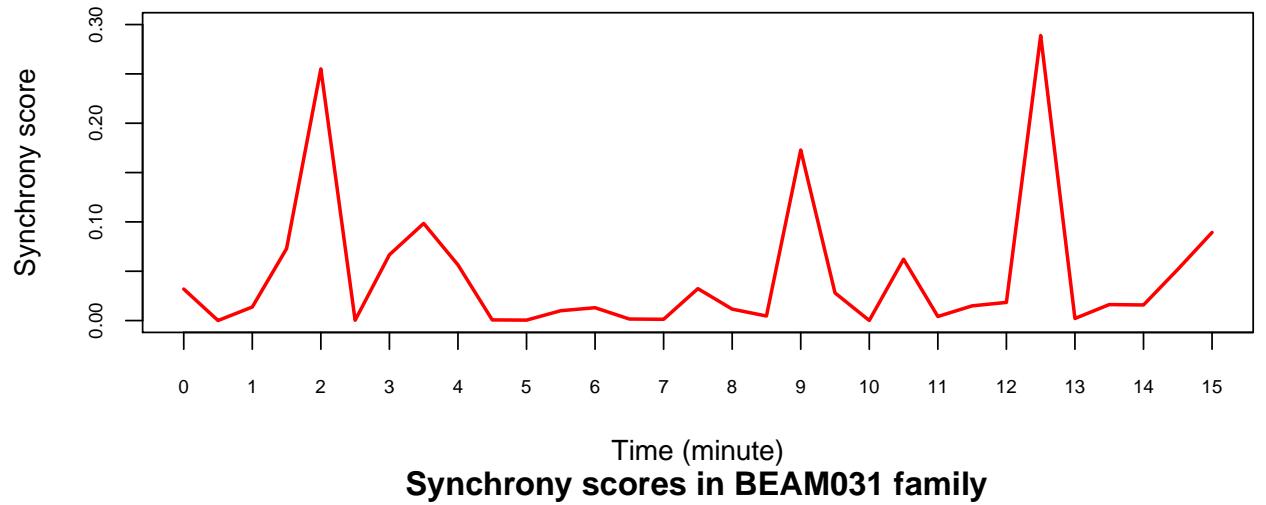
Synchrony scores in BAJE059 family



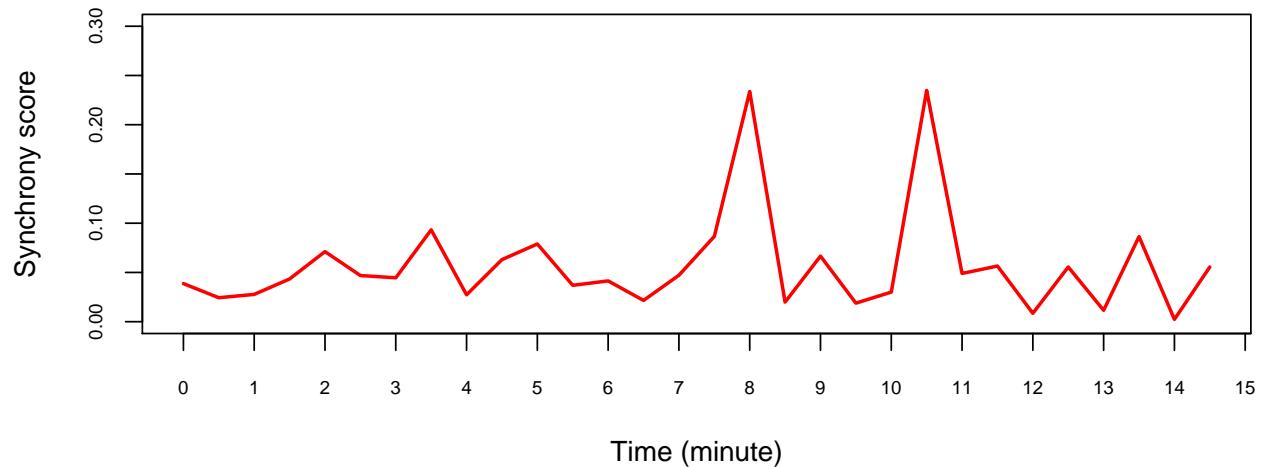
Synchrony scores in BALU062 family



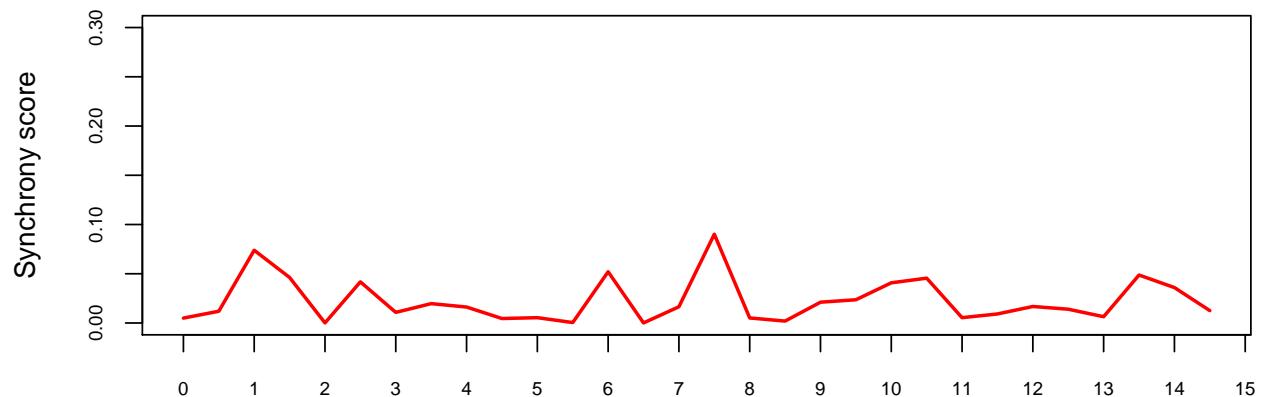
Synchrony scores in BEAL036 family



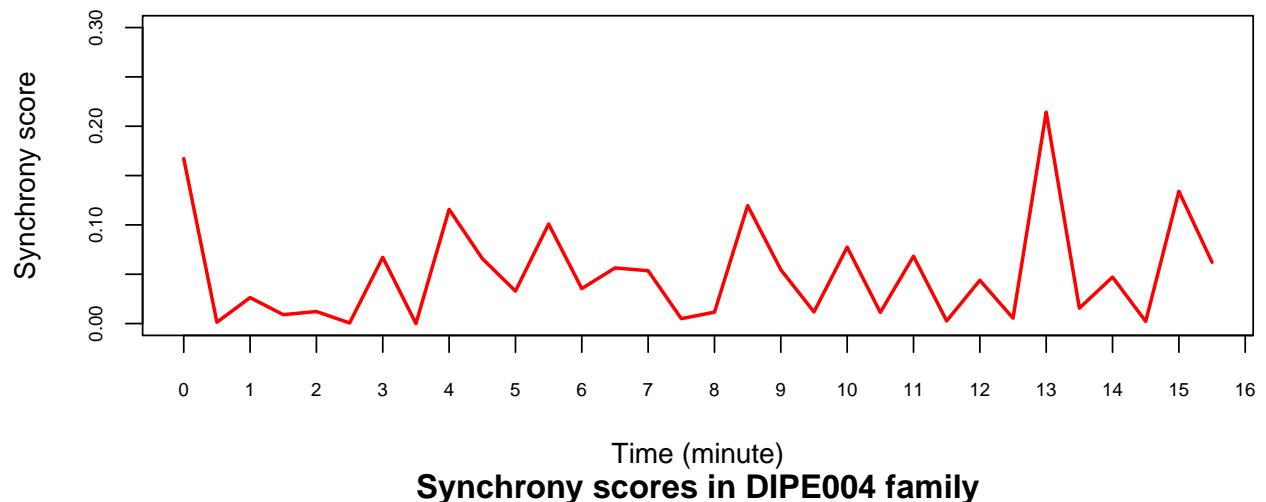
Synchrony scores in BEAM031 family



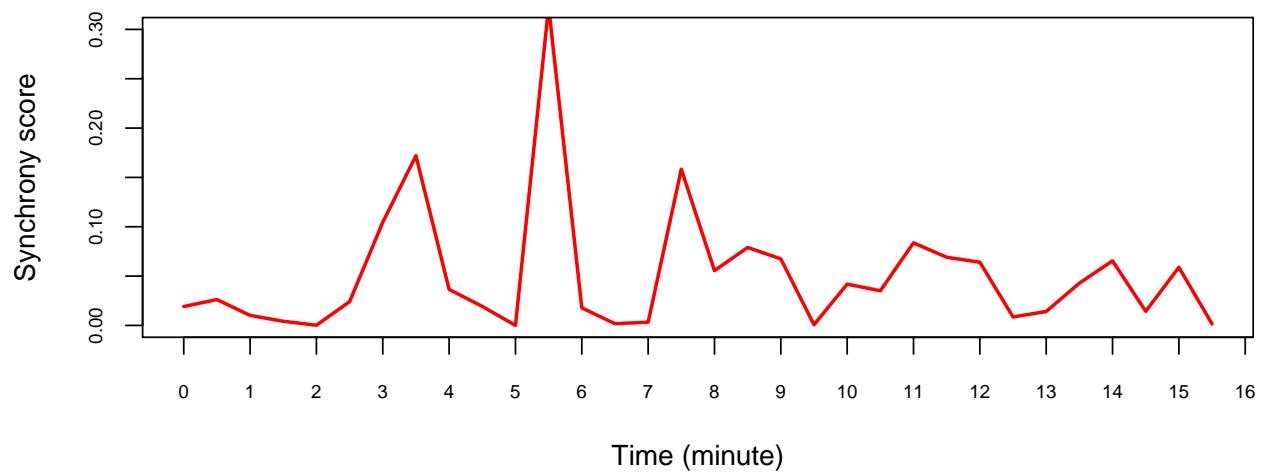
Synchrony scores in BRLO041 family



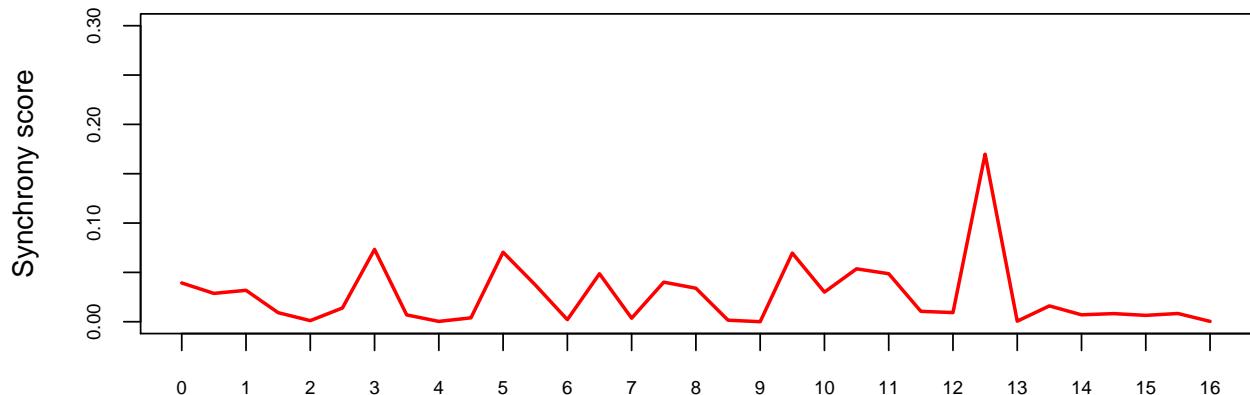
Synchrony scores in COLO022 family



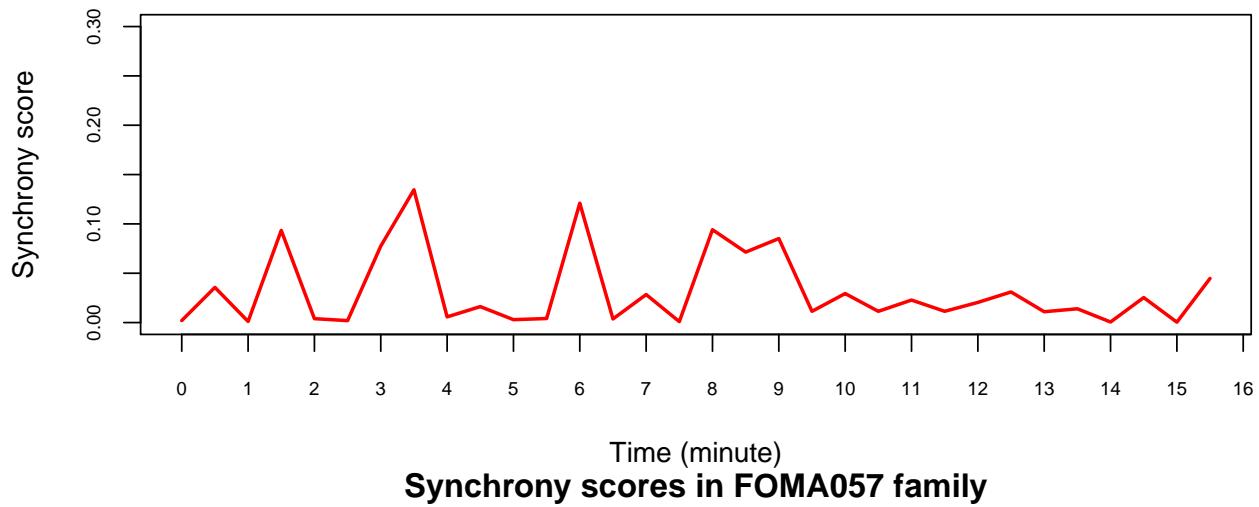
Synchrony scores in DIPE004 family



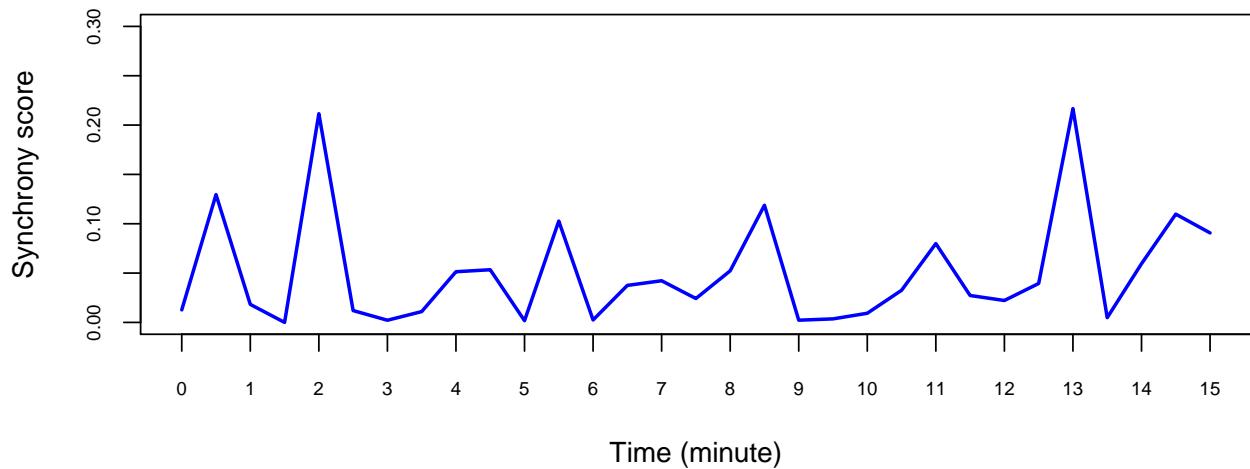
Synchrony scores in DOMA family



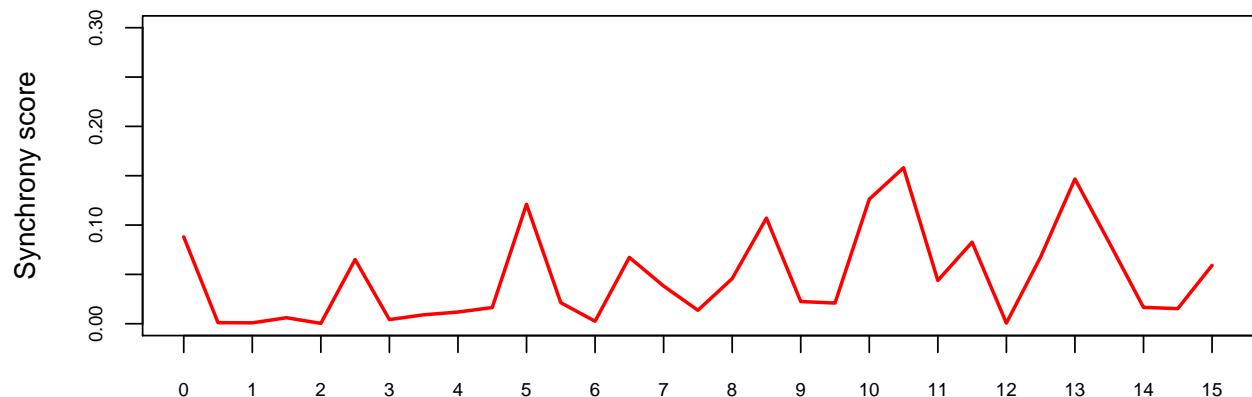
Synchrony scores in DRNE family



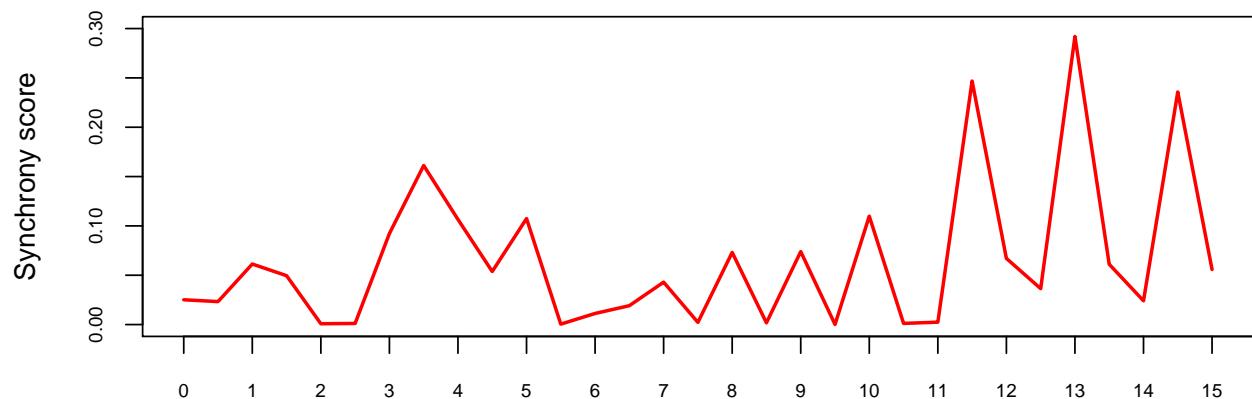
Synchrony scores in FOMA057 family



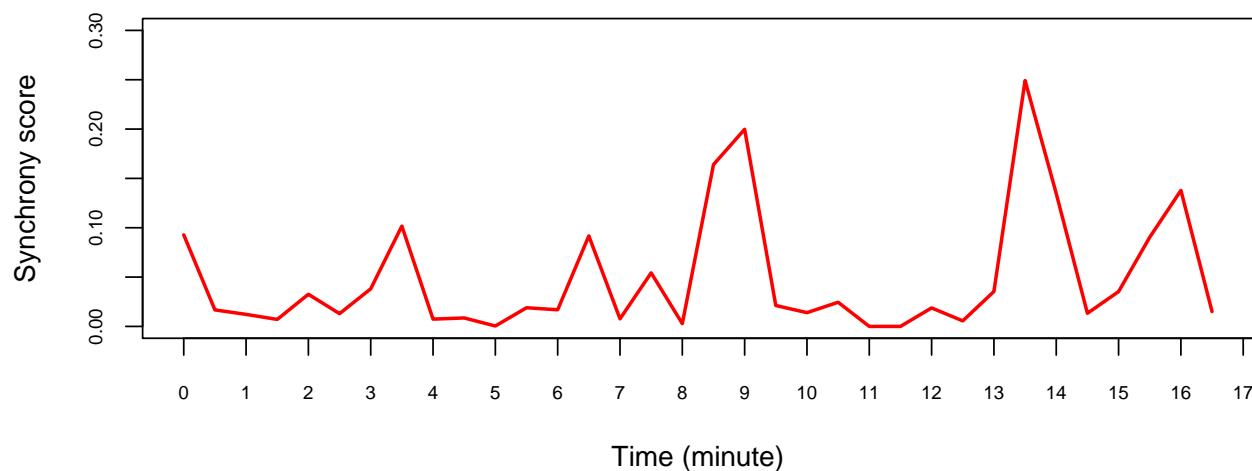
Synchrony scores in GROP039 family



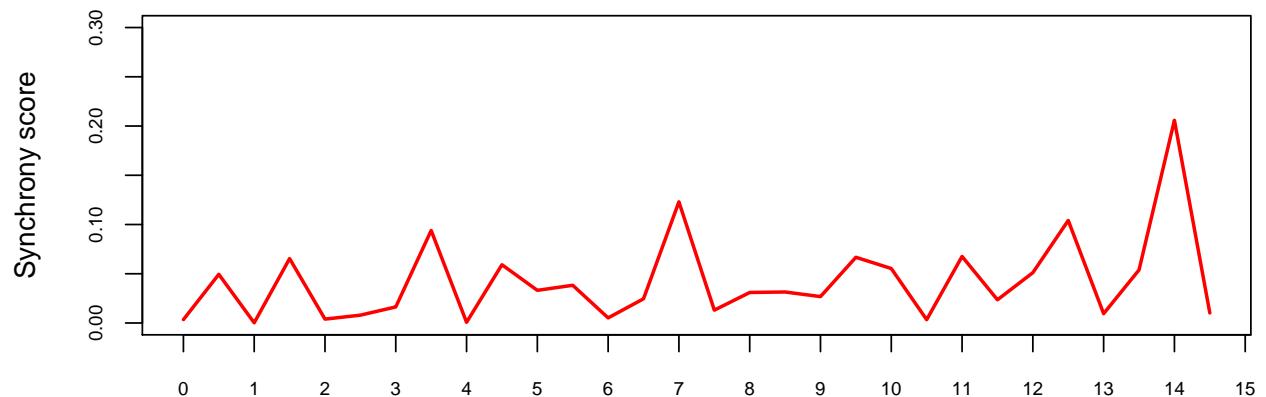
Synchrony scores in HAJA052 family



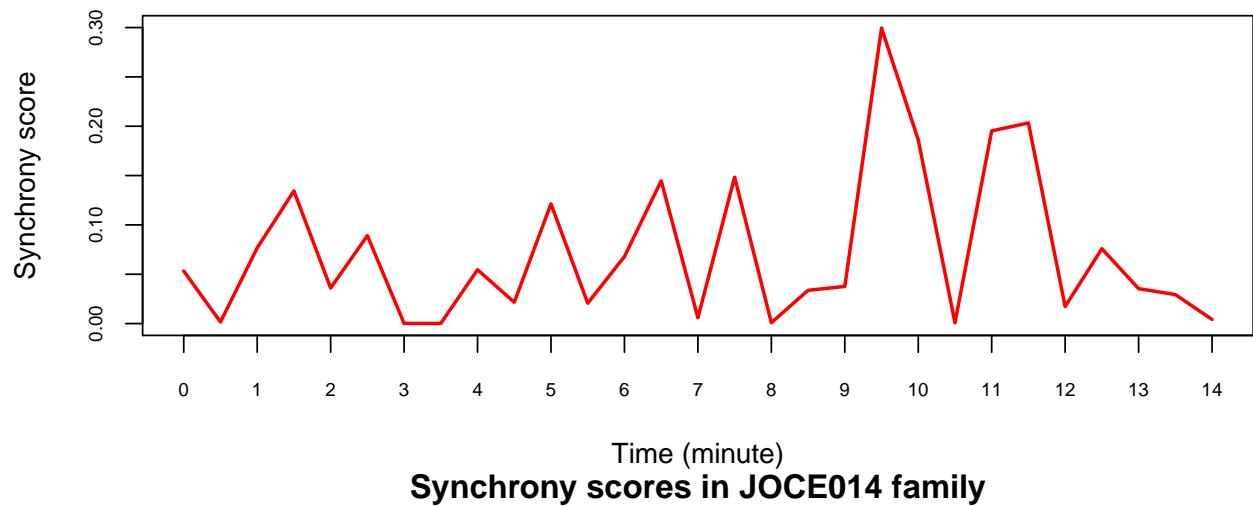
Synchrony scores in HUMA058 family



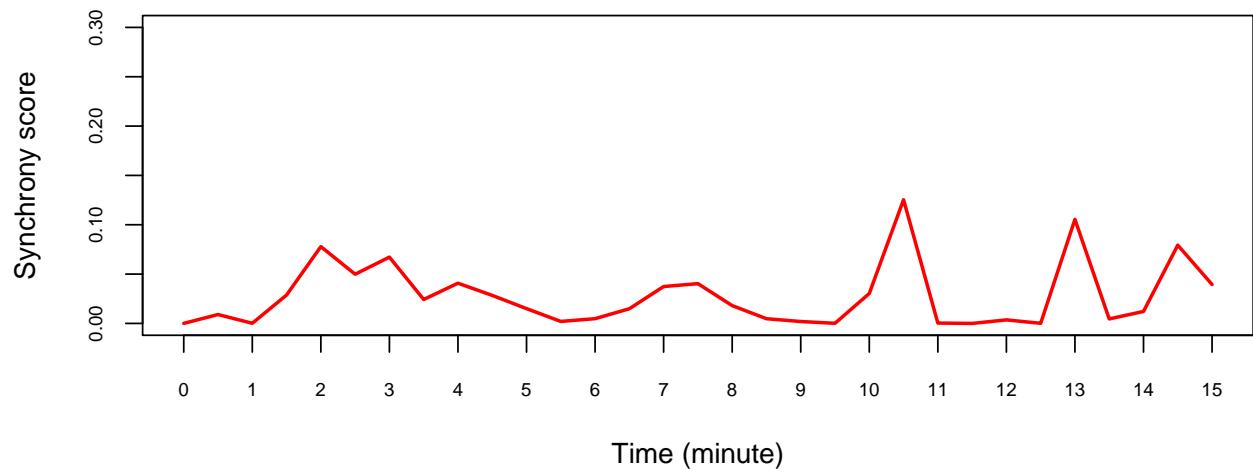
Synchrony scores in JAEM046 family



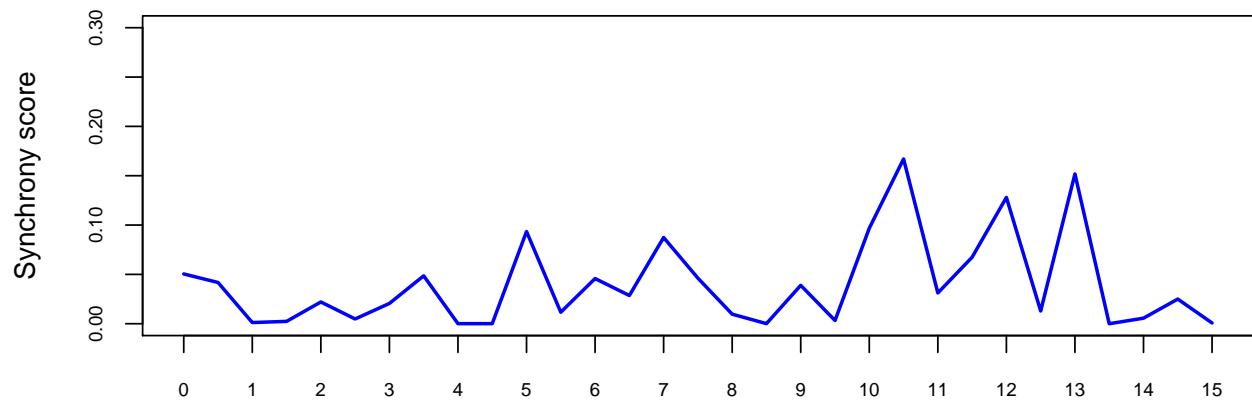
Synchrony scores in JEEO040 family



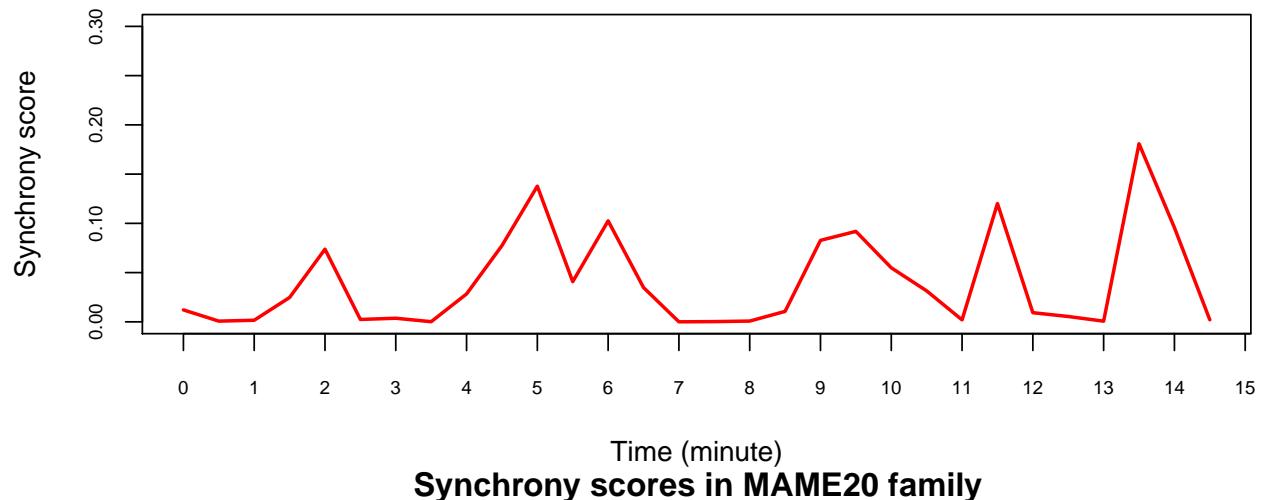
Synchrony scores in JOCE014 family



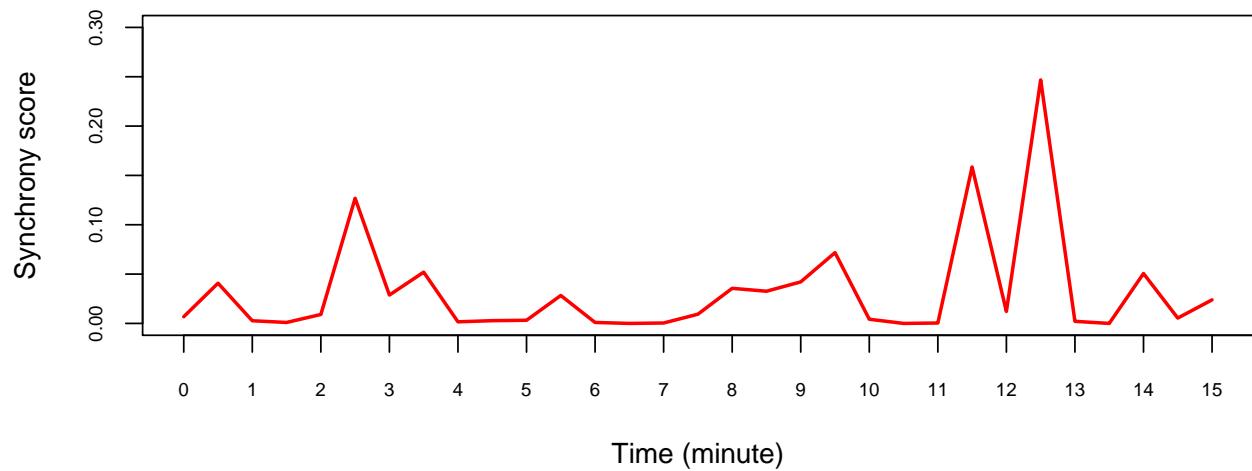
Synchrony scores in LACL family



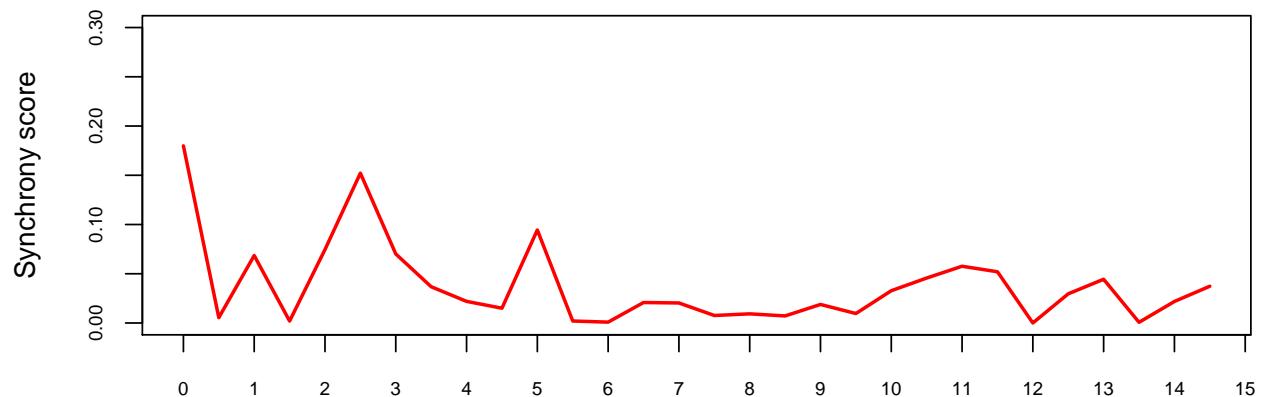
Synchrony scores in MAEL048 family



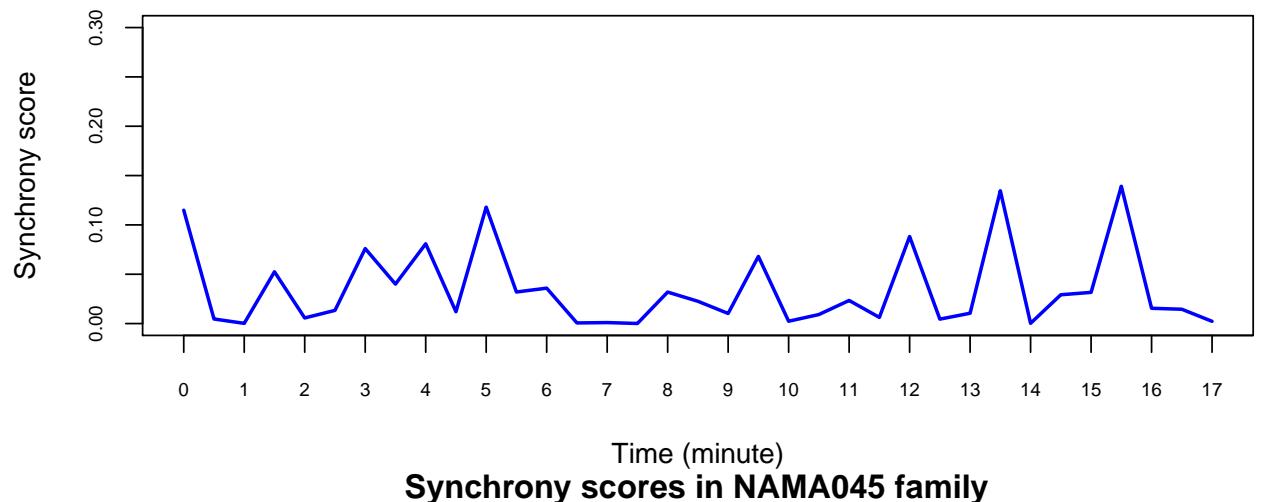
Synchrony scores in MAME20 family



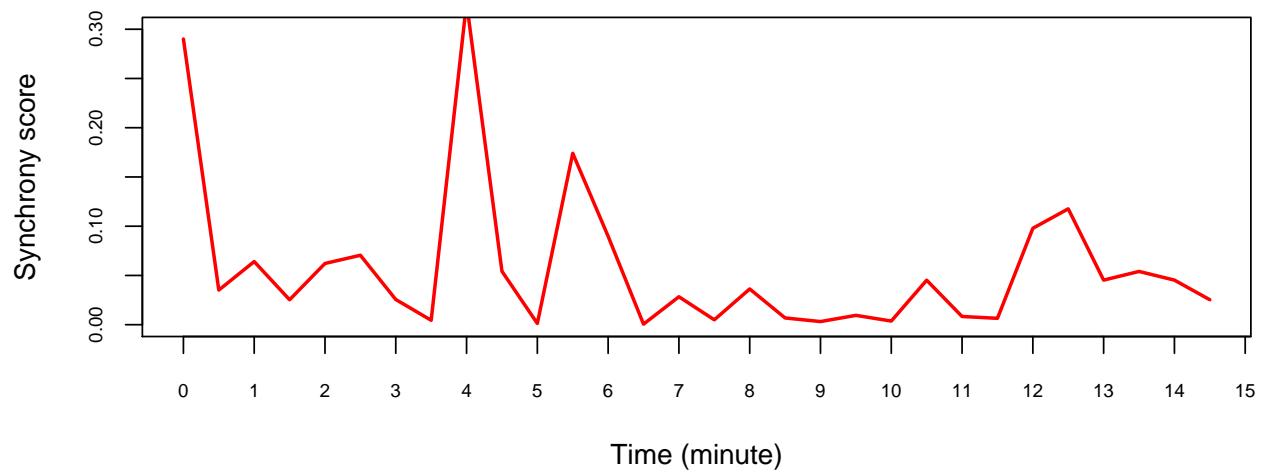
Synchrony scores in MIPH043 family



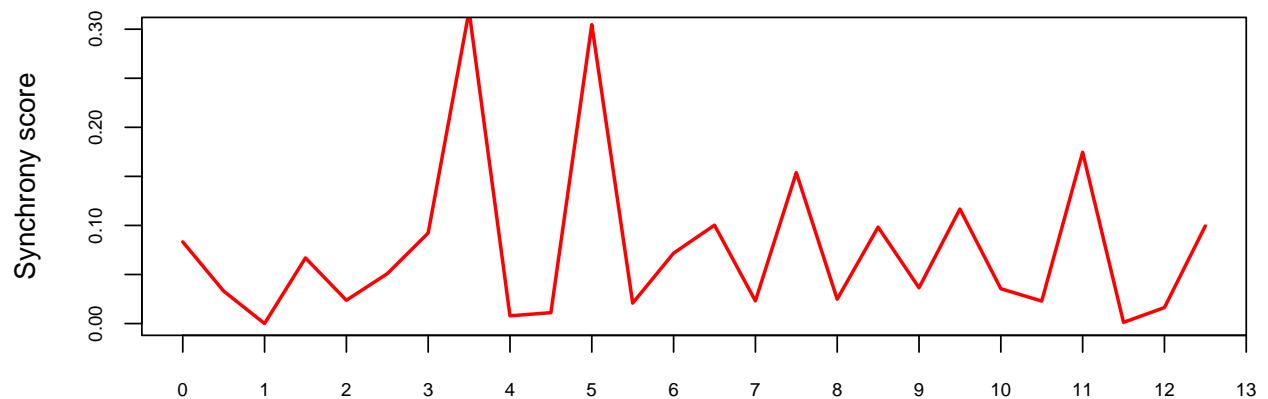
Synchrony scores in MOSA065 family



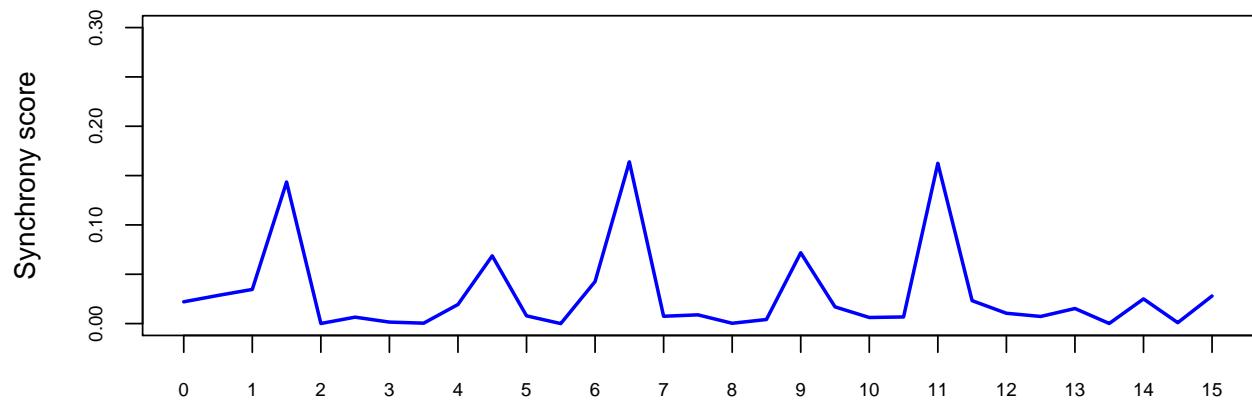
Synchrony scores in NAMA045 family



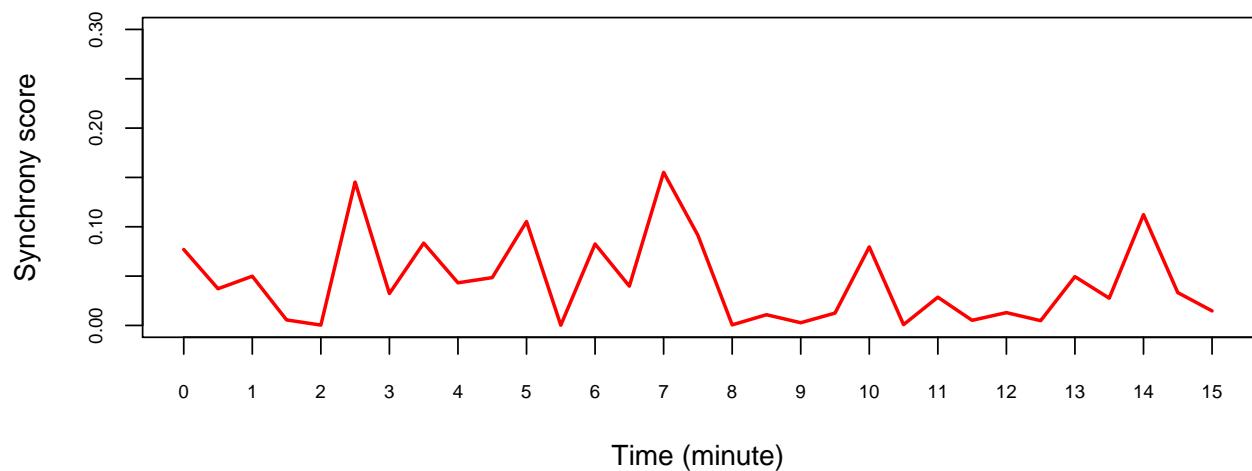
Synchrony scores in NUMA027 family



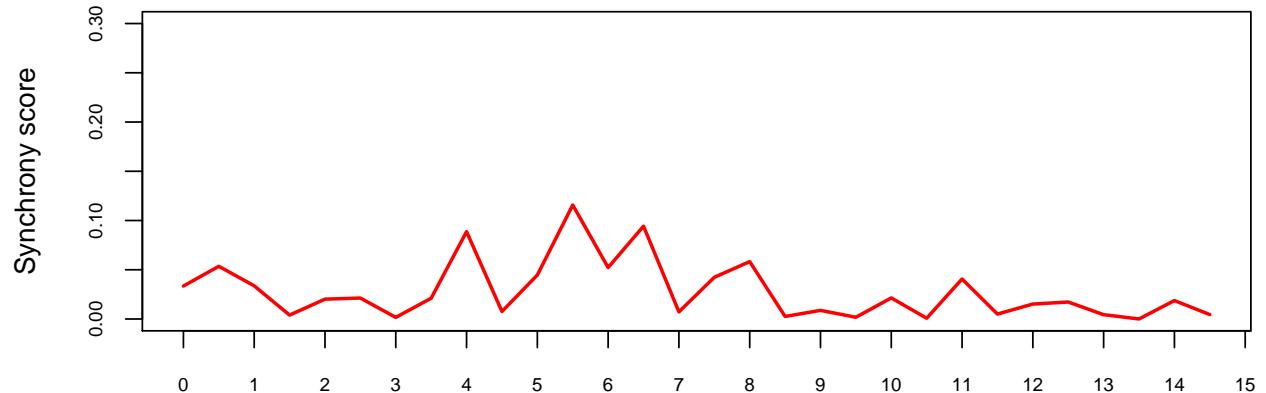
Synchrony scores in OGGA034 family



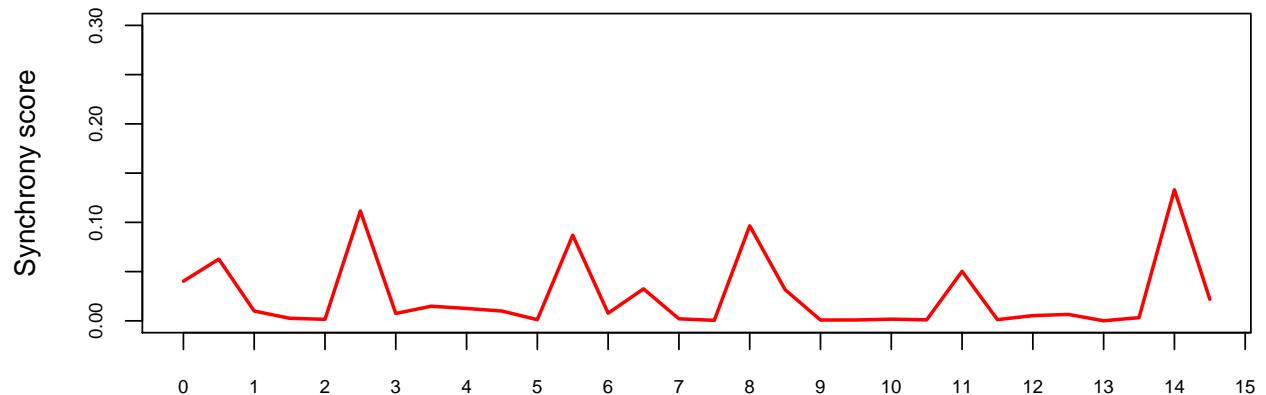
Synchrony scores in PAMA029 family



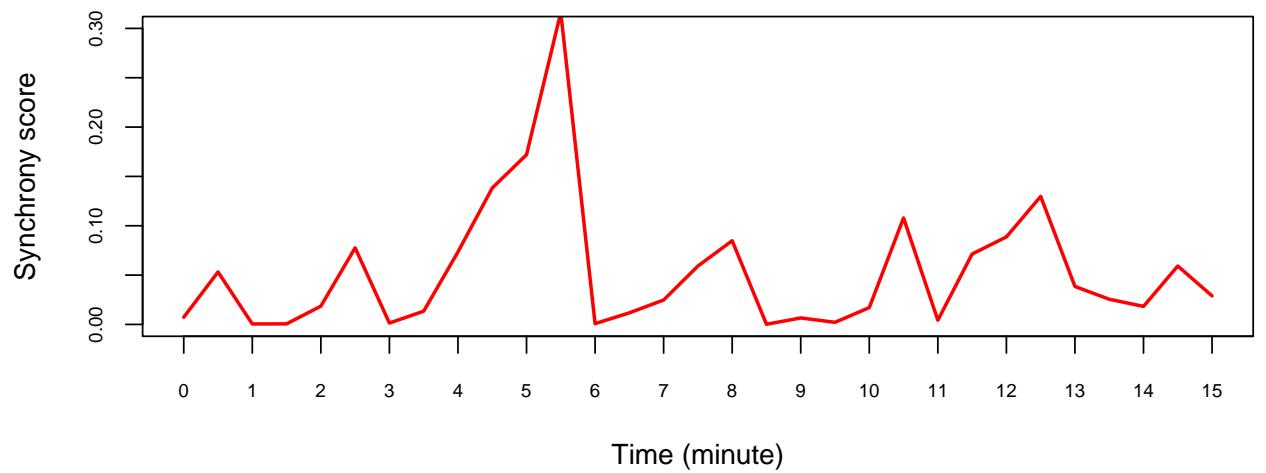
Synchrony scores in PELI020 family



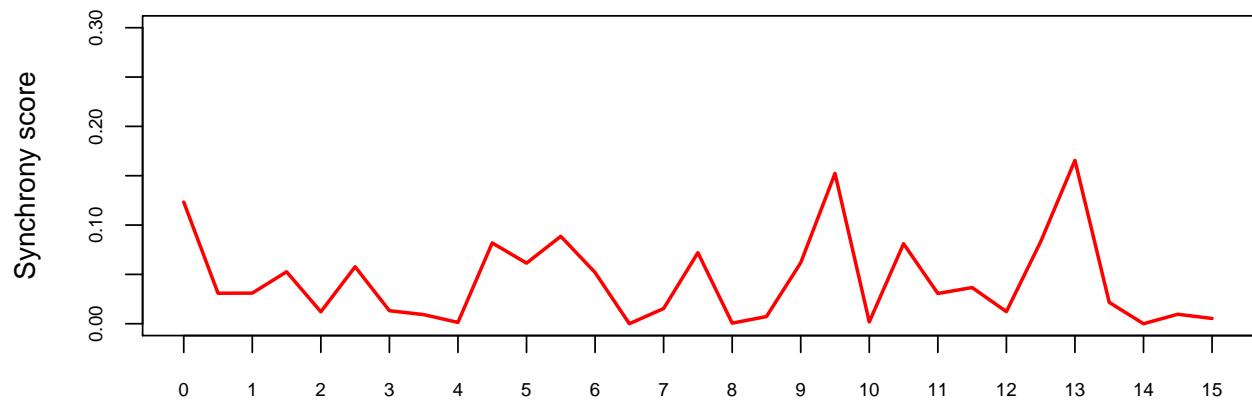
Synchrony scores in RAEM049 family



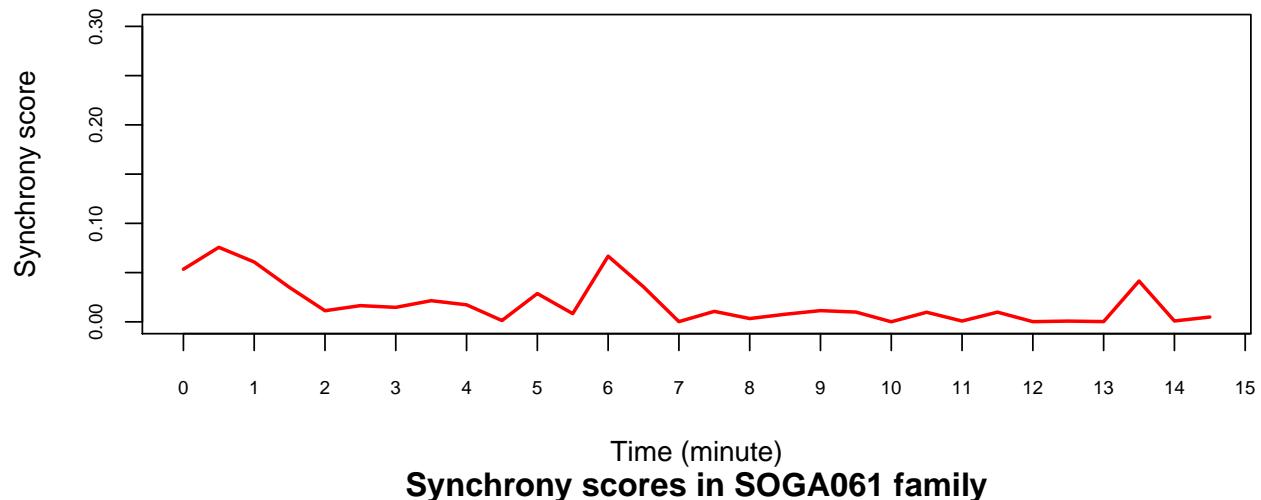
Synchrony scores in RAMA054 family



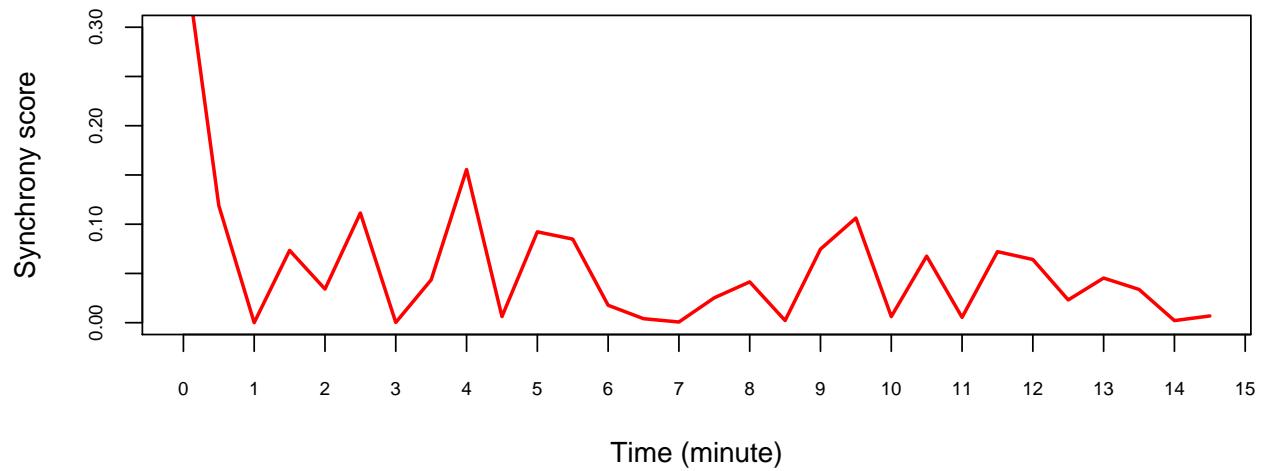
Synchrony scores in SEEM035 family



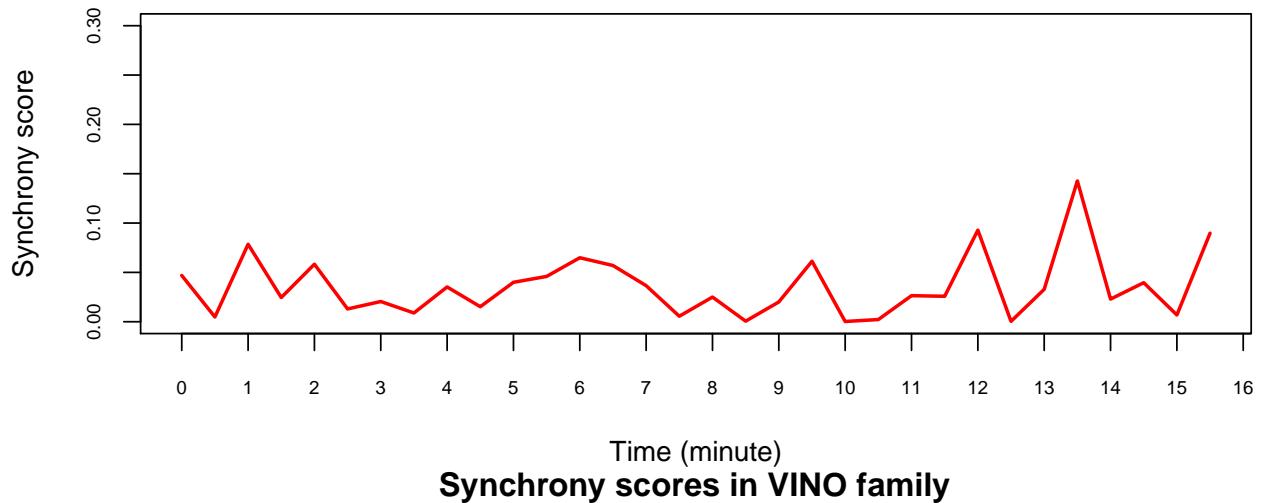
Synchrony scores in SHAN042 family



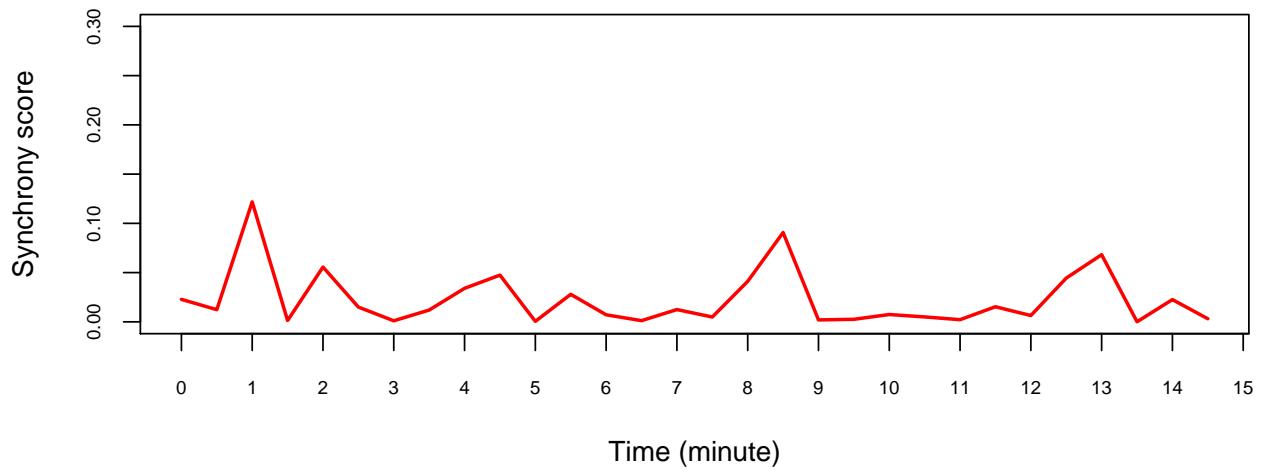
Synchrony scores in SOGA061 family



Synchrony scores in TIUG032 family



Synchrony scores in VINO family



```
str(SSInoLog)
```

```
## 'data.frame': 1054 obs. of 5 variables:
## $ family      : Factor w/ 35 levels "1606","BAJE059",...
## $ SSI-interval: int 0 1 2 3 4 5 6 7 8 9 ...
## $ Time_min    : num 0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 ...
## $ SSI_fa_ch   : num 0.06097 0.015151 0.008147 0.000207 0.023456 ...
## $ SSI_mo_ch   : num NA NA NA NA NA NA NA NA NA ...
SSInoLog <- merge(SSInoLog, cutFrames, by.x="family", by.y="family")
SSInoLog$LabelVideo <- rep(NA, nrow(SSInoLog))
SSInoLog[which(SSInoLog$Time_min < SSInoLog$CutBeforeMin),]$LabelVideo <- "Cut"

SSInoLog[which(SSInoLog$Time_min > SSInoLog$CutBeforeMin & SSInoLog$Time_min < SSInoLog$CutMiddle1Min),]
SSInoLog[which(SSInoLog$Time_min > SSInoLog$CutMiddle1Min & SSInoLog$Time_min < SSInoLog$CutMiddle2Min)]
SSInoLog[which(SSInoLog$Time_min > SSInoLog$CutMiddle2Min & SSInoLog$Time_min < SSInoLog$CutFinalMin),]
```

```

SSIConflict <- c(SSInoLog[which(SSInoLog$LabelVideo=="Conflict"),]$SSI_mo_ch, SSInoLog[which(SSInoLog$LabelVideo=="Conflict"),]$SSI_no_ch)
mean(SSIConflict, na.rm=TRUE)
## [1] 0.04153297

SSINoConflict <- c(SSInoLog[which(SSInoLog$LabelVideo=="No Conflict"),]$SSI_mo_ch, SSInoLog[which(SSInoLog$LabelVideo=="No Conflict"),]$SSI_no_ch)
mean(SSINoConflict, na.rm=TRUE)
## [1] 0.0406361

```

Demographic and Psychometric data

Demographic description

Sex

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

Age

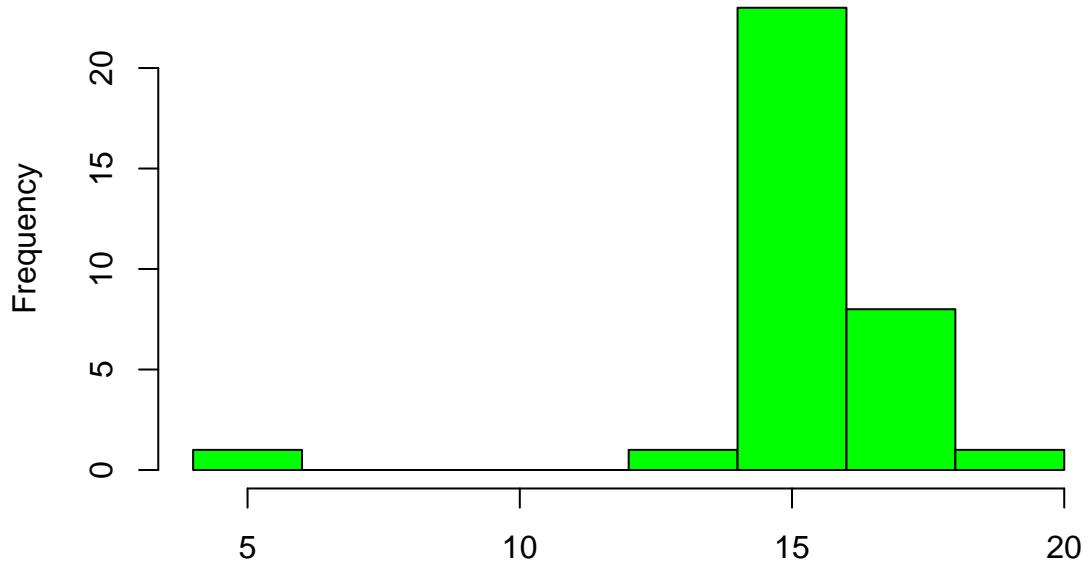
```
psycho$Birthday <- as.Date(psycho$Birthday, format="%d/%m/%y")
psycho$interview_date <- as.Date(psycho$interview_date, format="%d/%m/%y")
str(psycho$Birthday)

## Date[1:34], format: "1999-05-27" "1999-12-01" "2000-02-26" "1999-09-22" ...
str(psycho$interview_date)

## Date[1:34], format: "2015-03-26" "2015-03-27" "2015-03-18" "2014-12-03" ...
psycho$age <- (psycho$interview_date-psycho$Birthday)/365.25

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

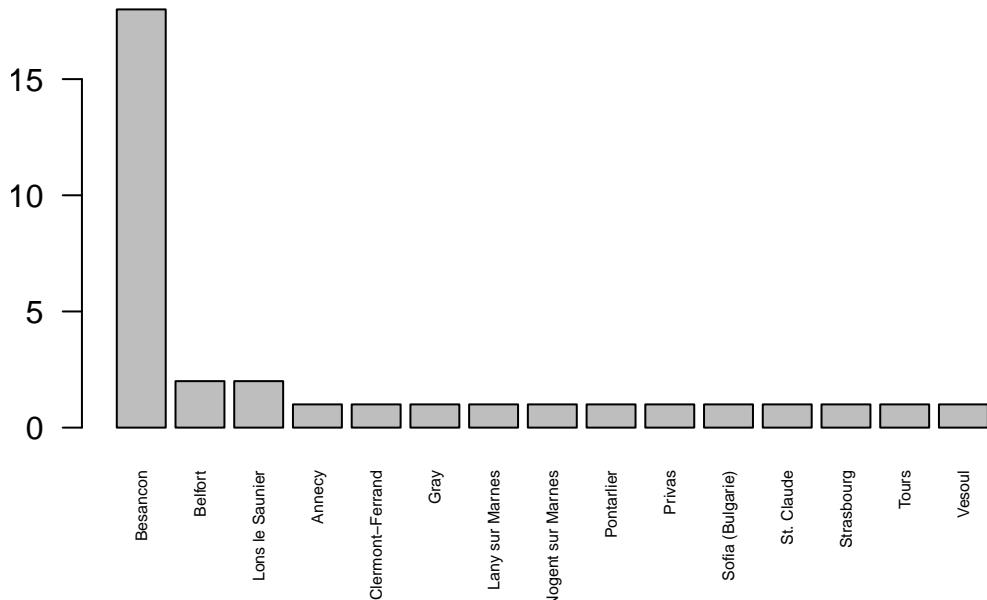
Histogram of as.numeric(psycho\$age)



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")
```

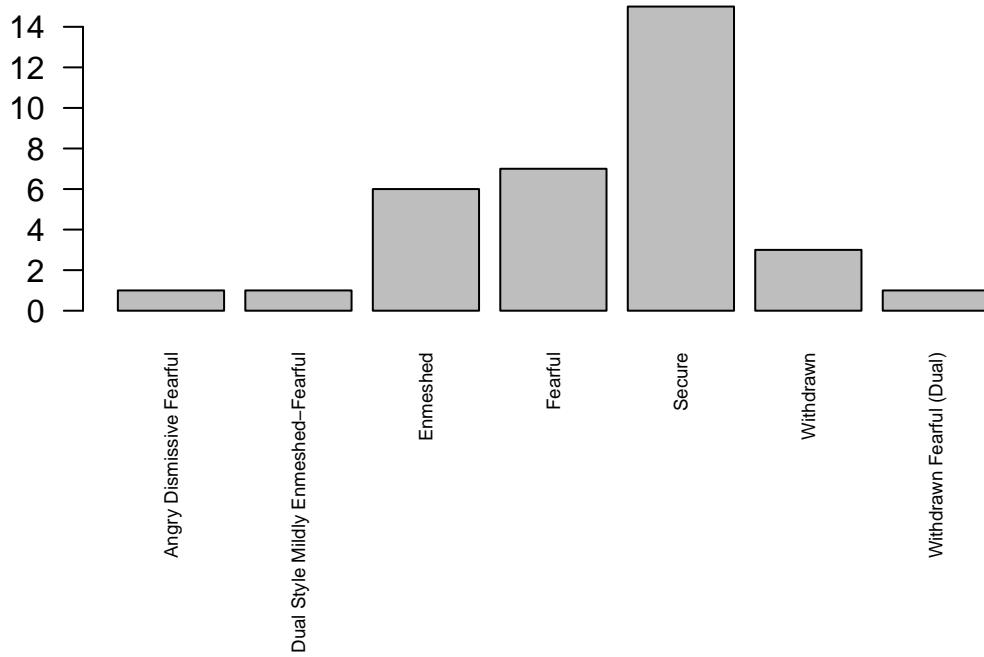
Birth place



Attachement styles

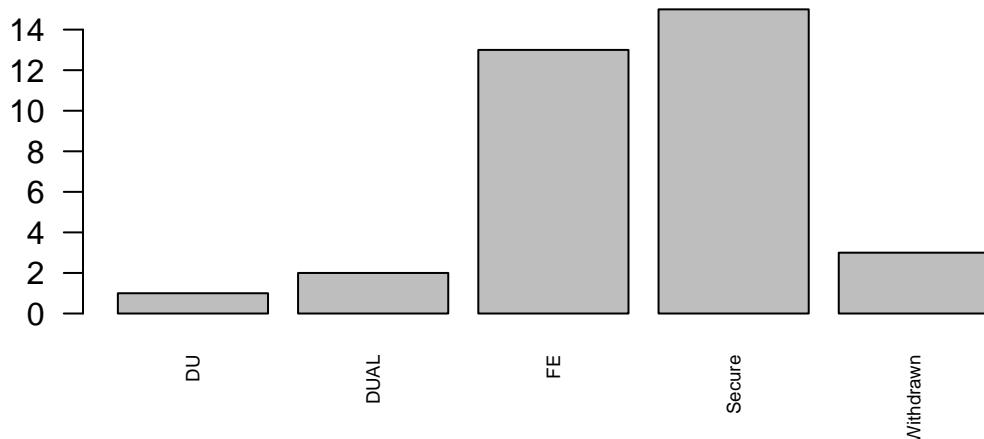
Raw

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



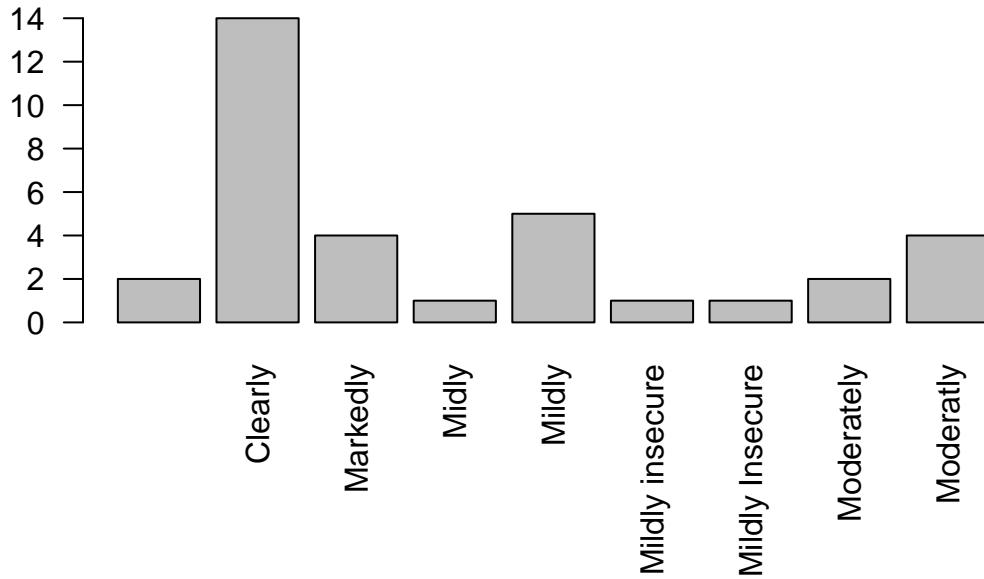
Clusters

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



Insecurity level

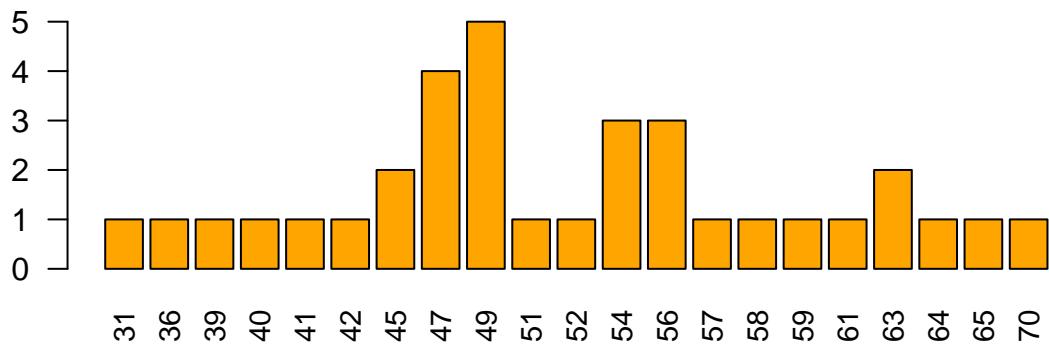
```
par(mar=c(9,5,3,3))
barplot(table(psycho$insecurite_level), las=2)
```



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)
```

Distribution of the TAS scores



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

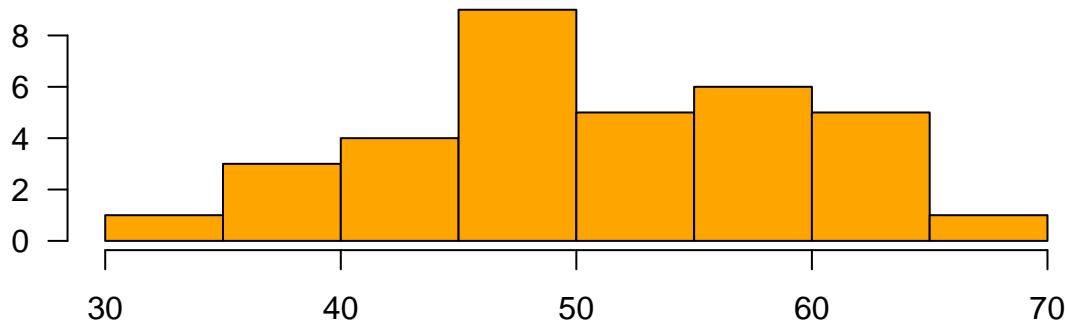
```
## Warning in plot.window(xlim, ylim, "", ...): "cex.name" is not a graphical
## parameter

## 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
```

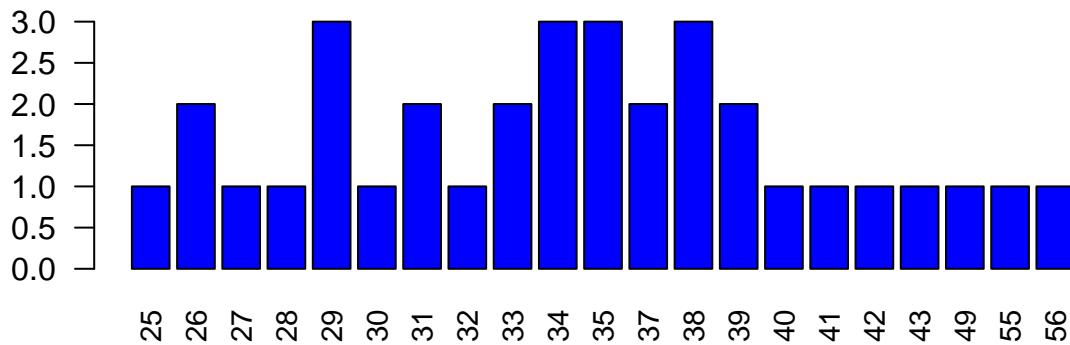
Distribution of the TAS scores



STAIYA

```
par(mar=c(3,3,3,3))
barplot(table(psycho$STAIYA_total), las=2, col="blue", main="Distribution of the STAIYA scores", cex.name=0.9)
```

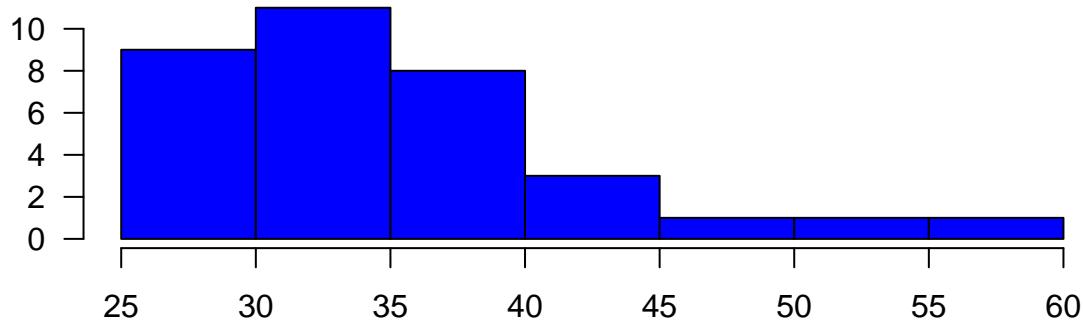
Distribution of the STAIYA scores



```
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
```

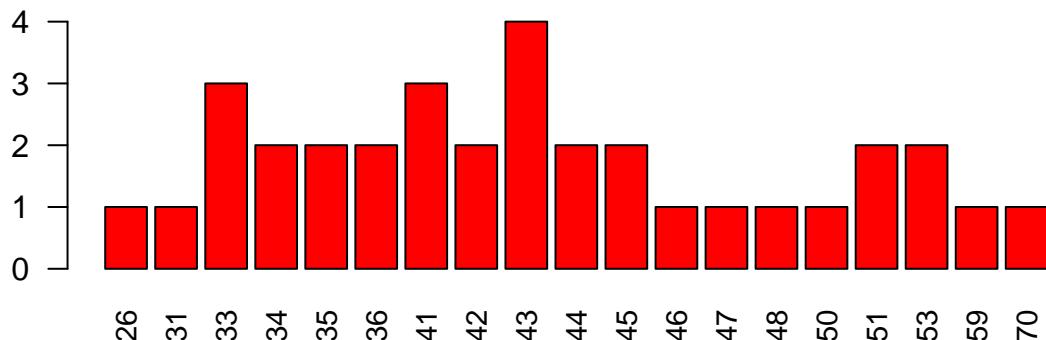
Distribution of the STAIYA scores



STAIYB

```
par(mar=c(3,3,3,3))
barplot(table(psycho$STAIYB_total), las=2, col="red", main="Distribution of the STAIYB scores", cex.name=0.9)
```

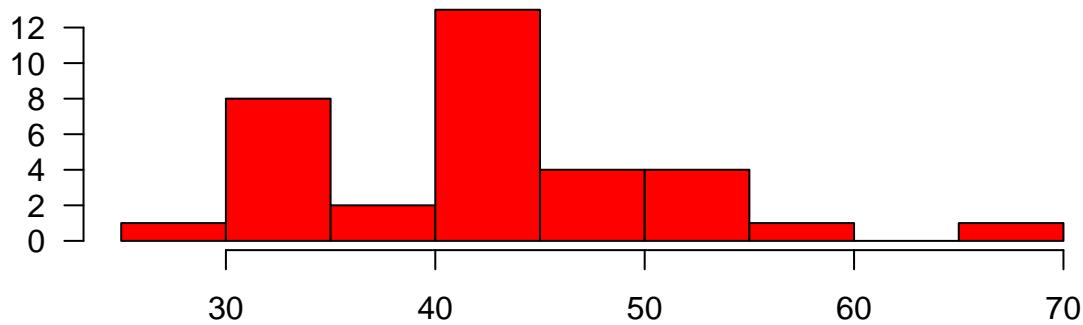
Distribution of the STAIYB scores



```
hist(psycho$STAIYB_total, las=1, col="red", main="Distribution of the STAIYB 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
```

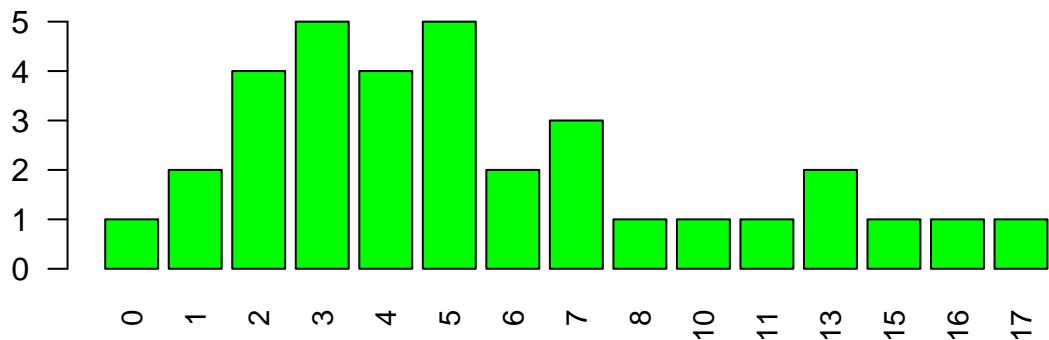
Distribution of the STAIYA scores



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=0.9)
```

Distribution of the BDI scores



```
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
```

Distribution of the BDI scores

