

Synchrony in Psychotherapy, example with F1044 patient data

Thomas Gargot

Mars, 24th 2016

Contents

Aim and Hypothesis	2
Nomenclature	2
Lists	2
Functions list	2
File lists	3
Participants list	4
Presentation of the data	4
Length of the videos in minutes	5
Length of the videos in number of frames	6
Number of Available (True) and Not Available (False) data for each participant	7
Global Motion history	8
Mean Motion history by video by participant	8
Raw data and mean of Motion History on sliding and non overlapping intervals on F1044C video	8
F1044C video	8
Sliding interval	9
Non overlapping interval	10
Focus on the motion history of the first 20 seconds of the first video(C)	10
Sliding interval function on a 5 frames interval	10
Motion history of the father during 10-20 seconds of the first video(C)	12
Mean motion history by minute plots	14
Export data in text files	24
SyncPy utilisation for creating synchrony dataframe	25
Description of SSI data frame	25

Synchrony scores for each dyad and for the whole group	26
Evolution of synchrony through time, raw each second	27
Evolution of synchrony through time, mean by minute	31
Evolution of synchrony through time, mean by 10 minutes	36

Aim and Hypothesis

The aim of this project is to evaluate if it is possible to detect automatic signals that could predict the outcomes of a familial psychotherapy.

Here is the analysis of the F1044 subject, his family (2 parents) and the therapist. Is there synchrony signals between himself, his parents and the therapist ? Is there synchrony signal between his parents and the therapist ?

Could this synchrony signal predict outcomes of the psychotherapy ?

We used this data since they come from a large european psychotherapy study and they come from almost real life practice of psychotherapy. This INCANT study aimed to evaluate the efficacy of the MultiDimensional Family Therapy for cannabis use disorders in adolescents.

Nomenclature

F1044 is the name of the subject studied (called pa for patient). He has a mother (mo) a father (fa) helped by a therapist (th). When a variable is referring to several participants, it is organised in alphabetical order separated by underscores, eg. SSI_fa_mo refers to the synchrony index (SSI) between the father and the mother. SSI_mo_fa doesn't exist. This family had several consultations with the psychotherapist. Some of them were video recorded. These videos are names with the name of the subject + an index letter. They can be subdivided after that with numbers (eg F1044C).

Lists

Functions list

MeanMotionByTime

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

Arguments:

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

```

MeanMotionByTime <- function(subject, indexOfvideos=1:NumberOfvideos, interval, data){
  x <- c()
  for (file in indexlist[indexOfvideos]){
    dataVector <- data[which(data$file==file), subject]
    ## with ceiling : superior limit of the round
    IntervalNumbersVideo <- ceiling(length(dataVector)/interval)
    for (i in 1:IntervalNumbersVideo){
      borneinf<- 1+(i-1)*interval
      bornesup <-i*interval
      dataVectorInterval <- dataVector[borneinf:bornesup]
      mean <- mean(dataVectorInterval, na.rm=TRUE)
      x <- c(x, mean)}}
  return (x)}

```

Slidinginterval

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

Arguments:

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

```

SlidingInterval <- function(subject, indexOfvideos=1:NumberOfvideos, interval, data)
{x <- c()
  for (file in indexlist[indexOfvideos]){
    dataVector <- data[which(data$file==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)}

```

MeanSynchronyByTime (TODO)

File lists

```

indexlist <- c("F1044C.VOB", "F1044D1.VOB", "F1044D2.VOB", "F1044E.VOB", "F1044F.VOB",
              "F1044G.VOB", "F1044H.VOB", "F1044I.VOB", "F1044L.VOB", "F1044M1.VOB",
              "F1044M2.VOB", "F1044N.VOB", "F1044O.VOB", "F1044P.VOB", "F1044Q.VOB",
              "F1044R1.VOB", "F1044R2.VOB")

labelvideolist<- c("C", "D1", "D2", "E", "F", "G", "H", "I", "L", "M1",

```

```

      "M2", "N", "O", "P", "Q", "R1", "R2")

NumberOfvideos <- length(indexlist)

colOrderList <- c("blue", "red", "green", "orange")

```

Participants list

```
## [1] "father"      "mother"      "patient"     "therapist"
```

Presentation of the data

```
str(data)
```

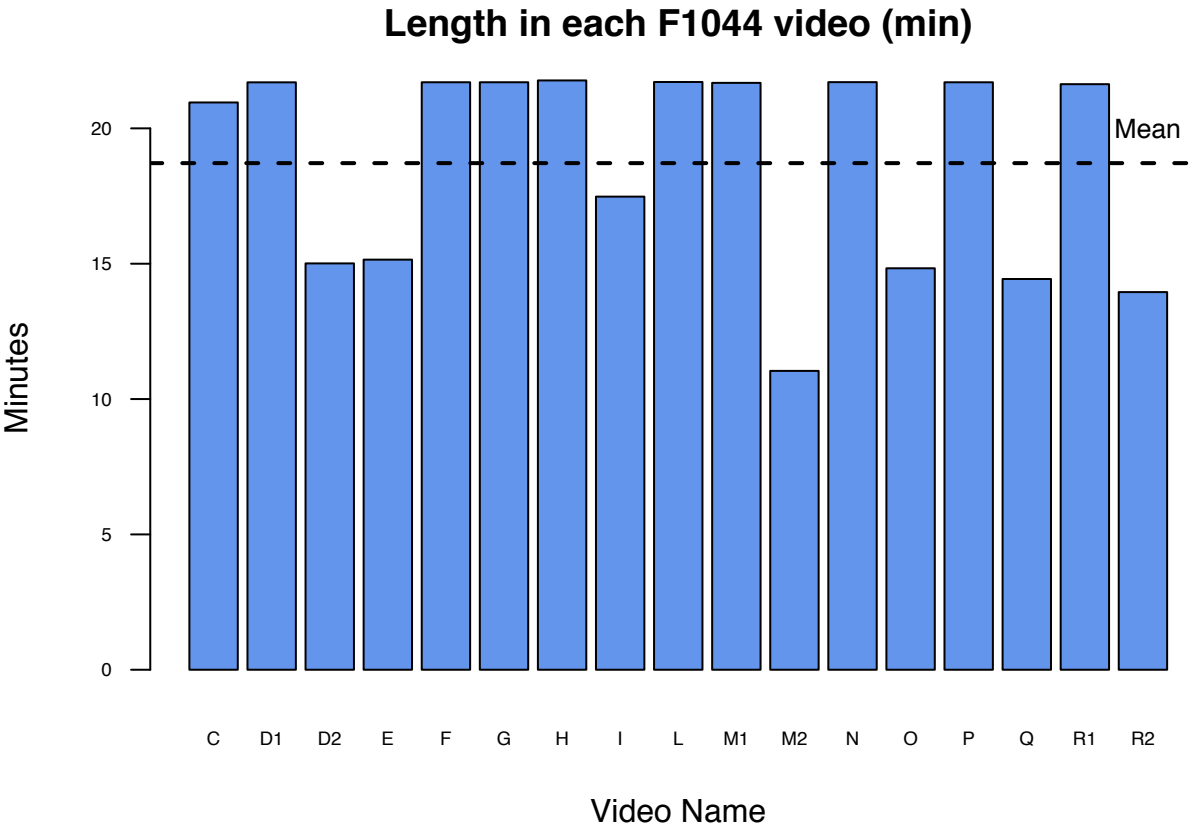
```
## 'data.frame':  477258 obs. of  7 variables:
## $ frame      : int  1 2 3 4 5 6 7 8 9 10 ...
## $ father     : num  0.01996 0.00915 0.01355 0.01787 0.01758 ...
## $ mother     : num  1.82e-05 1.82e-05 3.64e-05 1.82e-05 9.09e-05 ...
## $ patient    : num  NA NA NA NA NA NA NA NA NA NA ...
## $ therapist  : num  0.00162 0.00506 0.00349 0.00223 0.00249 ...
## $ file       : Factor w/ 17 levels "F1044C.VOB","F1044D1.VOB",...: 1 1 1 1 1 1 1 1 1 1 ...
## $ timeMin    : num  0.000667 0.001333 0.002 0.002667 0.003333 ...
```

```
summary(data)
```

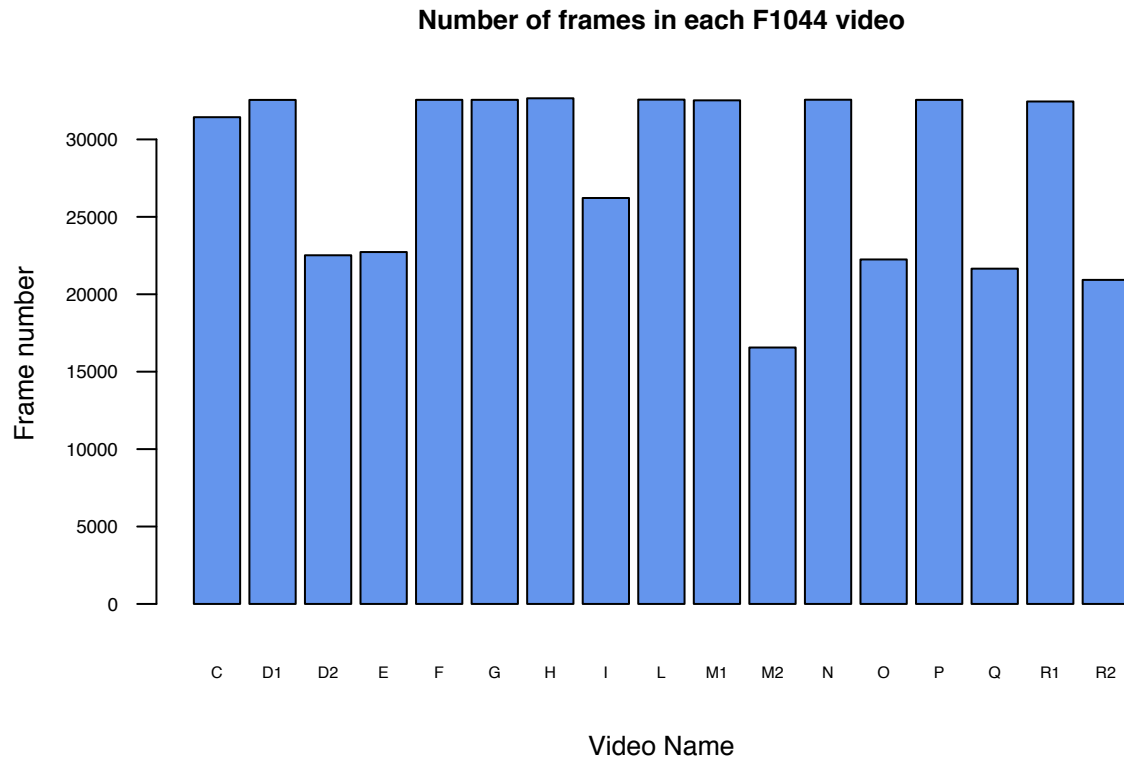
```
##      frame      father      mother      patient
## Min.   :    1  Min.   :0.00  Min.   :0.00  Min.   :0.00
## 1st Qu.: 7019  1st Qu.:0.00  1st Qu.:0.00  1st Qu.:0.00
## Median :14038  Median :0.00  Median :0.00  Median :0.00
## Mean   :14576  Mean   :0.00  Mean   :0.00  Mean   :0.01
## 3rd Qu.:21364  3rd Qu.:0.00  3rd Qu.:0.00  3rd Qu.:0.01
## Max.   :32656  Max.   :0.19  Max.   :0.49  Max.   :0.54
##      NA's      :265686  NA's   :91545  NA's   :189317
##      therapist      file      timeMin
## Min.   :0.0  F1044H.VOB: 32656  Min.   : 0.000667
## 1st Qu.:0.0  F1044L.VOB: 32570  1st Qu.: 4.679333
## Median :0.0  F1044N.VOB: 32562  Median : 9.358333
## Mean   :0.0  F1044G.VOB: 32556  Mean   : 9.717052
## 3rd Qu.:0.0  F1044F.VOB: 32555  3rd Qu.:14.242667
## Max.   :0.8  F1044P.VOB: 32554  Max.   :21.770667
## NA's    :77972  (Other)   :281805
```

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

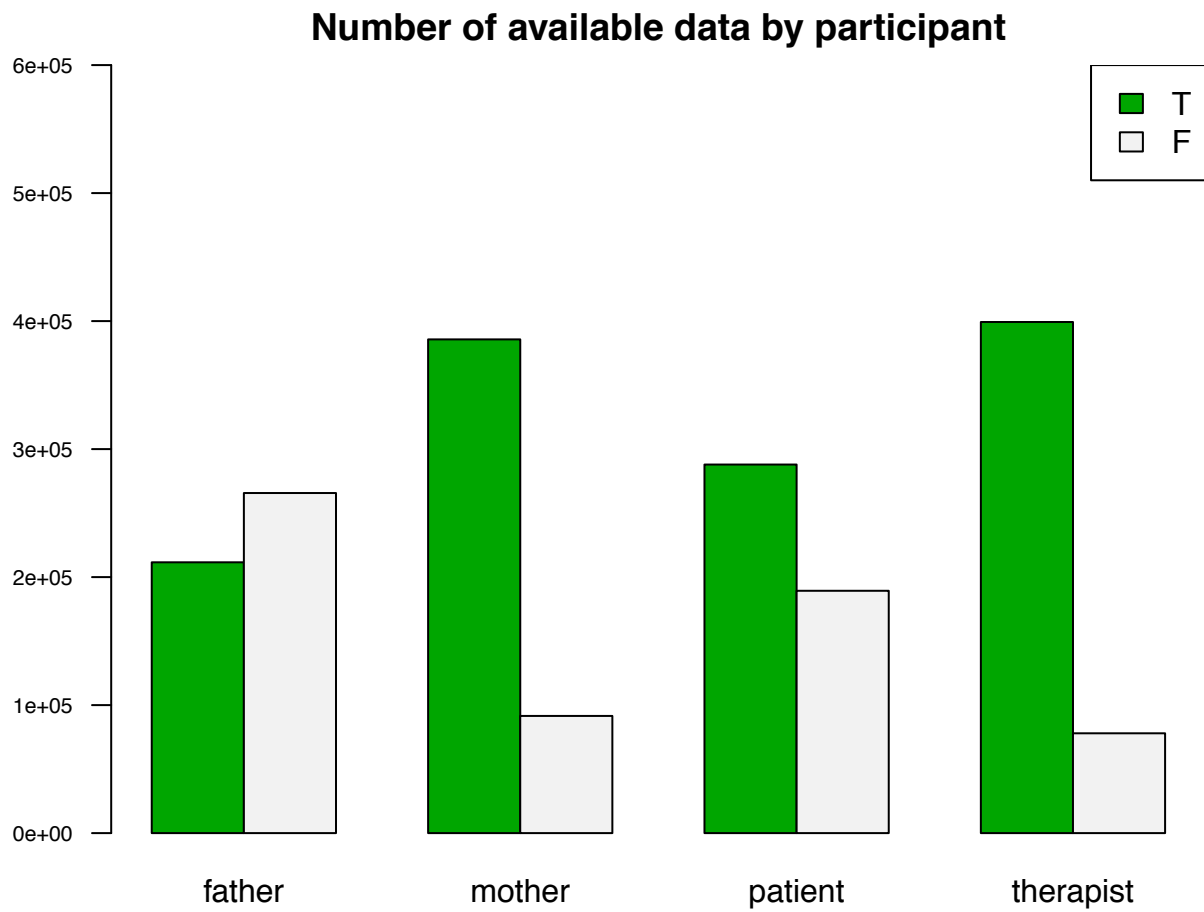
Length of the videos in minutes



Length of the videos in number of frames



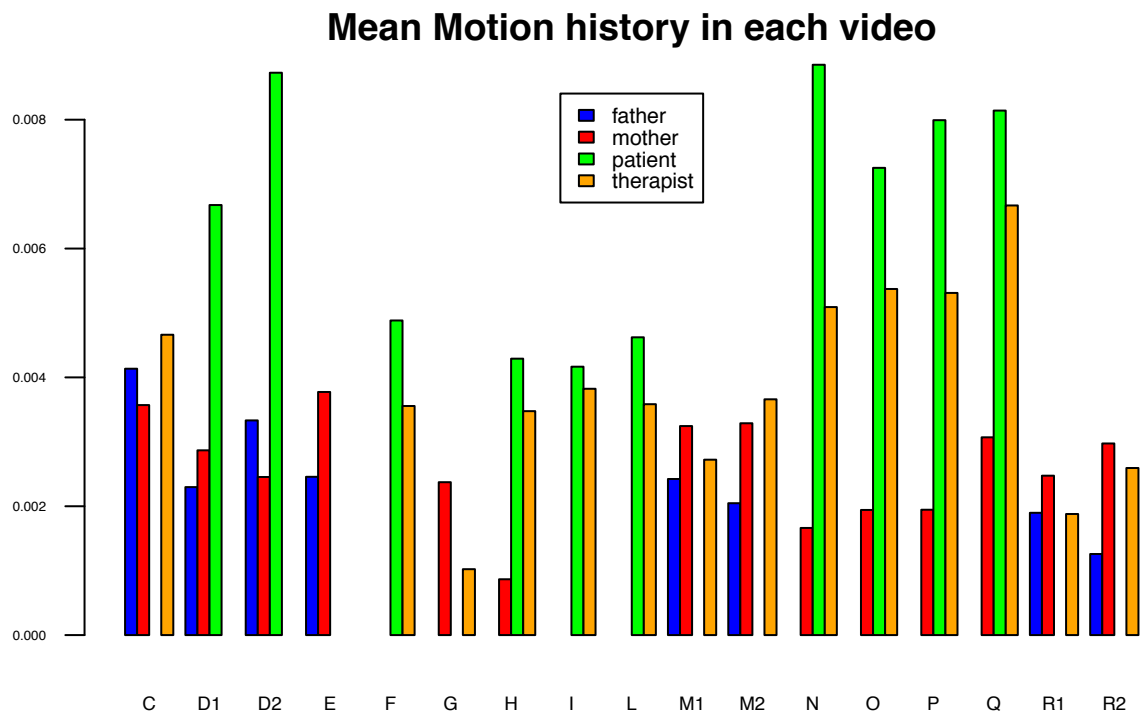
Number of Available (True) and Not Available (False) data for each participant



Some participants are not filmed (eg therapist) or don't come in the sessions. Mother and therapist are the more often present participants.

Global Motion history

Mean Motion history by video by participant



Raw data and mean of Motion History on sliding and non overlapping intervals on F1044C video

F1044C video

It is the first video of F10044C. The father, mother and therapist are present. The patient is absent. ##
Raw data

```
rawdatafather <- data[which(data$file=="F1044C.VOB"),]$father
rawdataMother <- data[which(data$file=="F1044C.VOB"),]$mother
rawdataTherapist <- data[which(data$file=="F1044C.VOB"),]$therapist

summary(rawdatafather)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.   NA's
## 0.000000 0.000000 0.000196 0.004135 0.003488 0.092340     10
```

```
summary(rawdataMother)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.   NA's
## 0.000000 0.000036 0.000127 0.003570 0.002200 0.159600     10
```



```
summary(rawdataTherapist)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.   NA's  
## 0.001179 0.002750 0.003405 0.004662 0.004234 0.236000    10
```

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
```

```
slidedfather <- SlidingInterval("father", 1 , 5, data)  
slidedmother <- SlidingInterval("mother", 1 , 5, data)  
slidedtherapist <- SlidingInterval("therapist", 1 , 5, data)  
slidedpatient <- SlidingInterval("patient", 1 , 5, data)
```

```
summary(slidedfather)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.   NA's  
## 0.000000 0.000005 0.000335 0.004139 0.003691 0.091450     6
```

```
summary(slidedmother)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.   NA's  
## 0.000000 0.000036 0.000156 0.003574 0.002520 0.145900     6
```

```
summary(slidedpatient)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.   NA's  
##      NA      NA      NA      NaN      NA      NA    31431
```

```
summary(slidedtherapist)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.   NA's  
## 0.001702 0.003029 0.003352 0.004672 0.003955 0.219600     6
```

```
par(mar=c(3,3,2,2))  
boxplot(slidedfather, slidedmother, slidedtherapist,  
        col=colOrderList[c(1,2,4)],  
        names=ParticipantsList[c(1,2,4)],  
        main= "Box plot of motion history sliding interval on F1044C video", las=1)  
par(mar=c(1,0.5,0.5,1))  
legend("topleft", ParticipantsList[c(1,2,4)], fill=colOrderList, cex=0.7)
```

Non overlapping interval

```
fatherFive<- MeanMotionByTime("father", indexOfvideos=1, interval=5, data)
motherFive <- MeanMotionByTime("mother", indexOfvideos=1, interval=5, data)
therapistFive <- MeanMotionByTime("therapist", indexOfvideos=1, interval=5, data)
summary(fatherFive)
```

```
##      Min.   1st Qu.   Median     Mean   3rd Qu.     Max.      NA's
## 0.0000000 0.0000049 0.0003305 0.0041350 0.0036840 0.0883000      2
```

```
summary(motherFive)
```

```
##      Min.   1st Qu.   Median     Mean   3rd Qu.     Max.      NA's
## 0.0000000 0.0000364 0.0001564 0.0035700 0.0025310 0.1459000      2
```

```
summary(therapistFive)
```

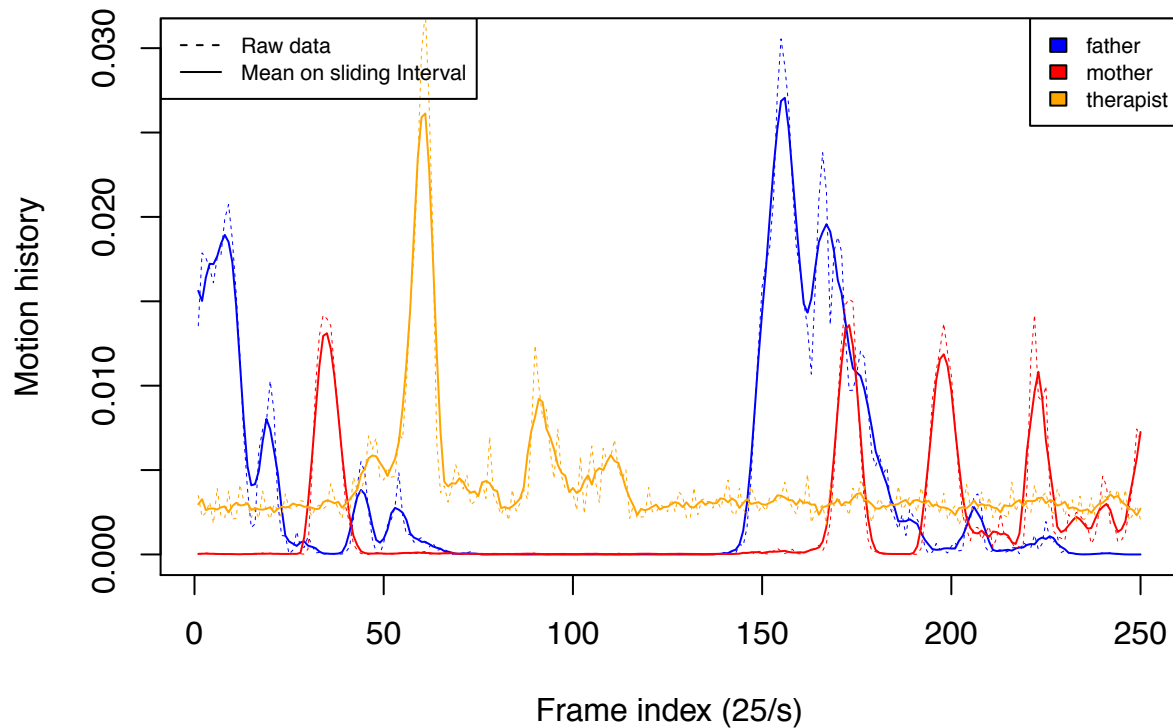
```
##      Min.   1st Qu.   Median     Mean   3rd Qu.     Max.      NA's
## 0.001807 0.003029 0.003352 0.004662 0.003972 0.219600      2
```

Focus on the motion history of the first 20 seconds of the first video(C)

Sliding interval function on a 5 frames interval

```
par(mar=c(4,4,4,2))
plot(1:250, data$father[3:252], main="Mean motion history (Sliding 5 frames interval)
     for father on F1044C video, 10 seconds ", xlab="Frame index (25/s)",
     ylab="Motion history",
     col="blue", type="l", lty=2, lwd=0.5)
lines(slidedfather[1:250], col="blue", lty=1)
lines(data$mother[3:252], col="red", lty=2, lwd=0.5)
lines(slidedmother[1:250], col="red", lty=1)
lines(data$therapist[3:252], col="orange", lty=2, lwd=0.5)
lines(slidedtherapist[1:250], col="orange", lty=1)
legend("topleft", c("Raw data", "Mean on sliding Interval") , lty=c(2, 1), cex=0.7)
legend("topright", ParticipantsList[c(1,2,4)], fill=colOrderList[c(1,2,4)], cex=0.7)
```

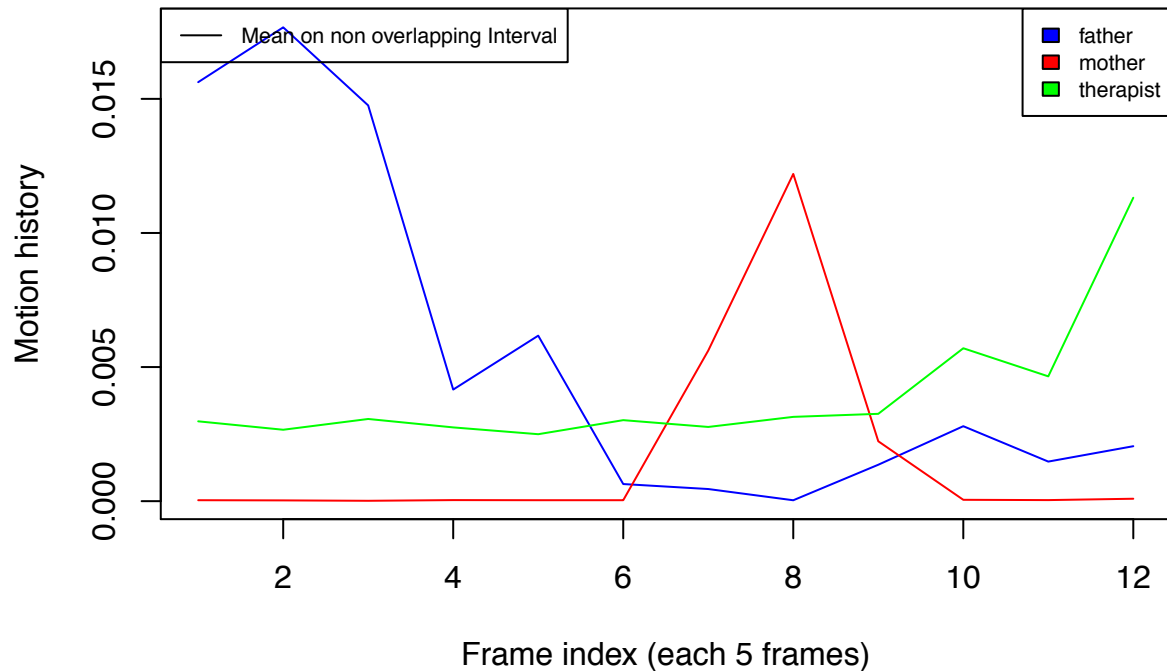
Mean motion history (Sliding 5 frames interval) for father on F1044C video, 10 seconds



Non overlapping interval function on a 5 frames interval

```
plot (1:12, fatherFive[1:12], type="l", col="blue",
main="Mean Motion history (non overlapping 5 frames
      intervals) for father on F1044C video, first 10 seconds",
ylab="Motion history", xlab="Frame index (each 5 frames)" )
lines(motherFive[1:12], col="red", lty=1)
lines(therapistFive[1:12], col="green", lty=1)
legend("topleft", "Mean on non overlapping Interval" , lty=1, cex=0.7)
legend("topright", ParticipantsList[c(1,2,4)], fill=colOrderList, cex=0.7)
```

Mean Motion history (non overlapping 5 frames intervals) for father on F1044C video, first 10 seconds



Non overlapping interval function on a 5 frames interval with shifting of therapist (substraction of min value of therapist)

```
plot(1:24, fatherFive[1:12], type="l", col="blue",
     main="Mean motion history (non overlapping 5 frames
           intervals) for father on F1044C video, first 10 seconds, data therapist shifted",
     ylab="Motion history", xlab="Frame index (each 5 frames)" )
lines(motherFive[1:12], col="red", lty=1)
lines(therapistFive[1:12]-min(slidedtherapist), col="green", lty=1)
legend("topleft", "Mean on non overlapping Interval", lty=1, cex=0.7)
legend("topright", ParticipantsList[c(1,2,4)], fill=colOrderList, cex=0.7)
```

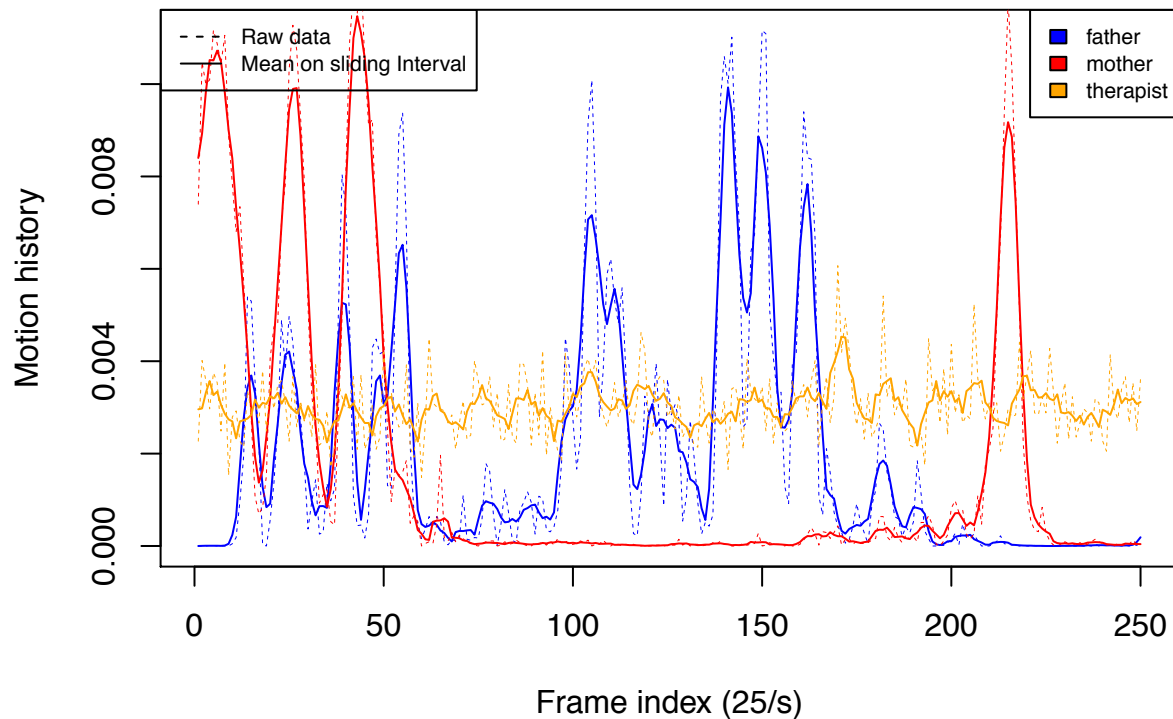
Motion history of the father during 10-20 seconds of the first video(C)

Non overlapping interval function on a 5 frames interval

```
par(mar=c(4,4,4,2))
plot(1:250, data$father[253:502], main="Mean motion history (Sliding 5 frames
     interval) for father on F1044C video, 10-20 seconds", xlab="Frame index (25/s)",
     ylab="Motion history", col="blue", type="l", lty=2, lwd=0.5)
lines(slidedfather[251:500], col="blue", lty=1)
lines(data$mother[253:502], col="red", lty=2, lwd=0.5)
lines(slidedmother[251:500], col="red", lty=1)
lines(data$therapist[253:502], col="orange", lty=2, lwd=0.5)
```

```
lines(slidedtherapist[251:500], col="orange", lty=1)
legend("topleft", c("Raw data", "Mean on sliding Interval"), lty=c(2, 1), cex=0.7)
legend("topright", ParticipantsList[c(1,2,4)], fill=colOrderList[c(1,2,4)], cex=0.7)
```

Mean motion history (Sliding 5 frames interval) for father on F1044C video, 10–20 seconds



```
### Non overlapping interval function on a 5 frames interval with shifting
### of therapist (substraction of min value of therapist)
par(mar=c(4,4,4,2))
plot(1:250, data$father[256:505], main="Mean motion history (Sliding 5 frames
interval) for father on F1044C video, 10-20 seconds,
data therapist shifted", xlab="Frame index (25/s)", ylab="Motion history", col="blue",
type="l", lty=2, lwd=0.5)
lines(slidedfather[251:500], col="blue", lty=1)
lines(data$mother[256:505], col="red", lty=2, lwd=0.5)
lines(slidedmother[251:500], col="red", lty=1)
lines(data$therapist[256:505], col="orange", lty=2, lwd=0.5)
lines(slidedtherapist[251:500]-min(slidedtherapist), col="orange", lty=1)
legend("topleft", c("Raw data", "Mean on sliding Interval"), lty=c(2, 1), cex=0.7)
legend("topright", ParticipantsList[c(1,2,4)], fill=colOrderList[c(1,2,4)], cex=0.7)
```

Non overlapping interval function on a 5 frames interval

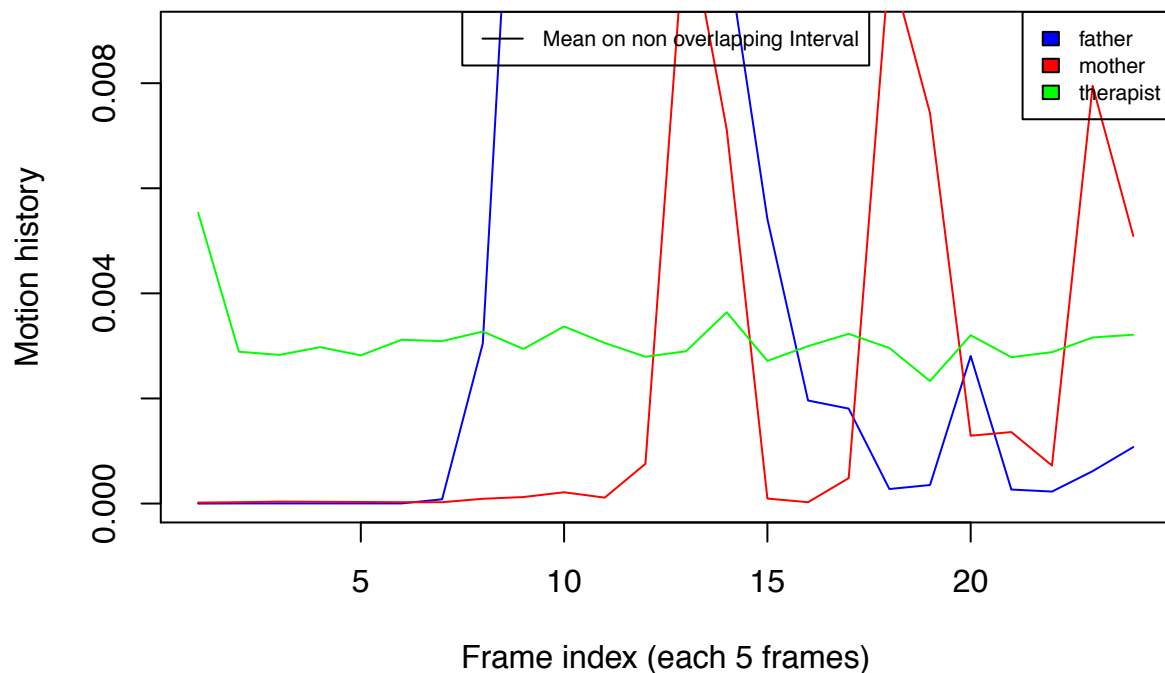
```
plot(1:24, fatherFive[23:46], type="l", col="blue",
main="Mean motion history (non overlapping 5 frames intervals) for
father on F1044C video, between 10-20 seconds",
```

```

ylab="Motion history", xlab="Frame index (each 5 frames)", ylim=c(0, 0.009))
lines(motherFive[23:46], col="red", lty=1)
lines(therapistFive[23:46], col="green", lty=1)
legend("top", "Mean on non overlapping Interval" , lty=1, cex=0.7)
legend("topright", ParticipantsList[c(1,2,4)], fill=colOrderList, cex=0.7)

```

Mean motion history (non overlapping 5 frames intervals) for father on F1044C video, between 10–20 seconds



```

### Non overlapping interval function on a 5 frames interval with shifting of therapist (subtraction o
plot(1:24, fatherFive[23:46], type="l", col="blue",
main="Mean motion history (non overlapping 5 frames intervals) for
father on F1044C video, between 10-20 seconds,
data therapist shifted",
ylab="Motion history", xlab="Frame index (each 5 frames)", ylim=c(0, 0.009))
lines(motherFive[23:46], col="red", lty=1)
lines(therapistFive[23:46]-min(slidedtherapist), col="green", lty=1)
legend("top", "Mean on non overlapping Interval" , lty=1, cex=0.7)
legend("topright", ParticipantsList[c(1,2,4)], fill=colOrderList, cex=0.7)

```

Mean motion history by minute plots

```

for (i in 1:NumberOfvideos){
  fatherMinute<- MeanMotionByTime("father", indexOfvideos=i, interval=1500, data)

  MotherMinute<- MeanMotionByTime("mother", indexOfvideos=i, interval=1500, data)

```

```

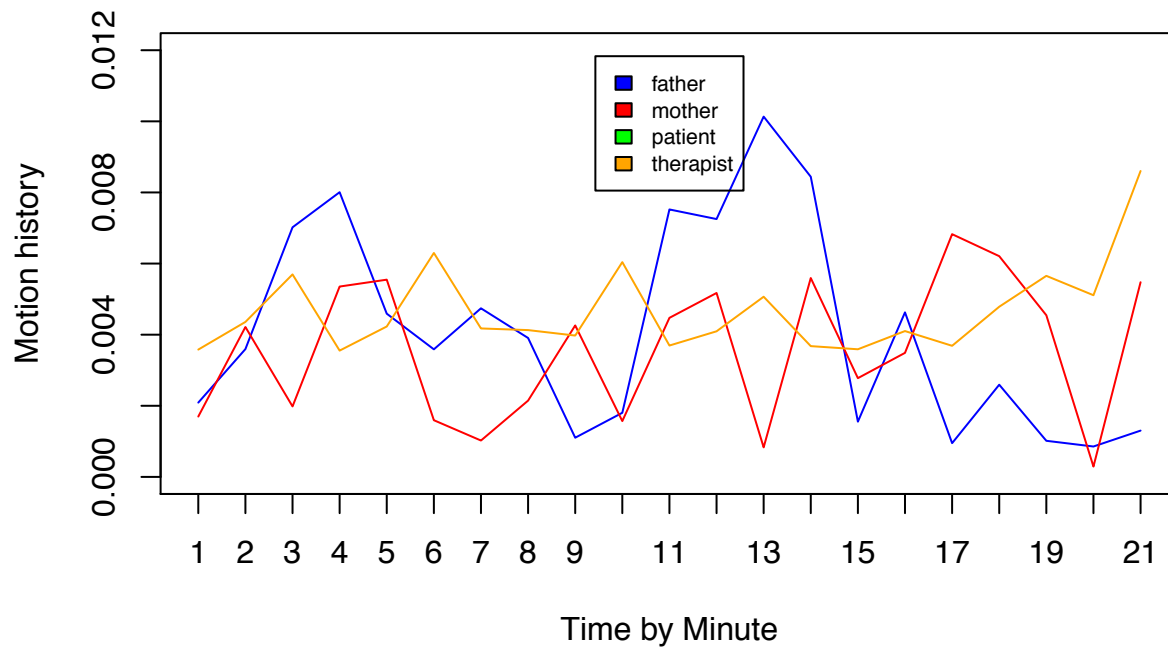
TherapistMinute<- MeanMotionByTime("therapist", indexOfvideos=i, interval=1500, data)

PatientMinute<- MeanMotionByTime("patient", indexOfvideos=i, interval=1500, data)

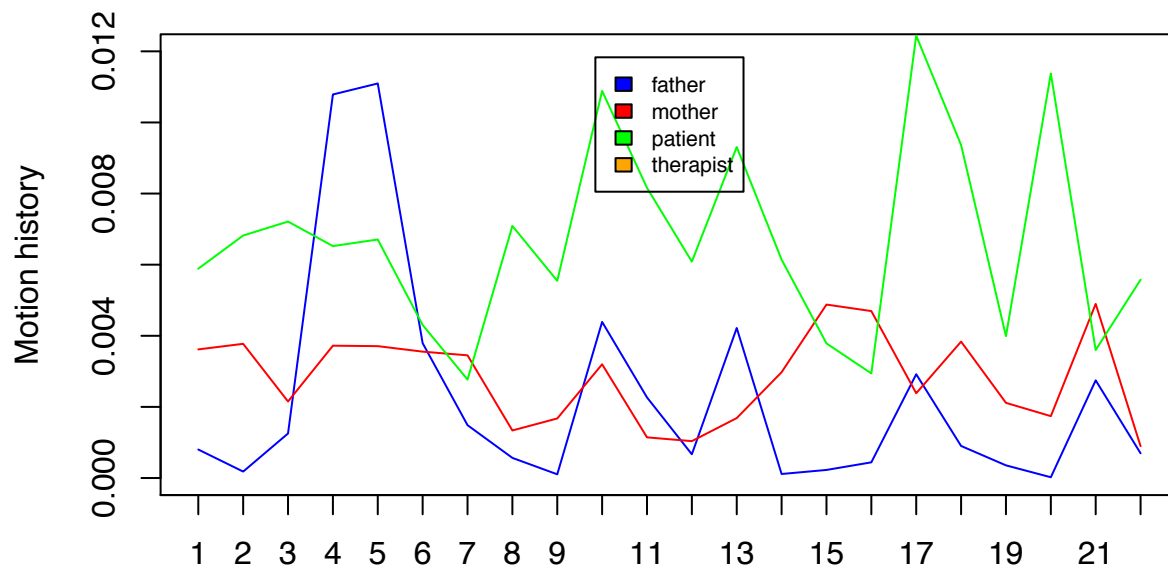
par(mar=c(4,4,4,2))
plot (1:length(fatherMinute), fatherMinute, type="l", col="blue",
main=paste("Mean motion history (non overlapping minute intervals)
on F1044", labelvideolist[i], " video" , sep=""),
ylab="Motion history", xlab="Time by Minute", ylim=c(0, 12E-03),
xaxp=c(0, length(fatherMinute), length(fatherMinute)))
lines(MotherMinute, col="red")
lines(TherapistMinute, col="orange")
lines(PatientMinute, col="green")
legend("top", inset=.05, ParticipantsList,
fill=colOrderList, cex=0.7)}

```

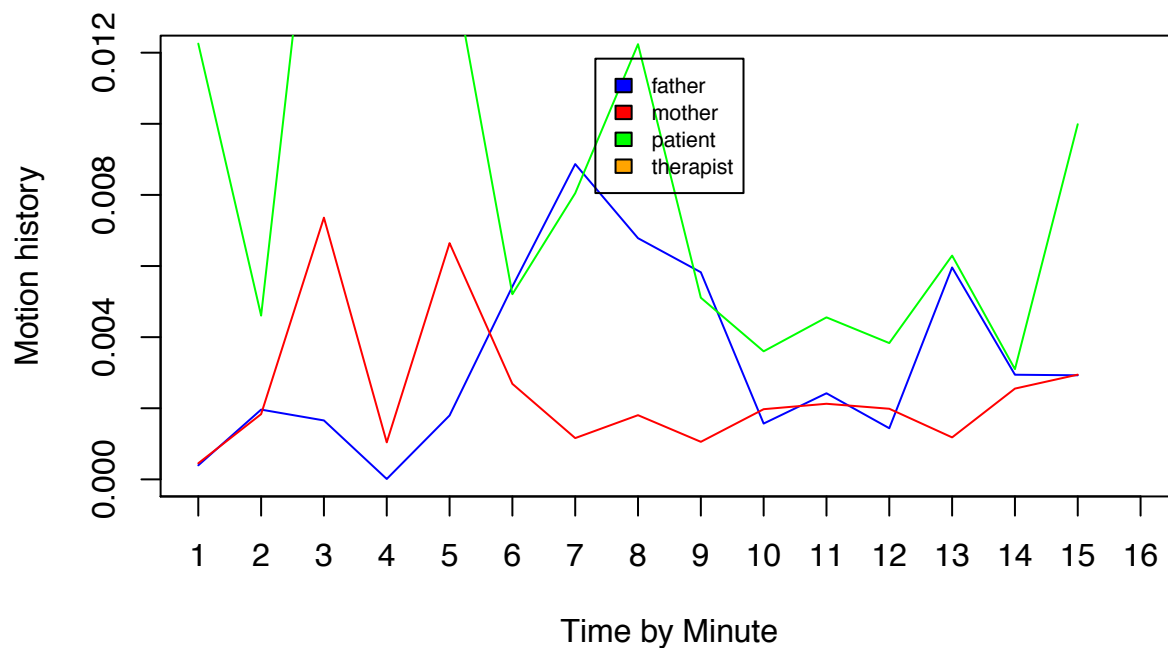
Mean motion history (non overlapping minute intervals) on F1044C video



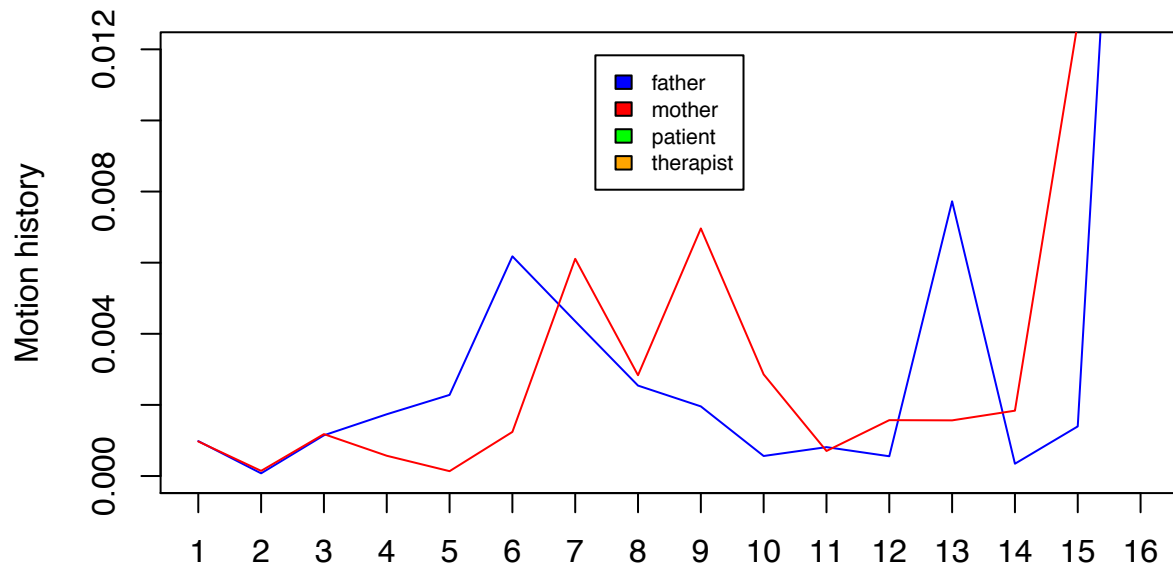
**Mean motion history (non overlapping minute intervals)
on F1044D1 video**



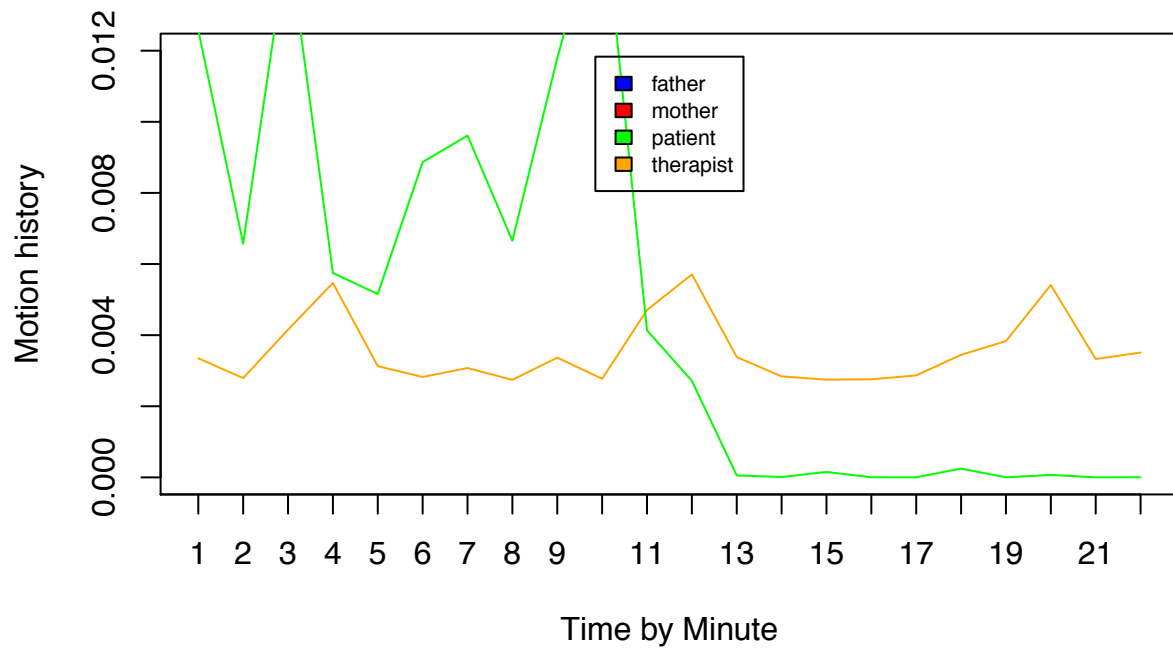
**Mean motion history (non overlapping minute intervals)
on F1044D2 video**



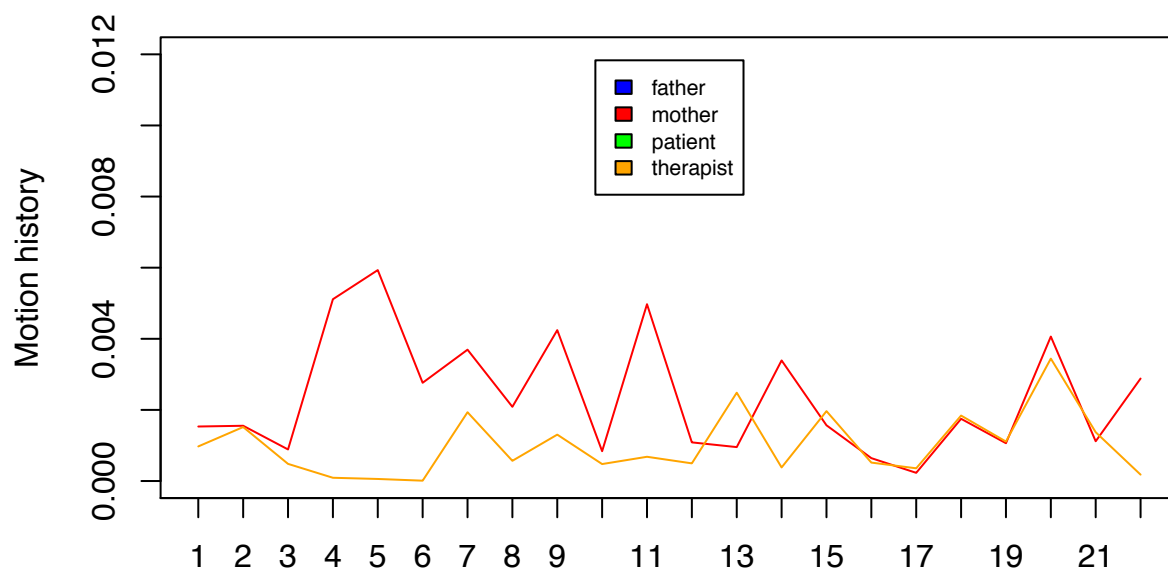
**Mean motion history (non overlapping minute intervals)
on F1044E video**



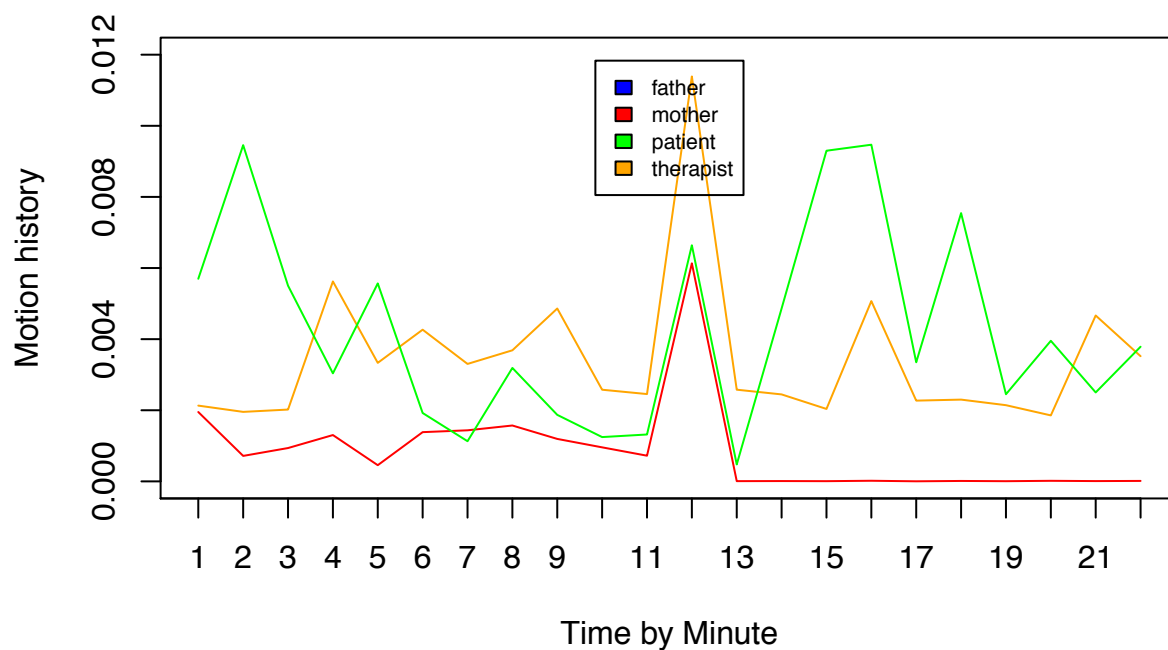
**Mean motion history (non overlapping minute intervals)
on F1044F video**



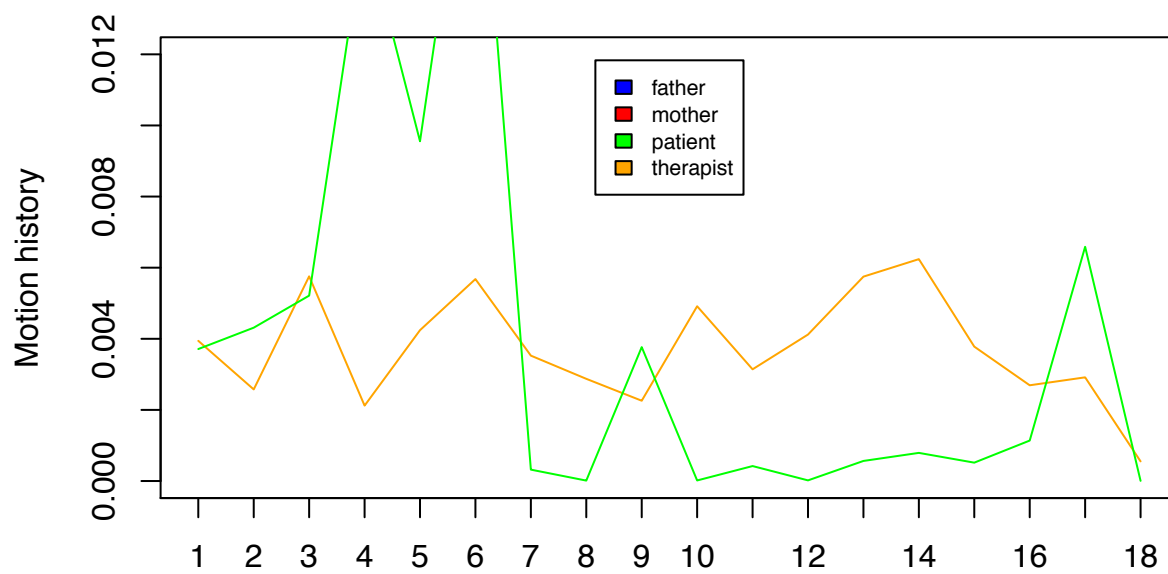
**Mean motion history (non overlapping minute intervals)
on F1044G video**



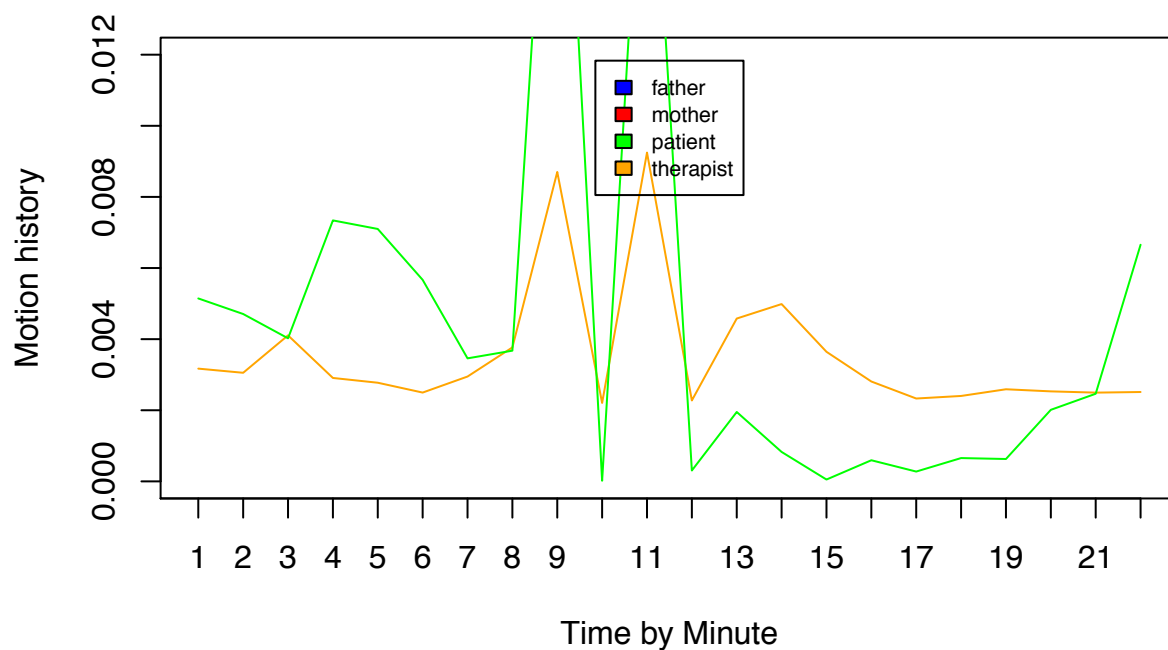
**Mean motion history (non overlapping minute intervals)
on F1044H video**



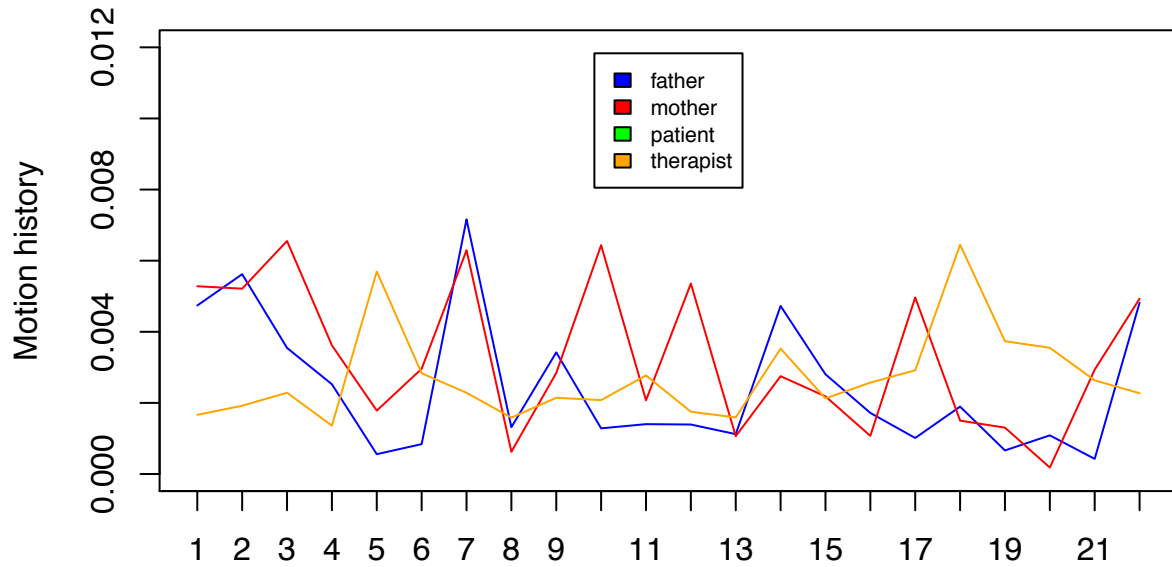
**Mean motion history (non overlapping minute intervals)
on F1044I video**



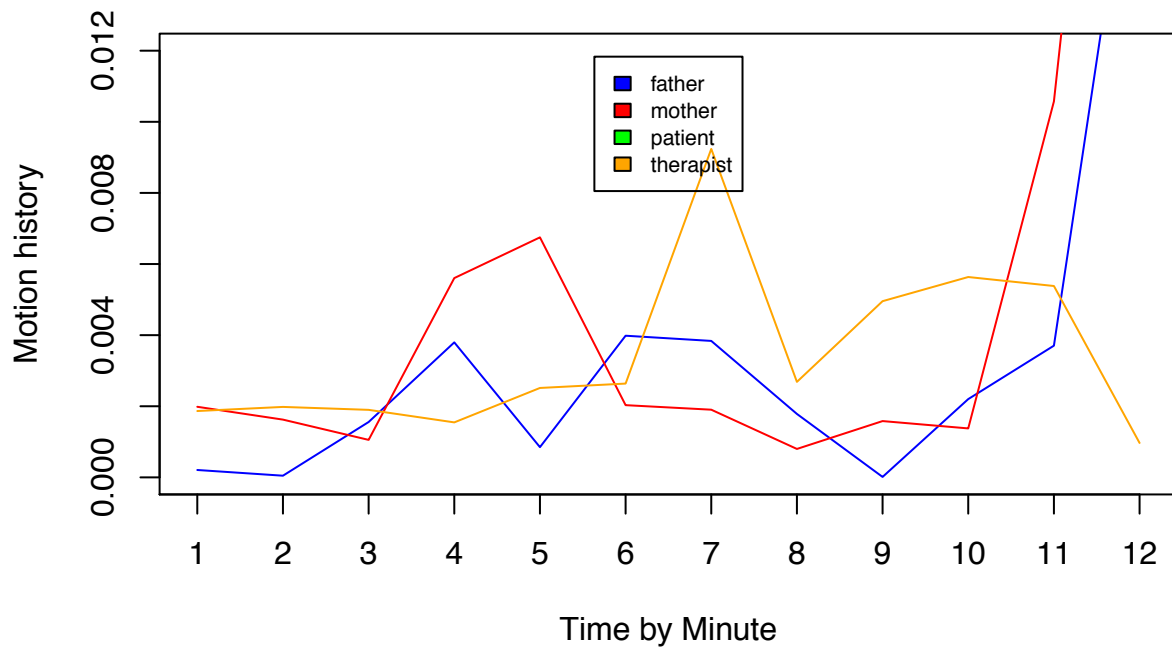
Time by Minute
**Mean motion history (non overlapping minute intervals)
on F1044L video**



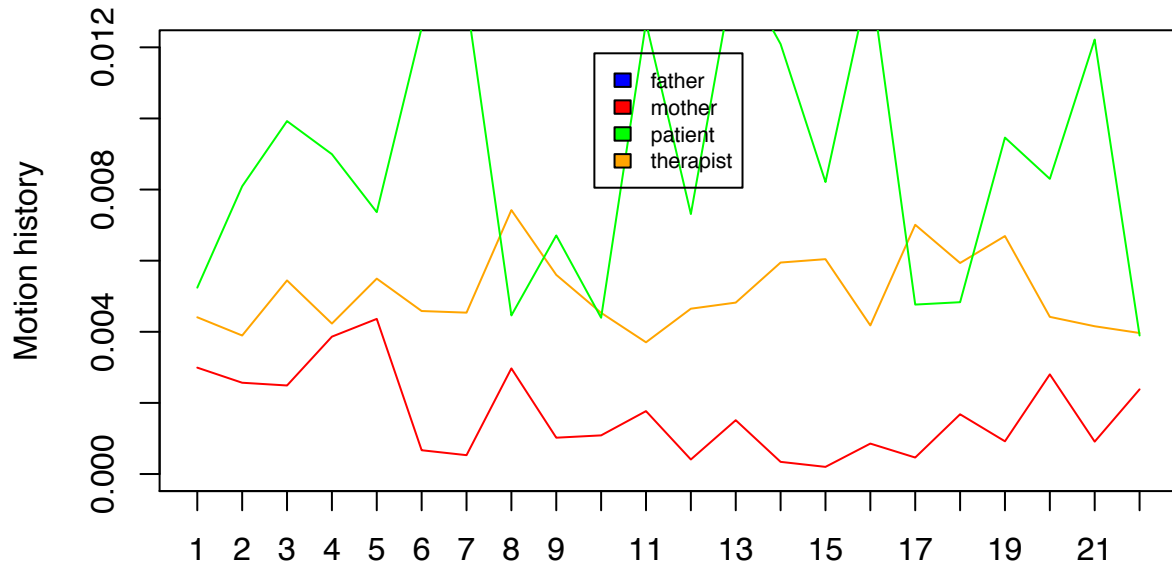
**Mean motion history (non overlapping minute intervals)
on F1044M1 video**



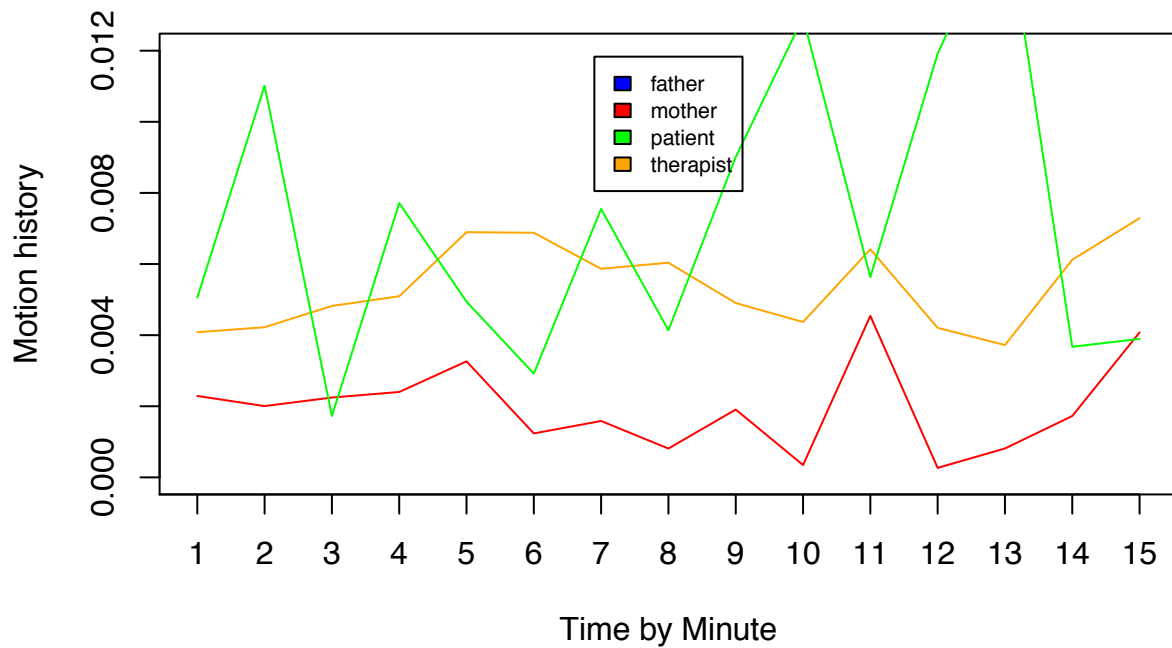
Time by Minute
**Mean motion history (non overlapping minute intervals)
on F1044M2 video**



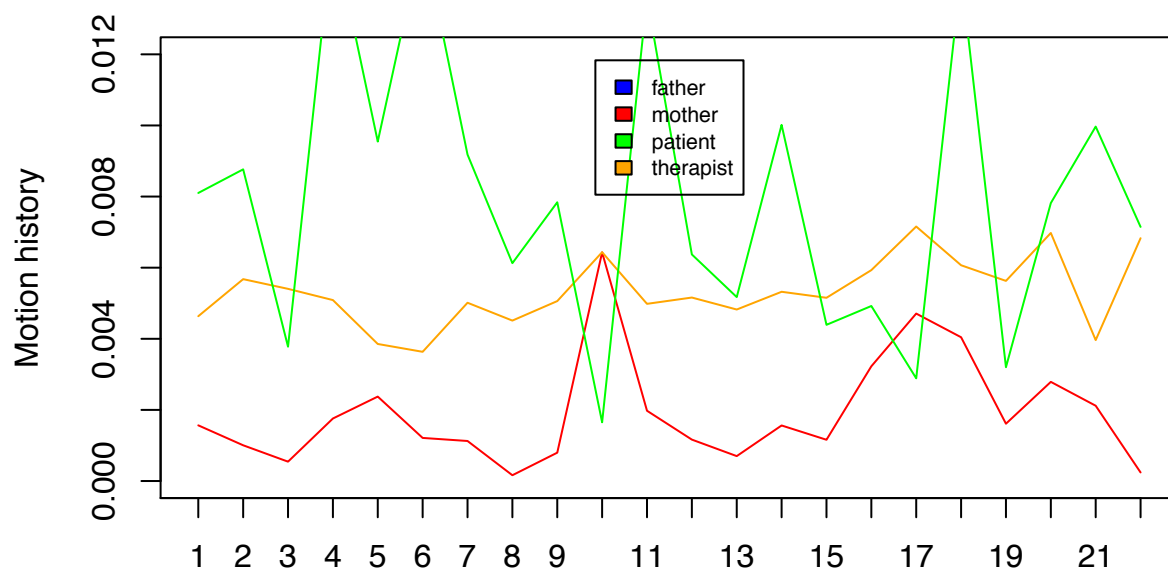
**Mean motion history (non overlapping minute intervals)
on F1044N video**



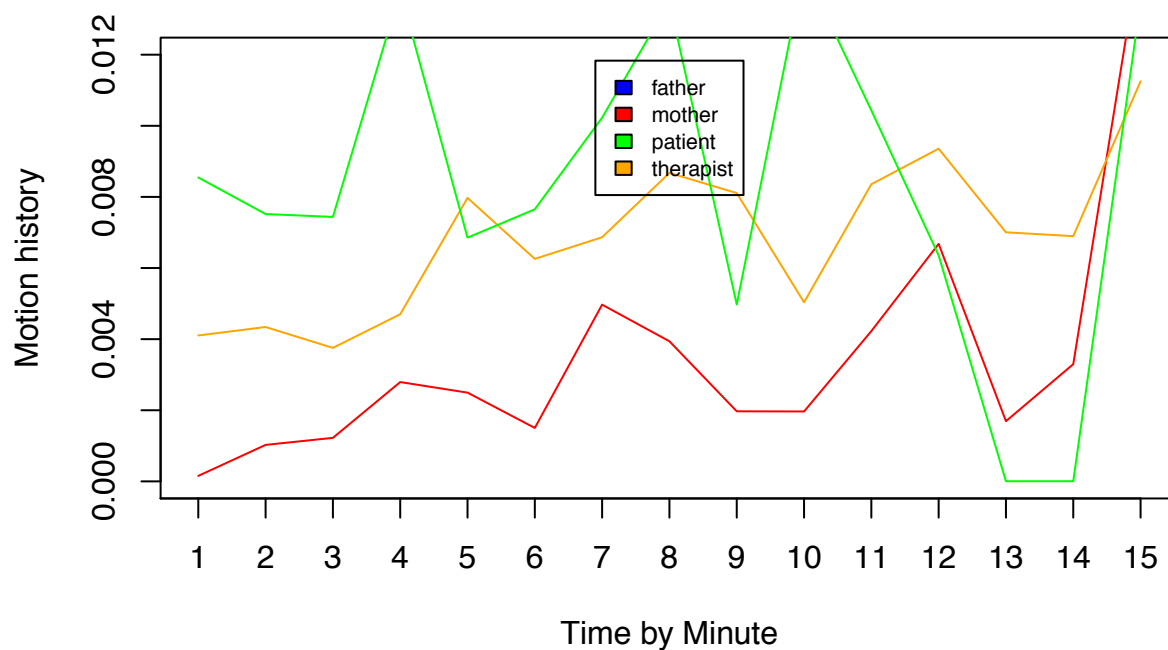
**Mean motion history (non overlapping minute intervals)
on F1044O video**



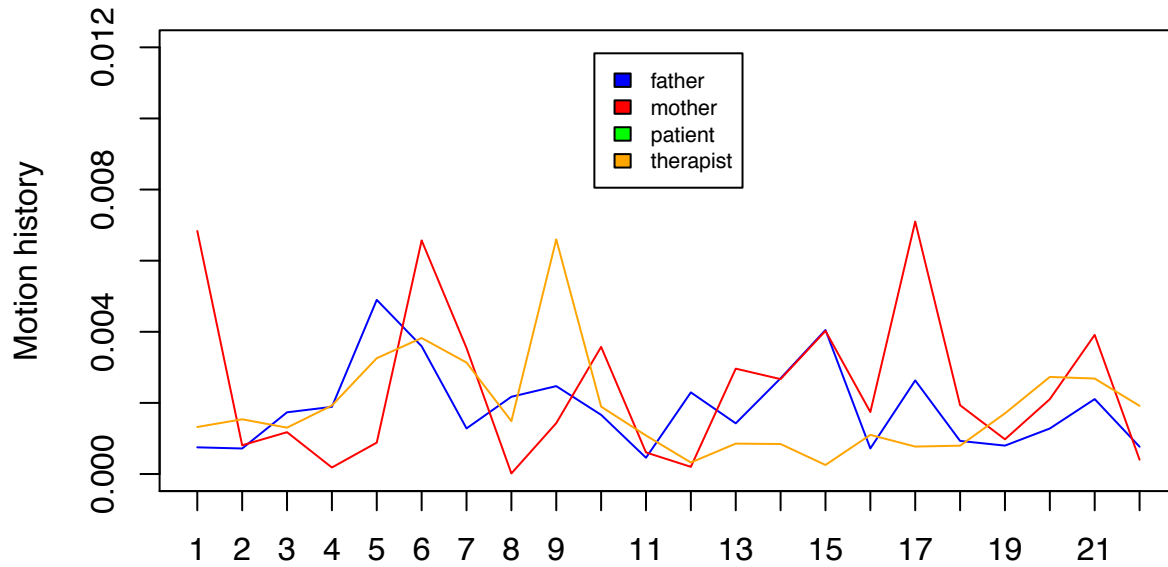
**Mean motion history (non overlapping minute intervals)
on F1044P video**



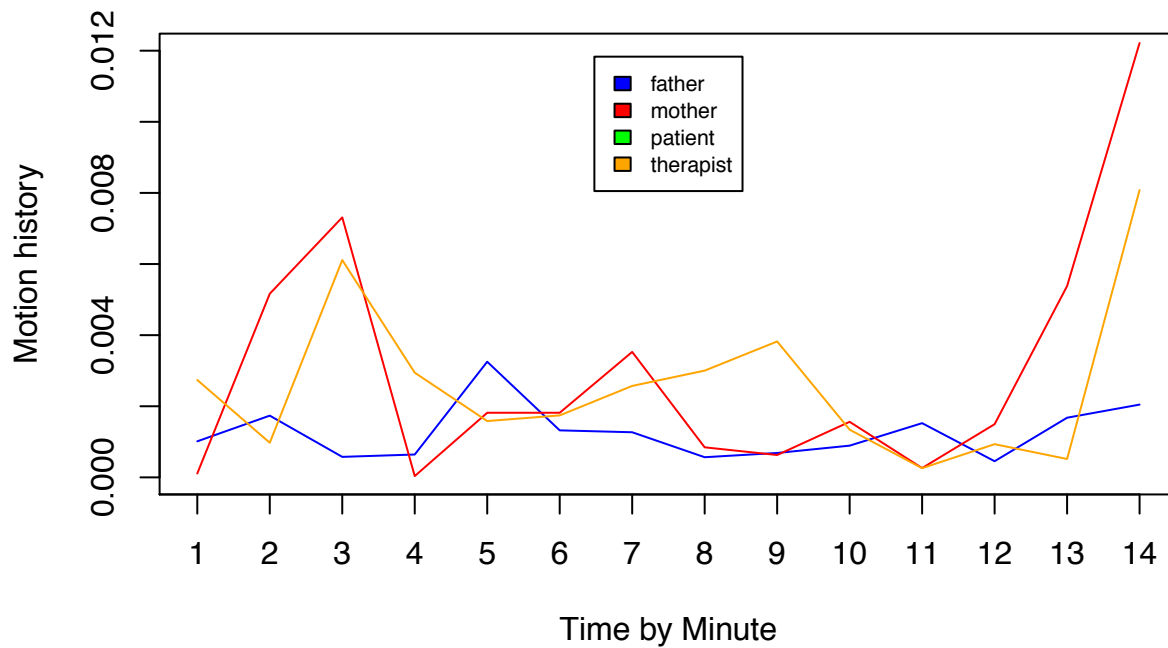
Time by Minute
**Mean motion history (non overlapping minute intervals)
on F1044Q video**



**Mean motion history (non overlapping minute intervals)
on F1044R1 video**



**Mean motion history (non overlapping minute intervals)
on F1044R2 video**



```
# slidedfatherwoNA <- slidedfather[which(is.na(slidedfather)==FALSE)]
# slidedmotherwoNA <- slidedmother[which(is.na(slidedmother)==FALSE)]
# slidedtherapistwoNA <- slidedtherapist[which(is.na(slidedtherapist)==FALSE)]
# slidedpatientwoNA <- slidedpatient[which(is.na(slidedpatient)==FALSE)]
```

Export data in text files

```
## 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

#index de la video de 1ere a la length de indexvideo
videoIndex <- 1
# videoName est le nom de la video actuelle
for (videoName in indexlist){
# Compute slinding interval for each participant
  print(paste("Computing slidedFather", videoName))
  slidedFather <- SlidingInterval("father", videoIndex, 5, data)

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

  print(paste("Computing slidedTherapist", videoName))
  slidedTherapist <- SlidingInterval("therapist", videoIndex, 5, data)

  print(paste("Computing slidedPatient", videoName))
  slidedPatient <- SlidingInterval("patient", videoIndex, 5, data)

# create a data frame to store temporarily this data with NA
  slidedVideo <- data.frame(slidedFather, slidedMother, slidedTherapist, slidedPatient)

##### Creating a data frame if the information is available #####
  dataframe <- FALSE
  dfSliding <- data.frame()
  for (participant in 1:4){
# If the colum is not empty, takes its length and begin a data frame with it
    if (dataframe==FALSE){
#if (length(slidedVideo[participant][!is.na(slidedVideo[participant])]) > 0 & dataframe==FALSE){
      dfSliding <- data.frame("video"=rep(indexlist[videoIndex],length(slidedVideo[participant]))
      dataframe <- TRUE}

# if (length(slidedVideo[participant][!is.na(slidedVideo[participant])]) > 0 & dataframe==TRUE){
    if (dataframe==TRUE){
      dfSliding <- cbind(dfSliding, slidedVideo[participant])}

    print(str(dfSliding))
##### Removing lines with only NA #####
# CCdfSliding <- complete.cases(dfSliding)
  emptyLine <- c()
  for (i in 1:nrow(dfSliding)){
    dfLine <- dfSliding[i,3:6]
    NaLine <- is.na(dfLine)
    if (all(NaLine)){
      emptyLine <- c(emptyLine, i)}}
  print (emptyLine)
```



```

    if (length(emptyLine)>0){
      dfSliding <- dfSliding[-emptyLine,]}

write.csv(dfSliding, paste("/Users/Ofix/Documents/Fac/internat/Recherche/projets/synchro/synchroD
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 (F10044C, F1044D1, etc) that we can import with R with

this following function

```

## [1] "SSI Files Directory"

## [1] "/Users/Ofix/Documents/Fac/internat/Recherche/projets/synchro/synchroData"

## [1] "SS Files List"

## [1] "SSIF1044C.csv" "SSIF1044D1.csv" "SSIF1044D2.csv" "SSIF1044E.csv"
## [5] "SSIF1044F.csv" "SSIF1044G.csv" "SSIF1044H.csv" "SSIF1044I.csv"
## [9] "SSIF1044L.csv" "SSIF1044M1.csv" "SSIF1044M2.csv" "SSIF1044N.csv"
## [13] "SSIF1044O.csv" "SSIF1044P.csv" "SSIF1044Q.csv" "SSIF1044R1.csv"
## [17] "SSIF1044R2.csv"

## [1] "This csv files are imported and merge in a unique data frame SSIDataFrame"

```

Description of SSI data frame

```
str(SSIDataFrame)
```

```

## 'data.frame':    1900 obs. of  13 variables:
## $ X              : int  0 1 2 3 4 5 6 7 8 9 ...
## $ Interval       : int  1 2 3 4 5 6 7 8 9 10 ...
## $ Time_min       : num  0 0.167 0.333 0.5 0.667 ...
## $ video          : Factor w/ 17 levels "F1044C.VOB","F1044D1.VOB",...: 1 1 1 1 1 1 1 1 1 1 ...
## $ SSI_fa_mo      : num  3.55e-04 4.27e-03 4.76e-05 1.61e-02 5.53e-02 ...
## $ SSI_fa_mo_th   : num  0.01428 0.00538 0.02604 0.02166 0.02527 ...

```

```
## $ SSI_fa_th : num 0.014475 0.000896 0.01422 0.005114 0.004301 ...
## $ SSI_mo_th : num 0.018606 0.007802 0.047529 0.031851 0.000883 ...
## $ SSI_fa_pa : num NA NA NA NA NA NA NA NA NA NA ...
## $ SSI_fa_mo_pa: num NA NA NA NA NA NA NA NA NA NA ...
## $ SSI_mo_pa : num NA NA NA NA NA NA NA NA NA NA ...
## $ SSI_pa_th : num NA NA NA NA NA NA NA NA NA NA ...
## $ SSI_mo_pa_th: num NA NA NA NA NA NA NA NA NA NA ...
```

Synchrony scores for each dyad and for the whole group

```
for (i in unique(SSIdataFrame$video))
{par(mar=c(4,4,4,3), mfrow=c(1,1))
plot(SSIdataFrame[which(SSIdataFrame$video==i),]$Time_min,
SSIdataFrame[which(SSIdataFrame$video==i),]$SSI_fa_mo,
type="l", col=rainbow(4)[1],
main=paste("Synchrony scores for each dyad and for \n the whole group in", i, "video"),
xlab = "Time (minute)", ylab="Synchrony score", lwd=2,
xaxp=c(0,length(SSIdataFrame$Time_min), length(SSIdataFrame$Time_min)))
#, ylim=c(0,0.5))
abline(h=mean(SSIdataFrame$SSI_fa_mo, na.rm=TRUE), col=rainbow(11)[1], lwd=2, lty=2)

lines(SSIdataFrame[which(SSIdataFrame$video==i),]$SSI_fa_mo_pa, col=rainbow(11)[2], lwd=2)
abline(h= mean(SSIdataFrame$SSI_fa_mo_pa, na.rm=TRUE), col=rainbow(11)[2], lwd=2, lty=2)

# lines(SSIdataFrame[which(SSIdataFrame$video==i),]$SSI_fa_mo_pa_th, col=rainbow(11)[3], lwd=2)
# abline(h= mean(SSIdataFrame$SSI_fa_mo_pa_th, na.rm=TRUE), col=rainbow(11)[3], lwd=2, lty=2)

lines(SSIdataFrame[which(SSIdataFrame$video==i),]$SSI_fa_mo_th, col=rainbow(11)[4], lwd=2)
abline(h= mean(SSIdataFrame$SSI_fa_mo_th, na.rm=TRUE), col=rainbow(11)[4], lwd=2, lty=2)

lines(SSIdataFrame[which(SSIdataFrame$video==i),]$SSI_fa_pa, col=rainbow(11)[5], lwd=2)
abline(h= mean(SSIdataFrame$SSI_fa_pa, na.rm=TRUE), col=rainbow(11)[5], lwd=2, lty=2)

# lines(SSIdataFrame[which(SSIdataFrame$video==i),]$SSI_fa_pa_th, col=rainbow(11)[6], lwd=2)
# abline(h= mean(SSIdataFrame$SSI_fa_pa_th, na.rm=TRUE), col=rainbow(11)[6], lwd=2, lty=2)

lines(SSIdataFrame[which(SSIdataFrame$video==i),]$SSI_fa_th, col=rainbow(11)[7], lwd=2)
abline(h= mean(SSIdataFrame$SSI_fa_th, na.rm=TRUE), col=rainbow(11)[7], lwd=2, lty=2)

lines(SSIdataFrame[which(SSIdataFrame$video==i),]$SSI_mo_pa, col=rainbow(11)[8], lwd=2)
abline(h= mean(SSIdataFrame$SSI_mo_pa, na.rm=TRUE), col=rainbow(11)[8], lwd=2, lty=2)

lines(SSIdataFrame[which(SSIdataFrame$video==i),]$SSI_mo_pa_th, col=rainbow(11)[9], lwd=2)
abline(h= mean(SSIdataFrame$SSI_mo_pa_th, na.rm=TRUE), col=rainbow(11)[9], lwd=2, lty=2)
lines(SSIdataFrame[which(SSIdataFrame$video==i),]$SSI_mo_th, col=rainbow(11)[10], lwd=2)
abline(h= mean(SSIdataFrame$SSI_mo_th, na.rm=TRUE), col=rainbow(11)[10], lwd=2, lty=2)

lines(SSIdataFrame[which(SSIdataFrame$video==i),]$SSI_pa_th, col=rainbow(11)[11], lwd=2)
abline(h= mean(SSIdataFrame$SSI_pa_th, na.rm=TRUE), col=rainbow(11)[11], lwd=2, lty=2)

legend("topleft", inset=.05, c("fa_mo", "fa_mo_pa", "fa_mo_pa_th",
```

```

"fa_mo_th", "fa_pa", "fa_pa_th", "fa_th",
"mo_pa", "mo_pa_th", "mo_th", "pa_th"),
col=rainbow(11), cex=0.6, lwd=2)

legend("topright", inset=.05, c(paste ("Mean fa_mo :",
                                         round(mean(SSIdataFrame$SSI_fa_mo, na.rm=TRUE),3)),
  paste ("Mean fa_mo_pa :", round(mean(SSIdataFrame$SSI_fa_mo_pa,na.rm=TRUE),3)),
#   paste ("Mean fa_mo_pa_th :", #round(mean(SSIdataFrame$SSI_fa_mo_pa_th),3)),
  paste ("Mean fa_mo_th :", round(mean(SSIdataFrame$SSI_fa_mo_th,na.rm=TRUE),3)),
  paste ("Mean fa_pa :", round(mean(SSIdataFrame$SSI_fa_pa, na.rm=TRUE),3)),
#   paste ("Mean fa_pa_th :", round(mean(SSIdataFrame$SSI_fa_pa_th,na.rm=TRUE),3)),
  paste ("Mean fa_th :", round(mean(SSIdataFrame$SSI_fa_th,na.rm=TRUE),3)),
  paste ("Mean mo_pa :", round(mean(SSIdataFrame$SSI_mo_pa,na.rm=TRUE),3)),
  paste ("Mean mo_pa_th :", round(mean(SSIdataFrame$SSI_mo_pa_th,na.rm=TRUE),3)),
  paste ("Mean mo_th :", round(mean(SSIdataFrame$SSI_mo_th,na.rm=TRUE),3)),
  paste ("Mean pa_th :", round(mean(SSIdataFrame$SSI_pa_th,na.rm=TRUE),3))),
col=rainbow(11), cex=0.5, lty=2, lwd=1)})

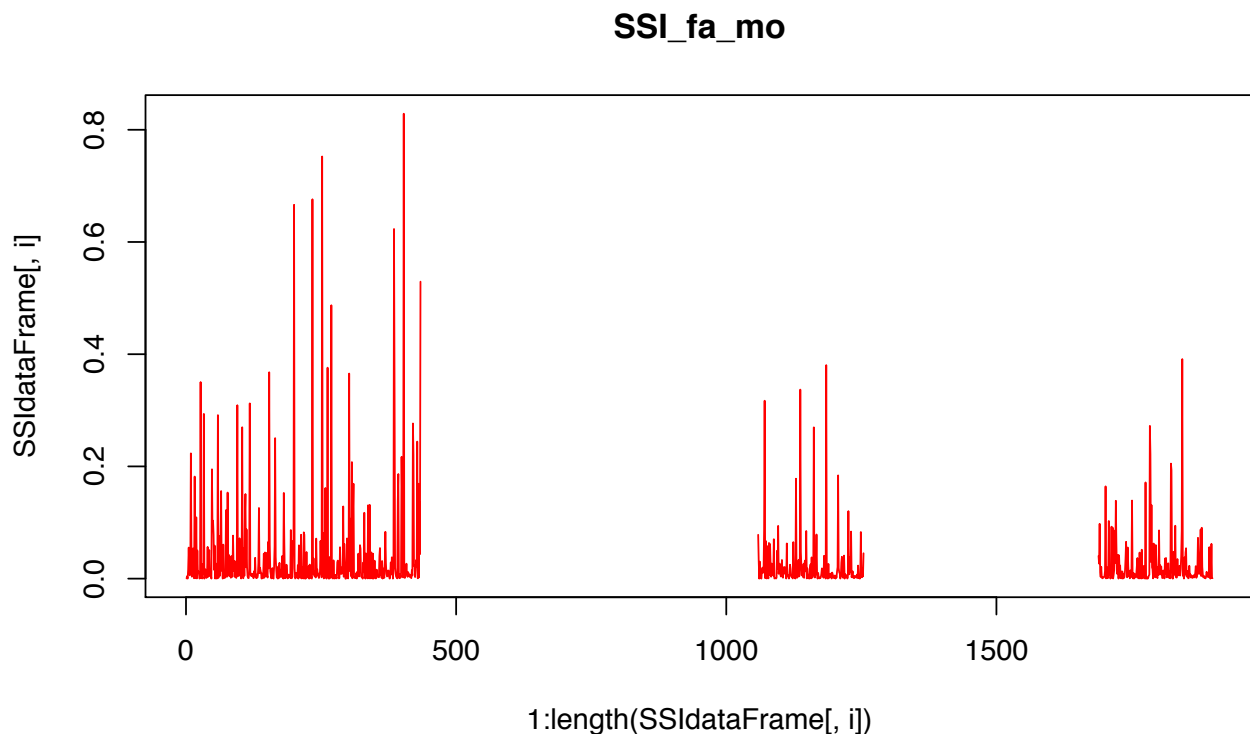
```

Evolution of synchrony through time, raw each second

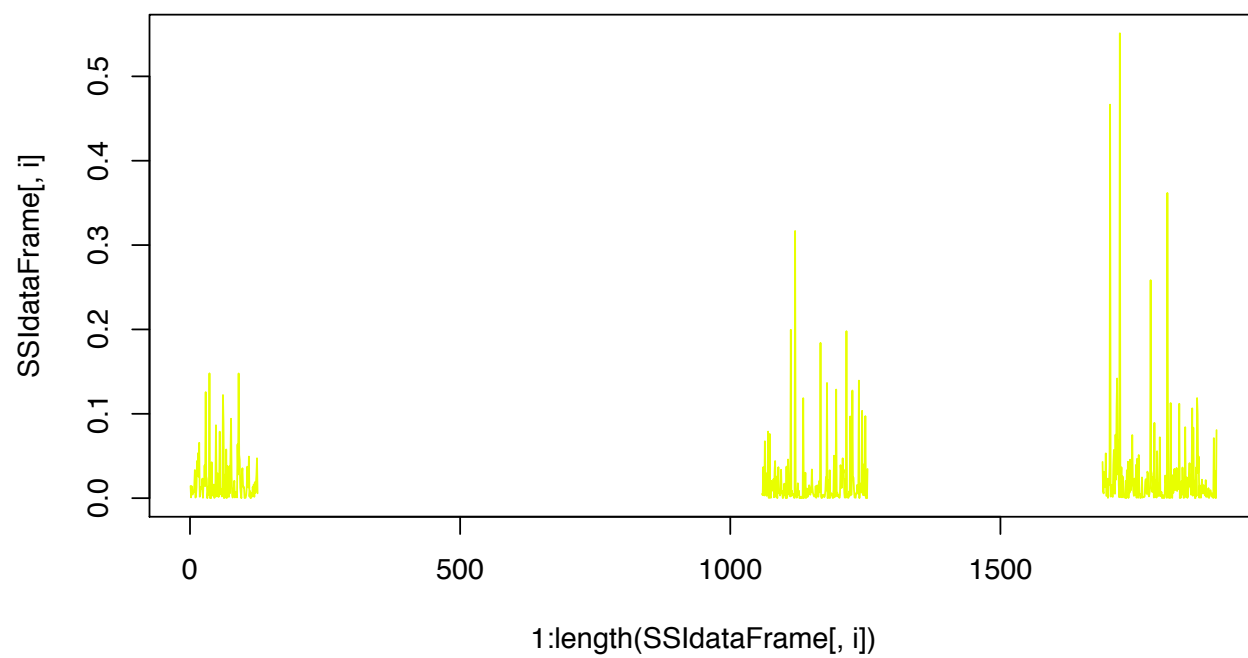
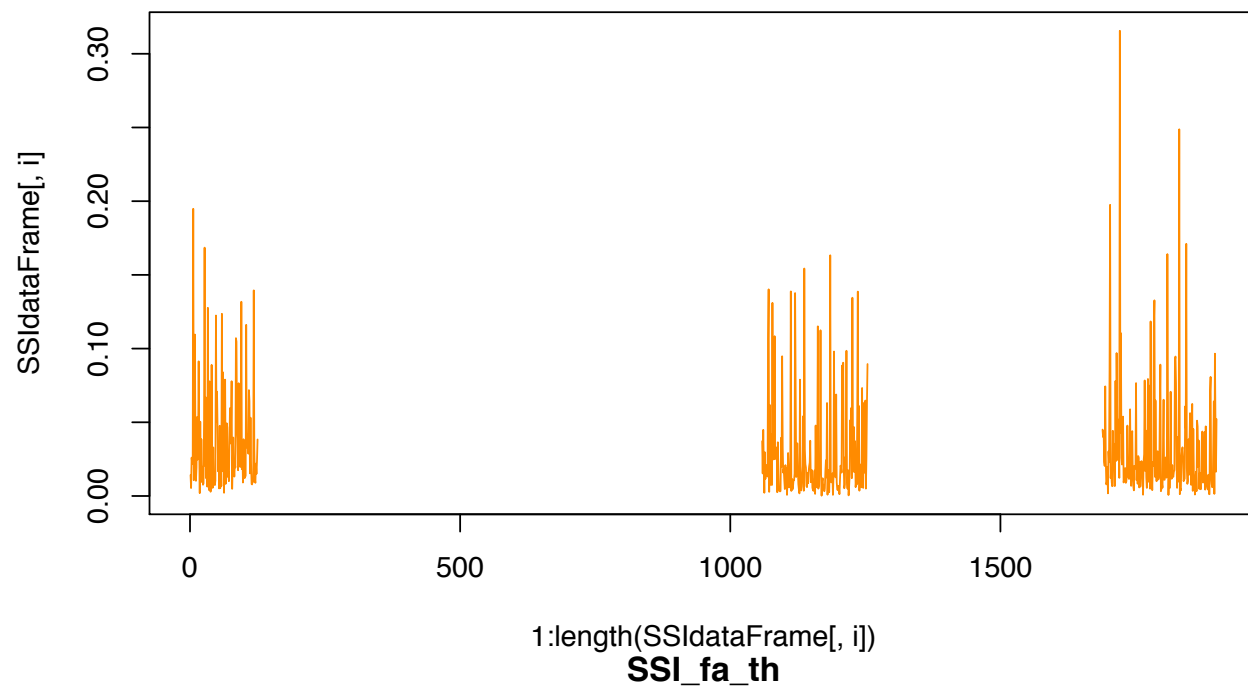
```

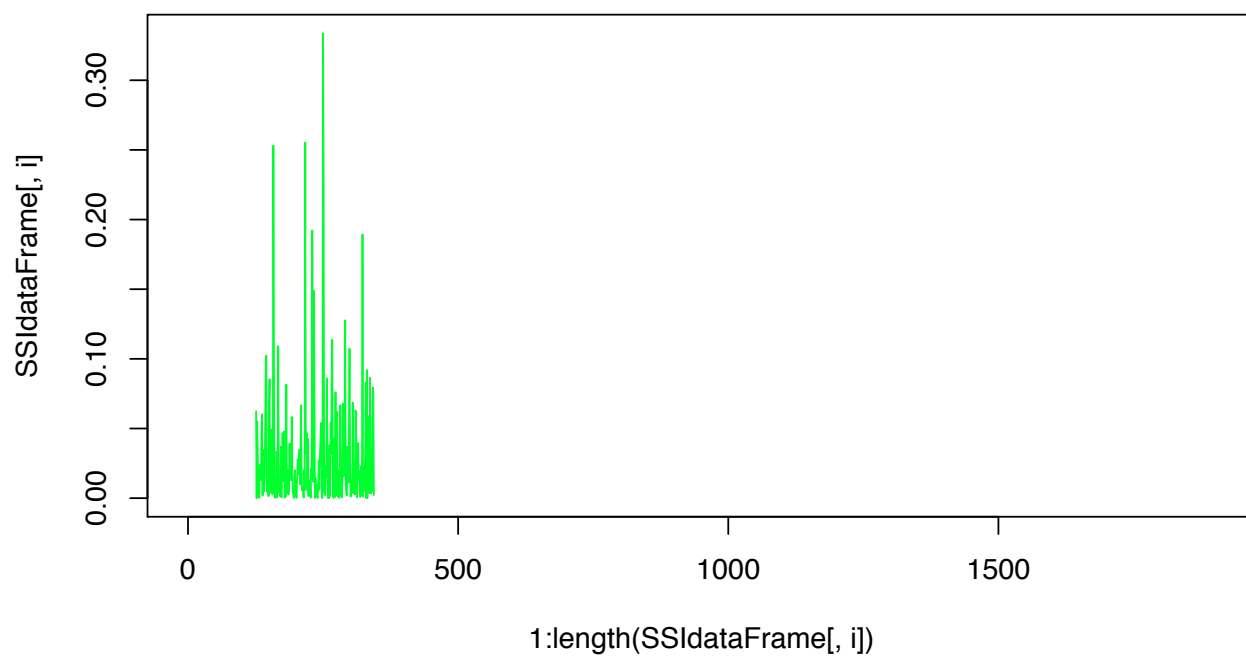
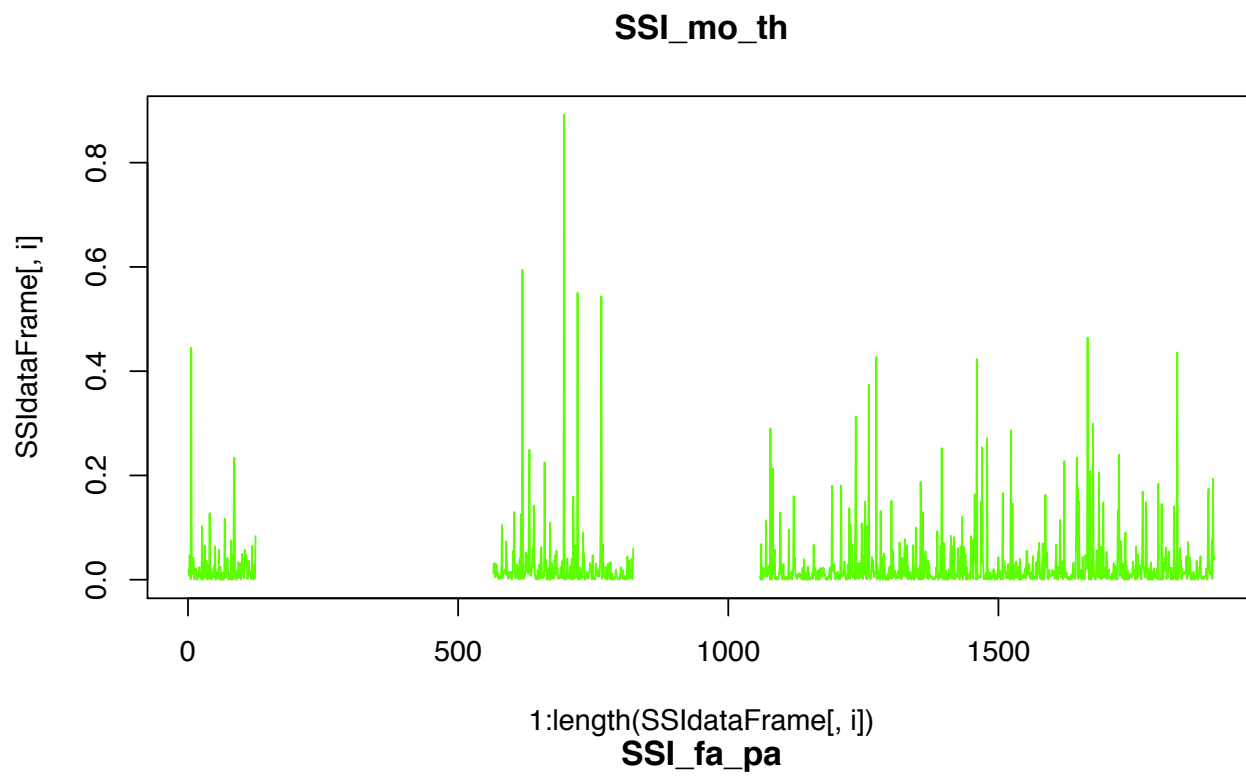
par(mar=c(4,4,4,4))
col= 1
for (i in 5:13){
  plot(1:length(SSIdataFrame[,i]), SSIdataFrame[,i], type="l",
  col=rainbow(11)[col], main = names(SSIdataFrame)[i])
  col <- col+1}

```

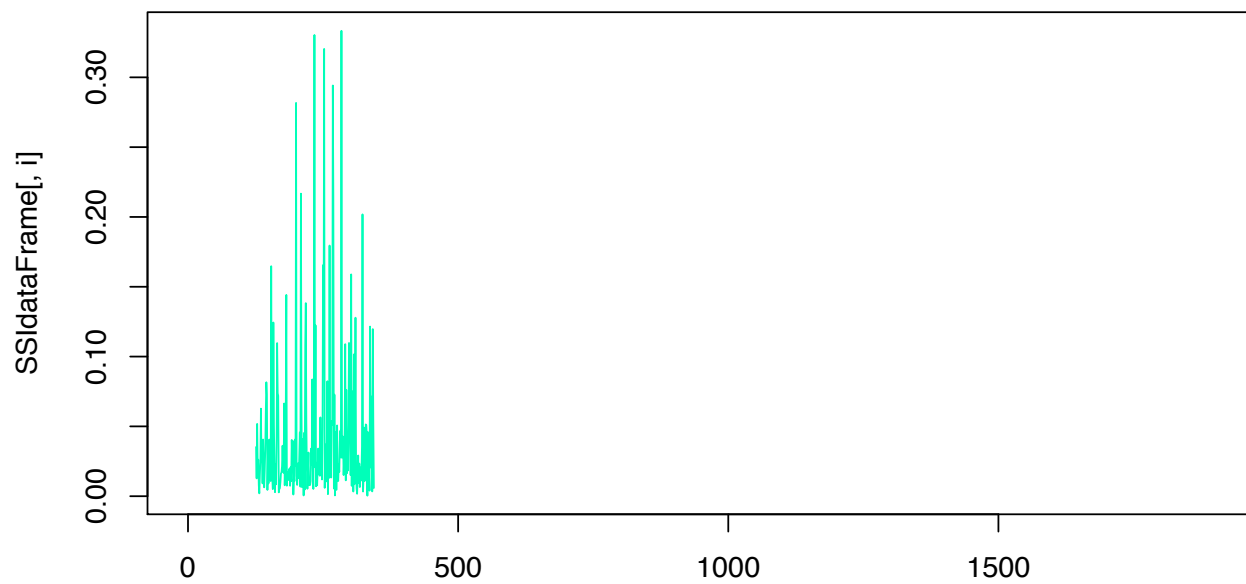


SSI_fa_mo_th

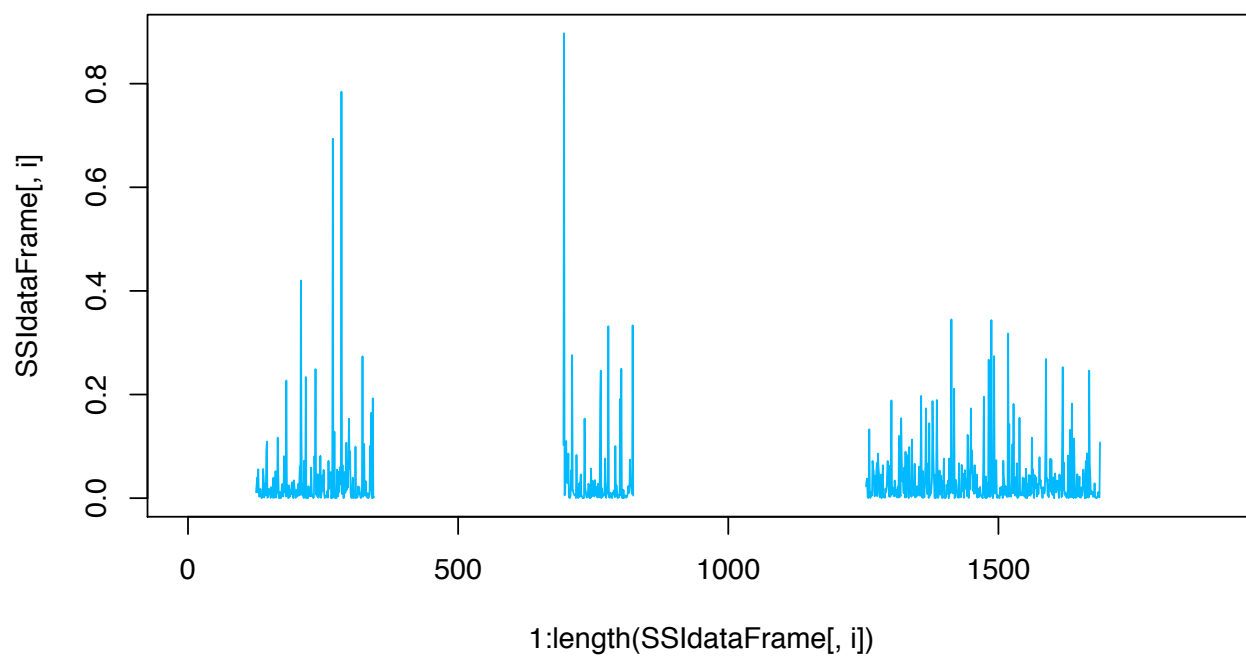


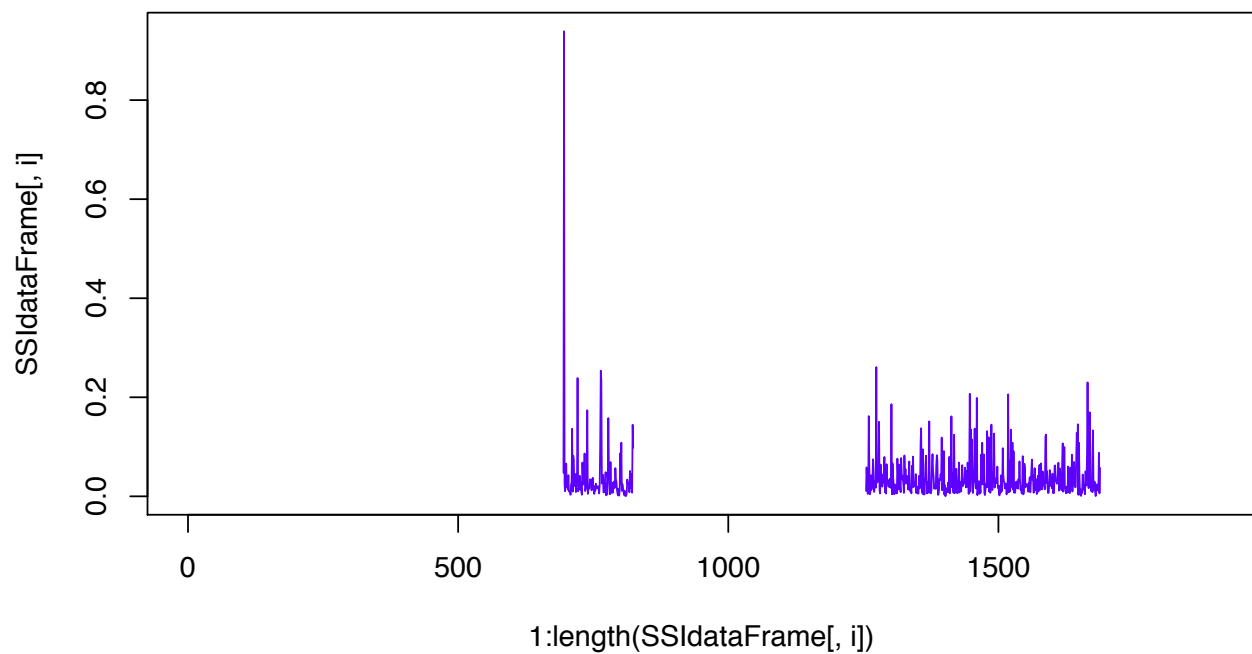
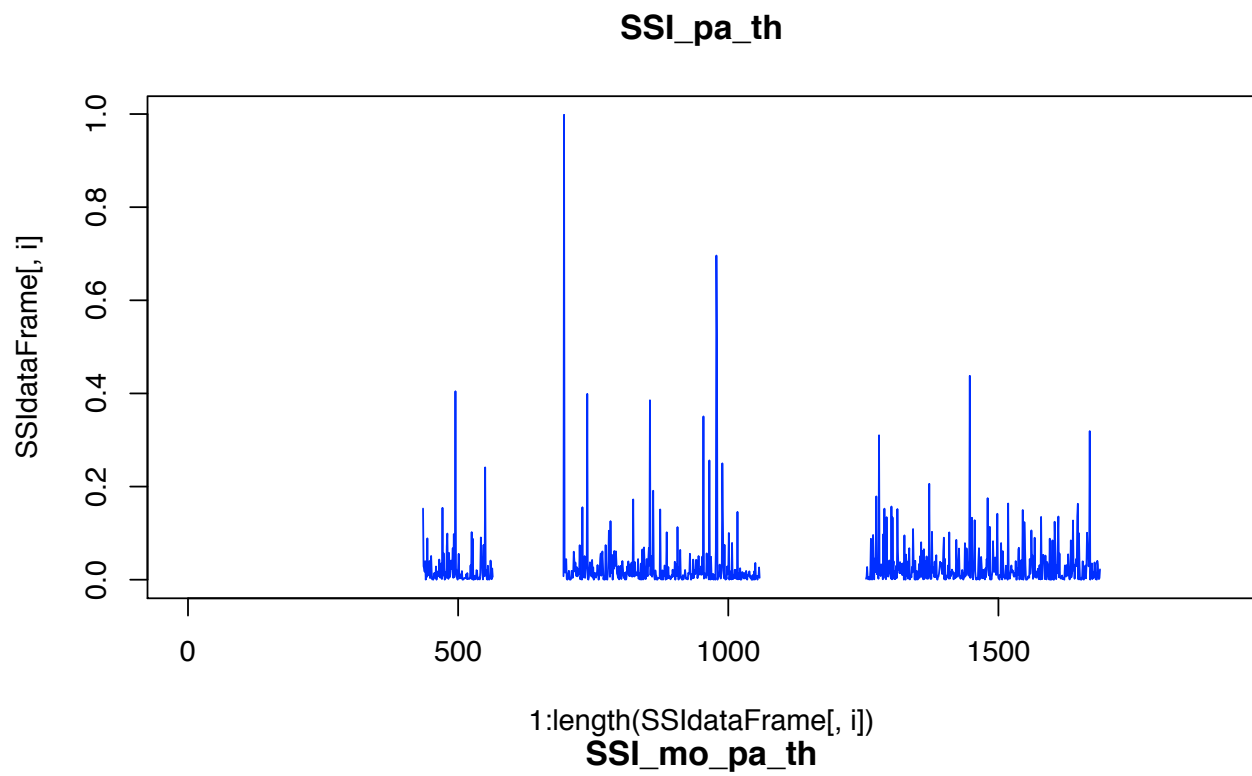


SSI_fa_mo_pa



SSI_mo_pa





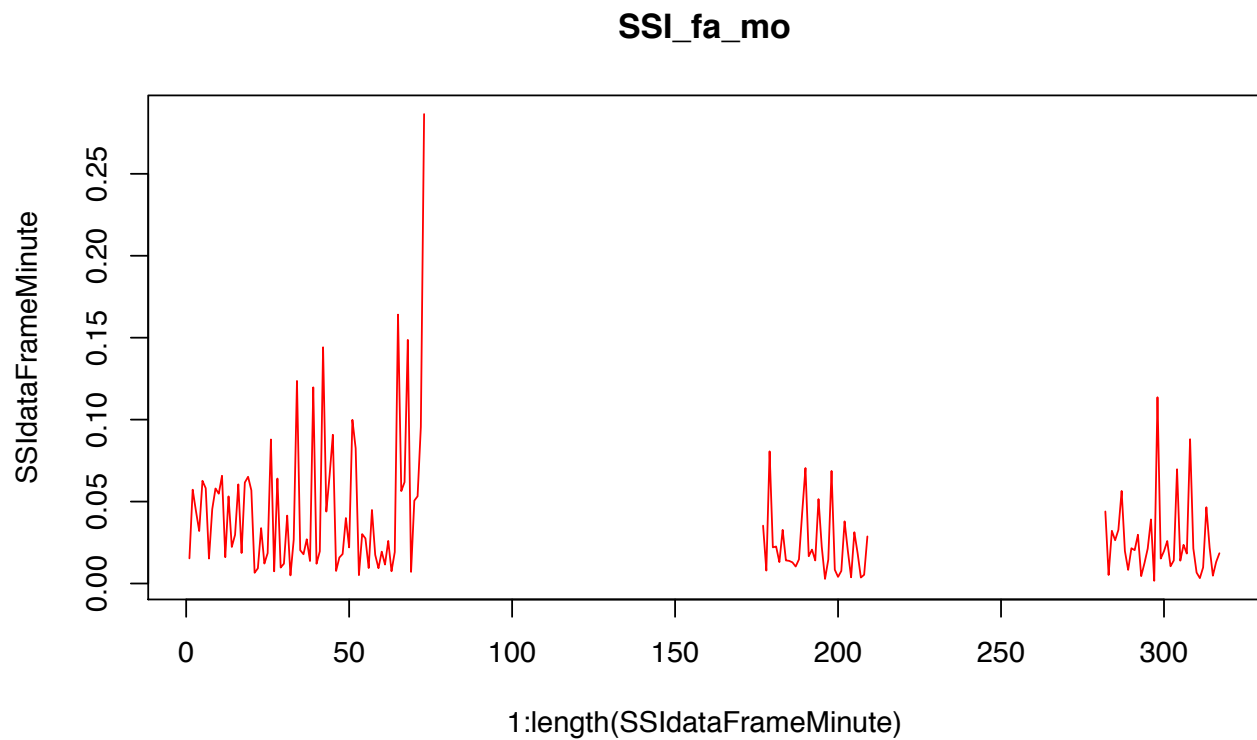
Evolution of synchrony through time, mean by minute

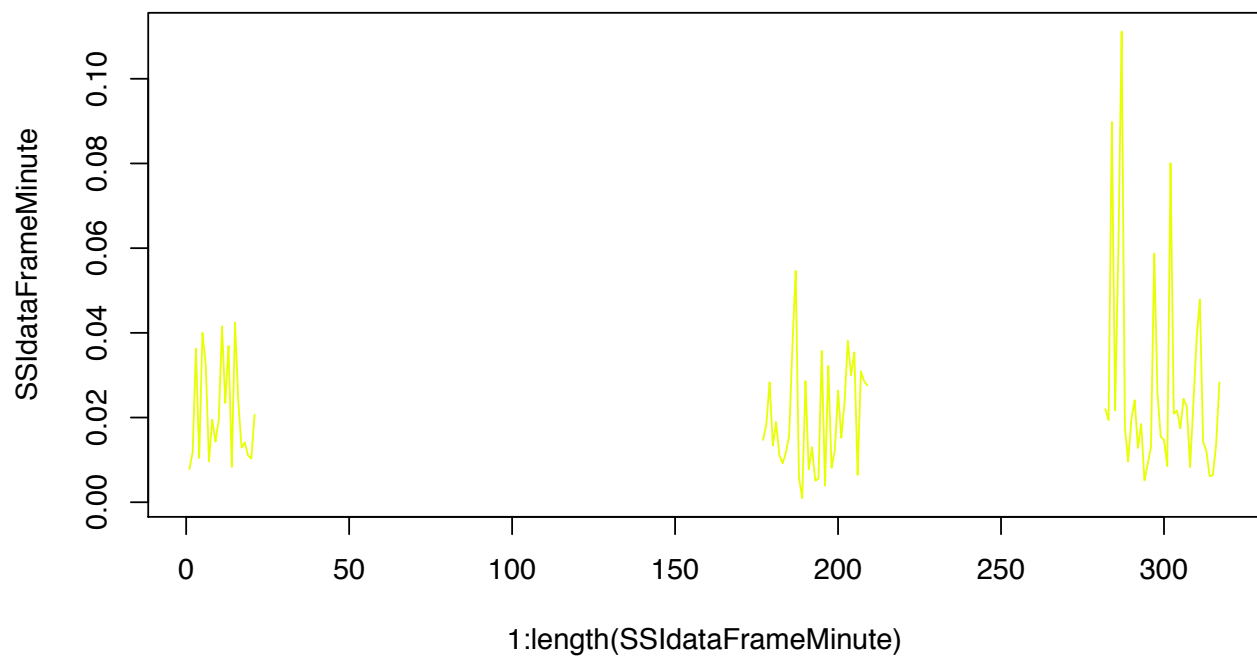
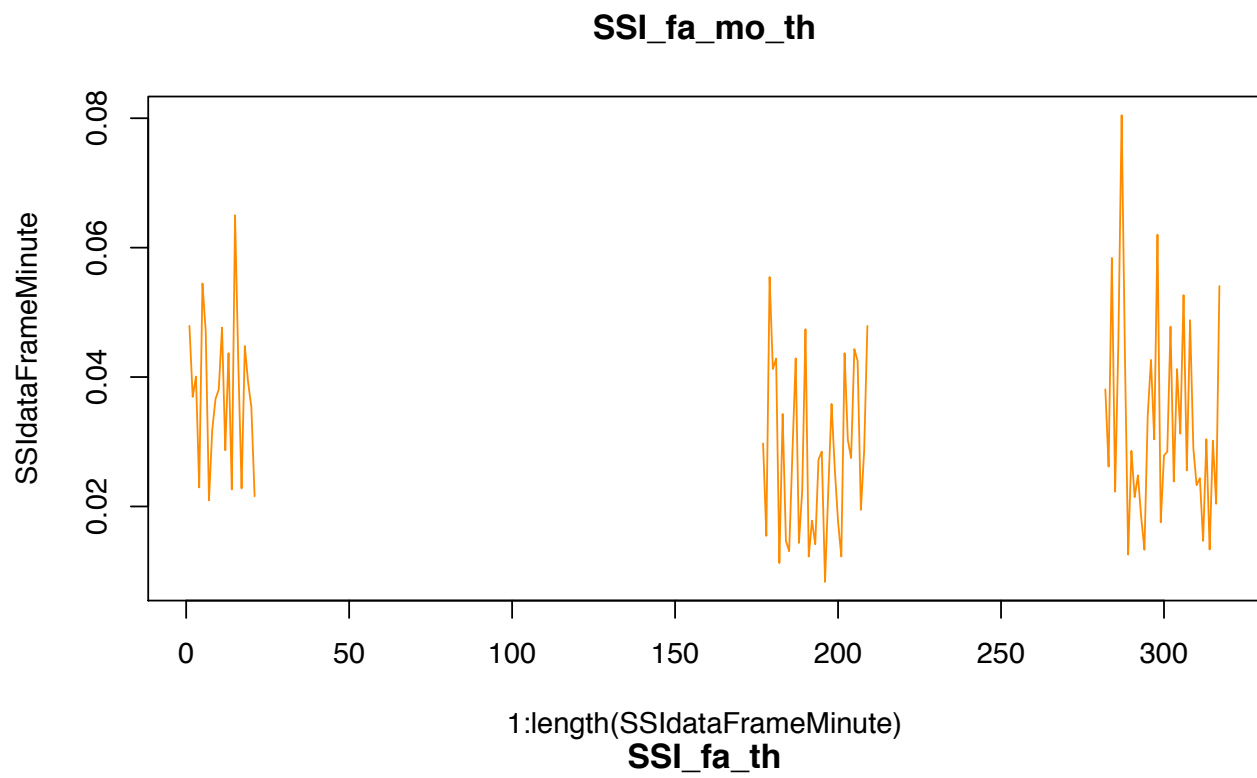
```
par(mar=c(4,4,4,4))
col = 1
for (indexSSI in 5:13){
```

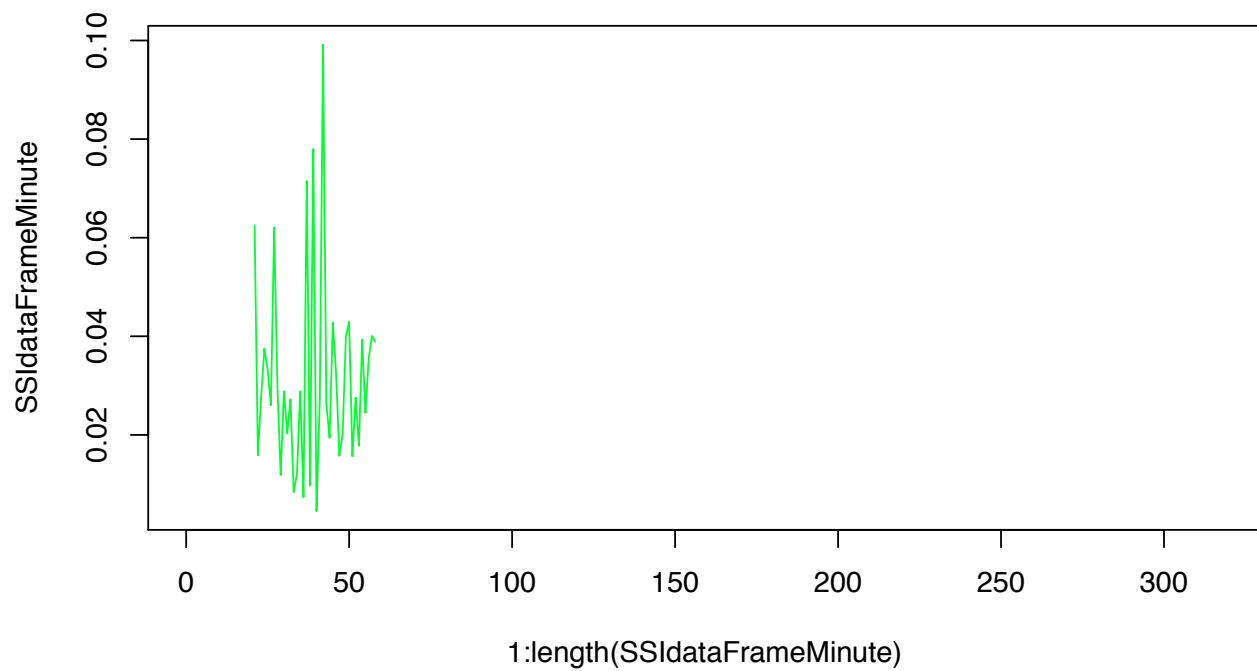
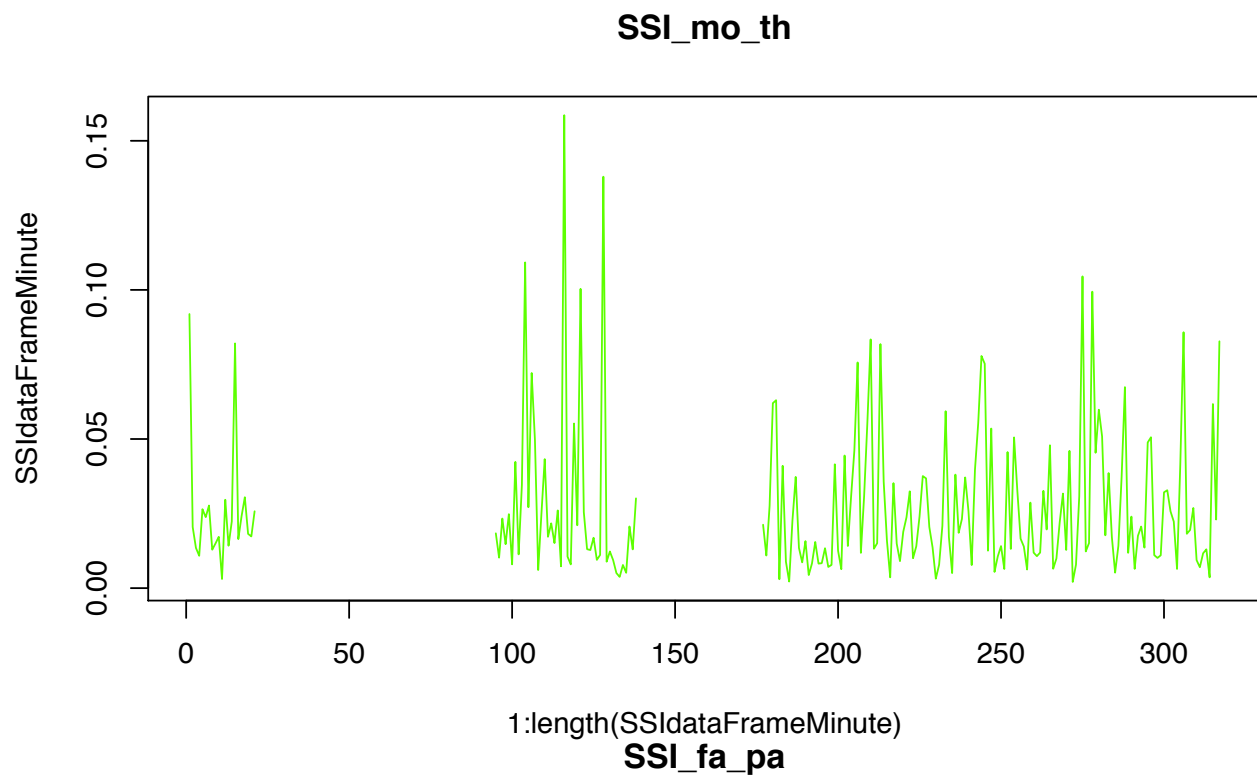
```

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

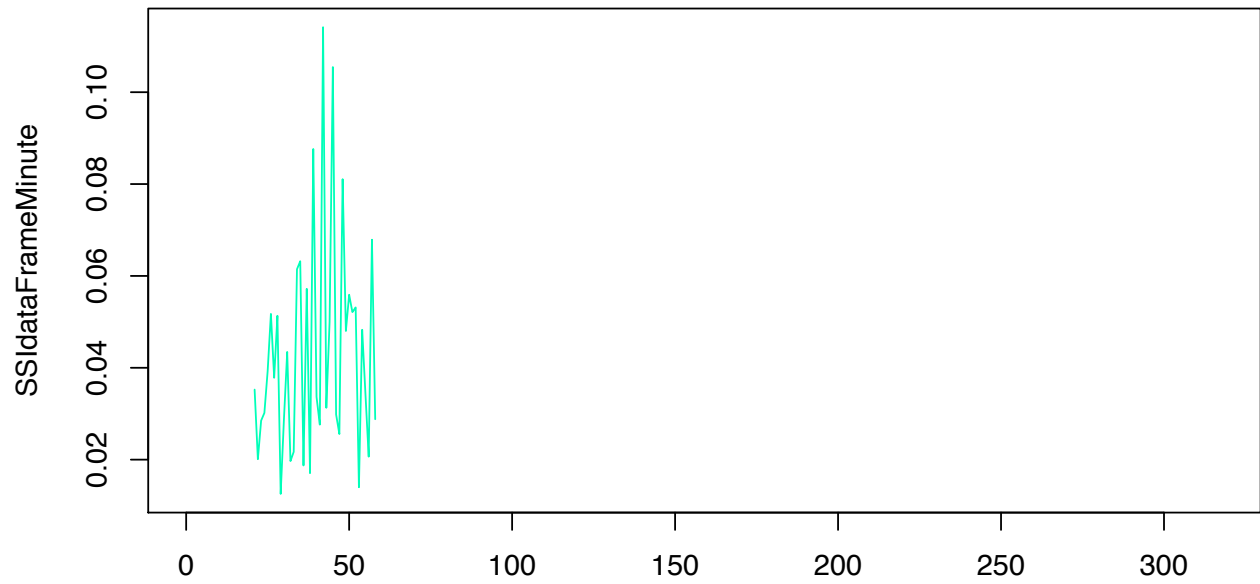
```



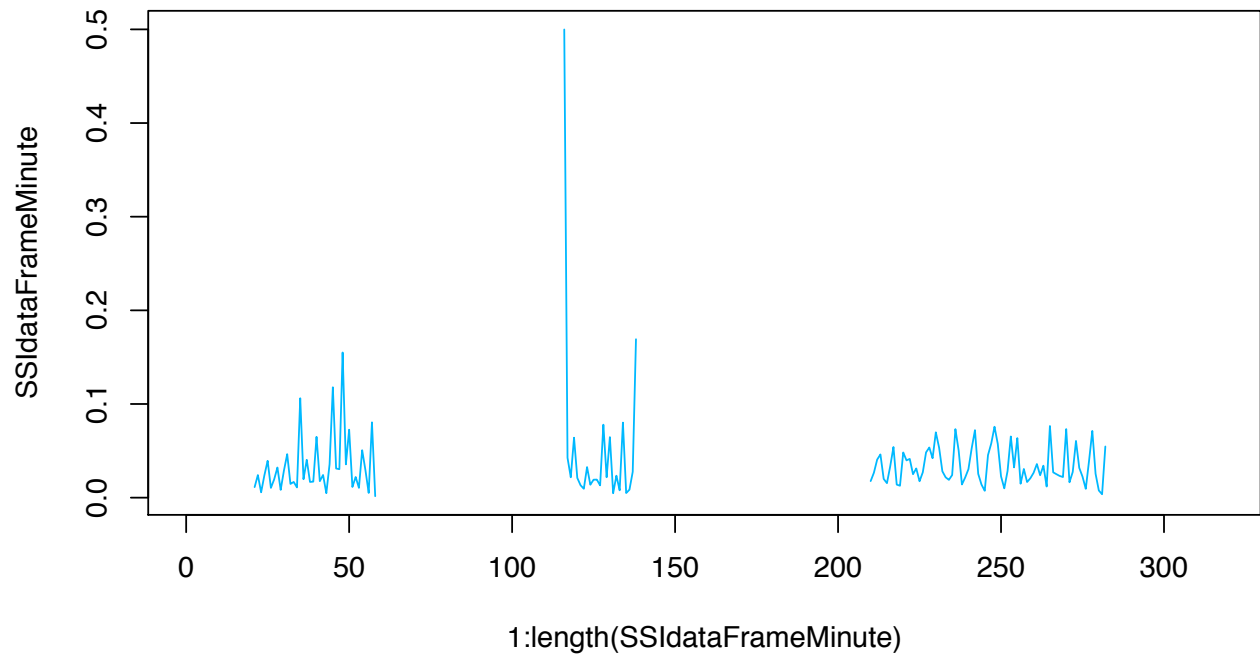


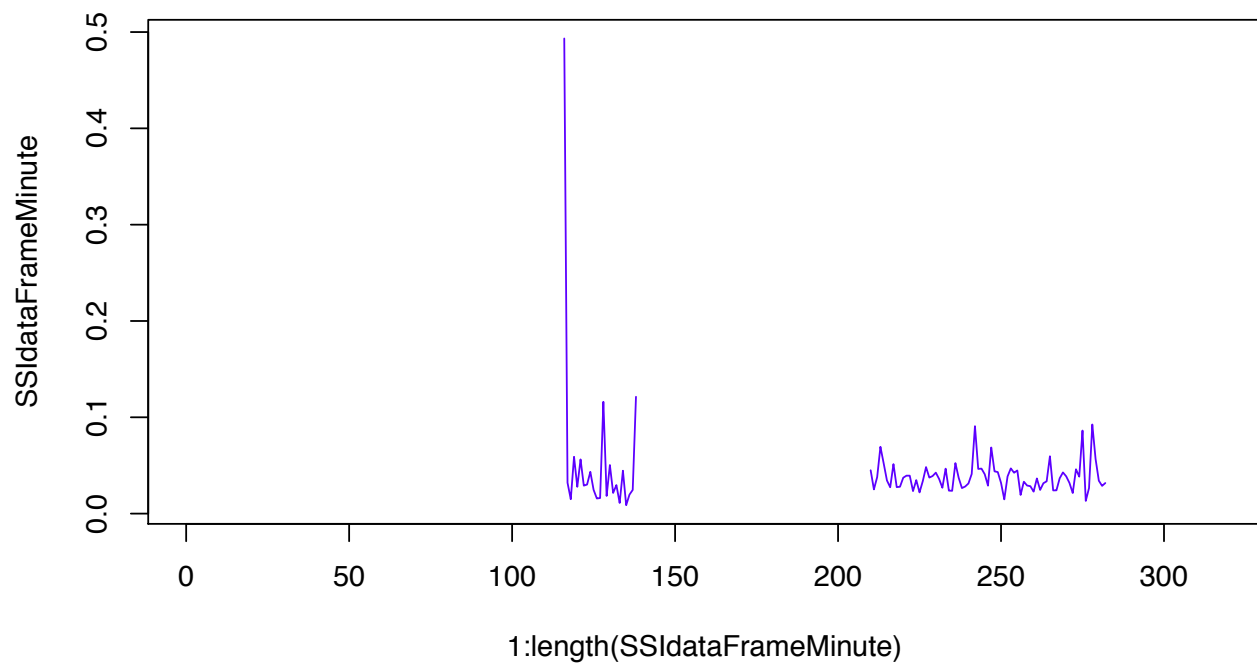
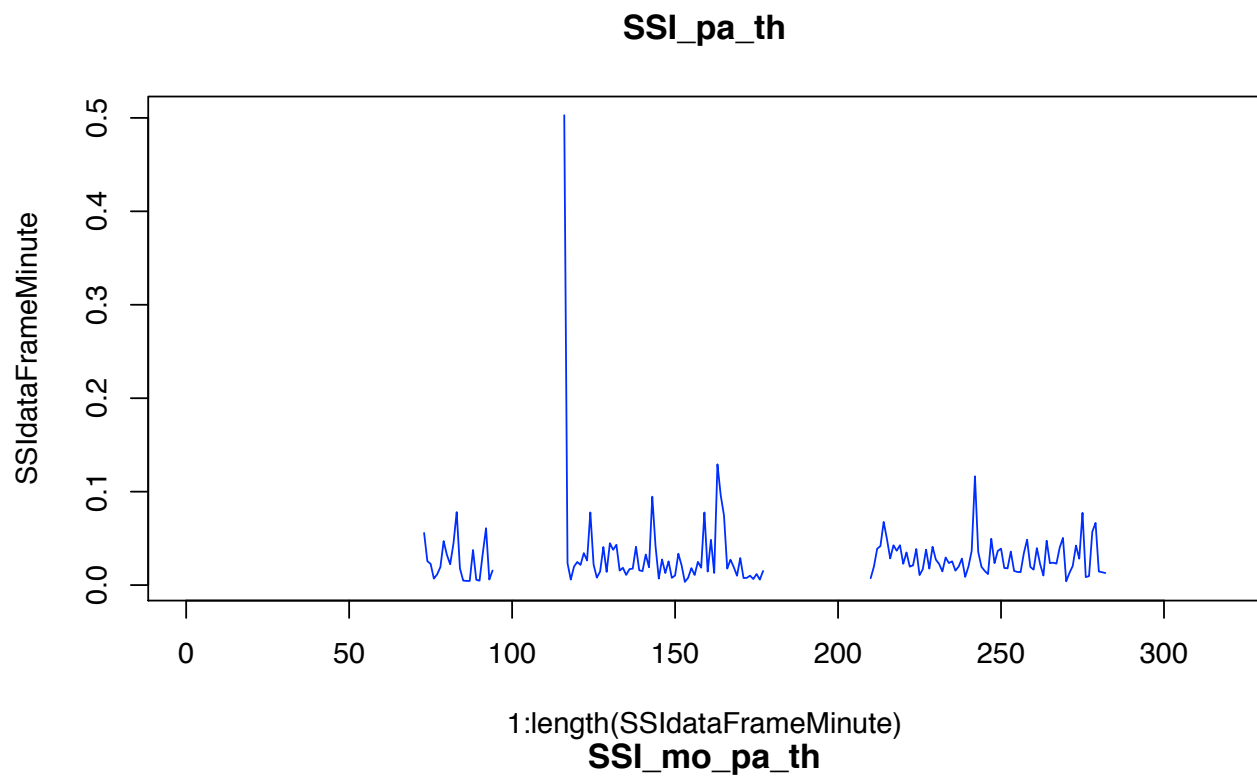


SSI_fa_mo_pa



1:length(SSIdataFrameMinute)
SSI_mo_pa





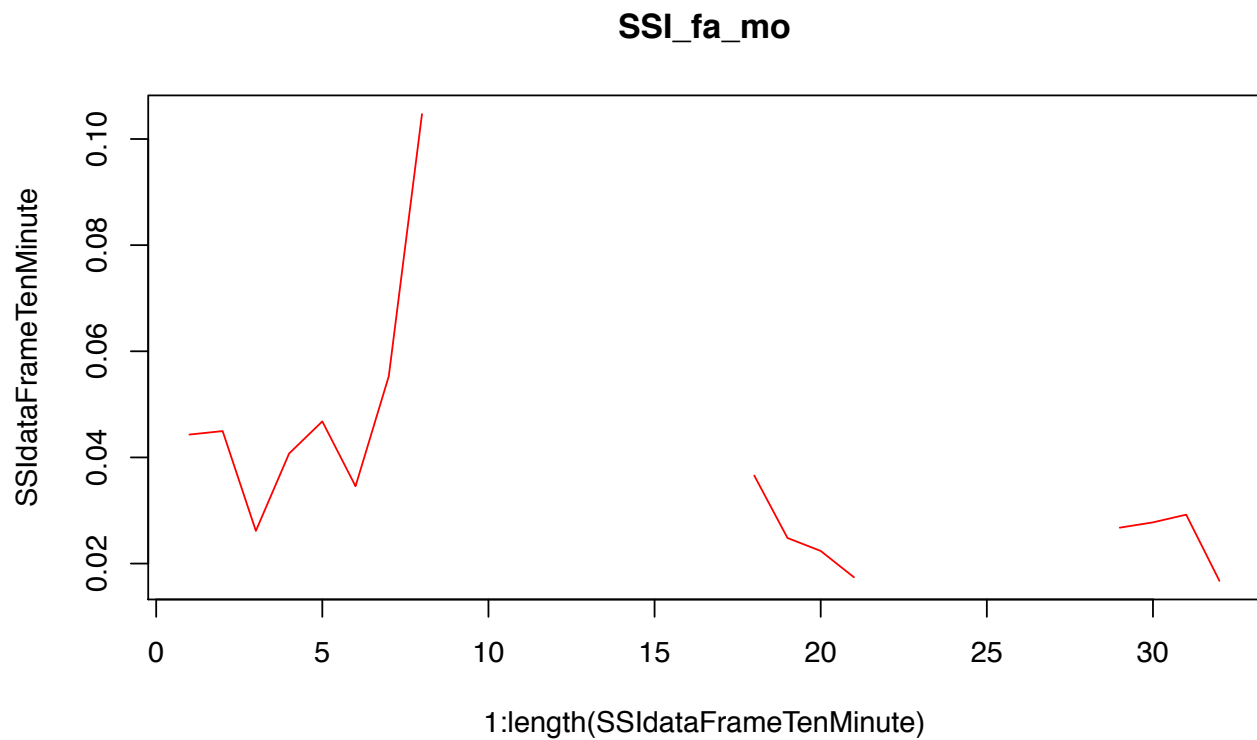
Evolution of synchrony through time, mean by 10 minutes

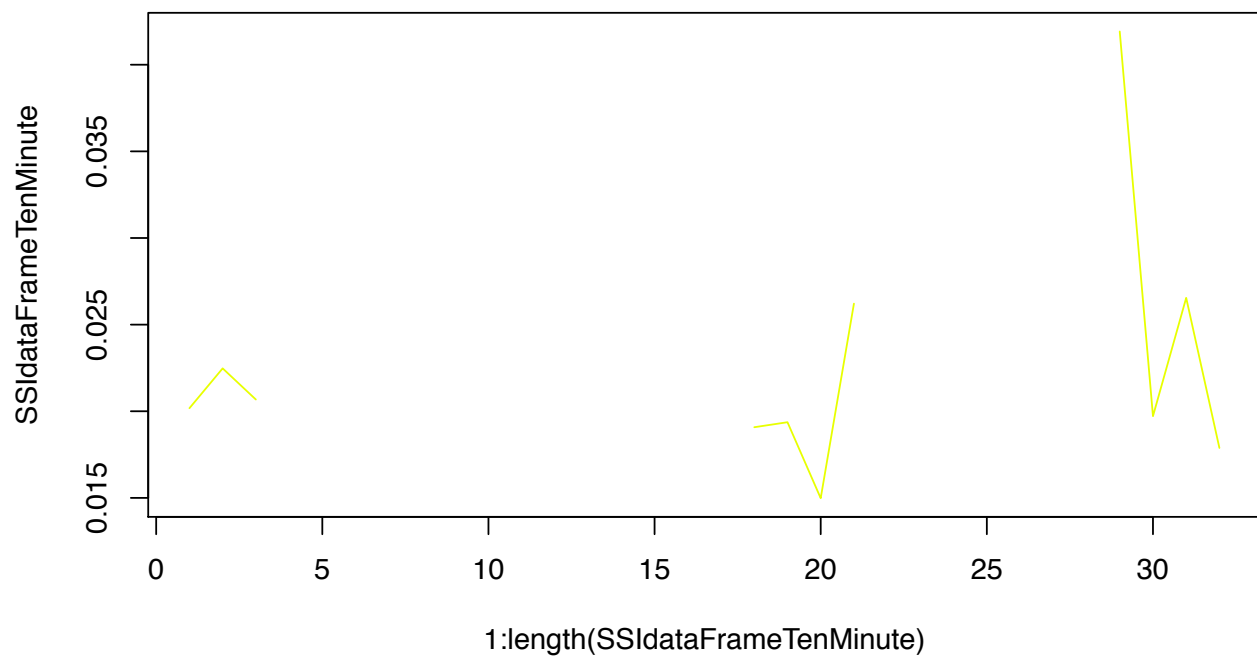
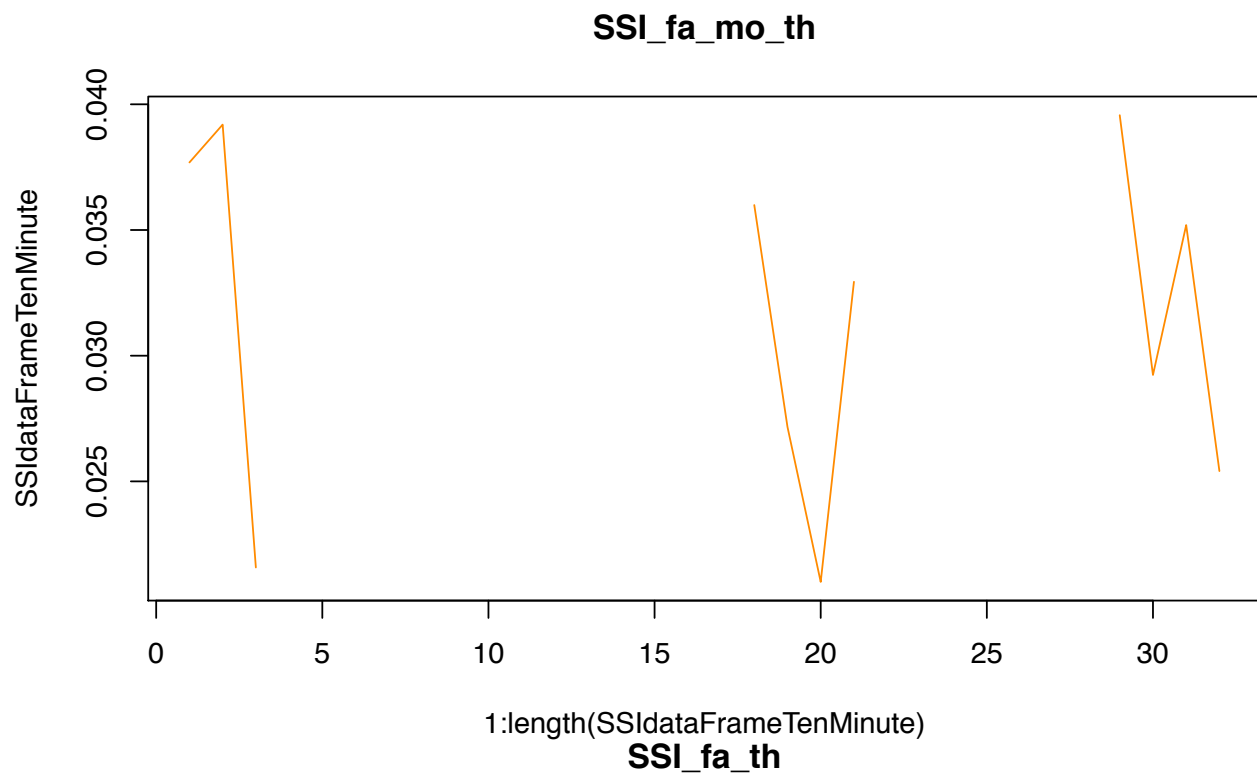
```
par(mar=c(4,4,4,4))
col = 1
for (indexSSI in 5:13){
```

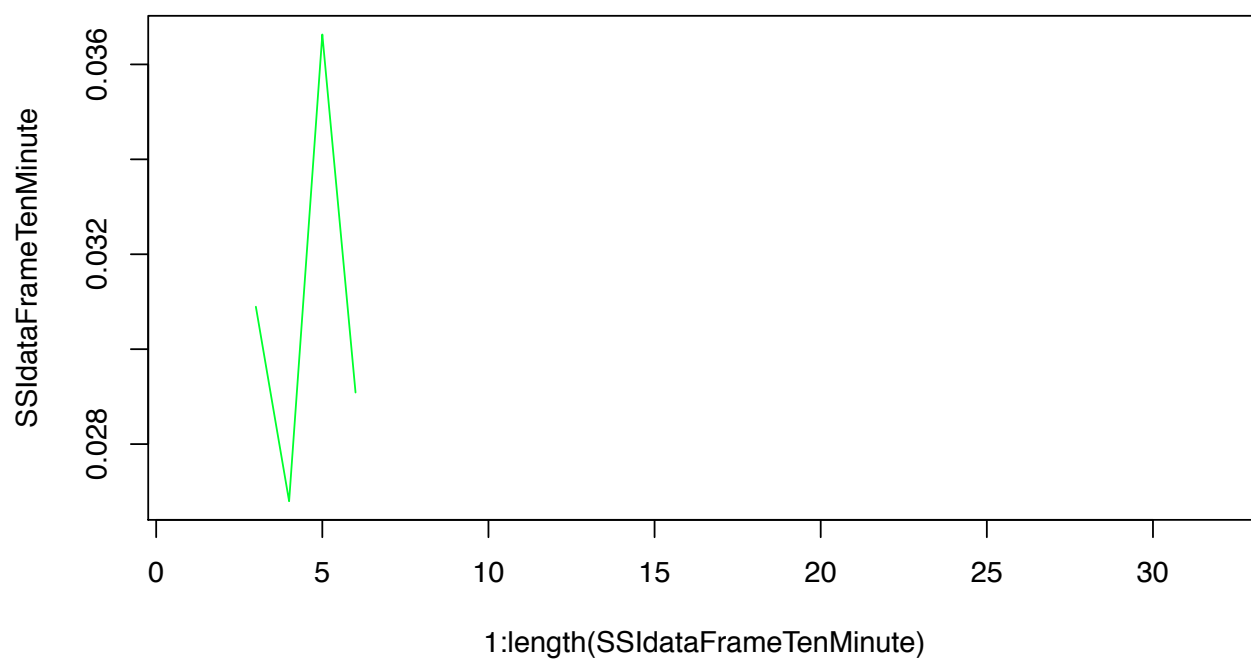
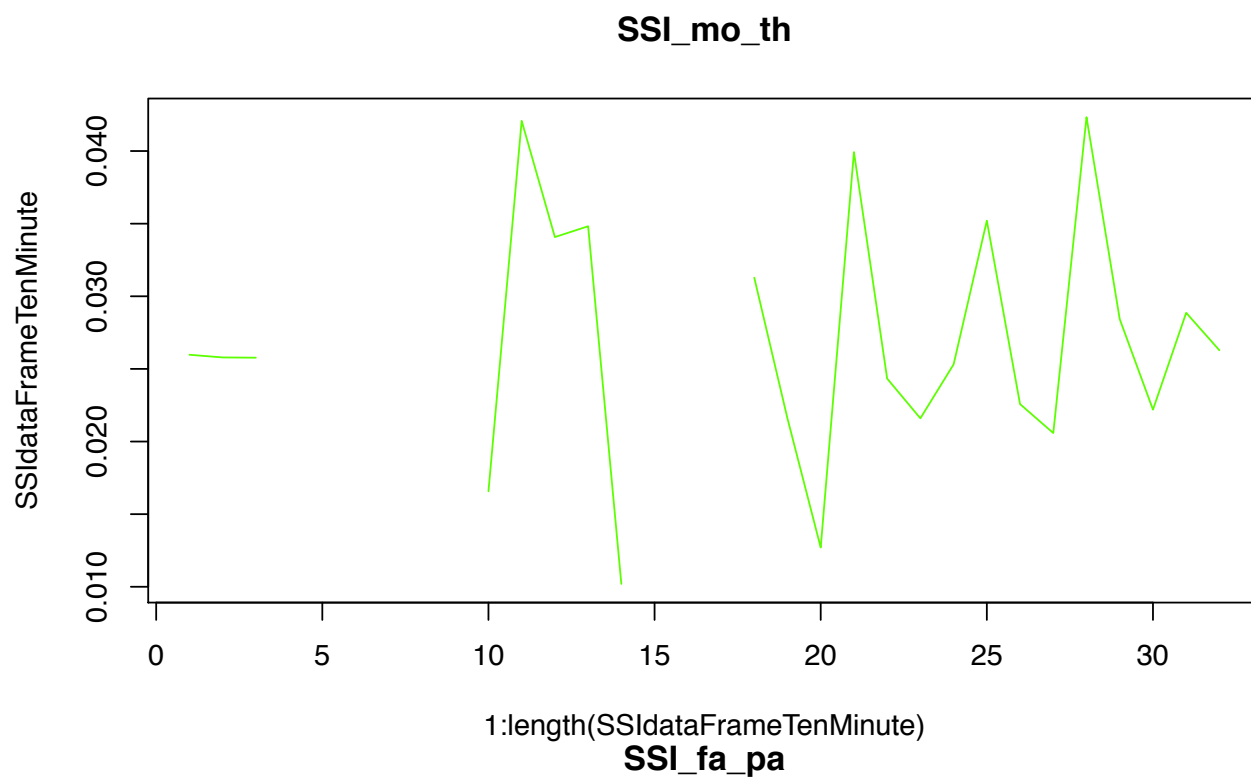
```

IntervalNumbersVideo <- ceiling(length(SSIdataFrame[,indexSSI])/60)
SSIColumn <- SSIdataFrame[,indexSSI]
SSIdataFrameTenMinute <- c()
for (i in 1:IntervalNumbersVideo){
  borneInf <- 1+(i-1)*60
  borneSup <- i * 60
  SSIVectorInterval <- SSIColumn[borneInf:borneSup]
  mean <- mean(SSIVectorInterval, na.rm=TRUE)
  SSIdataFrameTenMinute <- c(SSIdataFrameTenMinute, mean)}
plot(1:length(SSIdataFrameTenMinute), SSIdataFrameTenMinute, type="l", col=rainbow(11)[col], main =
col <- col+1}

```







SSI_fa_mo_pa

