```
#You need to run the following code to calculate synchrony/social entrainment scores


  library("synchrony")
  ## Warning: package 'synchrony' was built under R version 3.2.3
  ## synchrony 0.2.3 loaded.
  library("moments")
  ## Warning: package 'moments' was built under R version 3.2.3
  #Read the data you created from Line 115 of the TraMineR code
  dat<-read.csv("C:\\Users\\tmurase\\Dropbox (GT DELTA Lab)\\ARI MTS Coevolution Grant\\Book Chapter\\Data/Results\\MTSLevel Data20sec.csv")


  #Assign 0 to object run
  run<-0
  #Loop function
  for(mts_i in unique(dat[,"mts_i"])){
  #Get the rows which match mts_i
  mtsdat<-dat[dat[,"mts_i"]==mts_i,]
  #Assign 0 to memrun
  memrun<-0
  #From now, we have to select every pair of members. First, we select the first person and name the object as mem_i
  for(mem_i in unique(dat[,"member_i"])[-1*length(unique(dat[,"member_i"]))]){
   #memrun changes from 1 to 2 to 3 ….
   memrun<-memrun+1
   #If the member is the first person, object members will be assigned mem_i. Otherwise, run Line 21
   if(memrun==1){
     members<-mem_i
    }else{
     members<-c(members,mem_i)
    }
   #Another loop function. From this line, we will select the second person for the pair in which the other person is mem_i
   for(mem_ii in unique(mtsdat[,"member_i"])
      [-1*which(unique(mtsdat[,"member_i"])==members)])
     {
     run<-run+1
     #Selecting time-series data for the first person of the pair
     t1<-as.numeric(mtsdat[mtsdat[,"member_i"]==mem_i,-1:-2])
     #Selectin time-series data for the second person of the pair
     t2<-as.numeric(mtsdat[mtsdat[,"member_i"]==mem_ii,-1:-2])
     #Check how many unique actions the first time-series data contains
     div_i<-length(unique(t1))
     #Check how many unique actions the second time-series data contains
     div_ii<-length(unique(t2))
     #If both the data sets contain at least 3 action elements, calculate synchrony scores.
     if((div_i>=3)|(div_ii>=3)){
      sync.maxs<-phase.sync(t1,t2,mins=TRUE)
      k<-0
      s<-NA
      ave<-NA
      sds<-NA
```

```
            #The kurtosis function calculates the degree of peakedness from the distribution of cycle
            differences
37          k<-kurtosis(sync.maxs$deltaphase$mod_phase_diff_2pi,na.rm=TRUE)
            #The skewness function calculate how skewed the distribution is.
38          s<-skewness(sync.maxs$deltaphase$mod_phase_diff_2pi,na.rm=TRUE)
            #In addition, we calculate the average and standard deviation scores.
39          ave<-mean(sync.maxs$deltaphase$mod_phase_diff_2pi,na.rm=TRUE)
40          sds<-sd(sync.maxs$deltaphase$mod_phase_diff_2pi,na.rm=TRUE)
41        }else{
42          k<-NA
43          s<-NA
44          ave<-NA
45          sds<-NA
46        }
47        if(run==1){
48          finaldat<-cbind(mts_i,mem_i,mem_ii,div_i,div_ii,k,s,ave,sds)
49        }else{
50          finaldat<-rbind(finaldat,cbind(mts_i,mem_i,mem_ii,div_i,div_ii,k,s,ave,sds))
51        }
52      }
53    }
54  } #At the end, you have to save data finaldat in a specific file location.
```