Package: E4tools (via r-universe)

August 22, 2024

Type Package

Title Management and Processing Tools for Data Produced by the Empatica E4

Version 0.1.1

Description Process and manage the data from the Empatica E4. All functions operate on the EDA data stream, but other streams will be added soon. The Empatica E4 is a wearable physiological monitor made by Empatica (Empatica is not associated with any of this code). You can find more information about the E4 at Empatica's website <https://www.empatica.com/research/e4/>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports BBmisc, DataCombine, signal, stats, utils, anytime, chron, data.table, ggplot2, scales, accelerometry, hms, doParallel, doSNOW, parallel, foreach

RoxygenNote 6.1.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Evan Kleiman [aut, cre]

Maintainer Evan Kleiman <evan.kleiman@rutgers.edu>

Date/Publication 2019-07-07 13:50:03 UTC

Repository https://ekleiman1.r-universe.dev

RemoteUrl https://github.com/cran/E4tools

RemoteRef HEAD

RemoteSha 6f76df8e65158dee3cf495c47f9d805b3676834c

Contents

`E4.Acc_Process.part1.ExtractRawAcc`

Acc Processing Part 1: Extract raw acceleromter data

Description

This allows you extract acceleromter data. It will output raw acceleromter data (x,y,z). Inputs are: (1) List of participant numbers and (2) location where ZIP folders are stored. Outputs are: (1) one RDS file per participant with all data. A working example and vignette will be added later.

Usage

```
E4.Acc_Process.part1.ExtractRawAcc(participant_list, ziplocation,
  rdslocation.acc)
```

Arguments

`participant_list`
: list of participant numbers NOTE: This should match the names of the folders (e.g., participant 1001's data should be in a folder called "1001")

`ziplocation`
: folder location where the participant-level subfolders are (make sure that it ends in /)

`rdslocation.acc`
: folder location where you want the RDS outputs to go (make sure that it ends in /)

Examples

```
E4.Acc_Process.part1.ExtractRawAcc(
  participant_list=c(1001),
  ziplocation=paste(system.file(package="E4tools"),
  "/extdata/E4_demo_data/",sep=""),
  rdslocation.acc=paste(tempdir(),"/extdata/output/raw_acc/",sep=""))
```

E4.Acc_Process.part2.Filter_ConvertAcc

Accelerometer Processing Part 2: Extract and filter accelerometer data This function will allow you to filter acceleromter data (based on the EDA signal) and add metrics like g and the normalized Euclidian distance from origin vector.

Description

Accelerometer Processing Part 2: Extract and filter accelerometer data This function will allow you to filter acceleromter data (based on the EDA signal) and add metrics like g and the normalized Euclidian distance from origin vector.

Usage

```
E4.Acc_Process.part2.Filter_ConvertAcc(participant_list, rdslocation.EDA,
  rdslocation.acc, rdslocation.acc_filtered)
```

Arguments

`participant_list`
: list of participant numbers NOTE: This should match the names of the folders (e.g., participant 1001's data should be in a folder called "1001")

`rdslocation.EDA`
: folder location where the RDS files from the first step of the EDA processing are (make sure that it ends in /)

`rdslocation.acc`
: folder location where the RDS files from the first step of the accelerometer processing are

`rdslocation.acc_filtered`
: folder location where you want the filtered acc files to go.

Examples

```
E4.Acc_Process.part2.Filter_ConvertAcc(participant_list=c(1001),
                                   rdslocation.EDA=paste(system.file(package="E4tools"),
                                       "/extdata/output/raw_EDA/",sep=""),
                                   rdslocation.acc=paste(system.file(package="E4tools"),
                                       "/extdata/output/raw_acc/",sep=""),
                                       rdslocation.acc_filtered=paste(tempdir(),
                                       "/extdata/output/filtered_acc/",sep=""))
```

`E4.Diagnostics.EDAplot`

Diagnostics: Plot EDA data and button presses

Description

This will allow you to see all binned EDA data for a participant, along with which band they were wearing and when they pressed the event marker. One PDF file is made per participant. You must run E4.extras.BinEDA() first to prepare for this step.

Usage

```
E4.Diagnostics.EDAplot(participant_list, rdslocation.binnedEDA,
  rdslocation.buttonpress, plotlocation.EDA, RejectFlagCount = 48,
  Plot_E4s = TRUE, display_plot = FALSE)
```

Arguments

`participant_list`
list of participant numbers NOTE: This should match the names of the folders (e.g., participant 1001's data should be in a folder called "1001").

`rdslocation.binnedEDA`
folder location where binned EDA is stored (from E4.extras.BinEDA function).

`rdslocation.buttonpress`
location of folder where button press output is stored (from part extract raw EDA part 2). Set to FALSE if you do not want to plot the button presses.

`plotlocation.EDA`
Folder where you want to store the PDF plots. Set this to FALSE if you do not want to save the PDF output. You should only set to false if you are displaying the plot instead, and thus should also set display_plot to TRUE.

`RejectFlagCount`
What percent of samples in the bin must be bad for the entire bin to be marked bad? Default is 48, which is 10 percent of samples in a 2-minute bin.

`Plot_E4s` Do you want a line at the bottom of the plot showing which E4 the participant was wearing?

`display_plot` Do you want the plot to be displayed on screen in addition to saving the PDF file? Defaults to false. This is most useful if you are only looking at one participant's data.

Examples

```
E4.Diagnostics.EDAplot(participant_list=c(1001),
                       rdslocation.buttonpress=FALSE,
                       rdslocation.binnedEDA=paste(system.file(package=”E4tools”),
                       ”/extdata/plots/”,sep=””),
                       plotlocation.EDA=FALSE,display_plot=TRUE)
```

E4.Diagnostics.tempplot

Diagnostics: Plot Temperature data and button presses

Description

This will allow you to see all binned temperature data for a participant, along with which band they were wearing and when they pressed the event marker. One PDF file is made per participant. You must run E4.extras.BinEDA() first to prepare for this step.

Usage

```
E4.Diagnostics.tempplot(participant_list, rdslocation.binnedtemp,
  rdslocation.buttonpress, plotlocation.temp, Plot_E4s = TRUE,
  TempType = "C")
```

Arguments

`participant_list`
: list of participant numbers NOTE: This should match the names of the folders (e.g., participant 1001's data should be in a folder called "1001")

`rdslocation.binnedtemp`
: folder location where raw temperature data are stored

`rdslocation.buttonpress`
: location of folder where button press output is stored (from EDA part 2)

`plotlocation.temp`
: folder where you want PDF outputs to go.

`Plot_E4s`
: Do you want a line at the bottom of the plot showing which E4 the participant was wearing?

`TempType`
: Do you want Farenheit (TempType=F) or Celcius (TempType=C, default). If you did not elect to include Farenheit in the first temperature step, this step will calcuate it for you.

Examples

```
## Not run: E4.Diagnostics.EDAplot(participant_list=c(1001:1004),
rdslocation.buttonpress="~/study/data/tags/",
rdslocation.binnedtemp="~/study/data/Binned_EDA/",
plotlocation.EDA="~/study/data/EDAplots/")
## End(Not run)
```

E4.extras.BinEDA *Extras: Make EDA bins*

Description

Put EDA data in bins of X minutes length

Usage

```
E4.extras.BinEDA(participant_list, rdslocation.EDA, rdslocation.binnedEDA,
  BinLengthMin, RejectFlag = TRUE)
```

Arguments

`participant_list`
: list of participant numbers NOTE: This should match the names of the folders (e.g., participant 1001's data should be in a folder called "1001")

`rdslocation.EDA`
: folder location where raw EDA (from part 1) is saved.

`rdslocation.binnedEDA`
: folder location where you want the RDS outputs to go (make sure that it ends in /)

`BinLengthMin`
: folder location where you want the RDS outputs to go (make sure that it ends in /)

`RejectFlag`
: Did you include in step 1 the option to keep the flag that shows which data the high and low pass filters rejected (By default, these are included in step 1) AND do you want to include a summary in this dataset of how many samples in a bin were rejected? If you want to run the diagnostic steps, you must keep this. Defaults to TRUE.

Examples

```
E4.extras.BinEDA(participant_list=c(1001),
rdslocation.EDA=paste(system.file(package="E4tools"),"/extdata/output/raw_EDA/",sep=""),
rdslocation.binnedEDA=paste(tempdir(),"/extdata/output/binned_EDA/",sep=""),
BinLengthMin=2,RejectFlag=TRUE)
```

E4.extras.ButtonPressessPerDay *EDA Extra Processing: Get number of button presses per participant, per day from the combined "button_pressess.RDS" file*

Description

This function allows you extract button pressess per participant, per day. It will output a data frame (not an RDS file) that you can use for analyses. You must first extract button pressess using the E4_EDA_Process.part2.ExtractButtonPresses() function.

Usage

```
E4.extras.ButtonPressessPerDay(rdslocation.buttonpress,
  ImputeNAs = FALSE, ImputeZeros = FALSE)
```

Arguments

`rdslocation.buttonpress`
: location of folder where button press output is stored (the file is called "button_presses.RDS"). This should end in / .

`ImputeNAs`
: This will create NAs for any days between the first and last day of study data for each participant. If no data = no presses (which is likely the case, use the "ImputeZeros" option to make them zeros instead).

`ImputeZeros`
: Do you want to make the NAs for days without data zeros instead of NA?

Value

Dataframe with a three columns: ID, date, number of button pressess.

Examples

```
Presses_Per_Day<-
  E4.extras.ButtonPressessPerDay(rdslocation.buttonpress=
                                   paste(system.file(package="E4tools"),
                                   "/extdata/output/presses/",sep=""),
                                             ImputeNAs=TRUE,ImputeZeros=TRUE)
Presses_Per_Day
```

`E4.GGIR.Export` *GGIR Export*

Description

This function will allow you to export a CSV file that is compatible with GGIR. It will create one CSV (not RDS like other parts of E4Tools) per participant. The CSV file will contain a header compatiable with GGIR, the information in the header is: Header includes: 1. Participant ID, 2. Number of E4s used in the data file, 3. Time stamp type (unix, in miliseconds), 4. Time zone (using format that GGIR uses), 5. ACC sampling rate, 6. ACC dynamic range (in ±g), 7. ACC resolution (in bits), 8. Temp sampling rate, 9. Temp units, 10. Temp range min, 11. Temp range max, 12. Temp resolution The columns in the output file are: 1. Timestamp, 2. E4 Serial, 3. Raw ACC X in bits, 4. Raw ACC Y in bits, 5. Raw ACC Z in bits.

Usage

```
E4.GGIR.Export(participant_list, ziplocation, csvlocation.GGIRout, tz)
```

Arguments

`participant_list`
: list of participant numbers NOTE: This should match the names of the folders (e.g., participant 1001's data should be in a folder called "1001")

`ziplocation`
: folder location where the participant-level subfolders are (make sure that it ends in /). Enter ziplocation=ziplocation to use the prespecified folder structure from E4.Prep.FileHelper

`csvlocation.GGIRout`
: folder location where you want the CSV outputs to go (make sure that it ends in /). Enter csvlocation.GGIRout=csvlocation.GGIRout to use the prespecified folder structure from E4.Prep.FileHelper.

`tz`
: timezone where these data were collected (see https://en.wikipedia.org/wiki/List_of_tz_database_time_zo

Examples

```
## Not run: E4.Acc_Process.Part1.ExtractRawAcc(participant_list=c(1001:1002),
ziplocation="~/documents/study/data/",
csvlocation.GGIRout="~/documents/study/data/acc/")
## End(Not run)
```

`E4.Step0.FileHelper` *Set global file locations to make other functions easier*

Description

This function will allow you to pre-define file locations that are used in multiple functions so you only have to type them once and so that your folder structure will be well-organized.

Usage

```
E4.Step0.FileHelper(participant_list, ziplocation, dataroot)
```

Arguments

`participant_list`
: list of participant numbers NOTE: This should match the names of the folders (e.g., participant 1001's data should be in a folder called "1001")

`ziplocation`
: folder location where the participant-level subfolders are (make sure that it ends in /)

`dataroot`
: folder where you want your data to be stored.

Examples

```
## Not run: E4.Acc_Process.Part1.ExtractRawAcc(participant_list=c(1001:1002),
ziplocation="~/documents/study/data/",
rdslocation.acc="~/documents/study/data/acc/")
## End(Not run)
```

`E4.Temp.part1.extract_raw_temp` *Temperature Processing Part 1: Extract raw temperature data*

Description

Extract raw temperatuer data. Inputs are: (1) List of participant numbers and (2) location where ZIP folders are stored. Outputs are: (1) one RDS file per participant with all data. A working example and vignette will be added later.

Usage

```
E4.Temp.part1.extract_raw_temp(participant_list, ziplocation,
  rdslocation.temp, IncludeFarenheit = TRUE)
```

Arguments

`participant_list`
: list of participant numbers NOTE: This should match the names of the folders (e.g., participant 1001's data should be in a folder called "1001")

`ziplocation`
: folder location where the participant-level subfolders are (make sure that it ends in /)

`rdslocation.temp`
: folder location where you want the RDS outputs to go (make sure that it ends in /)

`IncludeFarenheit`
: do you want to include a column with temperature in Farenheit also? Defaults to true. Celcius, which is recorded by the E4, will always be included.

Examples

```
## Not run: E4.Temp.part1.extract_raw_temp(participant_list=c(1001:1002),
ziplocation="~/documents/study/data/",
rdslocation.temp="~/documents/study/data/TEMP/")
## End(Not run)
```

`E4.Temp.part2.bin_temp`

Temperature part 2: Make temperature bins

Description

Put temperature data in bins of X minutes length

Usage

```
E4.Temp.part2.bin_temp(participant_list, rdslocation.temp,
  rdslocation.binnedtemp, BinLengthMin)
```

Arguments

`participant_list`
list of participant numbers NOTE: This should match the names of the folders (e.g., participant 1001's data should be in a folder called "1001")

`rdslocation.temp`
folder location where raw temperature (from part 1) is saved.

`rdslocation.binnedtemp`
folder location where you want the RDS outputs to go (make sure that it ends in /)

`BinLengthMin`
folder location where you want the RDS outputs to go (make sure that it ends in /)

Examples

```
## Not run: E4.extras.BinEDA(participant_list=c(1001:1004),rdslocation.binnedtemp="~/study/data/EDA/",
rdslocation.binnedtemp="~/study/data/Binned_EDA/",
BinLengthMin=2,
RejectFlag=TRUE)
## End(Not run)
```

`E4_EDA_Process.part1.ExtractRawEDA`

EDA Processing Part 1: Extract and filter EDA data

Description

This function allows you extract and filter EDA data. It will output raw data, filtered data (using user-specified high and low pass filters + a butterworth filter), and filtered + feature-scaled ([0,1]) data. It will also provide summary data at the participant and session level. Inputs are: (1) List of participant numbers and (2) location where ZIP folders are stored. Outputs are: (1) one RDS file per participant with all data, (2) summary file that gives participant-level meta-data.

Usage

```
E4_EDA_Process.part1.ExtractRawEDA(participant_list, ziplocation,
  rdslocation.EDA, summarylocation, EDA_low_cut = 0, LowPctCutoff = 1,
  EDA_high_cut = 1000, HighPctCutoff = 1, KeepRejectFlag = TRUE,
  UseMultiCore = FALSE)
```

Arguments

`participant_list`
: list of participant numbers NOTE: This should match the names of the folders (e.g., participant 1001's data should be in a folder called "1001")

`ziplocation`
: folder location where the participant-level subfolders are (make sure that it ends in /)

`rdslocation.EDA`
: folder location where you want the RDS outputs to go (make sure that it ends in /)

`summarylocation`
: folder location where you want participant level summaries to be saved.

`EDA_low_cut`
: This is a HIGH PASS filter. What EDA value (in microsiemens) should be used as the minimum cutoff (0 = cuts off samples that have 0us)

`LowPctCutoff`
: what percentage of samples in a five-second block must contain the low cutoff in order to exclude that block? (e.g., if .5, there must be at least 50 percent of the samples below the low-cut value to exclude the 5-sec block)

`EDA_high_cut`
: This is a LOW PASS filter. What EDA value (in microsiemens) should be used as the maximum cutoff (100 = cuts off samples above 100us)

`HighPctCutoff`
: what percentage of samples in a five-second block must contain the high cutoff in order to exclude that block?

`KeepRejectFlag`
: Do you want to keep the flag that shows which data the high and low pass filters rejected? If you want to run the diagnostic steps, you must keep this. Defaults to TRUE.

`UseMultiCore`
: Do you want to use more than one core for processing? Defaults to FALSE.

Examples

```
E4_EDA_Process.part1.ExtractRawEDA(participant_list=c(1001:1003),
ziplocation=paste(system.file(package="E4tools"),"/extdata/E4_demo_data/",sep=""),
rdslocation.EDA=paste(tempdir(),"/extdata/output/raw_EDA/",sep=""),
summarylocation=paste(tempdir(),"/extdata/output/summaries/",sep=""),
EDA_low_cut=0.001,LowPctCutoff=.75,
EDA_high_cut=25,HighPctCutoff=.75)
```

E4_EDA_Process.part2.ExtractButtonPresses

EDA Processing Part 2: Extract button presses

Description

This function allows you extract button presses and remove presses that are within a certain number of minutes before the end of a session or that are too close to another button press. If the participant has not pressed the button at all, it will give you a warning and continue with the other participants.

Usage

```
E4_EDA_Process.part2.ExtractButtonPresses(participant_list, ziplocation,
  rdslocation.buttonpress, summarylocation, cutoff.ends = 0,
  cutoff.overlap = 0)
```

Arguments

`participant_list`
: list of participant numbers NOTE: This should match the names of the folders (e.g., participant 1001's data should be in a folder called "1001")

`ziplocation`
: folder location where the participant-level subfolders are (make sure that it ends in /)

`rdslocation.buttonpress`
: folder location where you want the RDS output to go (make sure that it ends in /). The file will be named "button_presses.RDS"

`summarylocation`
: location of folder where summaries from part 1 were saved (make sure that it ends in /)

`cutoff.ends`
: how close (in minutes) to the ends of a file do you want to cut off button presses (because they could be accidental e.g., when turning the band off). Default is 0, which will not remove button presses at all.

`cutoff.overlap`
: if you want to remove button presses within X number of minutes, enter that value here. Default is 0, which will not remove button presses at all.

Examples

```
E4_EDA_Process.part2.ExtractButtonPresses(participant_list=c(1001:1002),
                                   ziplocation=paste(system.file(package="E4tools"),
                                     "/extdata/E4_demo_data/",sep=""),
                                     rdslocation.buttonpress=paste(tempdir(),
                                     "/extdata/output/presses/",sep=""),
                               summarylocation=paste(system.file(package="E4tools"),
                                     "/extdata/output/summaries/",sep=""),
                                     cutoff.ends=2, cutoff.overlap=20)
```

`E4_EDA_Process.part3.MatchPressesToEDA` *Match EDA data to button pressess*

Description

This function allows you to extract the data that are within X minutes before and/or after a button press. If there are no button pressess for a participant, it will issue a warning and continue with the next participant. Inputs: (1) List of participant numbers, (2) location of individual EDA files from step 1, (3) location of button presses from step 2. Outputs: (1) RDS file with EDA data before and/or after button presses (and control data), for each participant and combined.

Usage

```
E4_EDA_Process.part3.MatchPressesToEDA(participant_list,
  rdslocation.MatchedEDA, rdslocation.EDA, rdslocation.buttonpress,
  min.before, min.after, control = TRUE)
```

Arguments

`participant_list`
: list of participant numbers NOTE: This should match the names of the folders (e.g., participant 1001's data should be in a folder called "1001")

`rdslocation.MatchedEDA`
: folder location where you want the RDS outputs to go (make sure that it ends in /). The combined data file will go into this directory. Individual participants' data will go into a subdirectory in this folder called "individual_participants"

`rdslocation.EDA`
: folder where rds files for individual Ps' EDA data are stored (from part 1)

`rdslocation.buttonpress`
: location of folder where button press output is stored (from part 2)

`min.before`
: how many minutes before a button press do you want EDA data? Enter 0 if you do not want ANY data before (i.e., you're using only data post-press)

`min.after`
: how many minutes after a button press do you want EDA data? Enter 0 if you do not want ANY data after (i.e., you're using only data pre-press)

`control`
: add in control cases, defaults to T (default is to specify controls from exactly 24 hours prior to the press, provided there was not a press then too)

Examples

```
E4_EDA_Process.part3.MatchPressesToEDA(participant_list=c(1001),
                        rdslocation.buttonpress=paste(system.file(package="E4tools"),
                                  "/extdata/output/presses/",sep=""),
                                  rdslocation.MatchedEDA=paste(tempdir(),
                                  "/extdata/output/matched_EDA/",sep=""),
                               rdslocation.EDA=paste(system.file(package="E4tools"),
                                  "/extdata/output/raw_EDA/",sep=""),
                                  min.before=20,min.after=20,control=TRUE)
```

E4_EDA_Process.part4.BinMatchedEDA

Bin the EDA data matched to button presses

Description

This function allows you to bin the data that has been matched to the button pressess (from step 3).

Usage

```
E4_EDA_Process.part4.BinMatchedEDA(participant_list,
  rdslocation.MatchedEDA, rdslocation.BinnedMatchedEDA, min.after,
  min.before, control = FALSE)
```

Arguments

`participant_list`
: list of participant numbers NOTE: This should match the names of the folders (e.g., participant 1001's data should be in a folder called "1001")

`rdslocation.MatchedEDA`
: folder location of the combined EDA file from step 3. (The file is called EDA_presses_COMBINED.RDS)

`rdslocation.BinnedMatchedEDA`
: location of folder where you want the binned data to be stored

`min.after`
: how many minutes after a button press do you want EDA data? Enter 0 if you do not want ANY data after (i.e., you're using only data pre-press). This should match what you entered in step 3!

`min.before`
: how many minutes before a button press do you want EDA data? Enter 0 if you do not want ANY data before (i.e., you're using only data post-press). This should match what you entered in step 3!

`control`
: does this dataset include control cases? This should match what you did in step 3.

Examples

```
E4_EDA_Process.part4.BinMatchedEDA(participant_list=c(1001:1002),
                            rdslocation.MatchedEDA=paste(system.file(package="E4tools"),
                                  "/extdata/output/matched_EDA/",sep=""),
                                  rdslocation.BinnedMatchedEDA=
                          paste(tempdir(),"/extdata/output/binned_matched_EDA/",sep=""),
                                  min.after = 20,min.before = 20,control=TRUE)
```

Index