Package ‘pawacc’

February 20, 2017

Type Package
Title Physical Activity with Accelerometers
Version 1.2.2
Date 2017-02-20
Author Marco Geraci
Maintainer Marco Geraci <geraci@mailbox.sc.edu>
Depends R (>= 3.0.0)
Imports stats, utils, graphics, SparseM
Description This is a collection of functions to process, format and store accelerometer data.
License GPL (>= 2)
URL https://CRAN.R-project.org/package=pawacc
LazyLoad yes
NeedsCompilation no
Repository CRAN
Date/Publication 2017-02-20 21:23:08

R topics documented:

pawacc-package ......................................................... 2
aggAccFile .............................................................. 2
collapse.accfile ....................................................... 4
dateSummary ............................................................ 7
errorChk ................................................................. 9
fun.collapse ............................................................ 10
gt1mAccDir ............................................................. 11
gt1mAccFile ........................................................... 12
gt1m_sample ........................................................... 13
gt3xAccDir ............................................................. 14
gt3xAccFile ............................................................ 15
gt3x_sample ........................................................... 16
markbouts.acclist ..................................................... 17
Description

This package provides processing and summary functions.

Details

<table>
<thead>
<tr>
<th>Package:</th>
<th>pawacc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>Version:</td>
<td>1.2.2</td>
</tr>
<tr>
<td>Date:</td>
<td>2017-02-20</td>
</tr>
<tr>
<td>License:</td>
<td>GPL (&gt;= 2)</td>
</tr>
<tr>
<td>LazyLoad:</td>
<td>yes</td>
</tr>
</tbody>
</table>

Author(s)

Marco Geraci
Maintainer: Marco Geraci <m.geraci@ucl.ac.uk>

References


Description

This function aggregates count values
aggAccFile

Usage

aggAccFile(object, by, which = "counts",
 x = NULL, keep.error = FALSE)

Arguments

object an object of class accfile.
by epoch by which count and steps are aggregated. Note: it cannot be less than the
accelerometer epoch (object$info$epoch).
which either 'counts' or 'steps' for gt1m files or one of c('x','y','z','steps') for
gt3x files.
x optional argument. If NULL, this is set to counts.
keep.error logical flag. Should errors be omitted?

Value

outcome aggregated values
ts_agg time stamping

Author(s)

Marco Geraci

References

on accelerometry data processing in the Millennium Cohort Study. London, UK: University College
London. Available at http://discovery.ucl.ac.uk/1361699

See Also

collapse.accfile

Examples

data(gt1m_sample)

# aggregate by 30-second epochs
aggAccFile(gt1m_sample, by = 30)

# aggregate by 5-minute epochs
aggAccFile(gt1m_sample, by = 300, keep.error = FALSE)
collapse.accfile  
**Collapse raw accelerometer files into a summary dataset**

**Description**

This function collapses accelerometer data into a dataframe with summary statistics.

**Usage**

```r
collapse(...)  
# S3 method for class 'accfile'  
collapse(object, which = "counts", palist = list(value = c(0, 100, 1000, 5000, 13000),  
rescale.epoch = 60, labels = NULL, extreme = NULL), mwlist = list(value = 20,  
nz = 0, rescale.epoch = 60), collapse.by = "%Y-%m-%d", collapse.epoch = 60, aggregate.by =  
NULL, FUN.list = list(mean = function(x) mean(x, na.rm = TRUE)),  
keep.extreme = FALSE, keep.error = FALSE, ...)  
```

**Arguments**

- **object**: an object of class `gt1m`.
- **which**: either 'counts' or 'steps' for `gt1m` files or one of c('x','y','z','steps') for `gt3x` files.
- **palist**: list of arguments for `markpa.accfile`.
- **mwlist**: list of arguments for `markwear.accfile`.
- **collapse.by**: dataset aggregation level. See argument format from `strptime` for options and details below.
- **collapse.epoch**: epoch by which time spent in different physical activity modes is summarized. See details.
- **aggregate.by**: pre-collapsing aggregation level for accelerometer values. See argument format from `strptime` for options.
- **FUN.list**: a named list of functions. See `fun.collapse`.
- **keep.extreme**: logical flag. If FALSE (default) extreme values will be replaced by NAs. See `markpa.accfile`.
- **keep.error**: logical flag. If FALSE (default) data errors as identified by `errorchk` will be replaced by NAs.
- **...**: arguments for `dateSummary`.

**Value**

A data frame containing the following variables

- **collapse.by**: aggregation factor
- **fileid**: file identifier
- **...**: named columns according to arguments `FUN` and labels of `palist`
Author(s)
Marco Geraci

References

See Also
markpa.accfile, markwear.accfile, fun.collapse

Examples
## Not run:
data(gt1m_sample)
collapse(gt1m_sample, palist = list(value = c(0, 100, 2000, 4000, 11000)), rescale.epoch = 60, labels = c("sedentary", "light", "moderate", "vigorous", "extreme_values"), extreme = "last"), mwlist = list(value = 20), nz = 0), collapse.by = "XY-%m-%d", collapse.epoch = 60,
FUN.list = list(mean = function(x) round(mean(x, na.rm = TRUE),2)),
keep.extreme = FALSE, keep.error = FALSE)
$outcome
collapse.by fileid outcome.mean sedentary light moderate vigorous
1 2011-12-08 test 117.63 293.75 243.75 29.75 4.00
2 2011-12-09 test 157.83 349.75 143.25 33.50 24.50
3 2011-12-10 test 79.75 468.25 177.25 24.25 6.25
4 2011-12-11 test 57.96 355.50 126.00 8.50 3.25
5 2011-12-12 test 70.05 455.50 157.25 19.50 6.00
6 2011-12-13 test 72.99 475.50 181.50 15.25 8.00
7 2011-12-14 test 79.94 476.25 210.50 20.25 8.50
8 2011-12-15 test 232.50 0.00 0.50 0.00 0.00
9 2011-12-16 test NaN 0.00 0.00 0.00 0.00
extreme_values non-wear
1 0.00 388.75
2 6.00 883.00
3 0.00 763.50
4 0.25 946.50
5 0.00 801.75
6 0.25 759.50
7 1.50 723.00
8 0.00 1439.50
9 0.00 187.50
$call
collapse.accfile(object = gt1m_sample, palist = list(value = c(0, 100, 2000, 4000, 11000)), rescale.epoch = 60, labels = c("sedentary", "light", "moderate", "vigorous", "extreme_values"), extreme = "last"), mwlist = list(value = 20, nz = 0), collapse.by = "
collapse.accfile

collapse.epoch = 60, FUN.list = list(mean = function(x) round(mean(x, na.rm = TRUE), 2)), keep.extreme = FALSE, keep.error = FALSE

attr("class")
[1] "accfile.collapse"
Warning message:
In collapse.accfile(gt1m_sample, palist = list(value = c(0, 100, : 
Nas imputed where extreme counts found

## End(Not run)

## Not run:
collapse(gt1m_sample, palist = list(value = c(0, 100, 2000, 4000, 11000), 
rescale.epoch = 60, labels = c("sedentary", "light", "moderate", "vigorous", "extreme_values"), 
exreme = "last"), mwlist = list(value = 20, 
nz = 0), collapse.by = "XY-%m-%d", collapse.epoch = 60, 
FUN.list = list(mean = function(x) round(mean(x, na.rm = TRUE),2), 
.sd = function(x) round(sd(x, na.rm = TRUE),2), 
"95th" = function(x) round(quantile(x, probs = .95, na.rm = TRUE),2)), 
keep.extreme = TRUE, keep.error = FALSE)

$soutcome

collapse.by fileid outcome.mean outcome.sd outcome.95th sedentary light
1 2011-12-08 test 117.63 216.12 529.40 293.75 243.75
2 2011-12-09 test 201.10 567.65 1085.60 349.75 143.25
3 2011-12-10 test 81.97 221.33 465.50 468.25 177.25
4 2011-12-11 test 59.80 172.08 320.35 355.50 126.00
5 2011-12-12 test 70.05 188.49 401.00 455.50 157.25
6 2011-12-13 test 74.08 207.55 386.85 475.50 181.50
7 2011-12-14 test 87.42 275.27 415.30 476.25 210.50
8 2011-12-15 test 232.50 222.74 374.25 0.00 0.50
9 2011-12-16 test NaN NA NaN 0.00 0.00

$sall

collapse.accfile(object = gt1m_sample, palist = list(value = c(0, 
100, 2000, 4000, 11000), rescale.epoch = 60, labels = c("sedentary", 
"light", "moderate", "vigorous", "extreme_values"), extreme = "last"), 
mwlist = list(value = 20, nz = 0), collapse.by = " 
collapse.epoch = 60, FUN.list = list(mean = function(x) round(mean(x, 
na.rm = TRUE), 2), sd = function(x) round(sd(x, na.rm = TRUE), 
2), "95th" = function(x) round(quantile(x, probs = .95, 
na.rm = TRUE), 2)), keep.extreme = TRUE, keep.error = FALSE)
**dateSummary**

```r
attr("class")
[1] "accfile.collapse"
```

```r
## End(Not run)
```

dateSummary  

*Date summary for accelerometer files*

**Description**

This function provides a date summary for Actigraph GT1M accelerometer files.

**Usage**

```r
dateSummary(object, wear, timestamp, minval = 0,
             rescale.epoch = 60, keep.error = FALSE)
```

**Arguments**

- `object`: an object of class `gt1m`.
- `wear`: a vector that classifies wear and non-wear time. See `markwear.accfile`.
- `timestamp`: a timestamp vector for accelerometer values that can be provided by `tsformat`.
- `minval`: threshold defining the minimum number of minutes to identify first and last days. See details.
- `rescale.epoch`: epoch expressed in the same unit as acceleromenter’s epoch to determine minutes of wear time (default is 60 and assumed to be in seconds).
- `keep.error`: logical flag. If FALSE (default) data errors as identified by `errorChk` will be replaced by NAs.

**Details**

Based on total wear time (in minutes) for each day, the threshold `minval` is applied to identify the first and last days. For example, if accelerometers are sent by post to collect survey data, the first and last days in which the accelerometer was worn might not be known. Days before the first and those after the last day (truncated days) are discarded by `collapse.accfile`.

**Value**

A data frame containing the following variables

- `fileid`  
  file identifier
- `days`  
  dates by calendar day
- `freq`  
  frequency of accelerometer observations in each day
- `hour_day`  
  total hours of accelerometer observations in each day
start_day  starting time of accelerometer observations
end  end time of accelerometer observations
valid_mins  wear time (minutes)
IsStartDate  dummy variable to define starting date (1 = yes)
IsEndDate  dummy variable to define end date (1 = yes)
IsTruncated  dummy variable to define truncated date (1 = yes)

Author(s)

Marco Geraci

References


See Also
collapse.accfile

Examples

data(gt1m_sample)

mw <- markwear.accfile(gt1m_sample, 20)
tsf <- tsformat(gt1m_sample)
dateSummary(gt1m_sample, mw, tsf)

## Not run:

fileid  days  freq  hour_day  start_day  end_day  valid_mins  IsStartDate
1  test  2011-12-08  3840  16.000  08  23  571.25  1
2  test  2011-12-09  5760  24.000  00  23  557.00  0
3  test  2011-12-10  5760  24.000  00  23  676.50  0
4  test  2011-12-11  5760  24.000  00  23  493.50  0
5  test  2011-12-12  5760  24.000  00  23  638.25  0
6  test  2011-12-13  5760  24.000  00  23  680.50  0
7  test  2011-12-14  5760  24.000  00  23  717.00  0
8  test  2011-12-15  5760  24.000  00  23  0.50  0
9  test  2011-12-16  750  3.125  00  03  0.00  0

IsEndDate  IsTruncated
1  0  0
2  0  0
3  0  0
4  0  0
5  0  0
6  0  0
7  0  0
8  0  0
9  1  0
errorChk

## End (Not run)

# at least 600 minutes per day to determine first and last day
dateSummary(gtlm_sample, mw, tsf, minval = 600)

## Not run:

<table>
<thead>
<tr>
<th>fileid</th>
<th>days</th>
<th>freq</th>
<th>hour_day</th>
<th>start_day</th>
<th>end_day</th>
<th>valid_mins</th>
<th>IsStartDate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>test</td>
<td>2011-12-08</td>
<td>3840</td>
<td>16.000</td>
<td>08</td>
<td>23</td>
<td>571.25</td>
</tr>
<tr>
<td>2</td>
<td>test</td>
<td>2011-12-09</td>
<td>5760</td>
<td>24.000</td>
<td>00</td>
<td>23</td>
<td>557.00</td>
</tr>
<tr>
<td>3</td>
<td>test</td>
<td>2011-12-10</td>
<td>5760</td>
<td>24.000</td>
<td>00</td>
<td>23</td>
<td>676.50</td>
</tr>
<tr>
<td>4</td>
<td>test</td>
<td>2011-12-11</td>
<td>5760</td>
<td>24.000</td>
<td>00</td>
<td>23</td>
<td>493.50</td>
</tr>
<tr>
<td>5</td>
<td>test</td>
<td>2011-12-12</td>
<td>5760</td>
<td>24.000</td>
<td>00</td>
<td>23</td>
<td>638.25</td>
</tr>
<tr>
<td>6</td>
<td>test</td>
<td>2011-12-13</td>
<td>5760</td>
<td>24.000</td>
<td>00</td>
<td>23</td>
<td>680.50</td>
</tr>
<tr>
<td>7</td>
<td>test</td>
<td>2011-12-14</td>
<td>5760</td>
<td>24.000</td>
<td>00</td>
<td>23</td>
<td>717.00</td>
</tr>
<tr>
<td>8</td>
<td>test</td>
<td>2011-12-15</td>
<td>5760</td>
<td>24.000</td>
<td>00</td>
<td>23</td>
<td>0.50</td>
</tr>
<tr>
<td>9</td>
<td>test</td>
<td>2011-12-16</td>
<td>750</td>
<td>3.125</td>
<td>00</td>
<td>03</td>
<td>0.00</td>
</tr>
</tbody>
</table>

IsEndDate IsTruncated
| 1      | 0    | 1   |
| 2      | 0    | 1   |
| 3      | 0    | 0   |
| 4      | 0    | 0   |
| 5      | 0    | 0   |
| 6      | 0    | 0   |
| 7      | 1    | 0   |
| 8      | 0    | 1   |
| 9      | 0    | 1   |

## End (Not run)

---

**errorChk**

**Perform error checking**

**Description**

These functions look for errors in the data. A code is returned.

**Usage**

`errorChk(x, fault = 32767)`

**Arguments**

- **x**: vector of accelerometer data.
- **fault**: numerical value that indicates voltage signal saturation (temporarily used for both accelerometer counts and steps).

**Details**

Error coded are as follow: 0, no error; 1, all values are 5-digit values or all one value; 2, negative values; 3, NAs.
fun.collapse

Value

a vector of the same length as x.

Author(s)

Marco Geraci

References


See Also

gt1mAccFile, gt3xAccFile

fun.collapse  Summary statistics

Description

Accessory function for collapsing accelerometer files.

Usage

fun.collapse(x, fun = list(mean = function(x) mean(x, na.rm = TRUE),
               median = function(x) median(x, na.rm = TRUE),
               sd = function(x) sd(x, na.rm = TRUE)))

Arguments

xnumeric vector.

funnamed list of functions to be applied to x.

Value

a list of named values of the same length as fun.

Author(s)

Marco Geraci

References

See Also

collapse.accfile

---

**gt1mAccDir**  
*Read list of accelerometer files*

**Description**

This function reads a list of Actigraph GT1M accelerometer files.

**Usage**

```
gt1mAccDir(accFileList, save, compress = "gzip", compression_level = 6, probar = TRUE)
```

**Arguments**

- `accFileList`: an object of type list.
- `save`: either logical or character. See `readAccDir` for details.
- `compress`: logical or character string specifying whether saving to a named file is to use compression if `save = TRUE`. See argument `compress` in `save`.
- `compression_level`: integer: the level of compression to be used. See argument `compression_level` in `save`.
- `progbar`: logical flag. Should a progress bar be used? Available for Windows only.

**Value**

An object of class `acclist`.

**Author(s)**

Marco Geraci

**References**

Actigraph (Pensacola, Florida).


**See Also**

`readAccDir`
gt1mAccFile  

*Read a single accelerometer file*

**Description**

This function reads an Actigraph GT1M accelerometer file.

**Usage**

```r
gt1mAccFile(file, path, fileid, counts.pos = 1, tz = "Europe/London", sparse = FALSE, fault = 32767)
```

**Arguments**

- `file`  file name including file extension.
- `path`  path to file.
- `fileid`  label for file identifier.
- `counts.pos`  when storage mode allows for accelerometer counts and steps to be recorded at the same time, this argument specifies the position of first measurement of accelerometer counts (default is `counts.pos = 1`).
- `tz`  a character string specifying the timezone to be used for the conversion (see `strptime`).
- `sparse`  logical flag: should data be stored in sparse format?
- `fault`  numerical value that indicates voltage signal saturation.

**Details**

Raw accelerometer data are processed according to the device data format. Several data checks are performed by `errorchk` and `infodate`. An additional check is performed on the length of the sequence of measurements when both accelerometer counts and steps are recorded. If the length is odd, a warning message is produced. See file `gt1m_sample.dat` in directory `\inst\extdata` of this package.

**Value**

These functions return an object of two classes: `accfile` and additional device-specific class (i.e., `gt1m`).

An object of class `accfile` is a list containing the following components:

- `df`  A `data.frame` object with accelerometer values in columns `counts` and `steps` (if present), and coded error for each accelerometer data column. See `errorChk` for error codes. If `sparse = TRUE`, all variables of the data frame `df` are returned as vectors of a matrix in sparse format (see `as.matrix.csr` for details).
gt1m_sample

info

A data.frame object with file identifier (fileid), device serial number (serial), number of recorded measurements (nobs), epoch (epoch), accelerometer mode (mode), start date and time (ts_start), time zone (tz), battery voltage (voltage), download date and time (ts_dl).

error_summary

A list object with file identifier (fileid), summary tables of error codes for each accelerometer data column, error code for date (date), and logical flag for odd number of measurements (odd_number) (see details).

Author(s)

Marco Geraci

References

Actigraph (Pensacola, Florida).


See Also

readAccDir, gt1mAccDir

Examples

```r
data(gt1m_sample)
class(gt1m_sample)
```

---

gt1m_sample GT1M accelerometer file

Description

This is the output of gt1mAccFile.

Format

See value in gt1mAccFile. The 'raw' file 'gt1m_sample.dat' can be found in the directory '\inst\extdata' of this package.

Source

Actigraph (Pensacola, Florida).
Examples

```r
## Not run:
gt1m_sample <- gtmAccFile("gt1m_sample.dat", path = ",", fileid = "test")
gt1m_sample
fileid serial nobs epoch mode ts_start tz voltage ts_dl
1 test xxxxx 44910 15 1 2011-12-08 08:00:00 GMT 3.76 2011-12-24 11:20:36

## End(Not run)
```

---

gt3xAccDir  

**Read list of accelerometer files**

### Description

This function reads a list of Actigraph GT3X accelerometer files.

### Usage

```r
gt3xAccDir(accFileList, save, compress = "gzip",
compression_level = 6, progbar = TRUE)
```

### Arguments

- **accFileList**  
an object of type list.
- **save**  
either logical or character. See `readAccDir` for details.
- **compress**  
logical or character string specifying whether saving to a named file is to use compression if `save = TRUE`. See argument `compress` in `save`.
- **compression_level**  
integer: the level of compression to be used. See argument `compression_level` in `save`.
- **progbar**  
logical flag. Should a progress bar be used? Available for Windows only.

### Value

An object of class acclist.

### Author(s)

Marco Geraci

### References

Actigraph (Pensacola, Florida).

### See Also

`readAccDir`
gt3xAccFile

Read a single accelerometer file

Description

This function reads Actigraph GT3X and ActiSleep accelerometer files.

Usage

gt3xAccFile(file, path, fileid, tz = "Europe/London",
sparse = FALSE, fault = 32767)

Arguments

file  file name including file extension.
path  path to file.
fileid label for file identifier.
tz    a character string specifying the timezone to be used for the conversion (see strftime).
sparse logical flag: should data be stored in sparse format?
fault numerical value that indicates voltage signal saturation.

Details

Raw accelerometer data are processed according to the device data format. See file ‘gt3x_sample.dat’ in directory ‘inst/extdata’ of this package.

Value

These functions return an object of two classes: accfile and additional device-specific class (i.e., gt3x).

An object of class accfile is a list containing the following components:

df  A data.frame object with accelerometer values in columns y, x, z, and steps (if present), and coded error for each accelerometer data column. See errorChk for error codes. If sparse = TRUE, all variables of the data frame df are returned as vectors of a matrix in sparse format (see as.matrix.csr for details).

info  A data.frame object with file identifier (fileid), device serial number (serial), number of recorded measurements (nobs), epoch (epoch), accelerometer mode (mode), start date and time (ts_start), time zone (tz), battery voltage (voltage), download date and time (ts_dl).

error_summary  A list object with file identifier (fileid), summary tables of error codes for each accelerometer data column, error code for date (date), and logical flag for odd number of measurements (odd_number) (see details).
**Author(s)**
Marco Geraci

**References**
Actigraph (Pensacola, Florida).

**See Also**
readAccDir

**Examples**
```
data(gt3x_sample)
class(gt3x_sample)
```

---

**gt3x_sample**

*GT3X accelerometer file*

**Description**
This is the output of `gt3xAccFile`.

**Format**
See value in `gt3xAccFile`. The 'raw' file 'gt3x_sample.dat' can be found in the directory 'inst/extdata' of this package.

**Source**
Actigraph (Pensacola, Florida).

**Examples**
```
## Not run:
gt3x_sample <- gt3xAccFile("gt3x_sample.dat", path = ", fileid = "test")
gt3x_sample
fileid serial nobs epoch mode ts_start tz voltage ts_dl
1 test xxxxx 2676 1 13 2009-03-03 10:40:00 GMT 4.09 2009-03-03 11:24:49

## End(Not run)
```
markbouts.acclist

Classify accumulation of physical activity in bouts

Description

This function identifies bouts of physical activity using user-defined breakpoints for accelerometer counts.

Usage

markbouts(object, value, which = "counts", bts = c(0, 10, 20, Inf), rescale.epoch = 60, collapse.by = "%Y-%m-%d", value.labels = NULL, bouts.labels = NULL, extreme = NULL, keep.error = FALSE, progbar = TRUE)

## S3 method for class 'acclist'
markbouts(object, value, which = "counts", bts = c(0, 10, 20, Inf), rescale.epoch = 60, collapse.by = "%Y-%m-%d", value.labels = NULL, bouts.labels = NULL, extreme = NULL, keep.error = FALSE, progbar = TRUE)

## S3 method for class 'accfile'
markbouts(object, value, which = "counts", bts = c(0, 10, 20, Inf), rescale.epoch = 60, collapse.by = "%Y-%m-%d", value.labels = NULL, bouts.labels = NULL, extreme = NULL, keep.error = FALSE, progbar = NULL)

Arguments

object an object of class acclist or accfile.
value vector of breaks to define physical activity modes. The lowest breakpoint must be 0.
which either 'counts' or 'steps' for gt1m files or one of c('x', 'y', 'z', 'steps') for gt3x files.
bts vector of breaks to define bouts duration (in minutes).
rescale.epoch epoch expressed in the same unit as accelerometer's epoch to determine multiplier to rescale value (default is 60). See details.
collapse.by dataset aggregation level. See argument format from strptime for options and details below.
value.labels labels for physical activity modes.
bouts.labels labels for bouts duration categories (NULL is recommended).
extreme
keep.error logical flag. If FALSE (default) data errors as identified by errorChk will be replaced by NAs.
progbar logical flag. Should a progress bar be used? Available for Windows only. Argument not used for class accfile.
Details

Breakpoints are specified as counts per seconds using `value`. If the epoch used for `value` is different from the accelerometer's epoch, a rescaling is applied. E.g., if epoch is 15 seconds and breakpoints are expressed as counts per 60 seconds, `value` is divided by \( \frac{60}{15} = 4 \). There can be \( n \) physical activity modes at maximum, where \( n \) is the length of `value`.

The argument `extreme` is NULL by default. Use either ‘last’ to select the last category or the category number 1 to \( n \).

Value

The function `markbouts.acclist` returns duration, frequency and mean duration of bouts by bout category, physical activity mode and \( t \) levels of collapse by (e.g., day) in array of dimension \( c(\text{length(bts)} - 1, \text{length(value)} + 1, 3, t) \). `markbouts.acclist` is applied to objects of class `acclist`, in which case a list of arrays of the same length as the number of accelerometer files in object is returned.

Author(s)

Marco Geraci

See Also

`readAccDir`

markpa.acclist

**Classify mode of physical activity**

Description

This function identifies modes of physical activity using user-defined breakpoints for accelerometer counts.

Usage

```r
markpa(object, value, which = "counts", rescale.epoch = 60, labels = NULL,
        extreme = NULL, keep.error = FALSE, progbar = TRUE)
```

## S3 method for class 'acclist'

```r
markpa(object, value, which = "counts", rescale.epoch = 60, labels = NULL,
        extreme = NULL, keep.error = FALSE, progbar = TRUE)
```

## S3 method for class 'accfile'

```r
markpa(object, value, which = "counts", rescale.epoch = 60, labels = NULL,
        extreme = NULL, keep.error = FALSE, progbar = NULL)
```
Arguments

- **object**: an object of class `acclist` or `accfile`.
- **value**: vector of breaks to define physical activity modes. The lowest breakpoint must be 0.
- **which**: either 'counts' or 'steps' for `gt1m` files or one of `c('x','y','z','steps')` for `gt3x` files.
- **rescale.epoch**: epoch expressed in the same unit as accelerometer epoch to determine multiplier to rescale value (default is 60). See details.
- **labels**: labels for physical activity modes.
- **extreme**: if `value` includes a threshold for defining the category of extreme values, this argument identifies such category among the physical activity modes defined by `value`. See details.
- **keep.error**: logical flag. If `FALSE` (default) data errors as identified by `errorchk` will be replaced by NAs.
- **progbars**: logical flag. Should a progress bar be used? Available for Windows only. Argument not used for class `accfile`.

Details

Breakpoints are specified as counts per seconds using `value`. If the epoch used for `value` is different from the accelerometer epoch, a rescaling is applied. E.g., if epoch is 15 seconds and breakpoints are expressed as counts per 60 seconds, `value` is divided by $60/15 = 4$. There can be $n$ physical activity modes at maximum, where $n$ is the length of `value`.

The argument `extreme` is `NULL` by default. Use either 'last' to select the last category or the category number 1 to $n$.

Value

If `object` is of class `acclist`, a list of factors of the same length as the number of accelerometer files in `object`. If `object` is of class `accfile`, a single factor will be given. The number of levels is equal to `length(value) + 1`.

Author(s)

Marco Geraci

See Also

`readAccDir`
markwear.acclist  

*Classify wear and non-wear time*

**Description**

This functions identifies sequences of zeroes of a given length to classify wear and non-wear time in accelerometer data files.

**Usage**

```r
markwear(object, value, which = "counts", rescale.epoch = 60,
nz = 0, keep.error = FALSE, progbar = TRUE)
## S3 method for class 'acclist'
markwear(object, value, which = "counts", rescale.epoch = 60,
nz = 0, keep.error = FALSE, progbar = TRUE)
## S3 method for class 'accfile'
markwear(object, value, which = "counts", rescale.epoch = 60,
nz = 0, keep.error = FALSE, progbar = NULL)
```

**Arguments**

- **object** an object of class acclist or accfile.
- **value** the length of the time window (in minutes) which contains zero values.
- **which** either 'counts' or 'steps' for gt1m files or one of c('x', 'y', 'z', 'steps') for gt3x files.
- **rescale.epoch** epoch expressed in the same unit as accelerometer epoch to determine multiplier to rescale value (default is 60). See details.
- **nz** the length of the time window (in minutes) of non-zero value sequences allowed between every two sequences of zero values.
- **keep.error** logical flag. If FALSE (default) data errors as identified by errorChk will be replaced by NAs.
- **progbar** logical flag. Should a progress bar be used? Available for Windows only. Argument not used for class accfile.

**Details**

The accelerometer epoch is assumed to be expressed in seconds. Therefore value is automatically rescaled to value * 60/object$info$epoch.

**Value**

If object is of class acclist, a list of factors with two levels of the same length as the number of accelerometers files in object. If object is of class accfile, a single factor will be given. Levels are 'non-wear' and 'wear'.

Author(s)

Marco Geraci

References


See Also

readAccDir

Examples

```r
## Not run:

data(gt1m_sample)

# 20-minute threshold
wear <- markwear(gt1m_sample, value = 20)
> table(wear)
  wear
 Non-wear    Wear  
    27572  17338

## End(Not run)

## Not run:

# lower threshold
wear <- markwear(gt1m_sample, value = 5)
> table(wear)
  wear
 Non-wear    Wear  
      30188  14722

## End(Not run)

## Not run:

# allow for some non-zero values within a 20-minute window
wear <- markwear.accfile(gt1m_sample, value = 20, nz = 2)
> table(wear)
  wear
 Non-wear    Wear  
       28198  16712

## End(Not run)
```
plot.gt1m  

Plot accelerometer file

Description
These functions plot data from Actigraph GT1M and GT3X accelerometer files.

Usage

## S3 method for class 'gt1m'
plot(x, y = NULL, xlab, ylab, main,
keep.error = TRUE, which = "counts", select = 1,...)

## S3 method for class 'gt3x'
plot(x, y = NULL, xlab, ylab, main,
keep.error = TRUE, which = "x", select = 1,...)

Arguments

x
an object of class gt1m. It can be either accfile or acclist.
y
ignored.
xlab
x-axis label (optional).
ylab
y-axis label (optional).
main
main title (optional).
keep.error
logical flag. If FALSE (default) data errors as identified by errorchk will be replaced by NAs.
which
either 'counts' or 'steps' for gt1m files or one of c('x', 'y', 'z', 'steps') for gt3x files.
select
numeric. If class(x) is acclist, this argument specifies the corresponding position of the accelerometer file in the list (first file by default).
...
Arguments to be passed to methods, such as graphical parameters (see par).

Author(s)
Marco Geraci

References
Actigraph (Pensacola, Florida).

See Also
plot, par, gt1mAccFile, gt1mAccDir
Examples

data(gt1m_sample)
data(gt3x_sample)

plot(gt1m_sample, which = "counts")
plot(gt1m_sample, which = "steps")
plot(gt3x_sample, which = "x")
plot(gt3x_sample, which = "steps")

print.acclist

Print an accfile or acclist Object

Description

Print an object generated by gt1mAccFile, gt3xAccFile or readAccDir.

Usage

## S3 method for class 'acclist'
print(x, ...)

## S3 method for class 'accfile'
print(x, ...)

Arguments

x an accfile or an acclist object.

... not used.

Author(s)

Marco Geraci

readAccDir

Read content of a directory with accelerometer files

Description

This function reads raw files downloaded from accelerometers and stores them in one directory.
readAccDir

Usage

```r
readAccDir(path, model, ext = "dat", counts.pos = 1,  
tz = "Europe/London", sparse = FALSE, 
fault = 32767, save = TRUE, compress = "gzip",  
compression_level = 6,...)
```

Arguments

- `path` : path to accelerometer files.
- `model` : accelerometer model, either 'gt1m' or 'gt3x'.
- `ext` : file extension without '.' (default is 'dat').
- `counts.pos` : position of first measurement of accelerometer counts. See `gt1mAccFile` for details.
- `tz` : a character string specifying the timezone to be used for the conversion (see `strptime`).
- `sparse` : logical flag: should data be stored in sparse format?
- `fault` : numerical value that indicates voltage signal saturation (temporarily used for both accelerometer counts and steps).
- `save` : either logical or character. If logical, `save = TRUE` stores accelerometer files as .Rdata objects in a new directory ('accfiles') in `path`. If the directory already exists, confirmation for overwriting will be prompted. `save = FALSE` stores accelerometer files in current R session. Alternatively, an existing folder can be specified.
- `compress` : logical or character string specifying whether saving to a named file is to use compression if `save = TRUE`. See argument `compress` in `save`.
- `compression_level` : integer: the level of compression to be used. See argument `compression_level` in `save`.
- `...` : not used.

Details

This is a wrapper function. It reads raw accelerometer files stored in one directory. The argument `ext` specifies the extension of the files to look for (e.g., 'dat' or 'txt'). Files with different extension and/or sub-directories are ignored. The data format must be consistent with the specification of `model`.

Value

Either a set of .Rdata files or a list of objects of class accfile. In both cases, each accelerometer data file is stored as an object of type list and labelled using the data file name. See `gt1mAccFile` or `gt3xAccFile` for details.

Author(s)

Marco Geraci


**References**

**See Also**
gt1mAccFile, gt3xAccFile, gt1mAccDir, gt3xAccDir

tsFormat

---

**Description**
Time stamping.

**Usage**
```r
tsFormat(object)
```

**Arguments**

- `object` an object of class accfile.

**Value**
a vector of timestamps.

**Author(s)**
Marco Geraci

**See Also**
tsFromEpoch

**Examples**
```r
data(gt1m_sample)
data(gt3x_sample)

tsFormat(gt1m_sample)
tsFormat(gt3x_sample)
```
**tsFromEpoch**

*Calculate timestamp from epoch number or epoch number from timestamp*

**Description**

Utility functions.

**Usage**

```r
tsFromEpoch(object, x)
epochFromTS(object, x)
```

**Arguments**

- `object`: an object of class `accfile`.
- `x`: either an integer giving the epoch number or the timestamp in a POSIX format (e.g., `%Y-%m-%d %H:%M:%S`).

**Value**

either a timestamp corresponding to an epoch number or the epoch number corresponding to a timestamp.

**Author(s)**

Marco Geraci

**See Also**

gt1mAccFile

**Examples**

```r
data(gt1m_sample)

# [1] "2011-12-10 01:39:45 GMT"

tsFromEpoch(gt1m_sample, 10000)
# [1] 2011-12-10 01:39:45 GMT

epochFromTS(gt1m_sample, as.POSIXlt(strptime('2011-12-10 01:39:45', '%Y-%m-%d %H:%M:%S')), 10000)
# [1] 2011-12-10 01:39:45 GMT
```
Index

*Topic Actigraph ActiSleep
  gt3x_sample, 16
  gt3xAccFile, 15
*Topic Actigraph GT1M
  collapse.accfile, 4
  dateSummary, 7
  errorChk, 9
  gt1m_sample, 13
  gt1mAccDir, 11
  gt1mAccFile, 12
  markbouts.acclist, 17
  markpaa.acclist, 18
  markwear.acclist, 20
  plot.gt1m, 22
  readAccDir, 23
  tsFormat, 25
  tsFromEpoch, 26
*Topic Actigraph GT3X
  collapse.accfile, 4
  errorChk, 9
  gt3x_sample, 16
  gt3xAccDir, 14
  gt3xAccFile, 15
  markbouts.acclist, 17
  markpaa.acclist, 18
  markwear.acclist, 20
  readAccDir, 23
  tsFormat, 25
  tsFromEpoch, 26
*Topic accelerometer
  gt1mAccDir, 11
  gt1mAccFile, 12
  gt3xAccDir, 14
  gt3xAccFile, 15
  pawacc-package, 2
  plot.gt1m, 22
  readAccDir, 23
*Topic collapse
  aggAccFile, 2
  collapse.accfile, 4
  fun.collapse, 10
*Topic datasets
  gt1m_sample, 13
  gt3x_sample, 16
*Topic error
  errorChk, 9
*Topic package
  pawacc-package, 2
*Topic physical activity mode
  markbouts.acclist, 17
  markpaa.acclist, 18
*Topic physical activity
  pawacc-package, 2
*Topic plot
  plot.gt1m, 22
*Topic print
  print.acclist, 23
*Topic read file
  gt1mAccFile, 12
  gt3xAccFile, 15
  readAccDir, 23
*Topic summary
  dateSummary, 7
*Topic timestamp
  tsFormat, 25
  tsFromEpoch, 26
*Topic wear time
  markwear.acclist, 20
  aggAccFile, 2
  as.matrix.csr, 12, 15
  class, 12, 15
  collapse(collapse.accfile), 4
  collapse.accfile, 3, 4, 7, 8, 11
  data.frame, 12, 13, 15
  dateSummary, 4, 7
  epochFromTS.tsFromEpoch, 26
errorchk, 4, 7, 9, 12, 15, 17, 19, 20, 22
fun.collapse, 4, 5, 10
gt1m_sample, 13
gt1mAccDir, 11, 13, 22, 25
gt1mAccFile, 10, 12, 13, 22–26
gt3x_sample, 16
gt3xAccDir, 14, 25
gt3xAccFile, 10, 15, 16, 23–25
infoDate, 12
list, 13, 15, 24
markbouts (markbouts.acclist), 17
markbouts.acclist, 17
markpa (markpa.acclist), 18
markpa.acccfile, 4, 5
markpa.acclist, 18
markwear (markwear.acclist), 20
markwear.acccfile, 4, 5, 7
markwear.acclist, 20
par, 22
pawacc (pawacc-package), 2
pawacc-package, 2
plot, 22
plot.gt1m, 22
plot.gt3x (plot.gt1m), 22
print.acccfile (print.acclist), 23
print.acclist, 23
readAccDir, 11, 13, 14, 16, 18, 19, 21, 23, 23
save, 11, 14, 24
strftime, 4, 12, 15, 17, 24
tsFormat, 25
tsFromEpoch, 25, 26