Package 'GGIRread'

October 28, 2026

Type Package

Title Wearable Accelerometer Data File Readers

Version 1.2.0

Date 2026-10-28

Maintainer Vincent T van Hees <v.vanhees@accelting.com>

Description Reads data collected from wearable acceleratometers as used in sleep and physical activity research. Currently supports file formats: binary data from 'GENEActiv' <https://activinsights.com/>, .bin-format from GENEA devices (not for sale), and .cwa-format from 'Axivity' <https://axivity.com>. Further, it has functions for reading text files with epoch level aggregates from 'Actical', 'Fitbit', 'Actiwatch', 'ActiGraph', and 'PhilipsHealthBand'. Primarily designed to complement R package GGIR <https://CRAN.R-project.org/package=GGIR>.

URL https://github.com/wadpac/GGIRread/

BugReports https://github.com/wadpac/GGIRread/issues

License Apache License (== 2.0)

Suggests testthat

Imports matlab, bitops, Rcpp (>= 1.0.10), data.table, readxl, jsonlite, digest

Depends stats, utils, R (>= 3.5.0)

NeedsCompilation yes

LinkingTo Rcpp (>= 1.0.10)

ByteCompile yes

Author Vincent T van Hees [aut, cre],
Patrick Bos [aut] (ORCID: <https://orcid.org/0000-0002-6033-960X>),
Lena Kushleyeva [ctb],
Jing Hua Zhao [ctb],
Evgeny Mirkes [ctb],
Dan Jackson [ctb],
Jairo H Migueles [ctb],
John Muschelli [ctb],
Jia Ying Chua [ctb],

Medical Research Council UK [cph, fnd],
Accelting [cph, fnd]

Repository CRAN

Date/Publication 2026-10-28 11:40:22 UTC

Contents

GGIRread-package 2
GENEActivReader 3
mergeFitbitData 4
mergePHBdata 4
readActicalCount 5
readActiGraphCount 6
readActiwatchCount 7
readAxivity 8
readFitbit 10
readGenea 11
readGENEActiv 12
readParmayMatrix 13
readPHBCount 16
readWav 17
resample 18

Index **19**

`GGIRread-package` *A package to read data files from wearable accelerometers*

Description

This package offers a collection of functions to read data files from wearable accelerometers. Some functions were migrated from R package GGIR to make GGIR more modular and to reduce it's complexity, while other functions such as readGENEActiv (R) and GENEActivReader (C++) are newly written.

Details

Package:	GGIRread
Type:	Package
Version:	1.2.0
Date:	2026-10-28
License:	LGPL (>= 2.0, < 3)

Author(s)

- Vincent T van Hees <v.vanhees@accelting.com> main creator and developer
- Patrick Bos developed function GENEActivReader
- Jing Hua Zhao <jinghua.zhao@mrc-epid.cam.ac.uk> co-developed function readGenea
- Evgeny Mirkes created function readAxivity
- Dan Jackson helped improve function readAxivity

`GENEActivReader` *Function (C++) to read binary files as produced by the GENEActiv accelerometer*

Description

Function to read binary files as produced by the GENEActiv accelerometer by ActivInsights Ltd https://activinsights.com/

Usage

```
GENEActivReader(filename, start = 0L, end = 0L, progress_bar = FALSE)
```

Arguments

`filename`	filename (required)
`start`	start page for reading data
`end`	end page for reading data
`progress_bar`	Boolean

Details

If only `start` page is defined then all data is read beyond `start` until the end of the file is reached

Value

`info`	List with ReadOK (good=0 or error=1), ReadErrors (Count of pages with read errors), SampleRate (Hertz), numBlocksTotal
`time`	Numeric vector with time in miliseconds since start page
`x`	Numeric vector with x-axis acceleration in gravitational units
`y`	Numeric vector with y-axis acceleration in gravitational units
`z`	Numeric vector with z-axis acceleration in gravitational units
`temperature`	matrix with battery voltage and corresponding timestamps
`lux`	Numeric vector with lux values in Volts

Author(s)

Patrick Bos <egpbos@gmail.com> Vincent T van Hees <v.vanhees@accelting.com>

mergeFitbitData *Merge Fitbit json files into one data.frame*

Description

Read and merge Fitbit json files into one data.frame, and give warning if time series do not intersect.

Usage

```
mergeFitbitData(filenames = NULL,
                  desiredtz = "", configtz = NULL)
```

Arguments

filenames	Character vector, specifying the filenames (at least two) to be merged.
desiredtz	Character, timezone name where the accelerometer was worn. Timezone names are expected to be the timezone database names, e.g. Europe/London. See also: https://en.wikipedia.org/wiki/List_of_tz_database_time_zones
configtz	Character, timezone name where the accelerometer was configured. Leave NULL if equal to desiredtz. Timezones name are expected to be the timezone database names, e.g. Europe/London. See also: https://en.wikipedia.org/wiki/List_of_tz_database_time_zones

Value

Merged data from the two or more files.

Author(s)

Vincent T van Hees <v.vanhees@accelting.com>

mergePHBdata *Merge Philips Health Band xlsx file pairs into one csv file*

Description

Read, merge, and save Philips Health Band xlsx file pairs to csv. If the pair is incomplete the function will still convert the available xlsx file to csv.

Usage

```
mergePHBdata(filenames = NULL,
                  timeformat = "%m/%d/%Y %H:%M:%S",
                  desiredtz = "", configtz = NULL,
                  timeformatName = "timeformat")
```

Arguments

filenames	Character vector, specifying the two filenames to be merged. Assumption is that there is one file starting with "Datalist_" and one starting with "Sleep_Wake_".
timeformat	Character, timestemp format.
desiredtz	Character, timezone name where the accelerometer was worn. Timezone names are expected to be the timezone database names, e.g. Europe/London. See also: https://en.wikipedia.org/wiki/List_of_tz_database_time_zones
configtz	Character, timezone name where the accelerometer was configured. Leave NULL if equal to desiredtz. Timezones name are expected to be the timezone database names, e.g. Europe/London. See also: https://en.wikipedia.org/wiki/List_of_tz_database_time_zones
timeformatName	Character, name of timeformat variable to print in error message when timeformat is incorrect, of use to GGIR where argument names can differ.

Value

Merged data from the two files.

Author(s)

Vincent T van Hees <v.vanhees@accelting.com>

---

readActicalCount *Read Actical Count data files (csv)*

Description

Reads Actical Count data file.

Usage

```
readActicalCount(filename = NULL,
                           timeformat = "%m/%d/%Y %H:%M:%S",
                           desiredtz = "", configtz = NULL,
                           timeformatName = "timeformat")
```

Arguments

filename	filename (required)
timeformat	Character, timestemp format.
desiredtz	Character, timezone name where the accelerometer was worn. Timezone names are expected to be the timezone database names, e.g. Europe/London. See also: https://en.wikipedia.org/wiki/List_of_tz_database_time_zones
configtz	Character, timezone name where the accelerometer was configured. Leave NULL if equal to desiredtz. Timezones name are expected to be the timezone database names, e.g. Europe/London. See also: https://en.wikipedia.org/wiki/List_of_tz_database_time_zones
timeformatName	Character, name of timeformat variable to print in error message when timeformat is incorrect, of use to GGIR where argument names can differ.

Value

data	Matrix with one or multiple columns
epochSize	epoch size in seconds of data
startTime	POSIXlt format timestamp on which recording starts

Author(s)

Vincent T van Hees <v.vanhees@accelting.com>

---

readActiGraphCount *Read ActiGraph Count data files (csv)*

Description

Reads ActiGraph Count data file. Currently a variety of csv format are facilitated.

Usage

```
readActiGraphCount(filename = NULL,
                           timeformat = "%m/%d/%Y %H:%M:%S",
                           desiredtz = "", configtz = NULL,
                           timeformatName = "timeformat")
```

Arguments

filename	filename (required)
timeformat	Character, timestemp format.
desiredtz	Character, timezone name where the accelerometer was worn. Timezone names are expected to be the timezone database names, e.g. Europe/London. See also: https://en.wikipedia.org/wiki/List_of_tz_database_time_zones
configtz	Character, timezone name where the accelerometer was configured. Leave NULL if equal to desiredtz. Timezones name are expected to be the timezone database names, e.g. Europe/London. See also: https://en.wikipedia.org/wiki/List_of_tz_database_time_zones
timeformatName	Character, name of timeformat variable to print in error message when timeformat is incorrect, of use to GGIR where argument names can differ.

Value

data	Matrix with one or multiple columns
epochSize	epoch size in seconds of data
startTime	POSIXlt format timestamp on which recording starts
deviceSerialNumber	Device serial number if it could be extracted from the file header

Author(s)

Vincent T van Hees <v.vanhees@accelting.com>

readActiwatchCount *Read Actiwatch Count data files (csv)*

Description

Reads Actiwatch Count data file. Currently a variety of csv format are facilitated.

Usage

```
readActiwatchCount(filename = NULL,
                           timeformat = "%m/%d/%Y %H:%M:%S",
                           desiredtz = "", configtz = NULL,
                           timeformatName = "timeformat")
```

Arguments

filename	filename (required)
timeformat	Character, timestemp format.
desiredtz	Character, timezone name where the accelerometer was worn. Timezone names are expected to be the timezone database names, e.g. Europe/London. See also: https://en.wikipedia.org/wiki/List_of_tz_database_time_zones
configtz	Character, timezone name where the accelerometer was configured. Leave NULL if equal to desiredtz. Timezones name are expected to be the timezone database names, e.g. Europe/London. See also: https://en.wikipedia.org/wiki/List_of_tz_database_time_zones
timeformatName	Character, name of timeformat variable to print in error message when timeformat is incorrect, of use to GGIR where argument names can differ.

Value

data	Matrix with one or multiple columns
epochSize	epoch size in seconds of data
startTime	POSIXlt format timestamp on which recording starts

Author(s)

Vincent T van Hees <v.vanhees@accelting.com>

readAxivity *Function to read .cwa-format files as produced by the accelerometer named 'Axivity'*

Description

For reading .cwa-format data with the Axivity AX3 and AX6 sensors.

Usage

```
readAxivity(filename, start = 0, end = 0, progressBar = FALSE,
  desiredtz = "", configtz = c(), interpolationType=1, loadbattery = FALSE,
  header = NULL, frequency_tol = 0.1, maxAllowedCorruptBlocks = 20)
```

Arguments

filename	filename (required) is name of cwa file to read
start	Zero-based, non-negative integer index of the first data block to read. The metadata header is not counted as a data block.
end	Exclusive data block index. Use Inf to read through the last complete data block. Values beyond the file's block count are limited to that count. If end <= start, only the header is returned, with data = NULL. The default start = end = 0 therefore reads only the header.
progressBar	Is trigger to switch on/off the text progress bar. If progressBar is TRUE then the function displays the progress bar but it works slightly slower
desiredtz	Desired timezone, a character with timezone database name.
configtz	Timezone in which the accelerometer was configured. Only use this argument if the timezone of configuration and timezone in which recording took place are different.
interpolationType	Integer to indicate type of interpolation to be used, 1=linear, 2=nearest neighbour.
loadbattery	Boolean to indicate whether battery voltage should be loaded
header	Header returned by an earlier call for the same file, to avoid reading it again when processing chunks. Use the same timezone settings as in that call. Warnings about incomplete trailing bytes are issued only when header = NULL.
frequency_tol	Numeric value representing the tolerance for frequency bias expressed as a fraction of the expected sampling frequency. For example, 0.10 indicates that we accept the sampling frequency configured at 100 Hertz to vary between 90 and 110 Hertz. If this condition is not met the data will be imputed with a constant value and the start and end time will be logged in the QClog output object. Regardless of the setting the function will always log instances when frequency differs by more than 5%, but if this is less than frequency_tol the block will not be imputed.
maxAllowedCorruptBlocks	Max number of consecutive blocks with a failed checksum that we'll tolerate.

Details

Only complete 512-byte data packets are decoded. An incomplete final packet is excluded with a warning when reading the header. An error is raised when no complete data packets remain. Corrupt packets at the start or end of a requested interval are skipped up to `maxAllowedCorruptBlocks`; internal corruption can cause the intervening interval to be imputed.

Acceleration is interpolated using the selected method. Rejected intervals are filled with the preceding acceleration orientation normalised to 1 g when its magnitude lies between 0.8 and 1.2 g, otherwise with `c(0, 0, 1)`. Gyroscope channels are filled with zero during imputation.

For failed-checksum rows in `QClog`, the packet's contents cannot be trusted: block IDs are inferred from file positions and timing and frequency fields are zero. Such rows have `imputed = FALSE`; a separate row records any imputation between the surrounding valid packets.

Value

- `data` dataframe with data from start inclusive till end exclusive. If start == end then data section of final structure is empty.
  - time numeric timestamp in seconds since the Unix epoch
  - gx gyroscope x-axis in degrees per second (AX6 with gyroscope enabled)
  - gy gyroscope y-axis in degrees per second (AX6 with gyroscope enabled)
  - gz gyroscope z-axis in degrees per second (AX6 with gyroscope enabled)
  - x accelerometer x-axis in g
  - y accelerometer y-axis in g
  - z accelerometer z-axis in g
  - temp temperature in Celsius (recorded once per block)
  - battery voltage in volts (recorded once per block), or zero when `loadbattery = FALSE`
  - light raw light sensor value (recorded once per block), without conversion to lux
- `header` file header. list of header information:
  - uniqueSerialCode is the device serial code
  - frequency is the nominal measurement frequency in Hz, including fractional rates such as 12.5 Hz. All data are resampled at this frequency
  - start is a `POSIXct` timestamp for the first valid data block, displayed in `desiredtz`
  - device is "Axivity"
  - firmwareVersion
  - blocks is the number of complete data blocks, including corrupt blocks
  - accrange is the accelerometer range in g (for example, 8 denotes +/-8 g)
  - hardwareType is `"AX3"` or `"AX6"`
  - blockLength is the sample count of the first valid data block, typically 40, 80 or 120 depending on the sensor configuration
- `QClog` dataframe logging failed checksums, imputed intervals and frequency deviations above 5%; `NULL` when there are no such events. It is omitted for header-only reads. Columns are:
  - checksum_pass Boolean to indicate whether checksum was zero.
  - blockID_current ID of the block whose interval is being checked
  - blockID_next ID of the following block

- start numeric start time of the checked interval, in seconds since the Unix epoch
- end numeric end time of the checked interval, in seconds since the Unix epoch
- blockLengthSeconds duration of the interval in seconds
- frequency_blockheader frequency derived from block header
- frequency_observed frequency as observed
- imputed Boolean indicating whether the interval was imputed

Author(s)

Evgeny Mirkes <em322@leicester.ac.uk> Lena Kushleyeva Vincent van Hees <v.vanhees@accelting.com>

Examples

```
 cwafile  = system.file("testfiles/ax3_testfile.cwa", package = "GGIRread")[1]
AX3 = readAxivity(filename = cwafile, desiredtz = "Europe/Berlin", start = 1, end = 2)
```

---

readFitbit *Read Fitbit data files (json)*

Description

Reads Fitbit data file (json) with sleep, steps or calories. Steps and calories are converted to 30 second resolution to match resolution of sleep data.

Usage

```
readFitbit(filename = NULL, desiredtz = "", configtz = NULL)
```

Arguments

filename	Character, filename (required) of json file
desiredtz	Character, timezone name where the accelerometer was worn. Timezone names are expected to be the timezone database names, e.g. Europe/London. See also: https://en.wikipedia.org/wiki/List_of_tz_database_time_zones
configtz	Character, timezone name where the accelerometer was configured. Leave NULL if equal to desiredtz. Timezones name are expected to be the timezone database names, e.g. Europe/London. See also: https://en.wikipedia.org/wiki/List_of_tz_database_time_zones

Value

Data.frame with data converted to time series

Author(s)

Vincent T van Hees <v.vanhees@accelting.com>

readGenea *Function to read binary files as produced by the accelerometer named 'Genea', not to be confused with the 'GENEActiv' (see package GENEAread for this)*

Description

For reading the binary data as collected with a Genea accelerometer (Unilever Discover, UK). For reading GENEActive binary data, see package GENEAread.

Usage

```
readGenea(filename, start = 0, end = 0)
```

Arguments

filename	filename (required)
start	start point for reading data, this can either be a timestamp in POSIXct format "year-month-day hr:min:sec" or a page number (optional)
end	end point for reading data, this can either be a timestamp in POSIXct format "year-month-day hr:min:sec" or a page number (optional)

Details

If only start is defined then readGenea will read all data beyond start until the end of the file is reached

Value

rawxyz	matrix with raw x, y, and, z acceleration values
header	file header
timestamps1	timestamps for rawxyz in seconds since 1970-01-01 00:00
timestamps2	timestamps for rawxyz in day time format
batt.voltage	matrix with battery voltage and corresponding timestamps

Author(s)

Vincent T van Hees <v.vanhees@accelting.com> Jing Hua Zhao <jinghua.zhao@mrc-epid.cam.ac.uk>

Examples

```
geneafile  = system.file("testfiles/genea_testfile.bin", package = "GGIRread")[1]
GENEA = readGenea(filename = geneafile, start = 0, end = 2)
```

`readGENEActiv` *Function (R) to read binary files as produced by the GENEActiv accelerometer*

Description

R function wrapper around GENEActivReader to read binary files as produced by the GENEActiv accelerometer by ActivInsights Ltd https://activinsights.com/

Usage

```
readGENEActiv(filename, start = 0L, end = 0L, progress_bar = FALSE,
              desiredtz = "", configtz = NULL)
```

Arguments

`filename`	filename (required)
`start`	start page for reading data
`end`	end page for reading data
`progress_bar`	Boolean
`desiredtz`	Character, timezone database name of the timezone where the accelerometer was worn.
`configtz`	Character, timezone database name of the timezone where the accelerometer was configured. Leave NULL if equal to timezone where experiment took place.

Details

If only `start` page is defined then all data is read beyond `start` until the end of the file is reached.

GENEActiv light measurements are calibrated according to the device model. For GENEActiv 1.1, the existing light calibration is applied. For GENEActiv 1.2 and above, an additional correction is applied to account for the light sensor's non-linear scaling with graded resolution. This provides greater accuracy at low light levels while supporting a higher range. This correction was provided by ActivInsights Ltd, the manufacturer of GENEActiv devices.

Value

header:

`serial_number`	Device serial number as extracted from file header
`firmware`	Firmware version
`tzone`	Time zone as extracted from file header
`ReadOK`	see GENEActivReader
`SampleRate`	matrix with battery voltage and corresponding timestamps
`ReadErrors`	matrix with battery voltage and corresponding timestamps

numBlocksTotal	matrix with battery voltage and corresponding timestamps
StartTime	Start time as extracted from file header

data.out:

time	Unix time in seconds with decimal places for miliseconds
x	see GENEActivReader
y	see GENEActivReader
z	see GENEActivReader
light	Ligth values express in lux units
temperature	Temperature in Celsius

Author(s)

Patrick Bos <egpbos@gmail.com> Vincent T van Hees <v.vanhees@accelting.com>

Examples

```
binfile  = system.file("testfiles/GENEActiv_testfile.bin", package = "GGIRread")[1]
rdata = readGENEActiv(filename = binfile, start = 1, end = 1, desiredtz = "Europe/London")
```

---

readParmayMatrix *Read and Process Binary Data from Matrix Devices*

Description

Reads a binary file generated by Parmay Tech Matrix devices, processes its header and packet data, validates data integrity using CRC32 checksums, and outputs structured sensor data and quality check information.

Usage

```
readParmayMatrix(filename, output = c("all", "sf", "dynrange")[1],
                 start = 1, end = NULL,
                 desiredtz = "", configtz = NULL, interpolationType = 1,
                 read_acc = TRUE, read_gyro = FALSE,
                 read_temp = TRUE, read_heart = FALSE)
```

Arguments

filename	Character. Path to the binary file to be read.
output	Character. Specifies the type of output. Options include:
	"all" Returns the full processed data.
	"sf" Returns the sampling frequency of the accelerometer data.
	"dynrange" Returns the dynamic range of the accelerometer.

`start`	Integer. The index of the starting packet to process. Default is 1.
`end`	Integer. The index of the ending packet to process. Default is `NULL`, which means all packets are processed.
`desiredtz`	Character. Time zone for the returned timestamps. Default is an empty string, which uses the system's default time zone.
`configtz`	Character. Time zone specified in the file's configuration. Default is `NULL`, which means that it uses `desiredtz`.
`interpolationType`	Integer. Specifies the type of interpolation (see `resample`) to use when resampling data: 1 for Linear interpolation, 2 for Nearest-neighbor interpolation.
`read_acc`	Logical. Indicates whether accelerometer data should be read.
`read_gyro`	Logical. Indicates whether gyroscope data should be read.
`read_temp`	Logical. Indicates whether temperature data should be read.
`read_heart`	Logical. Indicates whether heart rate data should be read.

Details

Matrix devices store binary data in packets, with varying lengths depending on the number of sensor recordings in each packet. The function processes the file's header to extract metadata such as the total number of packets and sensor ranges, validates data integrity using CRC32 checksums, and interpolates data to a consistent sampling frequency.

Header Information:

- Remarks: Bytes 1-512.
- Header string: Bytes 513-516. If not "MDTC", the file is considered corrupt.
- Total packets: Bytes 517-520.
- Accelerometer dynamic range: Bytes 521-522.
- Gyroscope range: Bytes 523-524.

Packet Structure:

Each packet contains accelerometer, gyroscope, temperature, and heart rate data.

- 8-byte package header.
- 4-byte CRC32 indicator.
- 4-byte start timestamp.
- 4-byte end timestamp.
- 4-byte number of accelerometer recordings in packet.
- 4-byte number of gyroscope recordings in packet.
- 4-byte number of temperature recordings in packet.
- 4-byte number of heart rate recordings in packet.
- Sensor data for accelerometer, gyroscope, temperature, and heart rate.

Value

A list containing the following elements (when `output = "all"`):

- `QClog`: A data frame with quality control information, including checksum validation and data gaps.
- `output`: A data frame with resampled sensor data, including:

  `time` Timestamps of the recordings.

  `acc_x`, `acc_y`, `acc_z` Resampled accelerometer data.

  `gyro_x`, `gyro_y`, `gyro_z` Resampled gyroscope data.

  `bodySurface_temp`, `ambient_temp` Resampled temperature data.

  `hr_raw`, `hr` Resampled heart rate data.

  `remarks` Remarks extracted from the file header.
- `header`: A list with the following elements:

  `sf` Sampling frequency of the accelerometer data.

  `acc_dynrange` Dynamic range of the accelerometer.

  `starttime` Start time of the first packet in POSIXct format.
- `lastchunk`: Logical, indicating if the processed data includes the last packet in the file.

If `output = "sf"`, the function returns only the sampling frequency.

If `output = "dynrange"`, it returns the dynamic range of the accelerometer.

Author(s)

Jairo H Migueles <jairo@jhmigueles.com>

References

For more details on Matrix devices', see: https://www.matrixwearable.net/ For additional details on Matrix BIN files structure, please contact manufacturer: https://www.matrixwearable.net/#!/contact

See Also

`resample` for resampling sensor data.

Examples

```
## Not run:
# Example usage:
binfile  = system.file("testfiles/mtx_12.5Hz_acc.BIN", package = "GGIRread")
# Read full data and process all packets
result <- readParmayMatrix(binfile)

# Get sampling frequency only
sf <- readParmayMatrix(binfile, output = "sf")

# Get accelerometer dynamic range
dynrange <- readParmayMatrix(binfile, output = "dynrange")
```

```
  # Process a subset of packets
  result_subset <- readParmayMatrix(binfile, start = 10, end = 20)

## End(Not run)
```

`readPHBCount` *Read PHB Count data files (xlsx)*

Description

Reads Philips Health Band (PHB) Count data file.

Usage

```
readPHBCount(filename = NULL,
             timeformat = "%m/%d/%Y %H:%M:%S",
             desiredtz = "", configtz = NULL,
             timeformatName = "timeformat")
```

Arguments

`filename`	filename (required)
`timeformat`	Character, timestemp format.
`desiredtz`	Character, timezone name where the accelerometer was worn. Timezone names are expected to be the timezone database names, e.g. Europe/London. See also: https://en.wikipedia.org/wiki/List_of_tz_database_time_zones
`configtz`	Character, timezone name where the accelerometer was configured. Leave NULL if equal to desiredtz. Timezones name are expected to be the timezone database names, e.g. Europe/London. See also: https://en.wikipedia.org/wiki/List_of_tz_database_time_zones
`timeformatName`	Character, name of timeformat variable to print in error message when timeformat is incorrect, of use to GGIR where argument names can differ.

Value

Data.frame with time series, exact column names can vary between xlsx files.

Author(s)

Vincent T van Hees <v.vanhees@accelting.com>

readWav *Deprecated function to read .wav files as can be stored with OMGUI software by Axivity Ltd*

Description

Originally designed to read the wav accelerometer data format as stored with the OMGUI software by Axivity Ltd and documented here.

This function has been deprecated as wav files are hardly used, their formatting is not consistent causing ongoing challenges, and the wav format is not well facilitated for AX6 recordings.

Usage

```
readWav(filename, start = 1, end = 100,units="minutes")
```

Arguments

filename	filename (required)
start	start point for reading data, see also units
end	end point for reading data, see also units
units	units used for defining start and end

Details

If only start is defined then g.binread will read all data beyond start until the end of the file is reached

Value

rawxyz	matrix with raw x, y, and, z acceleration values
header	file header
timestamps	local timestamps for rawxyz

Author(s)

Vincent T van Hees <v.vanhees@accelting.com>

`resample` *Resample timeseries*

Description

Resample single- or multi-variate time series using linear or nearest neighbour interpolation

Usage

```
resample(raw, rawTime, time, stop, type = 1)
```

Arguments

`raw`	stop-by-3 matrix with raw values of x, y and z.
`rawTime`	vector with stop elements of raw time.
`time`	array with required time points.
`stop`	Number of rows in raw
`type`	integer to indicate type of interpolation, 1=linear, 2=nearest neighbour

Examples

```
raw = cbind(1:10, 1:10, 1:10)
rawTime = seq(0.1, 1, by = 0.1)
time = seq(0.15001, 1.05001, by = 0.1)
stop = 10
dat_lin = resample(raw, rawTime, time, stop, type = 1)
```

Index

GENEActivReader, *3*, 3, *12, 13*
GGIRread-package, 2

mergeFitbitData, 4
mergePHBdata, 4

readActicalCount, 5
readActiGraphCount, 6
readActiwatchCount, 7
readAxivity, *3*, 8
readFitbit, 10
readGenea, *3*, 11
readGENEActiv, 12
readParmayMatrix, 13
readPHBCount, 16
readWav, 17
resample, *14, 15*, 18