Package: PhysioCore (via r-universe)

May 16, 2026

Title Core Data Structures for Physiological Signal Analysis

Version 0.2.0

Author Yusuke Matsui

Maintainer Yusuke Matsui `<you@example.com>`

Description Provides the PhysioExperiment class, a SummarizedExperiment extension for multi-modal physiological signal data. Includes core accessors, channel and event management, and basic S4 methods. Designed as a lightweight foundation for the Physio ecosystem.

Depends R (>= 4.2)

Imports methods, SummarizedExperiment, S4Vectors, stats

Suggests testthat (>= 3.2.0), abind, knitr, rmarkdown

VignetteBuilder knitr

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

Collate 'PhysioExperiment-class.R' 'PhysioExperiment-accessors.R' 'PhysioExperiment-methods.R' 'channels.R' 'events.R' 'utils-signal.R' 'utils-na.R' 'zzz.R'

RoxygenNote 7.3.3

Config/pak/sysreqs zlib1g-dev

Repository https://x-biosignal.r-universe.dev

Date/Publication 2026-03-16 11:30:55 UTC

RemoteUrl https://github.com/x-biosignal/PhysioCore

RemoteRef HEAD

RemoteSha cdd7f3bb4885b8c7bd95ec8f3552f99022d6a6d8

Contents

`.onLoad` *Package on-load hook*

Description

Silently registers the package namespace. This file is a placeholder for future registration routines (e.g. S3 methods or database initialisation).

Usage

```
.onLoad(libname, pkg)
```

Arguments

`libname`	Library path.
`pkg`	Package name.

`[,PhysioExperiment,ANY,ANY,ANY-method` *Subset PhysioExperiment by time indices*

Description

Extracts a subset of the `PhysioExperiment` by row (time) and/or column (channel) indices, preserving all metadata.

Usage

```
## S4 method for signature 'PhysioExperiment,ANY,ANY,ANY'
x[i, j, ..., drop = FALSE]
```

Arguments

`x`	A PhysioExperiment object.
`i`	Time indices (rows).
`j`	Channel indices (columns in first non-time dimension).
`...`	Additional arguments (not used).
`drop`	Logical. If TRUE, drops dimensions of size 1.

Value

A `PhysioExperiment` object containing only the selected time points and/or channels, with updated rowData and colData.

References

Huber, W., et al. (2015). "Orchestrating high-throughput genomic analysis with Bioconductor." *Nature Methods*, 12(2), 115-121. doi:10.1038/nmeth.3252

Morgan, M., et al. (2022). "S4Vectors: Foundation of vector-like and list-like containers in Bioconductor." R package.

See Also

`extractWindow` for subsetting by time in seconds, `pickChannels` for subsetting by channel name, `dropChannels` for removing specific channels

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  samplingRate = 100
)

# Subset by time
pe_subset <- pe[1:50, ]
dim(pe_subset)  # 50 4

# Subset by channels
pe_channels <- pe[, 1:2]
dim(pe_channels)  # 100 2
```

`addEvents` *Add events to a PhysioExperiment object*

Description

Add events to a PhysioExperiment object

Usage

```
addEvents(x, onset, duration = 0, type = "event", value = "")
```

Arguments

`x`	A PhysioExperiment object.
`onset`	Numeric vector of event onset times in seconds.
`duration`	Numeric vector of event durations in seconds.
`type`	Character vector of event types.
`value`	Character vector of event values/labels.

Value

The modified PhysioExperiment object.

References

Delorme A, Makeig S (2004). "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics." *Journal of Neuroscience Methods*, 134(1), 9-21.

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`setEvents` for replacing all events, `removeEvents` for removing events, `getEvents` for retrieving events, `nEvents` for event count

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(1000), nrow = 100)),
  samplingRate = 100
)

# Add stimulus events
pe <- addEvents(pe, onset = c(1, 2, 3), type = "stimulus")

# Add response events
pe <- addEvents(pe, onset = c(1.5, 2.5), type = "response", value = c("hit", "hit"))
nEvents(pe)  # 5 events total
```

`applyMontage` *Apply standard montage*

Description

Applies a standard electrode montage (e.g., 10-20 system).

Usage

```
applyMontage(x, system = c("10-20", "10-10", "10-5"))
```

Arguments

`x`	A PhysioExperiment object.
`system`	Montage system: "10-20", "10-10", or "10-5".

Value

Modified PhysioExperiment object with electrode positions.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`setElectrodePositions` for custom positions, `getElectrodePositions` for reading positions, `channelNames` for channel labels, `setReference` for reference electrode

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  colData = S4Vectors::DataFrame(label = c("Fz", "Cz", "Pz", "Oz")),
  samplingRate = 100
)

# Apply 10-20 system positions
pe <- applyMontage(pe, "10-20")
getElectrodePositions(pe)
```

`as.data.frame,PhysioExperiment-method`

Coerce to data.frame

Description

Converts the default assay of a `PhysioExperiment` to a `data.frame` with a `time` column (in seconds) followed by one column per channel.

Usage

```
## S4 method for signature 'PhysioExperiment'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

`x`	A PhysioExperiment object.
`row.names`	Unused.
`optional`	Unused.
`...`	Additional arguments.

Value

A `data.frame` with a `time` column and one column per channel. For 3D arrays, only the first sample (third dimension index 1) is returned. Returns an empty `data.frame` if no assays are present.

References

Huber, W., et al. (2015). "Orchestrating high-throughput genomic analysis with Bioconductor." *Nature Methods*, 12(2), 115-121. doi:10.1038/nmeth.3252

Morgan, M., et al. (2022). "S4Vectors: Foundation of vector-like and list-like containers in Bioconductor." R package.

See Also

`summary,PhysioExperiment-method` for summary statistics, `timeIndex` for the time vector, `channelNames` for column names

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(12), nrow = 3, ncol = 4)),
  colData = S4Vectors::DataFrame(label = c("Fz", "Cz", "Pz", "Oz")),
  samplingRate = 100
)
df <- as.data.frame(pe)
head(df)
```

`cbindPhysio` *Combine PhysioExperiment objects by channels*

Description

Combines two PhysioExperiment objects by concatenating along the channel (column) dimension. Both objects must have the same number of time points and matching sampling rates.

Usage

```
cbindPhysio(x, y)
```

Arguments

`x`	A PhysioExperiment object.
`y`	A PhysioExperiment object to combine.

Value

A `PhysioExperiment` object with channels from both `x` and `y`, combined colData, and metadata from `x`.

References

Huber, W., et al. (2015). "Orchestrating high-throughput genomic analysis with Bioconductor." *Nature Methods*, 12(2), 115-121. doi:10.1038/nmeth.3252

Morgan, M., et al. (2022). "S4Vectors: Foundation of vector-like and list-like containers in Bioconductor." R package.

See Also

`rbindPhysio` for combining along the time axis, `pickChannels` for selecting specific channels, `dropChannels` for removing channels

Examples

```
pe1 <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(200), nrow = 100, ncol = 2)),
  colData = S4Vectors::DataFrame(label = c("Fz", "Cz")),
  samplingRate = 100
)
pe2 <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(200), nrow = 100, ncol = 2)),
  colData = S4Vectors::DataFrame(label = c("Pz", "Oz")),
  samplingRate = 100
)

# Combine channels
pe_combined <- cbindPhysio(pe1, pe2)
nChannels(pe_combined)  # 4
```

`channelInfo` *Channel information management for PhysioExperiment*

Description

Functions for managing channel metadata including labels, types, units, and electrode positions. Get channel information

Usage

```
channelInfo(x)
```

Arguments

`x`	A PhysioExperiment object.

Details

Returns channel metadata as a DataFrame. Channel information is stored in colData (columns = channels).

Value

A DataFrame with channel information.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`channelNames` for channel labels, `nChannels` for channel count, `setChannelTypes` for assigning channel types

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  colData = S4Vectors::DataFrame(
    label = c("Fz", "Cz", "Pz", "Oz"),
    type = rep("EEG", 4)
  ),
  samplingRate = 100
)

# Get channel information
channelInfo(pe)

# Get channel names
channelNames(pe)

# Get number of channels
nChannels(pe)
```

`channelInfo<-` *Set channel information*

Description

Updates channel metadata. Channel information is stored in colData (columns = channels).

Usage

```
channelInfo(x) <- value
```

Arguments

`x`	A PhysioExperiment object.
`value`	A DataFrame with channel information.

Value

Modified PhysioExperiment object.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`channelInfo` for reading channel metadata, `channelNames` for channel labels, `setChannelTypes` for assigning channel types

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  samplingRate = 100
)

# Set channel info
channelInfo(pe) <- S4Vectors::DataFrame(
  label = c("Fz", "Cz", "Pz", "Oz"),
  type = rep("EEG", 4)
)
```

`channelNames` *Get channel names/labels*

Description

Get channel names/labels

Usage

```
channelNames(x)
```

Arguments

`x`	A PhysioExperiment object.

Value

Character vector of channel names.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`channelInfo` for full channel metadata, `renameChannels` for renaming channels, `nChannels` for channel count

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  colData = S4Vectors::DataFrame(label = c("Fz", "Cz", "Pz", "Oz")),
  samplingRate = 100
)
channelNames(pe)  # c("Fz", "Cz", "Pz", "Oz")
```

`channelNames<-` *Set channel names/labels*

Description

Set channel names/labels

Usage

```
channelNames(x) <- value
```

Arguments

`x`	A PhysioExperiment object.
`value`	Character vector of channel names.

Value

Modified PhysioExperiment object.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`channelNames` for reading channel labels, `renameChannels` for renaming specific channels, `channelInfo` for full channel metadata

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  samplingRate = 100
)
channelNames(pe) <- c("Fz", "Cz", "Pz", "Oz")
channelNames(pe)
```

`checkNA` *Check for NA values in assay data*

Description

Validates assay data for NA values and reports statistics.

Usage

```
checkNA(x, action = c("warn", "error", "none"))
```

Arguments

`x`	A PhysioExperiment object or numeric array.
`action`	Action to take: "warn" (default), "error", or "none".

Value

Invisibly returns a list with NA statistics.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`hasNA` for quick NA presence check, `naSummary` for per-assay NA statistics, `handleNA` for NA replacement strategies

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(c(1, NA, 3, 4), nrow = 2)),
  samplingRate = 100
)
checkNA(pe)
```

`defaultAssay` *Retrieve the default assay name*

Description

Returns the name of the first assay in the `PhysioExperiment` object, which is treated as the default assay for operations that do not specify an assay explicitly.

Usage

```
defaultAssay(x)
```

Arguments

`x`	A `PhysioExperiment` instance.

Value

Character scalar naming the first assay, or `NA_character_` when no assays are present.

References

Huber, W., et al. (2015). "Orchestrating high-throughput genomic analysis with Bioconductor." *Nature Methods*, 12(2), 115-121. doi:10.1038/nmeth.3252

Morgan, M., et al. (2022). "S4Vectors: Foundation of vector-like and list-like containers in Bioconductor." R package.

See Also

`PhysioExperiment` for the constructor, `samplingRate` for the sampling rate accessor, `timeIndex` for time point vector

---

`dim,PhysioExperiment-method` *Dim method for PhysioExperiment*

Description

Returns the dimensions of the default assay.

Usage

```
## S4 method for signature 'PhysioExperiment'
dim(x)
```

Arguments

`x`	A PhysioExperiment object.

Value

An integer vector of dimensions (time points x channels x samples), or `NULL` if no assays are present.

References

Huber, W., et al. (2015). "Orchestrating high-throughput genomic analysis with Bioconductor." *Nature Methods*, 12(2), 115-121. doi:10.1038/nmeth.3252

Morgan, M., et al. (2022). "S4Vectors: Foundation of vector-like and list-like containers in Bioconductor." R package.

See Also

`length,PhysioExperiment-method` for time point count, `nChannels` for channel count, `defaultAssay` for the assay being queried

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  samplingRate = 100
)
dim(pe)  # 100 4
```

`dropChannels` *Drop channels*

Description

Creates a new PhysioExperiment without specified channels.

Usage

```
dropChannels(x, channels)
```

Arguments

`x`	A PhysioExperiment object.
`channels`	Integer indices or character names of channels to drop.

Value

A new PhysioExperiment without specified channels.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`pickChannels` for keeping specific channels, `getChannelsByType` for finding channels by type, `nChannels` for channel count

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  colData = S4Vectors::DataFrame(label = c("Fz", "Cz", "Pz", "Oz")),
  samplingRate = 100
)

# Drop by index
pe_dropped <- dropChannels(pe, 1)
nChannels(pe_dropped)  # 3

# Drop by name
pe_dropped <- dropChannels(pe, c("Fz", "Oz"))
```

`duration` *Get signal duration*

Description

Computes the total duration of the signal in seconds from the number of time points and the sampling rate.

Usage

```
duration(x)
```

Arguments

`x`	A PhysioExperiment object.

Value

Numeric scalar giving the signal duration in seconds, or `NA_real_` if the sampling rate is not set.

References

Huber, W., et al. (2015). "Orchestrating high-throughput genomic analysis with Bioconductor." *Nature Methods*, 12(2), 115-121. doi:10.1038/nmeth.3252

Morgan, M., et al. (2022). "S4Vectors: Foundation of vector-like and list-like containers in Bioconductor." R package.

See Also

`samplingRate` for the sampling rate, `length,PhysioExperiment-method` for time point count, `timeIndex` for the time vector, `extractWindow` for time-based subsetting

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(1000), nrow = 1000, ncol = 4)),
  samplingRate = 100
)
duration(pe)  # 10 seconds
```

`extractWindow` *Extract time window*

Description

Extracts a time window from the signal based on start and end times in seconds.

Usage

```
extractWindow(x, tmin, tmax)
```

Arguments

`x`	A PhysioExperiment object.
`tmin`	Start time in seconds.
`tmax`	End time in seconds.

Value

A `PhysioExperiment` object containing only the samples within the specified time window, with preserved channel and event metadata.

References

Huber, W., et al. (2015). "Orchestrating high-throughput genomic analysis with Bioconductor." *Nature Methods*, 12(2), 115-121. doi:10.1038/nmeth.3252

Morgan, M., et al. (2022). "S4Vectors: Foundation of vector-like and list-like containers in Bioconductor." R package.

See Also

`duration` for total signal duration, `timeIndex` for the time vector, `[` for index-based subsetting, `timeToSamples` for converting times to indices

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(1000), nrow = 1000, ncol = 4)),
  samplingRate = 100
)

# Extract 2 to 5 seconds
pe_window <- extractWindow(pe, tmin = 2, tmax = 5)
duration(pe_window)  # approximately 3 seconds
```

`fillEdgeNA` *Fill NA values at edges*

Description

Fills NA values at the beginning and end of a signal that may result from filtering operations.

Usage

```
fillEdgeNA(x, method = c("extend", "zero"))
```

Arguments

`x`	Numeric vector.
`method`	Fill method: "extend" (extend nearest valid value) or "zero" (fill with zeros).

Value

Vector with edge NA values filled.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`handleNA` for general NA handling strategies, `replaceNA` for NA handling in PhysioExperiment assays

Examples

```
x <- c(NA, NA, 1, 2, 3, NA, NA)
fillEdgeNA(x, method = "extend")
```

`getChannelsByType` *Get channels by type*

Description

Returns indices of channels matching specified types.

Usage

```
getChannelsByType(x, types)
```

Arguments

`x`	A PhysioExperiment object.
`types`	Character vector of channel types to match.

Value

Integer vector of matching channel indices.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`setChannelTypes` for assigning channel types, `pickChannels` for subsetting by channel, `dropChannels` for removing channels

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  colData = S4Vectors::DataFrame(
    label = c("Fz", "EOG1", "EMG1", "Oz"),
    type = c("EEG", "EOG", "EMG", "EEG")
  ),
  samplingRate = 100
)

# Get EEG channels
eeg_idx <- getChannelsByType(pe, "EEG")  # c(1, 4)
```

`getElectrodePositions` *Get electrode positions*

Description

Get electrode positions

Usage

```
getElectrodePositions(x)
```

Arguments

`x`	A PhysioExperiment object.

Value

A data.frame with x, y, z columns or NULL if not set.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`setElectrodePositions` for setting positions, `applyMontage` for standard electrode layouts, `channelNames` for channel labels

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(300), nrow = 100, ncol = 3)),
  colData = S4Vectors::DataFrame(label = c("Fz", "Cz", "Pz")),
  samplingRate = 100
)
pe <- applyMontage(pe, "10-20")
getElectrodePositions(pe)
```

`getEvents` *Get events from a PhysioExperiment object*

Description

Get events from a PhysioExperiment object

Usage

```
getEvents(x, type = NULL)
```

Arguments

`x`	A PhysioExperiment object.
`type`	Optional character vector of event types to filter.

Value

A PhysioEvents object or DataFrame of events.

References

Delorme A, Makeig S (2004). "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics." *Journal of Neuroscience Methods*, 134(1), 9-21.

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`setEvents` for attaching events, `addEvents` for appending events, `removeEvents` for removing events, `nEvents` for event count

Examples

```
# Create PhysioExperiment with events
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(1000), nrow = 100)),
  samplingRate = 100
)
events <- PhysioEvents(
  onset = c(1, 2, 3),
  type = c("stimulus", "response", "stimulus")
)
pe <- setEvents(pe, events)

# Get all events
getEvents(pe)
```

```
# Get only stimulus events
getEvents(pe, type = "stimulus")
```

`getReference` *Get reference electrode*

Description

Get reference electrode

Usage

```
getReference(x)
```

Arguments

`x`	A PhysioExperiment object.

Value

Character string of reference electrode or NULL.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`setReference` for setting the reference, `channelInfo` for full channel metadata

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  samplingRate = 100
)
pe <- setReference(pe, "Cz")
getReference(pe)  # "Cz"
```

`handleNA` *Handle NA values in signal data*

Description

Provides various strategies for handling NA values in signal data.

Usage

```
handleNA(
  x,
  method = c("interpolate", "omit", "zero", "mean", "locf", "none"),
  ...
)
```

Arguments

`x`	Numeric vector or matrix.
`method`	Method for handling NA: "omit" (remove), "interpolate" (linear), "zero" (replace with 0), "mean" (replace with mean), "locf" (last observation carried forward), or "none" (no action).
`...`	Additional arguments passed to interpolation methods.

Value

Data with NA values handled according to the specified method.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`replaceNA` for handling NA in PhysioExperiment assays, `fillEdgeNA` for edge-specific NA filling, `checkNA` for NA validation

Examples

```
x <- c(1, NA, 3, NA, 5)

# Linear interpolation
handleNA(x, method = "interpolate")

# Replace with mean
handleNA(x, method = "mean")
```

```
# Last observation carried forward
handleNA(x, method = "locf")
```

`hasNA` *Check if data contains any NA values*

Description

Quick check for NA presence in PhysioExperiment data.

Usage

```
hasNA(x, assay_name = NULL)
```

Arguments

`x`	A PhysioExperiment object.
`assay_name`	Optional specific assay to check.

Value

Logical indicating presence of NA values.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`checkNA` for detailed NA statistics, `naSummary` for per-assay NA summary, `replaceNA` for handling NA values

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(1:4, nrow = 2)),
  samplingRate = 100
)
hasNA(pe)
```

`length,PhysioExperiment-method`

Length method for PhysioExperiment

Description

Returns the number of time points (rows) in the default assay.

Usage

```
## S4 method for signature 'PhysioExperiment'
length(x)
```

Arguments

`x`	A PhysioExperiment object.

Value

Integer scalar giving the number of time points (rows) in the default assay, or `0L` if no assays are present.

References

Huber, W., et al. (2015). "Orchestrating high-throughput genomic analysis with Bioconductor." *Nature Methods*, 12(2), 115-121. doi:10.1038/nmeth.3252

Morgan, M., et al. (2022). "S4Vectors: Foundation of vector-like and list-like containers in Bioconductor." R package.

See Also

`dim,PhysioExperiment-method` for full dimensions, `nChannels` for the number of channels, `duration` for duration in seconds

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  samplingRate = 100
)
length(pe)  # 100
```

`naSummary` *Get NA summary for all assays*

Description

Returns a summary of NA values across all assays.

Usage

```
naSummary(x)
```

Arguments

`x`	A PhysioExperiment object.

Value

A data.frame with NA statistics for each assay.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`hasNA` for quick NA check, `checkNA` for detailed NA validation, `replaceNA` for handling NA values

Examples

```
pe <- PhysioExperiment(
  assays = list(
    raw = matrix(c(1, NA, 3, 4), nrow = 2),
    filtered = matrix(1:4, nrow = 2)
  ),
  samplingRate = 100
)
naSummary(pe)
```

`nChannels` *Get number of channels*

Description

Get number of channels

Usage

```
nChannels(x)
```

Arguments

`x`	A PhysioExperiment object.

Value

Integer number of channels.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`channelNames` for channel labels, `channelInfo` for full channel metadata, `pickChannels` for subsetting channels

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  samplingRate = 100
)
nChannels(pe)  # 4
```

`nEvents` *Get number of events*

Description

Get number of events

Usage

```
nEvents(x)
```

Arguments

`x`	A PhysioEvents object.

Value

Integer count of events.

References

Delorme A, Makeig S (2004). "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics." *Journal of Neuroscience Methods*, 134(1), 9-21.

See Also

`getEvents` for retrieving events, `PhysioEvents` for the event constructor, `addEvents` for appending events

---

`PhysioEvents` *Create a PhysioEvents object*

Description

Create a PhysioEvents object

Usage

```
PhysioEvents(
  onset = numeric(0),
  duration = numeric(0),
  type = character(0),
  value = character(0)
)
```

Arguments

`onset`	Numeric vector of event onset times in seconds.
`duration`	Numeric vector of event durations in seconds.
`type`	Character vector of event types (e.g., "stimulus", "response").
`value`	Character vector of event values/labels.

Value

A PhysioEvents object.

References

Delorme A, Makeig S (2004). "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics." *Journal of Neuroscience Methods*, 134(1), 9-21.

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`setEvents` for attaching events to a PhysioExperiment, `getEvents` for retrieving events, `nEvents` for event count

Examples

```
# Create events for a simple experiment
events <- PhysioEvents(
  onset = c(1.0, 2.5, 4.0, 5.5),
  duration = c(0.5, 0.5, 0.5, 0.5),
  type = c("stimulus", "response", "stimulus", "response"),
  value = c("target", "hit", "distractor", "false_alarm")
)
events

# Create events with single type
stim_events <- PhysioEvents(
  onset = c(1, 2, 3, 4, 5),
  type = "stimulus"
)
```

`PhysioEvents-class` *Event management for PhysioExperiment*

Description

Functions for managing experimental events (triggers, markers, annotations) within PhysioExperiment objects. PhysioEvents class

Details

A simple S4 class to store event information as a DataFrame.

Slots

`events` A DataFrame containing event information with columns: onset (numeric), duration (numeric), type (character), value (character).

`PhysioExperiment` *Construct a PhysioExperiment object*

Description

Creates a new `PhysioExperiment` instance, which extends `SummarizedExperiment` with a `samplingRate` slot for physiological signal data.

Usage

```
PhysioExperiment(
  assays = S4Vectors::SimpleList(),
  rowData = NULL,
  colData = NULL,
  metadata = list(),
  samplingRate = as.numeric(NA)
)
```

Arguments

`assays`	A `SimpleList` (or coercible object) of assay arrays.
`rowData`	Feature-level metadata as a `DataFrame`.
`colData`	Sample-level metadata as a `DataFrame`.
`metadata`	Optional experiment-level metadata list.
`samplingRate`	Numeric scalar sampling rate in Hz.

Value

A `PhysioExperiment` object containing the supplied assays, row/column metadata, and sampling rate.

References

Huber, W., et al. (2015). "Orchestrating high-throughput genomic analysis with Bioconductor." *Nature Methods*, 12(2), 115-121. doi:10.1038/nmeth.3252

Morgan, M., et al. (2022). "S4Vectors: Foundation of vector-like and list-like containers in Bioconductor." R package.

See Also

samplingRate for accessing the sampling rate, defaultAssay for retrieving the first assay name, channelInfo for channel metadata, setEvents for attaching event information

Examples

```
# Create a simple PhysioExperiment with random EEG-like data
# 1000 time points, 4 channels
eeg_data <- matrix(rnorm(1000 * 4), nrow = 1000, ncol = 4)
colnames(eeg_data) <- c("Fz", "Cz", "Pz", "Oz")

pe <- PhysioExperiment(
  assays = list(raw = eeg_data),
  colData = S4Vectors::DataFrame(
    label = c("Fz", "Cz", "Pz", "Oz"),
    type = rep("EEG", 4)
  ),
  samplingRate = 250
)
pe

# Access sampling rate
samplingRate(pe)

# Create with multiple assays
pe2 <- PhysioExperiment(
  assays = list(raw = eeg_data, filtered = eeg_data * 0.5),
  samplingRate = 500
)
```

`PhysioExperiment-class`

PhysioExperiment class definition

Description

The `PhysioExperiment` class extends `SummarizedExperiment` to store multi-modal physiological signal data alongside metadata such as sampling rate. This file defines the class, its validity checks, and the user-facing constructor.

Slots

`samplingRate` Numeric scalar describing the acquisition frequency in Hz.

References

Huber, W., et al. (2015). "Orchestrating high-throughput genomic analysis with Bioconductor." *Nature Methods*, 12(2), 115-121. doi:10.1038/nmeth.3252

Morgan, M., et al. (2022). "S4Vectors: Foundation of vector-like and list-like containers in Bioconductor." R package.

See Also

`PhysioExperiment` for the constructor, `samplingRate` for accessing the sampling rate, `channelInfo` for channel metadata

---

`pickChannels` *Pick specific channels*

Description

Creates a new PhysioExperiment with only selected channels.

Usage

```
pickChannels(x, channels)
```

Arguments

`x`	A PhysioExperiment object.
`channels`	Integer indices or character names of channels to keep.

Value

A new PhysioExperiment with selected channels.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`dropChannels` for removing channels, `getChannelsByType` for finding channels by type, `channelNames` for available channel labels

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  colData = S4Vectors::DataFrame(label = c("Fz", "Cz", "Pz", "Oz")),
  samplingRate = 100
)

# Pick by index
pe_subset <- pickChannels(pe, c(1, 3))
nChannels(pe_subset)  # 2

# Pick by name
pe_frontal <- pickChannels(pe, c("Fz", "Cz"))
```

`rbindPhysio` *Combine PhysioExperiment objects by time*

Description

Concatenates two PhysioExperiment objects along the time (row) axis. Both objects must have the same number of channels and matching sampling rates. Event onsets in `y` are offset by the duration of `x`.

Usage

```
rbindPhysio(x, y)
```

Arguments

`x`	A PhysioExperiment object.
`y`	A PhysioExperiment object to concatenate.

Value

A `PhysioExperiment` object with time points from both `x` and `y` concatenated, combined rowData, and merged events.

References

Huber, W., et al. (2015). "Orchestrating high-throughput genomic analysis with Bioconductor." *Nature Methods*, 12(2), 115-121. doi:10.1038/nmeth.3252

Morgan, M., et al. (2022). "S4Vectors: Foundation of vector-like and list-like containers in Bioconductor." R package.

See Also

`cbindPhysio` for combining along the channel axis, `extractWindow` for extracting a time window, `[` for general subsetting

Examples

```
pe1 <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  samplingRate = 100
)
pe2 <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  samplingRate = 100
)

# Concatenate in time
pe_concat <- rbindPhysio(pe1, pe2)
length(pe_concat)  # 200
```

`removeEvents` *Remove events from a PhysioExperiment object*

Description

Remove events from a PhysioExperiment object

Usage

```
removeEvents(x, type = NULL, indices = NULL)
```

Arguments

`x`	A PhysioExperiment object.
`type`	Optional event types to remove. If NULL, removes all events.
`indices`	Optional integer indices of events to remove.

Value

The modified PhysioExperiment object.

References

Delorme A, Makeig S (2004). "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics." *Journal of Neuroscience Methods*, 134(1), 9-21.

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`getEvents` for retrieving events, `addEvents` for appending events, `setEvents` for replacing all events

`renameChannels` *Rename channels*

Description

Rename channels

Usage

```
renameChannels(x, old_names, new_names)
```

Arguments

`x`	A PhysioExperiment object.
`old_names`	Character vector of current names.
`new_names`	Character vector of new names.

Value

Modified PhysioExperiment object.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`channelNames` for reading channel labels, `channelInfo` for full channel metadata, `pickChannels` for subsetting channels

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  colData = S4Vectors::DataFrame(label = c("Fz", "Cz", "Pz", "Oz")),
  samplingRate = 100
)

# Rename channels
pe <- renameChannels(pe, c("Fz", "Cz"), c("F3", "C3"))
channelNames(pe)
```

`replaceNA` *Replace NA values in assay*

Description

Creates a new assay with NA values handled according to the specified method.

Usage

```
replaceNA(
  x,
  method = "interpolate",
  input_assay = NULL,
  output_assay = "na_handled"
)
```

Arguments

`x`	A PhysioExperiment object.
`method`	Method for handling NA (see handleNA).
`input_assay`	Input assay name. If NULL, uses default assay.
`output_assay`	Output assay name. Default is "na_handled".

Value

Modified PhysioExperiment with new assay.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`handleNA` for the underlying NA handling strategies, `checkNA` for NA validation, `hasNA` for quick NA presence check

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(c(1, NA, 3, NA, 5, 6), nrow = 3)),
  samplingRate = 100
)

# Interpolate NA values
pe <- replaceNA(pe, method = "interpolate")
```

`samplesToTime` *Convert sample indices to times*

Description

Convert sample indices to times

Usage

```
samplesToTime(x, samples)
```

Arguments

`x`	A PhysioExperiment object.
`samples`	Integer vector of sample indices.

Value

Numeric vector of times in seconds.

References

Delorme A, Makeig S (2004). "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics." *Journal of Neuroscience Methods*, 134(1), 9-21.

See Also

`timeToSamples` for the inverse conversion, `samplingRate` for the sampling rate, `timeIndex` for the full time vector

---

`samplingRate` *Accessors for PhysioExperiment*

Description

These helper functions expose common slots and derived quantities for `PhysioExperiment` objects. Get or set sampling rate

Usage

```
samplingRate(x)

samplingRate(x) <- value

## S4 replacement method for signature 'PhysioExperiment'
samplingRate(x) <- value
```

Arguments

`x`	A PhysioExperiment object.
`value`	Numeric scalar for the new sampling rate in Hz.

Value

For `samplingRate(x)`: a numeric scalar giving the sampling rate in Hz. For `samplingRate(x) <- value`: the modified `PhysioExperiment` object (returned invisibly).

References

Huber, W., et al. (2015). "Orchestrating high-throughput genomic analysis with Bioconductor." *Nature Methods*, 12(2), 115-121. doi:10.1038/nmeth.3252

Morgan, M., et al. (2022). "S4Vectors: Foundation of vector-like and list-like containers in Bioconductor." R package.

See Also

`PhysioExperiment` for the constructor, `defaultAssay` for the default assay name, `duration` for signal duration, `timeIndex` for time point vector

Examples

```
# Create example data
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(100), nrow = 10)),
  samplingRate = 250
)

# Get sampling rate
samplingRate(pe)

# Set sampling rate
samplingRate(pe) <- 500
samplingRate(pe)
```

`setChannelTypes` *Set channel types*

Description

Assigns types (EEG, EMG, EOG, etc.) to channels.

Usage

```
setChannelTypes(x, types)
```

Arguments

`x`	A PhysioExperiment object.
`types`	Named character vector or list mapping channel names/indices to types. If unnamed, applies types in order.

Value

Modified PhysioExperiment object.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`getChannelsByType` for querying channels by type, `channelInfo` for full channel metadata, `setChannelUnits` for assigning physical units

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  colData = S4Vectors::DataFrame(label = c("Fz", "EOG1", "EMG1", "Oz")),
  samplingRate = 100
)

# Set all channels to same type
pe <- setChannelTypes(pe, "EEG")

# Set specific channel types by name
pe <- setChannelTypes(pe, c(EOG1 = "EOG", EMG1 = "EMG"))
```

`setChannelUnits` *Set channel units*

Description

Assigns physical units to channels.

Usage

```
setChannelUnits(x, units)
```

Arguments

`x`	A PhysioExperiment object.
`units`	Character vector or named list of units.

Value

Modified PhysioExperiment object.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`setChannelTypes` for assigning channel types, `channelInfo` for full channel metadata

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  samplingRate = 100
)

# Set all channels to same unit
pe <- setChannelUnits(pe, "uV")
```

`setElectrodePositions` *Set electrode positions*

Description

Assigns 3D electrode positions to channels.

Usage

```
setElectrodePositions(x, positions)
```

Arguments

`x`	A PhysioExperiment object.
`positions`	A data.frame or matrix with columns x, y, z and rows matching channels. Row names should match channel names.

Value

Modified PhysioExperiment object.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`getElectrodePositions` for reading positions, `applyMontage` for standard electrode layouts, `channelInfo` for full channel metadata

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(300), nrow = 100, ncol = 3)),
  colData = S4Vectors::DataFrame(label = c("Fz", "Cz", "Pz")),
  samplingRate = 100
)

# Set electrode positions
positions <- data.frame(
  x = c(0, 0, 0),
  y = c(0.71, 0, -0.71),
  z = c(0.71, 1, 0.71)
)
pe <- setElectrodePositions(pe, positions)
```

`setEvents` *Set events for a PhysioExperiment object*

Description

Set events for a PhysioExperiment object

Usage

```
setEvents(x, events)
```

Arguments

`x`	A PhysioExperiment object.
`events`	A PhysioEvents object or a data.frame with columns: onset, duration, type, value.

Value

The modified PhysioExperiment object.

References

Delorme A, Makeig S (2004). "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics." *Journal of Neuroscience Methods*, 134(1), 9-21.

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`getEvents` for retrieving events, `addEvents` for appending events, `PhysioEvents` for the event constructor

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(1000), nrow = 100)),
  samplingRate = 100
)

# Set events using PhysioEvents object
events <- PhysioEvents(onset = c(1, 2, 3), type = "stimulus")
pe <- setEvents(pe, events)

# Set events using data.frame
pe <- setEvents(pe, data.frame(onset = c(1, 2), type = "response"))
```

`setReference` *Set reference electrode*

Description

Records the reference electrode used for the recording.

Usage

```
setReference(x, reference)
```

Arguments

`x`	A PhysioExperiment object.
`reference`	Character string naming the reference electrode.

Value

Modified PhysioExperiment object.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`getReference` for reading the reference, `channelInfo` for full channel metadata, `applyMontage` for applying standard montages

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  samplingRate = 100
)

# Set reference electrode
pe <- setReference(pe, "average")
getReference(pe)  # "average"
```

`show,PhysioEvents-method` *Show method for PhysioEvents*

Description

Show method for PhysioEvents

Usage

```
## S4 method for signature 'PhysioEvents'
show(object)
```

Arguments

`object`	A PhysioEvents object.

See Also

`PhysioEvents` for the constructor, `nEvents` for event count

`show,PhysioExperiment-method` *S4 Methods for PhysioExperiment*

Description

Standard S4 methods for PhysioExperiment objects including show, subsetting, and combining. Show method for PhysioExperiment

Usage

```
## S4 method for signature 'PhysioExperiment'
show(object)
```

Arguments

`object`	A PhysioExperiment object.

Details

Displays a summary of the PhysioExperiment object.

Value

Invisibly returns `NULL`. Called for its side effect of printing a human-readable summary to the console.

References

Huber, W., et al. (2015). "Orchestrating high-throughput genomic analysis with Bioconductor." *Nature Methods*, 12(2), 115-121. doi:10.1038/nmeth.3252

Morgan, M., et al. (2022). "S4Vectors: Foundation of vector-like and list-like containers in Bioconductor." R package.

See Also

`PhysioExperiment` for the constructor, `summary,PhysioExperiment-method` for channel-level statistics, `as.data.frame,PhysioExperiment-method` for conversion to data.frame

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  colData = S4Vectors::DataFrame(label = c("Fz", "Cz", "Pz", "Oz")),
  samplingRate = 100
)
pe  # Displays summary
```

`summary,PhysioExperiment-method`

Summary statistics for PhysioExperiment

Description

Computes per-channel summary statistics (min, max, mean, sd, median) for the default assay. For 3D arrays, values are first averaged across the third dimension.

Usage

```
## S4 method for signature 'PhysioExperiment'
summary(object, ...)
```

Arguments

`object`	A PhysioExperiment object.
`...`	Additional arguments (not used).

Value

A `data.frame` with columns `channel`, `min`, `max`, `mean`, `sd`, and `median`, with one row per channel. Returns an empty `data.frame` if no assays are present.

References

R Core Team (2024). "R: A Language and Environment for Statistical Computing." R Foundation for Statistical Computing, Vienna, Austria.

See Also

`PhysioExperiment` for the constructor, `as.data.frame,PhysioExperiment-method` for full data export, `channelNames` for channel labels

Examples

```
pe <- PhysioExperiment(
  assays = list(raw = matrix(rnorm(400), nrow = 100, ncol = 4)),
  colData = S4Vectors::DataFrame(label = c("Fz", "Cz", "Pz", "Oz")),
  samplingRate = 100
)
summary(pe)
```

`timeIndex` *Time index helper*

Description

Computes a time vector for the default assay using the object's sampling rate.

Usage

```
timeIndex(x)
```

Arguments

`x`	A `PhysioExperiment` instance.

Value

Numeric vector of time points in seconds.

References

Oostenveld R, Fries P, Maris E, Schoffelen JM (2011). "FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data." *Computational Intelligence and Neuroscience*, 2011, 156869.

See Also

`samplingRate` for the sampling rate, `duration` for signal duration, `timeToSamples` for converting times to sample indices

---

`timeToSamples` *Convert event times to sample indices*

Description

Convert event times to sample indices

Usage

```
timeToSamples(x, times)
```

Arguments

`x`	A PhysioExperiment object.
`times`	Numeric vector of times in seconds.

Value

Integer vector of sample indices.

References

Delorme A, Makeig S (2004). "EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics." *Journal of Neuroscience Methods*, 134(1), 9-21.

See Also

`samplesToTime` for the inverse conversion, `samplingRate` for the sampling rate, `timeIndex` for the full time vector

Index