Package 'musclesyneRgies'

August 26, 2026

Title Extract Muscle Synergies from Electromyography

Version 1.3.1

Description Provides a framework to factorise electromyography (EMG) data. Tools are provided for raw data pre-processing, non negative matrix factorisation, classification of factorised data and plotting of obtained outcomes. In particular, reading from ASCII files is supported, along with wide-used filtering approaches to process EMG data. All steps include one or more sensible defaults that aim at simplifying the workflow. Yet, all functions are largely tunable at need. Example data sets are included.

License MIT + file LICENSE

URL https://github.com/alesantuz/musclesyneRgies

BugReports https://github.com/alesantuz/musclesyneRgies/issues

Depends R (>= 4.1.0)

Imports FNN, ggplot2, graphics, grDevices, gridExtra, plyr, proxy, reshape2, signal, stats, umap

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Language en-GB

LazyData true

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Alessandro Santuz [aut, cre] (ORCID: <https://orcid.org/0000-0002-6577-5101>)

Maintainer Alessandro Santuz <alessandro.santuz@gmail.com>

Repository CRAN

Date/Publication 2026-08-26 09:00:10 UTC

Contents

act_pattern . . . 2
act_patterns . . . 3
classify_kmeans . . . 3
CoA . . . 5
cossim . . . 6
filtEMG . . . 7
FILT_EMG . . . 8
FWHM . . . 9
HFD . . . 10
Hurst . . . 11
normEMG . . . 13
plot_classified_syns . . . 14
plot_classified_syns_UMAP . . . 15
plot_meanEMG . . . 16
plot_rawEMG . . . 17
plot_syn_trials . . . 19
rawdata . . . 20
RAW_DATA . . . 22
sMLE . . . 23
subsetEMG . . . 24
SYNS . . . 25
synsNMF . . . 26

Index **29**

`act_pattern` *Single activation pattern example (30 cycles)*

Description

A data frame containing one activation pattern extracted from one wild type mouse walking on a treadmill.

Usage

```
act_pattern
```

Format

A data frame of two columns:

`time` Normalised time in points.

`signal` Activation pattern

Source

doi:10.1152/jn.00360.2020

`act_patterns` *All activation patterns of one synergy example (30 cycles)*

Description

A demo and incomplete `musclesyneRgies` object containing time info and three activation patterns extracted from one wild type mouse walking on a treadmill.

Usage

```
act_patterns
```

Format

A data frame of four columns:

`time` Normalised time in points.

`Syn1` Activation pattern of synergy 1

`Syn2` Activation pattern 2

`Syn3` Activation pattern of synergy 3

Source

doi:10.1152/jn.00360.2020

`classify_kmeans` *Muscle synergy classification with k-means*

Description

Muscle synergy classification with k-means

Usage

```
classify_kmeans(
  x,
  clusters = NA,
  MSE_lim = 0.001,
  inspect = FALSE,
  show_plot = FALSE
)
```

Arguments

x	A list of musclesyneRgies objects
clusters	The number of clusters can be specified a priori if needed
MSE_lim	Mean squared error threshold for determining the minimum number of clusters
inspect	Logical, ask for interactive re-ordering or go fully automated?
show_plot	Logical, to decide whether plots should be plotted in the active graphic device

Details

This function must be applied to a list with a sufficient amount of trials, otherwise the classification will not work. Typically, at least 10 trials for the same condition are needed for satisfactory classification. If show_plot is TRUE (default) plots are also shown in the active graphic device. Plots can then be saved with the preferred export method, such as ggplot2::ggsave. The algorithm used is the default for stats::kmeans (Hartigan and Wong, 1979), which is known for its robustness to local minima. Nonetheless, the stochastic nature of the algorithm should prompt the user to attempt a few classifications and analyse their stability, before drawing conclusions on e.g. the number of fundamental synergies and/or their function. While the default parameters are optimised for human locomotion, it is suggested to test the function with different mean squared error thresholds, which is a crucial quantity to determine the number of clusters. Inspection and plotting are as well highly recommended to gain more insight into the classification process.

Value

List of musclesyneRgies objects, each with elements:

- syns factorisation rank or minimum number of synergies
- M muscle weights (time-invariant coefficients)
- P activation patterns (time-dependent coefficients)
- V original data
- Vr reconstructed data
- iterations number of iterations to convergence
- R2 quality of reconstruction (coefficient of determination)
- rank_type was the rank fixed or variable?
- classification classification type (k-means)

Examples

```
# Load some data
data(SYNS)
# Classify synergies
SYNS_classified <- classify_kmeans(SYNS)
```

CoA *Centre of activity*

Description

Centre of activity

Usage

```
CoA(x, tol_rel = 1e-06)
```

Arguments

x	A numeric time series of length n (samples represent bin centres at 1..n).
tol_rel	Relative tolerance (0–1). If the mean resultant length is less than this value, the CoA is considered undefined and NA is returned. Default is 1e-6.

Value

The centre of activity of the time series, calculated with circular statistics: a numeric scalar in (0, n] giving the centre of activity on the 1..n axis.

References

Martino, G. et al. Locomotor patterns in cerebellar ataxia. J. Neurophysiol. 112, 2810–2821 (2014).

Examples

```
# Number of users connected to the Internet through a server every minute
ts <- datasets::WWWusage[1:80]

# Calculate CoA
ts_CoA <- CoA(ts)

# Plot
plot(ts, ty = "l", xlab = "Time", ylab = "Number of users")
graphics::abline(v = ts_CoA, lwd = 2, lty = 2)
```

`cossim` *Cosine similarity*

Description

Cosine similarity

Usage

```
cossim(x, y)
```

Arguments

x	A numeric vector
y	A numeric vector

Value

The cosine similarity between two vectors

References

Leydesdorff, L. (2005) Similarity Measures, Author Cocitation Analysis,and Information Theory. JASIST 56(7), pp.769-772.

Fridolin Wild (2022) lsa: Latent Semantic Analysis, R package version 0.73.3 https://CRAN.R-project.org/package=lsa

Examples

```
data("act_pattern")

# Calculate the cosine similarity between an activation pattern and itself
similarity <- cossim(
x = act_pattern$signal,
y = act_pattern$signal
)

# Calculate the cosine similarity between a non-negative activation pattern and its negative
similarity <- cossim(
x = act_pattern$signal,
y = -act_pattern$signal
)
```

`filtEMG` *To filter raw EMG*

Description

To filter raw EMG

Usage

```
filtEMG(
  x,
  demean = TRUE,
  rectif = "fullwave",
  HPf = 50,
  HPo = 4,
  LPf = 20,
  LPo = 4,
  min_sub = TRUE,
  ampl_norm = TRUE
)
```

Arguments

`x`	Object of class `EMG` with elements `cycles` and `emg`
`demean`	Logical: should EMG be demeaned?
`rectif`	Rectification type: "fullwave", "halfwave" or "none"
`HPf`	High-pass filter cut-off frequency, use 0 to exclude high-pass filtering
`HPo`	High-pass filter order
`LPf`	Low-pass filter cut-off frequency, use 0 to exclude Low-pass filtering
`LPo`	Low-pass filter order
`min_sub`	Logical: should the minimum be subtracted?
`ampl_norm`	Logical: should amplitude be normalised?

Details

Lists in the correct format can be created with the function `rawdata()`. The first column of each `emg` element must be time in the same units as those used for `cycles` (e.g., [s] or [ms]).

Value

Object of class `EMG` with elements:

- `cycles` data frame containing cycle timings, with as many columns as many cycle subdivisions are wanted
- `emg` data frame containing filtered EMG data in columns, first column is time

References

Santuz, A., Ekizos, A., Janshen, L., Baltzopoulos, V. & Arampatzis, A. On the Methodological Implications of Extracting Muscle Synergies from Human Locomotion. Int. J. Neural Syst. 27, 1750007 (2017).

Examples

```
# Load some data
data("RAW_DATA")
# Filter raw EMG
filtered_EMG <- lapply(
  RAW_DATA,
  function(x) {
    filtEMG(x,
      HPf = 50,
      HPo = 4,
      LPf = 20,
      LPo = 4
    )
  }
)
```

FILT_EMG *Filtered EMG example*

Description

A list containing filtered and time-normalised electromyographic (EMG) human data from the right-side lower limb recorded during one walking trial.

Usage

```
FILT_EMG
```

Format

A list containing one object of class `EMG` with elements `cycles` and `emg`, both data frames.

`ID0012_TW_01` Object of class `EMG` containing the two following data frames:

`cycles` Gait cycle-timings, in seconds.

`emg` Filtered and time-normalised EMG, first column is time in points, muscles named as:

ME=gluteus medius
MA=gluteus maximus
FL=tensor fasciae latae
RF=rectus femoris
VM=vastus medialis
VL=vastus lateralis

ST=semitendinosus
BF=biceps femoris
TA=tibialis anterior
PL=peroneus longus
GM=gastrocnemius medialis
GL=gastrocnemius lateralis
SO=soleus

Source

doi:10.1016/j.isci.2019.100796

`FWHM` *Full width at half maximum*

Description

Full width at half maximum

Usage

```
FWHM(x, sub_minimum = TRUE)
```

Arguments

`x`	A time series (numeric)
`sub_minimum`	Logical; should the minimum be subtracted before amplitude normalisation?

Value

The full width at half maximum of the time series.

References

Martino, G. et al. Locomotor patterns in cerebellar ataxia. J. Neurophysiol. 112, 2810–2821 (2014).

Examples

```
# Number of users connected to the Internet through a server every minute
ts <- datasets::WWWusage

# Calculate FWHM
ts_FWHM <- FWHM(ts)

# Half maximum (for the plots)
hm <- min(ts) + (max(ts) - min(ts)) / 2
hm_plot <- ts
hm_plot[which(hm_plot > hm)] <- hm
hm_plot[which(hm_plot < hm)] <- NA
```

```
# Plots
plot(ts, ty = "l", xlab = "Time", ylab = "Number of users")
lines(hm_plot, lwd = 3, col = 2)
```

HFD *Higuchi's fractal dimension*

Description

Higuchi's fractal dimension

Usage

```
HFD(P, k_max = 10)
```

Arguments

P	A time series (numeric)
k_max	Maximum window length in points

Details

The Higuchi's fractal dimension is a measure of local complexity and it increases together with the "roughness" of the time series at a single cycle level (thus the term "local"). Higuchi's fractal dimension values range from 1 to 2, with increasing values correlating to increasingly complex data and Higuchi's fractal dimension = 1.5 indicating random Gaussian noise (Higuchi, 1988; Anmuth et al., 1994; Kesić & Spasić, 2016) For locomotor activation patterns, only the most linear part of the log-log plot should be used, as reported in Santuz, Akay (2020).

Value

A list with elements:

- `loglog` containing the log-log plot from which the HFD is calculated
- `Higuchi` containing the Higuchi's fractal dimension of the time series.

References

Higuchi, T. Approach to an irregular time series on the basis of the fractal theory. Phys. D Nonlinear Phenom. 31, 277–283 (1988).

Anmuth C. J., Goldberg G. & Mayer N. H. Fractal dimension of electromyographic signals recorded with surface electrodes during isometric contractions is linearly correlated with muscle activation. Muscle Nerve 17, 953–954 (1994).

Kesić S. & Spasić S. Z. Application of Higuchi's fractal dimension from basic to clinical neurophysiology: A review. Comput Methods Programs Biomed 133, 55–70 (2016).

Santuz, A. & Akay, T. Fractal analysis of muscle activity patterns during locomotion: pitfalls and how to avoid them. J. Neurophysiol. 124, 1083–1091 (2020).

Examples

```
# Measurements of the annual flow of the river Nile at Aswan
flow <- datasets::Nile

# Calculate HFD
fractal_dimension <- HFD(flow)$Higuchi
message("Higuchi's fractal dimension: ", round(fractal_dimension, 3))

# Thirty-cycle activation pattern from Santuz & Akay (2020)
data(act_pattern)
fractal_dimension <- HFD(act_pattern$signal)$Higuchi
message("Higuchi's fractal dimension: ", round(fractal_dimension, 3))
```

Hurst *Hurst exponent*

Description

Hurst exponent

Usage

```
Hurst(P, min_win = 2)
```

Arguments

P	A time series (numeric)
min_win	Minimum window length in points

Details

`Hurst` calculates the Hurst exponent based on the R/S approach as in Hurst (1951). The Hurst exponent is a measure of global complexity and it increases if the "accuracy" of the time series decreases across several cycles (thus the term "global"). The Hurst exponent can vary between 0 and 1. For 0.5 < Hurst exponent < 1, in the long-term high values in the time series (the activation pattern in our case) will be probably followed by other high values and a positive or negative trend is visible (Mandelbrot, 1983; Gneiting & Schlather, 2004). For 0 < Hurst exponent < 0.5, in the long term high values in the series will be probably followed by low values, with a frequent switch between high and low values (Mandelbrot, 1983; Gneiting & Schlather, 2004). Hurst exponent = 0.5 corresponds to a completely random series (Mandelbrot, 1983; Qian & Rasheed, 2004). In other words, values of Hurst exponent approaching 0.5 from both ends indicate more complex (or

random) behaviour of the time series (Hurst, 1951). For locomotor activation patterns, the minimum window length should be bigger than the period (i.e. the length of each cycle), as reported in Santuz, Akay (2020).

Value

A list with elements:

- `loglog` containing the log-log plot from which the HFD is calculated
- `Hurst` containing the Higuchi's fractal dimension of the time series.

References

Hurst, H. E. Long-term storage capacity of reservoirs. Trans. Am. Soc. Civ. Eng. 116, 770-808 (1951).

Mandelbrot B. B. The Fractal Geometry of Nature. W. H. Freeman and Co., New York (1983).

Gneiting T .& Schlather M. Stochastic Models That Separate Fractal Dimension and the Hurst Effect. SIAM Rev 46, 269–282 (2004).

Qian B. & Rasheed K. Hurst exponent and financial market predictability. In Proceedings of the Second IASTED International Conference on Financial Engineering and Applications, pp. 203–209 (2004).

Santuz, A. & Akay, T. Fractal analysis of muscle activity patterns during locomotion: pitfalls and how to avoid them. J. Neurophysiol. 124, 1083-1091 (2020).

Examples

```
# Measurements of the annual flow of the river Nile at Aswan
flow <- datasets::Nile

# Calculate Hurst exponent
H <- Hurst(flow)$Hurst
message("Hurst exponent: ", round(H, 3))

# Thirty-cycle activation pattern from Santuz & Akay (2020)
data(act_pattern)
H <- Hurst(act_pattern$signal, min_win = max(act_pattern$time))$Hurst
message("Hurst exponent: ", round(H, 3))
```

normEMG *To time-normalise filtered EMG*

Description

To time-normalise filtered EMG

Usage

```
normEMG(x, trim = TRUE, cy_max = NA, cycle_div = NA)
```

Arguments

x	Object of class EMG with elements cycles and emg
trim	Logical: should first and last cycle be trimmed to remove filtering effects?
cy_max	Maximum number of cycles to be considered
cycle_div	A vector or one dimensional array with the number of points each cycle should be normalised to

Details

Lists in the correct format can be created with the function rawdata(). The first column of each emg element must be time in the same units as those used for cycles (e.g., [s] or [ms]).

Value

Object of class EMG with elements:

- cycles data frame containing cycle timings, with as many columns as many cycle subdivisions are wanted
- emg data frame containing filtered and time-normalised EMG data in columns, first column is time

References

Santuz, A., Ekizos, A., Janshen, L., Baltzopoulos, V. & Arampatzis, A. On the Methodological Implications of Extracting Muscle Synergies from Human Locomotion. Int. J. Neural Syst. 27, 1750007 (2017).

Examples

```
# Load some data
data("RAW_DATA")
# Filter raw EMG
filtered_EMG <- lapply(RAW_DATA, function(x) {
  filtEMG(x, HPf = 50, HPo = 4, LPf = 20, LPo = 4)
})
# Time-normalise filtered EMG, including three cycles and trimming first and last
filt_norm_EMG <- lapply(filtered_EMG, function(x) {
  normEMG(
  x,
  cy_max = 3,
  cycle_div = c(100, 100))
})
```

---

plot_classified_syns *Plot muscle synergies*

Description

Plot muscle synergies

Usage

```
plot_classified_syns(
  x,
  dark_mode = FALSE,
  line_size = 0.9,
  dot_size = 0.1,
  line_col = "black",
  sd_col = "grey80",
  ylim_min_M = 0,
  ylim_max_M = 1,
  ylim_min_P = -0.2,
  ylim_max_P = 1.2,
  condition = NA,
  show_plot = TRUE
)
```

Arguments

x	List of objects of class musclesyneRgies (must be classified)
dark_mode	To enable dark mode
line_size	Line thickness
dot_size	Dot size on muscle weights (NA to remove dots)
line_col	Line colour

sd_col	Standard deviation ribbon colour (NA to remove ribbon)
ylim_min_M	y-axis lower limit for muscle weights
ylim_max_M	y-axis upper limit for muscle weights
ylim_min_P	y-axis lower limit for activation patterns
ylim_max_P	y-axis upper limit for activation patterns
condition	Character: the condition that is being analysed, for archiving purposes
show_plot	Logical, to decide whether plots should be plotted in the active graphic device

Details

If show_plot is TRUE (default) plots are also shown in the active graphic device. Plots can then be saved with the preferred export method, such as ggplot2::ggsave.

Value

Global plot containing the average classified muscle synergies and individual trials (muscle weights) or standard deviations (activation patterns)

Examples

```
# Load some data
data(SYNS)

# Classify synergies with k-means
SYNS_classified <- classify_kmeans(SYNS)

# Save plot of classified synergies
pp <- plot_classified_syns(
  SYNS_classified,
  dark_mode = TRUE,
  line_col = "tomato1",
  sd_col = "tomato4",
  condition = "TW",
  show_plot = FALSE
)
```

---

plot_classified_syns_UMAP

Plot 2D UMAP of muscle synergies

Description

Plot 2D UMAP of muscle synergies

Usage

```
plot_classified_syns_UMAP(x, condition, show_plot = TRUE)
```

Arguments

x	List of objects of class musclesyneRgies (must be classified)
condition	Character: the condition that is being analysed, for archiving purposes
show_plot	Logical, to decide whether plots should be plotted in the active graphic device

Details

If show_plot is TRUE (default) plots are also shown in the active graphic device. Plots can then be saved with the preferred export method, such as ggplot2::ggsave.

Value

2D UMAP plot of classified synergies.

Examples

```
# Load some data
data(SYNS)

# Classify synergies with k-means
SYNS_classified <- classify_kmeans(SYNS)

# Save plot
pp <- plot_classified_syns_UMAP(SYNS_classified,
  condition = "TW",
  show_plot = FALSE
)
```

---

plot_meanEMG *Plot EMG averaged across all cycles*

Description

Plot EMG averaged across all cycles

Usage

```
plot_meanEMG(
  x,
  trial,
  row_number = NA,
  col_number = 1,
  dark_mode = FALSE,
  line_size = 0.6,
  line_col = "black",
  show_plot = TRUE
)
```

Arguments

x	A data frame containing filtered EMG organised in columns
trial	Character: the name of the considered trial, for archiving purposes
row_number	How many rows should the final plot be divided into?
col_number	How many columns should the final plot be divided into?
dark_mode	To enable dark mode
line_size	Line thickness
line_col	Line colour
show_plot	Logical, to decide whether plots should be plotted in the active graphic device

Details

If show_plot is TRUE (default) plots are also shown in the active graphic device. Plots can then be saved with the preferred export method, such as ggplot2::ggsave.

Value

Exports average filtered and normalised EMG.

Examples

```
# Load some data
data(FILT_EMG)

# Save a plot of the only present trial with the average filtered and time-normalised EMG
pp <- plot_meanEMG(FILT_EMG[[1]],
  trial = names(FILT_EMG)[1],
  row_number = 4,
  col_number = 4,
  dark_mode = TRUE,
  line_col = "tomato3",
  show_plot = FALSE
)
```

---

plot_rawEMG *Plot raw EMG*

Description

Plot raw EMG

Usage

```
plot_rawEMG(
  x,
  trial,
  plot_time = 3,
  start = 1,
  row_number = NA,
  col_number = 1,
  dark_mode = FALSE,
  line_size = 0.3,
  line_col = "black",
  show_plot = TRUE
)
```

Arguments

x	Object of class EMG with elements cycles and emg
trial	Character: the name of the considered trial, for archiving purposes
plot_time	How many seconds of data should be plotted?
start	At which data point should the plot start?
row_number	How many rows should the final plot be divided into?
col_number	How many columns should the final plot be divided into?
dark_mode	To enable dark mode
line_size	Line thickness
line_col	Line colour
show_plot	Logical, to decide whether plots should be plotted in the active graphic device

Details

If show_plot is TRUE (default) plots are also shown in the active graphic device. Plots can then be saved with the preferred export method, such as ggplot2::ggsave.

Value

Plots raw EMG trials of the specified length.

Examples

```
# Load some data
data(RAW_DATA)

# Save a plot with the first (and only) trial in RAW_DATA, first three seconds, in dark mode
plot_rawEMG(RAW_DATA[[1]],
  trial = names(RAW_DATA)[1],
  row_number = 4,
  col_number = 4,
  dark_mode = TRUE,
```

```
  line_col = "tomato3",
  show_plot = FALSE
)
```

`plot_syn_trials` *Plot muscle synergies (individual trials)*

Description

Plot muscle synergies (individual trials)

Usage

```
plot_syn_trials(
  x,
  max_syns,
  trial,
  dark_mode = FALSE,
  line_size = 0.6,
  line_col = "black",
  sd_col = "grey80",
  show_plot = TRUE
)
```

Arguments

`x`	Object of class `musclesyneRgies`
`max_syns`	Number of synergies to be plotted or how many rows should the final panel be divided into
`trial`	Character: the name of the considered trial, for archiving purposes
`dark_mode`	To enable dark mode
`line_size`	Line thickness
`line_col`	Line colour
`sd_col`	Standard deviation ribbon colour
`show_plot`	Logical, to decide whether plots should be plotted in the active graphic device

Details

If `show_plot` is TRUE (default) plots are also shown in the active graphic device. Plots can then be saved with the preferred export method, such as `ggplot2::ggsave`.

Value

Plots of the unclassified synergies, trial by trial.

Examples

```
# Load some data
data(SYNS)

# Find maximum number of synergies
max_syns <- max(unlist(lapply(SYNS, function(x) x$syns)))

# Save a plot with the first (and only, in this case) trial in the list
pp <- plot_syn_trials(SYNS[[1]],
  max_syns = max_syns,
  trial = names(SYNS)[1],
  dark_mode = TRUE,
  line_size = 0.8,
  line_col = "tomato1",
  sd_col = "tomato4",
  show_plot = FALSE
)
```

---

rawdata *Import RData or ASCII data into R*

Description

Import RData or ASCII data into R

Usage

```
rawdata(path_cycles = NA, path_emg = NA, header_cycles, header_emg = TRUE)
```

Arguments

path_cycles	Optional, path where cycle timing files are located
path_emg	Optional, path where raw EMG files are located
header_cycles	Logical, are the cycle files containing a named header (the header is optional)?
header_emg	Logical, are the raw EMG files containing a named header (they should)?

Details

Supported are R lists saved as RData files or tab- or comma-separated files readable through `read.table()` or `read.csv()`. The first column of each raw emg file must be time in the same units as those used for the cycle timings (e.g., [s] or [ms]). If reading from RData files, please call cycles `CYCLE_TIMES.RData` and raw EMG `RAW_EMG.RData`. Lists must be saved with `save()`.

Value

List of objects of class EMG, each with elements:

- cycles data frame containing cycle timings, with as many columns as many cycle subdivisions are wanted
- emg data frame containing raw EMG data in columns, first column must be time in the same units as in the cycle timings

Examples

```
# Load built-in data set
data("RAW_DATA")

# Get current working directory
data_path <- getwd()
data_path <- paste0(data_path, .Platform$file.sep)

# Create two conveniently-named subfolders if they don't already exist
# (if they exist, please make sure they're empty!)
dir.create("cycles", showWarnings = FALSE)
dir.create("emg", showWarnings = FALSE)

# Export ASCII data from built-in data set to the new subfolders
write.table(RAW_DATA[[1]]$cycles,
  file = paste0(data_path, "cycles", .Platform$file.sep, names(RAW_DATA)[1], ".txt"),
  sep = "\t", row.names = FALSE, col.names = FALSE
)
write.table(RAW_DATA[[1]]$emg,
  file = paste0(data_path, "emg", .Platform$file.sep, names(RAW_DATA)[1], ".txt"),
  sep = "\t", row.names = FALSE
)

# Run the function to parse ASCII files into objects of class `EMG`
raw_data_from_files <- rawdata(
  path_cycles = paste0(data_path, "/cycles/"),
  path_emg = paste0(data_path, "/emg/"),
  header_cycles = FALSE
)

# Check data in the new folders if needed before running the following (will delete!)

# Delete folders
unlink("cycles", recursive = TRUE)
unlink("emg", recursive = TRUE)
```

RAW_DATA *Raw EMG example*

Description

A list containing electromyographic (EMG) human data from the right-side lower limb recorded during one walking trial.

Usage

```
RAW_DATA
```

Format

A list containing one object of class `EMG` with elements `cycles` and `emg`, both data frames.

`ID0012_TW_01` Object of class `EMG` containing the two following data frames:

`cycles` Gait cycle-timings, in seconds.

`emg` Raw EMG, first column is time in seconds, muscles named as:

ME=gluteus medius
MA=gluteus maximus
FL=tensor fasciae latae
RF=rectus femoris
VM=vastus medialis
VL=vastus lateralis
ST=semitendinosus
BF=biceps femoris
TA=tibialis anterior
PL=peroneus longus
GM=gastrocnemius medialis
GL=gastrocnemius lateralis
SO=soleus

Source

doi:10.1016/j.isci.2019.100796

sMLE *Short-term maximum Lyapunov exponents*

Description

Short-term maximum Lyapunov exponents

Usage

```
sMLE(synergies, mean_period, future_pts, norm, pts, R2_threshold = 0.9)
```

Arguments

synergies	A musclesyneRgies object
mean_period	To locate the nearest neighbour of each point on the state space trajectory
future_pts	To limit the number of points "in the future" that are being searched
norm	Type of normalisation ("u" for minimum subtraction and normalisation to the maximum, "z" for subtracting the mean and then divide by the standard deviation)
pts	Minimum number of points needed to linearly approximate the first part of the divergence curve
R2_threshold	Threshold for calculating the slope of the divergence curve

Details

The mean period is intended to exclude temporally close points. In gait, values are usually plus/minus half gait cycle. Future points usually correspond in gait to one to two gait cycles. Please consider that a sufficient amount of cycles in order to compute meaningful sMLE. For locomotor activation patterns, 30 gait cycles have been shown to be sensitive to perturbations (Santuz et al. 2020). However, in the more classical and widespread use on kinematic data, more are usually needed (Kang and Dingwell, 2006).

Value

A list with elements:

- divergences containing the average logarithmic divergence curve
- sMLE the short-term Maximum Lyapunov exponent
- R2 the goodness of fit of the most linear part of the divergence curve

References

Rosenstein, M.T., Collins, J.J., and De Luca, C.J. (1993). A practical method for calculating largest Lyapunov exponents from small data sets. Phys. D 65, 117–134.

Santuz A, Brüll L, Ekizos A, Schroll A, Eckardt N, Kibele A, et al. Neuromotor Dynamics of Human Locomotion in Challenging Settings. iScience. 2020;23: 100796.

Kang H.G., and Dingwell J.B. (2006). Intra-session reliability of local dynamic stability of walking. Gait Posture. 24(3) 386-390.

Examples

```
# Load some activation patterns
data("act_patterns")
# Calculate sMLE of activation patterns in the muscle synergy space
short_term_MLE <- sMLE(act_patterns,
  mean_period = 80,
  future_pts = 200,
  norm = "z",
  pts = 30
)
```

`subsetEMG` *Subset raw EMG*

Description

Subset raw EMG

Usage

```
subsetEMG(x, cy_max, cy_start = 1)
```

Arguments

`x`	Objects of class `EMG` with elements `cycles` and `emg`
`cy_max`	Maximum number of cycles to be considered
`cy_start`	From which cycle should the subset begin?

Details

Lists in the correct format can be created with the function `rawdata()`. The first column of each `emg` element must be time in the same units as those used for `cycles` (e.g., [s] or [ms]). For locomotion, thirty cycles are enough for proper synergy extraction (Oliveira et al. 2014).

Value

Object of class EMG with elements:

- cycles data frame containing cycle timings, with as many columns as many cycle subdivisions are wanted
- emg data frame containing raw EMG data in columns, first column is time

References

Oliveira, A. S. C., Gizzi, L., Farina, D. & Kersting, U. G. Motor modules of human locomotion: influence of EMG averaging, concatenation, and number of step cycles. Front. Hum. Neurosci. 8, 335 (2014).

Examples

```
# Load some data
data("RAW_DATA")
# Subset example raw data to the first 3 cycles
RAW_DATA_sub <- lapply(
  RAW_DATA,
  function(x) {
    subsetEMG(x,
      cy_max = 3,
      cy_start = 1
    )
  }
)
```

SYNS *Muscle synergies example*

Description

A list created by synsNMF containing muscle synergies extracted from 15 humans walking on a treadmill.

Usage

```
SYNS
```

Format

A list containing 15 objects of class musclesyneRgies, each of which represents a walking trial from a different person.

ID0012_TW_01 ID0001_TW_01 ID0002_TW_01 ID0003_TW_01 ID0004_TW_01 ID0005_TW_01 ID0006_TW_01 ID0007_TW_01
: Objects of class musclesyneRgies containing the following items:

`syns` Factorisation rank or minimum number of synergies.

`M` Muscle weights (time-invariant coefficients)

`P` Activation patterns (time-dependent coefficients)

`V` Original data, muscles named as:

ME=gluteus medius
MA=gluteus maximus
FL=tensor fasciae latae
RF=rectus femoris
VM=vastus medialis
VL=vastus lateralis
ST=semitendinosus
BF=biceps femoris
TA=tibialis anterior
PL=peroneus longus
GM=gastrocnemius medialis
GL=gastrocnemius lateralis
SO=soleus

`Vr` Reconstructed data, muscles named as in `Vr`

`iterations` Number of iterations to convergence

`R2` Quality of reconstruction (coefficient of determination)

`classification` Classification type (e.g., `none`, `k-means`, `NMF`, etc.)

Source

doi:10.1016/j.isci.2019.100796

`synsNMF` *Non-negative matrix factorisation*

Description

Non-negative matrix factorisation

Usage

```
synsNMF(
  V,
  R2_target = 0.01,
  runs = 5,
  max_iter = 1000,
  last_iter = 20,
  MSE_min = 1e-04,
  fixed_syns = NA
)
```

Arguments

V	EMG data frame to be reconstructed, usually filtered and time-normalised
R2_target	Threshold to stop iterations for a certain factorisation rank
runs	Number of repetitions for each rank to avoid local minima
max_iter	Maximum number of iterations allowed for each rank
last_iter	How many of the last iterations should be checked before stopping?
MSE_min	Threshold on the mean squared error to choose the factorisation rank or minimum number of synergies
fixed_syns	To impose the factorisation rank or number of synergies

Details

The first column of V must always contain time information.

Value

Object of class musclesyneRgies with elements:

- syns factorisation rank or minimum number of synergies
- M muscle weights (time-invariant coefficients)
- P activation patterns (time-dependent coefficients)
- V original data
- Vr reconstructed data
- iterations number of iterations to convergence
- R2 quality of reconstruction (coefficient of determination)
- rank_type was the rank fixed or variable?
- classification classification type (e.g., none, k-means, NMF, etc.)

References

Lee, D. D. & Seung, H. S. Learning the parts of objects by non-negative matrix factorization. Nature 401, 788-91 (1999).

Santuz, A., Ekizos, A., Janshen, L., Baltzopoulos, V. & Arampatzis, A. On the Methodological Implications of Extracting Muscle Synergies from Human Locomotion. Int. J. Neural Syst. 27, 1750007 (2017).

Févotte, C., Idier, J. Algorithms for Nonnegative Matrix Factorization with the Beta-Divergence Neural Computation 23, 9 (2011).

Examples

```
# Note that for bigger data sets one might want to run computation in parallel
# Load some data
data(FILT_EMG)
# Extract synergies (careful, rank is imposed here!)
SYNS <- lapply(FILT_EMG, synsNMF, fixed_syns = 4)
```

Index

∗ **datasets**
act_pattern, 2
act_patterns, 3
FILT_EMG, 8
RAW_DATA, 22
SYNS, 25

act_pattern, 2
act_patterns, 3

classify_kmeans, 3
CoA, 5
cossim, 6

FILT_EMG, 8
filtEMG, 7
FWHM, 9

HFD, 10
Hurst, 11

normEMG, 13

plot_classified_syns, 14
plot_classified_syns_UMAP, 15
plot_meanEMG, 16
plot_rawEMG, 17
plot_syn_trials, 19

RAW_DATA, 22
rawdata, 20

sMLE, 23
subsetEMG, 24
SYNS, 25
synsNMF, 26