

Package ‘rticulate’

February 20, 2025

Type Package

Title Articulatory Data Processing in R

Version 2.0.1

Date 2025-02-20

Maintainer Stefano Coretta <stefano.coretta@gmail.com>

Description A tool for processing Articulate Assistant Advanced™ (AAA) ultrasound tongue imaging data and Carstens AG500/1 electro-magnetic articulographic data.

URL <https://github.com/stefanocoretta/rticulate>,
<https://stefanocoretta.github.io/rticulate/>

BugReports <https://github.com/stefanocoretta/rticulate/issues>

Depends R (>= 4.1.0)

Encoding UTF-8

LazyData true

Imports cli, dplyr, ggplot2, glue, gsignal, magrittr, mgcv, pracma,
purrr, readr, rlang, stats, tibble, tidyr, tidyselect

RoxygenNote 7.3.2

Suggests knitr, rcartocolor, rmarkdown, stringr, tidyverse

VignetteBuilder knitr

Language en-US

License MIT + file LICENSE

NeedsCompilation no

Author Stefano Coretta [aut, cre]

Repository CRAN

Date/Publication 2025-02-20 12:20:02 UTC

Contents

filter_signal	2
get_acceleration	3
get_landmarks	3
get_origin	4
get_velocity	5
palate	5
plot_tongue	6
polar_gam	7
predict_polar_gam	8
read_aaa	9
read_ag500_pos	10
resample_signal	11
stimuli	11
tongue	12
transform_coord	13

Index	15
--------------	-----------

filter_signal	<i>Filter a signal</i>
---------------	------------------------

Description

Filter a signal

Usage

```
filter_signal(
  signal,
  filter = "sgolay",
  order = 2,
  window_length = NULL,
  cutoff_freq = NULL,
  sampling_freq = NULL,
  type = "low",
  apply = 1
)
```

Arguments

signal	Signal to filter.
filter	Type of filter (default is "sgolay", or "butter").
order	Order of the filter.
window_length	Window length of the Savitzky-Golay filter.
cutoff_freq	Cut-off frequency of the Butterworth filter.

sampling_freq Sampling frequency of the signal.
type Butterworth band type (default is "low").
apply Apply the filter N times (default is 1).

Value

The filtered signal.

get_acceleration *Get acceleration of displacement*

Description

Get acceleration of displacement

Usage

`get_acceleration(signal)`

Arguments

signal The signal to get the acceleration of.

Value

A vector with the second derivative of the signal.

get_landmarks *Get gestural landmarks*

Description

Get gestural landmarks

Usage

`get_landmarks(signal_vel, time, start, end, threshold = 0.2)`

Arguments

signal_vel The velocity of the displacement signal.
time The time of the signal.
start Start time of interval in which to search for maximum displacement.
end End time of interval in which to search for maximum displacement.
threshold The velocity threshold (default is 0.2, corresponding to 20 percent velocity.)

Value

A tibble with one row and a column for each gestural landmark.

get_origin	<i>Get the origin of spline data</i>
------------	--------------------------------------

Description

It returns the Cartesian x , y coordinates of the virtual origin of the ultrasonic waves/probe surface (see Details).

Usage

```
get_origin(data, fan_lines = c(10, 25))
```

Arguments

data	The spline data (the cartesian coordinates must be in two columns named X and Y).
fan_lines	A numeric vector with two fan lines (the default is $c(10, 25)$).

Details

The function estimates the origin of the ultrasound waves from the probe using the spline data and the provided fan lines. The estimation method is based on Heyne, Matthias & Donald Derrick (2015) Using a radial ultrasound probe's virtual origin to compute midsagittal smoothing splines in polar coordinates. *The Journal of the Acoustical Society of America* 138(6), EL509–EL514, DOI:10.1121/1.4937168.

Value

A numeric vector with the Cartesian (x , y) coordinates of the virtual origin of the ultrasonic waves/probe surface.

Origin estimation

The equations of the two fan lines (10 and 25 by default) are set equal to find their intersection. The intersection is the origin. In some cases, the linear estimation of the equation fails, and an error related to fit is returned. In these cases, try different fan lines by increasing the minimum fan line and/or changing the maximum fan line (for example, if $c(10, 25)$ returns an error, try $c(15, 30)$).

get_velocity	<i>Get velocity of displacement</i>
--------------	-------------------------------------

Description

Get velocity of displacement

Usage

```
get_velocity(signal)
```

Arguments

signal The signal to get the velocity of.

Value

A vector with the first derivative of the signal.

palate	<i>Palate profile dataset.</i>
--------	--------------------------------

Description

A dataset containing the palate profile of a single speaker.

Usage

```
palate
```

Format

A data frame with 42 rows and 14 variables.

speaker speaker ID

seconds time of coordinate, in seconds

rec_date date and time of recording

prompt prompt string

label label of annotation

TT_displacement smoothed displacement of tongue tip

TT_velocity velocity of tongue tip displacement

TT_abs_velocity absolute velocity of tongue tip displacement

TD_displacement smoothed displacement of tongue dorsum

TD_velocity velocity of tongue dorsum displacement
TD_abs_velocity absolute velocity of tongue dorsum displacement
fan_line fan line number
X horizontal coordinate at time seconds
Y vertical coordinate at time seconds

plot_tongue	<i>Plot tongue contours from spline data.</i>
-------------	---

Description

It plots tongue contours from data imported from AAA.

Usage

```
plot_tongue(data, geom = "line", ..., palate = NULL, palate_col = "green")
```

Arguments

data	A data frame with splines data.
geom	Type of geom to plot. Possible values are: line (the default), point, path.
...	List of arguments to be passed to geom.
palate	An optional data frame with the palate spline. If provided, the palate is plotted.
palate_col	The colour of the palate spline (the default is green).

Value

An object of class `ggplot`.

Examples

```
plot_tongue(tongue, geom = "point")
```

polar_gam	<i>Polar generalised additive model (polar GAM)</i>
-----------	---

Description

It fits a generalised additive model (GAM) to transformed polar tongue data and it returns a model in polar coordinates. Use `plot_polar_smooths()` for plotting.

Usage

```
polar_gam(  
  formula,  
  data,  
  origin = NULL,  
  fan_lines = c(10, 25),  
  AR_start = NULL,  
  ...  
)
```

Arguments

formula	A GAM formula.
data	A data set containing the spline coordinates (cartesian coordinates must be in columns named X and Y, polar coordinates in columns named angle and radius; these are the defaults in data imported with <code>read_aaa()</code>).
origin	The coordinates of the origin as a vector of <code>c(x, y)</code> coordinates.
fan_lines	A numeric vector with two fan lines (the default is <code>c(10, 25)</code>).
AR_start	The <code>AR.start</code> argument to be passed to <code>mgcv::bam()</code> .
...	Arguments to be passed to <code>mgcv::bam()</code> .

Details

It is advised to fit a separate model per speaker, unless you have a working method for inter-speaker normalisation of the coordinates.

Value

An object of class "gam" as described in [gamObject](#).

Examples

```
library(dplyr)  
tongue_it01 <- filter(tongue, speaker == "it01")  
pgam <- polar_gam(Y ~ s(X, by = c2_place) + s(X, word, bs = "fs"),  
  data = tongue_it01)
```

predict_polar_gam *Get all predictions from a polar GAM model*

Description

It returns a tibble with the predictions from all the terms in a [polar_gam](#) model.

Usage

```
predict_polar_gam(  
  model,  
  origin = NULL,  
  exclude_terms = NULL,  
  length_out = 50,  
  values = NULL,  
  return_ci = FALSE,  
  ci_z = 1.96  
)
```

Arguments

model	A polar_gam model object.
origin	The coordinates of the origin as a vector of $c(x, y)$ coordinates.
exclude_terms	Terms to be excluded from the prediction. Term names should be given as they appear in the model summary (for example, " $s(x_0, x_1)$ ").
length_out	An integer indicating how many values along the numeric predictors to use for predicting the outcome term (the default is 50).
values	User supplied values for numeric terms as a named list.
return_ci	Whether to return a tibble with cartesian confidence intervals (for use with geom_polar_ci).
ci_z	The z-value for calculating the CIs (the default is 1.96 for 95 percent CI).

Details

The function converts the coordinates from polar to cartesian automatically.

To see an example of plotting, see the examples in [geom_polar_ci](#).

Value

A tibble with predictions from a [polar_gam](#) model.

Examples

```
library(dplyr)
tongue_it01 <- filter(tongue, speaker == "it01")
it01_pol <- polar_gam(Y ~ s(X, by = c2_place) + s(X, word, bs = "fs"),
  data = tongue_it01)

# get predictions
it01_pred <- predict_polar_gam(it01_pol)

# get predictions excluding the random smooth for word (the coefficient for
# the random smooth is set to 0)
it01_excl_rand <- predict_polar_gam(it01_pol, exclude_terms = "s(X,word)")
```

read_aaa	<i>Read tab separated files with AAA spline data.</i>
----------	---

Description

It reads a file or a list of files with data exported from AAA. The data are automatically transformed from a wide to a long format (each row has values of X or Y axes for each fan line). The imported tibble can then be used for plotting and statistical analysis.

Usage

```
read_aaa(
  file,
  coordinates = "cartesian",
  format = "long",
  na_rm = FALSE,
  knots = NULL,
  column_names = NULL,
  fan_lines = NULL
)
```

Arguments

file	The path of the file with AAA data. It can also be a character vector with multiple paths as separate strings..
coordinates	A string specifying the coordinate system. Possible values are "cartesian" (the default) and "polar".
format	A string specifying the data format. Possible values are "long" and "wide" (the default is "long").
na_rm	Remove NAs (the default is FALSE).
knots	The number of spline knots or fan lines.
column_names	The names of the columns without including the splines columns.
fan_lines	The number of fan lines in legacy fan-line data.

Value

A tibble.

Examples

```
columns <- c("speaker", "seconds", "rec_date", "prompt", "label",
  "TT_displacement", "TT_velocity", "TT_abs_velocity", "TD_displacement",
  "TD_velocity", "TD_abs_velocity")
file_path <- system.file("extdata", "it01.tsv", package = "rticulate")

tongue <- read_aaa(file_path, knots = 42, column_names = columns)
```

read_ag500_pos	<i>Read EMA data from AG500 pos files</i>
----------------	---

Description

Read EMA data from AG500 pos files

Usage

```
read_ag500_pos(path, channels = 12, ch_values = 7, bytes = 4, fs = 200)
```

Arguments

path	Path to the .pos file.
channels	Number of channels (default 12).
ch_values	Number of values per channel (default 7).
bytes	Number of bytes per value (default 4).
fs	Sampling frequency (default 200 Hz).

Value

A tibble.

resample_signal	<i>Resample (up or down) a signal</i>
-----------------	---------------------------------------

Description

Resample (up or down) a signal

Usage

```
resample_signal(
  signal,
  time,
  by = 2,
  to = NULL,
  from = NULL,
  method = "interpolation"
)
```

Arguments

signal	The signal to resample.
time	The time vector of the signal to resample.
by	The factor by which to resample the signal (default is 2).
to	The frequency to resample to.
from	The original sampling frequency.
method	Resampling method (default is interpolate which uses approx).

Value

A list with the resampled signal and time if methdo = "interpolate".

stimuli	<i>Stimuli dataset.</i>
---------	-------------------------

Description

A dataset with linguistic information on the stimuli.

Usage

```
stimuli
```

Format

A data frame with 12 rows and 11 variables.

item item ID

word words of the form CVCV

ipa IPA transcription of the words

c1 first consonant

c1_phonation phonation of the first consonant, voiceless

vowel first and second vowel

anteropost backness of the vowel, back or central

height height of the vowel, high, mid or low

c2 second consonant

c2_phonation phonation of the second consonant, voiceless or voiced

c2_place place of the second consonant, coronal or velar

tongue

Tongue contours dataset.

Description

A dataset containing tongue contour coordinates of a single speaker.

Usage

tongue

Format

A data frame with 3612 rows and 28 variables.

speaker speaker ID

seconds time of coordinate, in seconds

rec_date date and time of recording

prompt prompt string

label label of annotation

TT_displacement smoothed displacement of tongue tip

TT_velocity velocity of tongue tip displacement

TT_abs_velocity absolute velocity of tongue tip displacement

TD_displacement smoothed displacement of tongue dorsum

TD_velocity velocity of tongue dorsum displacement

TD_abs_velocity absolute velocity of tongue dorsum displacement

TR_displacement smoothed displacement of tongue root
TR_velocity velocity of tongue root displacement
TR_abs_velocity absolute velocity of tongue root displacement
fan_line fan line number
X horizontal coordinate at time seconds
Y vertical coordinate at time seconds
word words of the form CVCV
item item ID
ipa IPA transcription of the words
c1 first consonant
c1_phonation phonation of the first consonant, voiceless
vowel first and second vowel
anteropost backness of the vowel, back or central
height height of the vowel, high, mid or low
c2 second consonant
c2_phonation phonation of the second consonant, voiceless or voiced
c2_place place of the second consonant, coronal or velar

transform_coord

Transform the coordinates of spline data

Description

This function transforms the coordinates of spline data between Cartesian and polar coordinate systems. The origin x and y coordinates can be supplied by the user, or calculated automatically (see Details).

Usage

```

transform_coord(
  data,
  to = "polar",
  origin = NULL,
  fan_lines = c(10, 25),
  use_XY = FALSE
)

```

Arguments

data	A data set containing the spline coordinates (cartesian coordinates must be in columns named X and Y, polar coordinates in columns named angle and radius; these are the defaults in data imported with read_aaa()).
to	Which system to convert to, as a string, either "polar" or "cartesian" (the default is "polar").
origin	The coordinates of the origin as a vector of c(x, y) coordinates.
fan_lines	A numeric vector with two fan lines (the default is c(10, 25)).
use_XY	Whether to use the column names X and Y when converting to and from polar coordinates, rather than the default angle and radius (the default is FALSE. If TRUE, the columns X and Y are overwritten with the converted values. If converting to polar, X is the angle and Y the radius.

Details

The transformation between the coordinate systems require the selection of an origin in Cartesian coordinates (x and y). The origin ideally corresponds to the virtual origin of the ultrasound waves from the probe. The origin coordinates can be supplied by the user as a vector with the `origin` argument, or they can be estimated automatically if `origin = NULL` (the default). The estimation is performed by [get_origin](#) (see that function documentation for details).

Value

An object of class `tbl_df-class` (a tibble).

Index

* datasets

- palate, 5
- stimuli, 11
- tongue, 12

filter_signal, 2

gamObject, 7

geom_polar_ci, 8

get_acceleration, 3

get_landmarks, 3

get_origin, 4, 14

get_velocity, 5

ggplot, 6

palate, 5

plot_tongue, 6

polar_gam, 7, 8

predict_polar_gam, 8

read_aaa, 9

read_ag500_pos, 10

resample_signal, 11

stimuli, 11

tongue, 12

transform_coord, 13