Package ‘rticulate’

October 14, 2022

Type Package
Title Ultrasound Tongue Imaging
Version 1.7.3
Date 2022-09-04
Maintainer Stefano Coretta <stefano.coretta@gmail.com>
Description A tool for processing Articulate Assistant Advanced™ (AAA) export files and plot tongue contour data from any system.
URL https://github.com/stefanocoretta/rticulate
BugReports https://github.com/stefanocoretta/rticulate/issues
Depends R (>= 3.0.0)
Encoding UTF-8
LazyData true
Imports dplyr, ggplot2, glue, magrittr, mgcv, purrr, readr, rlang, stats, tibble, tidymv, tidyr, tidyselect
RoxygenNote 7.2.1
Suggests knitr, rmarkdown, stringr
VignetteBuilder knitr
Language en_GB
License MIT + file LICENSE
NeedsCompilation no
Author Stefano Coretta [aut, cre]
Repository CRAN
Date/Publication 2022-09-04 16:20:02 UTC
**get_origin**

**R topics documented:**

- get_origin ........................................... 2
- palate ................................................ 3
- plot_polar_smooths ................................. 4
- plot_tongue .......................................... 5
- polar_gam ........................................... 6
- predict_polar_gam ................................. 7
- read_aaa ............................................. 8
- stimuli ................................................ 9
- tongue ............................................... 10
- transform_coord ................................... 11

**Index**

- 12

---

**get_origin**

*Get the origin of spline data*

**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**


**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)).

palate

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_polar_smooths**  
*Plot smooths from a polar gam*

**Description**

It plots the smooths of a polar GAM fitted with `polar_gam()`.

**Usage**

```r
plot_polar_smooths(
  model,
  series,
  comparison = NULL,
  origin = NULL,
  facet_terms = NULL,
  conditions = NULL,
  exclude_random = TRUE,
  series_length = 100,
  split = NULL,
  sep = "\.",
  time_series
)
```

**Arguments**

- `model` A gam or bam model object.
- `series` An unquoted expression indicating the model term that defines the series on which smoothing is applied. This is the term that is displayed on the x-axis when plotting.
- `comparison` An unquoted expression indicating the model term for which the comparison will be plotted.
- `origin` The coordinates of the origin as a vector of c(x, y) coordinates.
- `facet_terms` An unquoted formula with the terms used for faceting.
- `conditions` A list of quosures with quos specifying the levels to plot from the model terms not among `series`, `comparison`, or `facet_terms`.
- `exclude_random` Whether to exclude random smooths (the default is TRUE).
- `series_length` An integer indicating how many values along the time series to use for predicting the outcome term.
- `split` Columns to separate as a named list.
- `sep` Separator between columns (default is "\.", which is the default with ). If character, it is interpreted as a regular expression.
- `time_series` Deprecated, use `series` instead.
plot_tongue

Value

An object of class ggplot.

Examples

library(dplyr)
tongue_it01 <- filter(tongue, speaker == "it01")
pgam <- polar_gam(Y ~ s(X, by = as.factor(label)), data = tongue_it01)
plot_polar_smooths(pgam, X, label)

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")
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 read_aaa()).
- **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)).
- **AR_start**: The AR.start argument to be passed to mgcv::bam().
- **...**: Arguments to be passed to mgcv::bam().

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(x0, x1)").
- **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 behaves like predict_gam but it converts the coordinates from polar to cartesian automatically. Check vignette("predict-gam", package = "tidymv") to an overview of the predict method.

To see an example of plotting, see the examples in geom_polar_ci.

Value

A tibble with predictions from a polar_gam model.
Examples

```r
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")
```

### Description

Read tab separated files with AAA spline data.

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

```r
read_aaa(
  file,
  column_names,
  fan_lines = 42,
  coordinates = "cartesian",
  na_rm = FALSE,
  format = "long"
)
```

### Arguments

- **file**: The path of the file with AAA data. It can also be a character vector with multiple paths as separate strings.
- **column_names**: The names of the columns without including the splines columns.
- **fan_lines**: The number of fan lines (the default is 42).
- **coordinates**: A string specifying the coordinate system. Possible values are "cartesian" (the default) and "polar".
- **na_rm**: Remove NAs (the default is FALSE).
- **format**: A string specifying the data format. Possible values are "long" and "wide" (the default is "long").
### Value

A tibble. An .index column is added which indexes (groups) each tongue contour.

### Examples

```r
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, columns, na_rm = TRUE)
```

---

**Stimuli**

**Stimulus dataset.**

**Description**

A dataset with linguistic information on the stimuli.

**Usage**

`stimuli`

**Format**

A data frame with 12 rows and 11 variables.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>item</td>
<td>item ID</td>
</tr>
<tr>
<td>word</td>
<td>words of the form CVCV</td>
</tr>
<tr>
<td>ipa</td>
<td>IPA transcription of the words</td>
</tr>
<tr>
<td>c1</td>
<td>first consonant</td>
</tr>
<tr>
<td>c1_phonation</td>
<td>phonation of the first consonant, voiceless</td>
</tr>
<tr>
<td>vowel</td>
<td>first and second vowel</td>
</tr>
<tr>
<td>anteropost</td>
<td>backness of the vowel, back or central</td>
</tr>
<tr>
<td>height</td>
<td>height of the vowel, high, mid or low</td>
</tr>
<tr>
<td>c2</td>
<td>second consonant</td>
</tr>
<tr>
<td>c2_phonation</td>
<td>phonation of the second consonant, voiceless or voiced</td>
</tr>
<tr>
<td>c2_place</td>
<td>place of the second consonant, coronal or velar</td>
</tr>
</tbody>
</table>
**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, 3
  stimuli, 9
  tongue, 10

  gamObject, 6
  geom_polar_ci, 7
  get_origin, 2, 11
  ggplot, 5

  palate, 3
  plot_polar_smooths, 4
  plot_tongue, 5
  polar_gam, 6, 7
  predict_gam, 7
  predict_polar_gam, 7

  read_aaa, 8

  stimuli, 9

  tongue, 10
  transform_coord, 11