Package: card (via r-universe)

November 7, 2024

Type Package

Title Cardiovascular Applications in Research Data

Version 0.1.1

Description A collection of cardiovascular research datasets and analytical tools, including methods for cardiovascular procedural data, such as electrocardiography, echocardiography, and catheterization data. Additional methods exist for analysis of procedural billing codes.

License MIT + file LICENSE

URL https://cran.r-project.org/package=card

BugReports https://github.com/shah-in-boots/card/issues

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Depends R (>= 4.1),

Imports stats, generics, ggplot2, tidyr, dplyr, purrr, tibble, rlang, parsnip, hardhat,

Suggests kableExtra, knitr, rmarkdown, covr, testthat, broom.mixed, ggrepel

VignetteBuilder knitr

NeedsCompilation no

Author Anish S. Shah [aut, cre, cph] (<https://orcid.org/0000-0002-9729-1558>)

Maintainer Anish S. Shah <shah.in.boots@gmail.com>

Repository CRAN

Date/Publication 2024-11-06 07:40:02 UTC

Config/pak/sysreqs libicu-dev

Contents

`augment.cosinor` *Augment data with information from a* `cosinor` *object*

Description

Augment accepts a `cosinor` model object and adds information about each observation in the dataset. This includes the predicted values in the `.fitted` column and the residuals in the `.resid` column. New columns always begin with a `.` prefix to avoid overwriting columns in original dataset.

Usage

```
## S3 method for class 'cosinor'
augment(x, ...)
```

Arguments

`x`	A `cosinor` object created by `cosinor()`
`...`	For extensibility

Value

a `tibble` object

See Also

Other cosinor: `cosinor()`, `ggcosinor()`

`cosinor` *Fit a* `cosinor`

Description

`cosinor()` fits a regression model of a time variable to a continuous outcome use trigonometric features. This approaches uses the linearization of the parameters to assess their statistics and distribution.

Usage

```
cosinor(t, ...)

## Default S3 method:
cosinor(t, ...)

## S3 method for class 'data.frame'
cosinor(t, y, tau, population = NULL, ...)

## S3 method for class 'matrix'
cosinor(t, y, tau, population = NULL, ...)

## S3 method for class 'formula'
cosinor(formula, data, tau, population = NULL, ...)

## S3 method for class 'recipe'
cosinor(t, data, tau, population = NULL, ...)
```

Arguments

`t` Represents the *ordered* time indices that provide the positions for the cosine wave. Depending on the context:

- A `data frame` of a time-based predictor/index.
- A `matrix` of time-based predictor/index.
- A `recipe` specifying a set of preprocessing steps created from `recipes::recipe()`.

`...` Not currently used, but required for extensibility.

`y` When **t** is a `data frame` or `matrix`, **y** is the outcome specified as:

- A `data frame` with 1 numeric column.
- A `matrix` with 1 numeric column.
- A numeric `vector`.

`tau` A vector that determines the periodicity of the time index. The number of elements in the vector determine the number of components (e.g. single versus multiple cosinor).

- A `vector` with a single element = single-component cosinor, e.g. period = c(24)

- A `vector` with multiple elements = multiple-component cosinor, e.g. period = c(24, 12)

`population` Represents the population to be analyzed with a population-mean cosinor. Defaults to NULL, assuming individual cosinors are being generated. When a `recipe` or `formula` is used, **population** is specified as:

- A `character` name of the column contained in **data** that contains identifiers for each subject. Every row will have a subject name which should be duplicated for each time index given.

When a `data frame` or `matrix` is used, **population** is specified as:

- A `vector` of the same length as **t**, with values representing each subject at the correct indices.

`formula` A formula specifying the outcome terms on the left-hand side, and the predictor terms on the right-hand side.

`data` When a `recipe` or `formula` is used, **data** is specified as:

- A `data frame` containing both the predictors and the outcome.

Value

A `cosinor` object.

See Also

Other cosinor: `augment.cosinor()`, `ggcosinor()`

Examples

```
# Data setup
data("twins")

# Formula interface
model <- cosinor(rDYX ~ hour, twins, tau = 24)
```

`cosinor_area` *Area of Ellipse*

Description

Formulas for creating the area of the ellipse to identify confidence intervals, direction, and graphing purposes.

Usage

```
cosinor_area(object, level = 0.95, ...)
```

Arguments

`object`	Model of class `cosinor`
`level`	Confidence level requested
`...`	Not currently used, but required for extensibility.

Value

Area of potential cosinor for graphical analysis as matrix stored in a list.

`cosinor_features` *Multiple Component Cosinor Features*

Description

Extract the special/global features of a multiple component cosinor. In a multiple component model, there are specific parameters that are not within the model itself, but must be extracted from the model fit. When extracted, can be used to improve the plot of a multiple component cosinor. However, this is only possible if the cosinor is harmonic (see `details`). For single-component models, the orthophase is the same as the acrophase and the global amplitude

- Global Amplitude (Ag) = the overall amplitude is defined as half the difference between the peak and trough values
- Orthophase (Po) = the lag until the peak time
- Bathyphase (Pb) = the lag until the trough time

Usage

```
cosinor_features(object, population = TRUE, ...)
```

Arguments

`object` Model of class `cosinor` with multiple periods

`population` If the object is a population cosinor, should the features be calculated for the individual cosinors or for the population-cosinors. Default is TRUE. This has no effect on "Individual" cosinor objects.

- If TRUE, then will calculate features for entire population.
- If FALSE, then will calculate features for every individual cosinor in the population.

`...` For extensibility

Details

These calculations can only occur if the periods of the cosinor are harmonic - as in, the longest period is a integer multiple of the smallest period (known as the fundamental frequency). Otherwise, these statistics are not accurate or interpretable.

Value

When returning the cosinor features for a single model, will return an object of class `list`. When returning the cosinor features for every individual in a population cosinor, will return an object of class `tibble`.

Examples

```
data(twins)
model <- cosinor(rDYX ~ hour, twins, c(24, 8), "patid")
results <- cosinor_features(model, population = FALSE)
head(results)
```

cosinor_goodness_of_fit *Goodness of Fit of Cosinor*

Description

Goodness of fit of a cosinor from data that has multiple collections at different timepoints or from multiple cycles. The RSS is partitioned into pure error (SSPE) and lack of fit (SSLOF). An F-test compares the SSPE and SSLOF to detect appropriateness of model.

$$SSLOF = RSS - SSPE$$

$$SSPE = \sum_i \sum_l (Y_{il} - \overline{Y}_i)^2$$

The fitted values for each time point are:

$$\overline{Y}_i = \frac{\sum_l Y_{il}}{n_i}$$

Usage

```
cosinor_goodness_of_fit(object, level = 0.95, ...)
```

Arguments

`object`	requires cosinor model generated with cosinor to calculate statistics.
`level`	confidence level desired
`...`	additional parameters may be needed for extensibility

Value

f-statistic as result of goodness of fit

`cosinor_reg` *General Interface for Cosinor Regression Models*

Description

`cosinor_reg()` is a *parsnip* friendly method for specification of cosinor regression model before fitting.

Usage

```
cosinor_reg(mode = "regression", period = NULL)

## S3 method for class 'cosinor_reg'
update(object, period = NULL, fresh = FALSE, ...)

## S3 method for class 'cosinor_reg'
print(x, ...)
```

Arguments

`mode` A character string that describes the type of model. In this case, it only supports type of "regression".

`period` A non-negative number or vector of numbers that represent the expected periodicity of the data to be analyzed.

`object` Cosinor model specification

`fresh` A logical for whether the arguments should be modified in place or replaced altogether

`...` Extensible

`x` Cosinor model specification

Examples

```
library(parsnip)
cosinor_reg(period = c(24, 8)) |>
 parsnip::set_engine("card") |>
 parsnip::set_mode("regression")
```

`cosinor_zero_amplitude` *Zero Amplitude Test*

Description

Zero amplitude test assesses how well the circadian pattern fits the data, essentially detecting the present of a rhythm to the data.

Usage

```
cosinor_zero_amplitude(object, level = 0.95)
```

Arguments

`object`	model of class `cosinor`
`level`	confidence level

Value

Returns a list of test statistics, as well prints out a report of analysis.

`geh` *GEH parameters in a large clinical cohort*

Description

Used in the model-building examples for repeat testing.

Usage

```
geh
```

Format

A tibble

ggcosinor *ggplot of cosinor model*

Description

ggplot of cosinor model that can visualize a variety of cosinor model subtypes, including single-component, multiple-component, individual, and population cosinor models, built using cosinor. For single component cosinor, the following values are plotted:

- M = midline estimating statistic of rhythm
- A = amplitude
- P = phi or acrophase (shift from 0 to peak)

If using a multiple-component cosinor, the terms are different. If the periods or frequencies resonate or are harmonic, then the following are calculated. If the periods are not harmonic, the values are just descriptors of the curve.

- M = midline estimating statistic of rhythm
- Ag = global amplitude, which is the distance between peak and trough (this is the same value as the amplitude from single component)
- Po = orthophase (the equivalent of the acrophase in a single component), the lag time to peak value
- Pb = bathyphase, the lag time to trough value

Usage

```
ggcosinor(object, labels = TRUE, ...)
```

Arguments

`object`	Model of class `cosinor`. If instead of a single cosinor model, multiple objects are to be plotted, can provide a list of cosinor models. Plotting multiple models simultaneously is preferred if the outcome variable is similar in scale.
`labels`	Logical value if annotations should be placed on plot, default = TRUE. The labels depend on the type of plot. The labels are attempted to be placed "smartly" using the `ggrepel::geom_label_repel()` function.
`...`	For extensibility. This function will use different implementations based on the type of model (single or multiple component). Attributes of the object will be passed down, or calculated on the fly.

Value

Object of class `ggplot` that can be layered

See Also

Other cosinor: `augment.cosinor()`, `cosinor()`

Examples

```
data(triplets)
m1 <- cosinor(rDYX ~ hour, twins, tau = 24)
m2 <- cosinor(rDYX ~ hour, twins, tau = c(24, 12))
ggcosinor(m1, labels = FALSE)
ggcosinor(m2)
ggcosinor(list(single = m1, multiple = m2))
```

ggellipse *Graphical Assessment of Amplitude and Acrophase*

Description

This is a ggplot-styled graphical representation of the ellipse region generated by the cosinor analysis. It requires the same data used by cosinor model to be fit with the model cosinor. This includes the amplitude, acrophase,

Usage

```
ggellipse(object, level = 0.95, ...)
```

Arguments

object	Requires a cosinor model to extract the correct statistics to generate the plot.
level	Confidence level for ellipse
...	Additional parameters may be needed for extensibility

Value

Object of class `ggplot` to help identify confidence intervals

Examples

```
data("twins")
m <- cosinor(rDYX ~ hour, twins, tau = 24)
ggellipse(m)
```

hrv *Output from MATLAB HRV Toolbox*

Description

Data is a single patient data output from HRV Toolbox. It contains granular data of calculated HRV in 5-second sliding windows.

Usage

```
hrv
```

Format

An tibble data frame

predict.cosinor *Predict from a* `cosinor`

Description

Predict from a `cosinor`

Usage

```
## S3 method for class 'cosinor'
predict(object, new_data, type = "numeric", ...)
```

Arguments

`object`	A `cosinor` object.
`new_data`	A data frame or matrix of new predictors.
`type`	A single character. The type of predictions to generate. Valid options are: • `"numeric"` for numeric predictions.
`...`	Additional arguments passed to the prediction function

Value

A tibble of predictions. The number of rows in the tibble is guaranteed to be the same as the number of rows in `new_data`.

procedure_codes *Center for Medicare and Medicaid Services (CMS) Procedure Codes*

Description

This is a generative function used to call CMS procedure codes. It is used to create a dataset that can be generally used to map procedure codes to their descriptions, allowing for understanding of interventions performed. The currently supported codes are explained in the details.

The following procedure codes are currently supported:

- ICD9 procedure codes, most recently updated on `2014-10-01`
- ICD10 procedure codes, most recently updated on `2023-01-11`
- HCPCS prcoedure codes, most recently updated on `2023-11-29`
- CPT procedure codes, most recently updated on `2023-11-29`

Usage

```
procedure_codes(format, version)
```

Arguments

`format` The format of the procedure codes, written as a `character`. Currently supported formats are: `c("icd9", "icd10", "hcpcs", "cpt")` (case-insensitive).

`version` The version of the procedure codes, which are generally written as a year. Currently supported: `c("2014", "2023")`

Details

CMS will usually release updated version of these codes on an annual basis. Each dataset that is supported below can be identified by the year it was published (not the *go-live* date, but the *publically-available* date). The previous versions that are included in the package are as below.

- ICD9: 2014
- ICD10: 2023
- HCPCS: 2023
- CPT: 2023

Value

A `tbl_df` with two columns: `code` and `description`. The `code` refers to the procedure code, while the `description` refers to the description of the procedure.

Examples

```
# Procedure codes from the 2014 version of ICD-9
icd9 <- procedure_codes(format = "icd9", version = 2014)
```

`stress` *Recurrent event sample data*

Description

Data is from a outcomes study on cardiovascular outcomes. It contains the first visit date, the last known date, and times of various events that have happened. They document death at right censoring as well. These events are non-ordered.

Usage

```
stress
```

Format

An tibble data frame

`tidy.cosinor` *Tidy a(n) cosinor object*

Description

Tidy summarizes information about the components of a `cosinor` model.

Usage

```
## S3 method for class 'cosinor'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

`x` A `cosinor` object created by `cosinor()`

`conf.int` Logical indicating whether or not to include confidence interval in tidied output

`conf.level` The confidence level to use if `conf.int` = `TRUE`. Must be between 0 and 1, with default to 0.95 (the 95% confidence interval).

`...` For extensibility

Details

`cosinor` objects do not necessarily have a T-statistic as the standard error is not based on a mean value, but form a joint-confidence interval. The standard error is generated using Taylor series expansion as the object is a subspecies of harmonic regressions.

Value

a `tibble` object

`triplets` *Hourly time series data with clinical covariates*

Description

Clinical data is also available for visualization and comparison. Other HRV measures are used here for comparison and testing out functions.

Usage

```
triplets
```

Format

A `tbl_df`

`twins` *Hourly time series data with clinical covariates*

Description

Data is from an algorithm that generates a summary HRV measure using the Poincare phase-space plot, generated from kurtoses of the x and y axis. Clinical data is also available for visualization and comparison. There are repeat rows for each hour that Dyx was taken.

Usage

```
twins
```

Format

An tibble data frame

`zipcode` *Zipcodes with Associated Latitude and Longitude*

Description

This is a dataset from the archived/orphaned `zipcode` package.

Usage

```
zipcode
```

Format

A data frame with character vector zipcodes and latitude/longitude

Index