# Package 'circacompare'

January 9, 2024

**Title** Analyses of Circadian Data

**Version** 0.2.0

**Description** Uses non-linear regression to statistically compare two circadian rhythms.
Groups are only compared if both are rhythmic (amplitude is non-zero).
Performs analyses regarding mesor, phase, and amplitude, reporting on estimates and statistical differences, for each, between groups.
Details can be found in Parsons et al (2020) <doi:10.1093/bioinformatics/btz730>.

**License** MIT + file LICENSE

**Encoding** UTF-8

**RoxygenNote** 7.2.3

**Imports** ggplot2 (>= 2.2.1), stats, withr

**Suggests** testthat (>= 3.0.0), nlme, knitr, rmarkdown

**Config/testthat/edition** 3

**VignetteBuilder** knitr

**URL** `https://rwparsons.github.io/circacompare/`

**Language** en-US

**NeedsCompilation** no

**Author** Rex Parsons [aut, cre] (<`https://orcid.org/0000-0002-6053-8174`>),
Alexander Bender [ctb]

**Maintainer** Rex Parsons <`Rex.Parsons94@gmail.com`>

**Repository** CRAN

**Date/Publication** 2024-01-09 21:43:03 UTC

## R topics documented:

---

`circacompare` *circacompare*

---

### Description

`circacompare` performs a comparison between two rhythmic groups of data. It tests for rhythmicity and then fits a nonlinear model with parametrization to estimate and statistically support differences in mesor, amplitude, and phase between groups.

### Usage

```
circacompare(
  x,
  col_time,
  col_group,
  col_outcome,
  period = 24,
  alpha_threshold = 0.05,
  timeout_n = 10000,
  control = list(),
  weights = NULL,
  suppress_all = FALSE
)
```

### Arguments

| | |
|---|---|
| `x` | data.frame. This is the data.frame which contains the rhythmic data for two groups in a tidy format. |
| `col_time` | The name of the column within the data.frame, x, which contains time in hours at which the data were collected. |
| `col_group` | The name of the column within the data.frame, x, which contains the grouping variable. This should only have two levels. |
| `col_outcome` | The name of the column within the data.frame, x, which contains outcome measure of interest. |
| `period` | The period of the rhythm. For circadian rhythms, leave this as the default value, 24. |
| `alpha_threshold` | The level of alpha for which the presence of rhythmicity is considered. Default is 0.05. |
| `timeout_n` | The upper limit for the model fitting attempts. Default is 10,000. |
| `control` | `list`. Used to control the parameterization of the model. |
| `weights` | An optional numeric vector of (fixed) weights. When present, the objective function is weighted least squares. |
| `suppress_all` | Logical. Set to `TRUE` to avoid seeing errors or messages during model fitting procedure. Default is `FALSE`. |

### Value

list

### Examples

```
df <- make_data(phi1 = 6)
out <- circacompare(
  x = df, col_time = "time", col_group = "group",
  col_outcome = "measure"
)
out

# with sample weights (arbitrary weights for demonstration)
sw <- runif(n = nrow(df))
out2 <- circacompare(
  x = df, col_time = "time", col_group = "group",
  col_outcome = "measure", weights = sw
)
out2
```

---

## `circacompare_mixed` *circacompare_mixed*

### Description

`circacompare_mixed` is similar to `circacompare` but allows for some simple, user-specified random-effects on the rhythmic parameters of choice.

### Usage

```
circacompare_mixed(
  x,
  col_time,
  col_group,
  col_outcome,
  col_id,
  randomeffects = c(),
  period = 24,
  alpha_threshold = 0.05,
  nlme_control = list(),
  nlme_method = "REML",
  weights = NULL,
  suppress_all = FALSE,
  timeout_n = 10000,
  control = list()
)
```

## Arguments

| | |
|---|---|
| `x` | `data.frame`. This is the data.frame which contains the rhythmic data for two groups in a tidy format. |
| `col_time` | The name of the column within the data.frame, `x`, which contains time in hours at which the data were collected. |
| `col_group` | The name of the column within the data.frame, `x`, which contains the grouping variable. This should only have two levels. |
| `col_outcome` | The name of the column within the data.frame, `x`, which contains outcome measure of interest. |
| `col_id` | The name of the column within the data.frame, `x`, which contains the identifying values for the random effect, such as `subject_id`. |
| `randomeffects` | which rhythmic parameters to allow random effects. The default is to include no rhythmic parameters. |
| `period` | The period of the rhythm. For circadian rhythms, leave this as the default value, `24`. |
| `alpha_threshold` | The level of alpha for which the presence of rhythmicity is considered. Default is to `0.05`. |
| `nlme_control` | A list of control values for the estimation algorithm to replace the default values returned by the function nlme::nlmeControl. Defaults to an empty list. |
| `nlme_method` | A character string. If "REML" the model is fit by maximizing the restricted log-likelihood. If "ML" the log-likelihood is maximized. Defaults to "REML". |
| `weights` | An optional numeric vector of (fixed) weights internally passed to `nlme::nlme()` via `nlme::varPower()`. When present, the objective function is weighted least squares. |
| `suppress_all` | Logical. Set to `TRUE` to avoid seeing errors or messages during model fitting procedure. Default is `FALSE`. If `FALSE`, also runs `nlme()` with `verbose = TRUE`. |
| `timeout_n` | The upper limit for the model fitting attempts. Default is `10000`. |
| `control` | `list`. Used to control the parameterization of the model. |

## Value

list

## Examples

```
# Generate some data with within-id correlation for phase-shift (phi1)

set.seed(99)
phi1_in <- 3.15

mixed_data <- function(n) {
  counter <- 1
  for (i in 1:n) {
   x <- make_data(k1 = 0, alpha1 = 0, phi1 = rnorm(1, phi1_in, 0.5), hours = 72, noise_sd = 1)
```

```
    x$id <- counter
    counter <- counter + 1
    if (i == 1) {
      res <- x
    } else {
      res <- rbind(res, x)
    }
  }
  return(res)
}
df <- mixed_data(20)
out <- circacompare_mixed(
  x = df,
  col_time = "time",
  col_group = "group",
  col_outcome = "measure",
  col_id = "id",
  control = list(grouped_params = c("phi"), random_params = c("phi1"))
)

# with sample weights (arbitrary weights for demonstration)
sw <- runif(n = nrow(df))
out2 <- circacompare_mixed(
  x = df,
  col_time = "time",
  col_group = "group",
  col_outcome = "measure",
  col_id = "id",
  control = list(grouped_params = c("phi"), random_params = c("phi1")),
  weights = sw
)
```

---

## `circa_single` *circa_single*

### Description

`circa_single` performs an analysis on a single rhythmic dataset. It estimates the mesor, amplitude and phase of the data provided.

### Usage

```
circa_single(
  x,
  col_time,
  col_outcome,
  period = 24,
  alpha_threshold = 0.05,
```

```
  timeout_n = 10000,
  return_figure = TRUE,
  control = list(),
  weights = NULL,
  suppress_all = FALSE
)
```

## Arguments

| | |
|---|---|
| `x` | data.frame. This is the data.frame which contains the rhythmic data in a tidy format. |
| `col_time` | The name of the column within the data.frame, x, which contains time in hours at which the data were collected. |
| `col_outcome` | The name of the column within the data.frame, x, which contains outcome measure of interest. |
| `period` | The period of the rhythm. For circadian rhythms, leave this as the default value, 24. |
| `alpha_threshold` | The level of alpha for which the presence of rhythmicity is considered. Default is 0.05. |
| `timeout_n` | The upper limit for the model fitting attempts. Default is 10,000. |
| `return_figure` | Whether or not to return a ggplot graph of the rhythm and cosine model. |
| `control` | `list`. Used to control the parameterization of the model. |
| `weights` | An optional numeric vector of (fixed) weights. When present, the objective function is weighted least squares. |
| `suppress_all` | Logical. Set to `TRUE` to avoid seeing errors or messages during model fitting procedure. Default is `FALSE`. |

## Value

list

## Examples

```
df <- make_data()
df <- df[df$group == "g1", ]
out <- circa_single(x = df, col_time = "time", col_outcome = "measure")
out

# with sample weights (arbitrary weights for demonstration)
sw <- runif(n = nrow(df))
out2 <- circa_single(
  x = df,
  col_time = "time",
  col_outcome = "measure",
  weights = sw,
  suppress_all = TRUE
)
```

```
out2
```

---

## `circa_single_mixed` *circa_single_mixed*

### Description

`circa_single_mixed` is similar to `circa_single` but allows for some simple, user-specified random-effects on the rhythmic parameters of choice.

### Usage

```
circa_single_mixed(
  x,
  col_time,
  col_outcome,
  col_id,
  randomeffects = c("k", "alpha", "phi"),
  period = 24,
  alpha_threshold = 0.05,
  nlme_control = list(),
  nlme_method = "ML",
  weights = NULL,
  suppress_all = FALSE,
  timeout_n = 10000,
  return_figure = TRUE,
  control = list()
)
```

### Arguments

| | |
|---|---|
| `x` | data.frame. This is the data.frame which contains the rhythmic data in a tidy format. |
| `col_time` | The name of the column within the data.frame, x, which contains time in hours at which the data were collected. |
| `col_outcome` | The name of the column within the data.frame, x, which contains outcome measure of interest. |
| `col_id` | The name of the column within the data.frame, `x`, which contains the identifying values for the random effect, such as `subject_id`. |
| `randomeffects` | which rhythmic parameters to allow random effects. The default is to include all rhythmic parameters. |
| `period` | The period of the rhythm. For circadian rhythms, leave this as the default value, `24`. |

| | |
|---|---|
| `alpha_threshold` | The level of alpha for which the presence of rhythmicity is considered. Default is to `0.05`. |
| `nlme_control` | A list of control values for the estimation algorithm to replace the default values returned by the function nlme::nlmeControl. Defaults to an empty list. |
| `nlme_method` | A character string. If "REML" the model is fit by maximizing the restricted log-likelihood. If "ML" the log-likelihood is maximized. Defaults to "ML". |
| `weights` | An optional numeric vector of (fixed) weights internally passed to `nlme::nlme()` via `nlme::varPower()`. When present, the objective function is weighted least squares. |
| `suppress_all` | Logical. Set to `TRUE` to avoid seeing errors or messages during model fitting procedure. Default is `FALSE`. If `FALSE`, also runs `nlme()` with `verbose = TRUE`. |
| `timeout_n` | The upper limit for the model fitting attempts. Default is `10000`. |
| `return_figure` | Whether or not to return a ggplot graph of the rhythm and cosine model. |
| `control` | `list`. Used to control the parameterization of the model. |

## Value

list

## Examples

```
set.seed(42)
mixed_data <- function(n) {
  counter <- 1
  for (i in 1:n) {
    x <- make_data(k1 = rnorm(1, 10, 2), alpha1 = 0, phi1 = 0)
    x$id <- counter
    counter <- counter + 1
    if (i == 1) {
      res <- x
    } else {
      res <- rbind(res, x)
    }
  }
  return(res)
}
df <- mixed_data(n = 50)
out <- circa_single_mixed(
  x = df, col_time = "time", col_outcome = "measure",
  col_id = "id", randomeffects = c("k")
)

# with sample weights (arbitrary weights for demonstration)
sw <- runif(n = nrow(df))
out2 <- circa_single_mixed(
  x = df, col_time = "time", col_outcome = "measure",
  col_id = "id", randomeffects = c("k"), weights = sw
```

```
)
```

---

`make_data` *make_data*

---

### Description

Generate example circadian data with specified phase shift between groups

### Usage

```
make_data(
  k = 0,
  k1 = 3,
  alpha = 10,
  alpha1 = 4,
  phi = 0,
  phi1 = 3.15,
  tau = 24,
  hours = 48,
  noise_sd = 0.1,
  seed = NULL
)
```

### Arguments

| | |
|---|---|
| `k` | mesor of group 1. |
| `k1` | change in mesor in group 2 from group 1. |
| `alpha` | amplitude rhythm for group 1. |
| `alpha1` | change in amplitude in group 2 from group 1 |
| `phi` | phase of rhythm, in radian-hours, in group 1. |
| `phi1` | change in phase, in radian-hours, in group 2 from group 1 |
| `tau` | period of the rhythm, shared between both groups. |
| `hours` | the number of hours/datapoints to sample. |
| `noise_sd` | the standard deviation of the noise term. |
| `seed` | random seed for generating data. |

### Value

data.frame

### Examples

```
data <- make_data(k1 = 3, alpha1 = 4, phi1 = 6)
```

# Index