

Package: GeneSCARAB (via r-universe)

October 10, 2026

Title Gene Set Circular Analysis for Rhythms Ascertainment in Bioprocesses

Version 0.99.3

Description GeneSCARAB includes a range of features designed to enable a robust and comprehensive analysis of the circular distributions of biomolecules (genes, proteins, metabolites, ...) associated with bioprocesses in any species. It is specifically designed for circadian omics studies and facilitates data organization, statistical analysis, and the visualization of results. It works with the output of standard rhythm detection software, such as CircaCompare, MetaCycle or RAIN.

License Artistic-2.0

Encoding UTF-8

Roxygen list(markdown = TRUE)

Imports BiocGenerics, circlize, CircMLE, circular, ComplexHeatmap, Directional, GO.db, grid, KEGGREST, MetBrewer, OptCirClust, AnnotationDbi, methods

Depends R (>= 4.5.0)

LazyData false

Suggests BiocStyle, knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/roxygen2/version 8.0.0

biocViews Genetics, MultipleComparison, Proteomics, Software, SystemsBiology, TimeCourse, Transcriptomics, StatisticalMethod

URL <https://github.com/ramosgonzmarc/GeneSCARAB>

BugReports <https://github.com/ramosgonzmarc/GeneSCARAB>

Config/testthat/edition 3

Config/pak/sysreqs libabsl-dev cmake libfreetype6-dev libgdal-dev gdal-bin libgeos-dev libglu1-mesa-dev make libgsl0-dev texlive libicu-dev libpng-dev libuv1-dev libg11-mesa-dev libssl-dev perl libproj-dev libsqlite3-dev libudunits2-dev zlib1g-dev

Repository <https://biocstaging.r-universe.dev>

Date/Publication 2026-09-03 14:17:25 UTC

RemoteUrl <https://github.com/BiocStaging/GeneSCARAB>

RemoteRef HEAD

RemoteSha ea64178b43536454a2db5ca289608cd20a39a178

Contents

.create_circular_table	3
.create_circular_table_grouped	4
.get_ancestor_for_gene_lists	5
.helper_init_plot	5
.HermansRasson2PGroupedRad	6
.HermansRasson2T	6
.input_complete_circular_table	7
.input_create_circular_table	7
.input_create_gene_list_go	8
.input_create_gene_list_kegg	8
.input_gene_contribution	9
.KuiperPGroupedRad	10
.legend_builder	10
.legend_builder_boxplot	11
.multimodal_modes_test	11
.pathways_from_kos	12
.RaoPGroupedRad	12
.RaoTestUngroupedRad	13
.RaoTestValue	13
.table_builder	14
.track_box_box_cut	14
.track_box_no_cut	15
.track_box_segments_cut	16
.track_dot	16
circa_table_genescarab	17
circular_boxplot	17
circular_dotplot	18
circular_histogram	20
complete_circular_table	21
complete_circular_table_grouped	23
create_gene_list_go	25
create_gene_list_kegg	26
gene_contribution_to_set	27
gene_list_to_phases	28
GeneSCARAB	29
load_example_annot	31
multimodal_analysis	31
test_against_gen_dist	32
test_two_dist	34

`.create_circular_table`*Creation of the circular table and deviation from the uniform distribution for a single set*

Description

Function that takes a vector of gene phases (continuous measurements or parametric estimates) and returns statistical measures of their circular distribution and their deviations from a circular uniform distribution based on the Rayleigh, Kuiper, Hermans-Rasson and Rao tests

Usage

```
.create_circular_table(  
  circa_vector,  
  hr.on.large.sets.th = 400,  
  iter.hr = 999,  
  force.hr.th = 0.05,  
  rao.on.large.sets.th = 400,  
  iter.rao = 999,  
  force.rao.th = 0.05  
)
```

Arguments

<code>circa_vector</code>	Phases vector in hours
<code>hr.on.large.sets.th</code>	Threshold indicating the maximum size of a set to perform HR test on
<code>iter.hr</code>	Number of iterations for HR test
<code>force.hr.th</code>	Significance level to execute HR test if Rayleigh and Kuiper p-values are higher
<code>rao.on.large.sets.th</code>	Threshold indicating the maximum size of a set to perform Rao test on
<code>iter.rao</code>	Number of iterations for Rao test
<code>force.rao.th</code>	Significance level to execute Rao test if Rayleigh, Kuiper and HR p-values are higher

Value

A table indicating number of phases used for computation (n); mean, rho, variance, first, second and third quantiles of the circular distribution in hours; and p-values for each of the four tests

`.create_circular_table_grouped`

Creation of the circular table and deviation from the uniform distribution for a single set of grouped data

Description

Function that takes a vector of gene phases (discrete measurements or non-parametric estimates) and returns statistical measures of their circular distribution and their deviations from a circular uniform distribution based on Rayleigh and adapted versions of Kuiper, Hermans-Rasson and Rao tests.

Usage

```
.create_circular_table_grouped(
  circa_vector,
  n_bins,
  hr.on.large.sets.th = 400,
  iter.hr = 999,
  force.hr.th = 0.05,
  rao.on.large.sets.th = 400,
  iter.rao = 999,
  force.rao.th = 0.05,
  kuiper.on.large.sets.th = 400,
  iter.kuiper = 999,
  force.kuiper.th = 0.05
)
```

Arguments

<code>circa_vector</code>	Discrete phases vector in hours
<code>n_bins</code>	Number of bins
<code>hr.on.large.sets.th</code>	Threshold indicating the maximum size of a set to perform HR test on
<code>iter.hr</code>	Number of iterations for HR test
<code>force.hr.th</code>	Significance level to execute HR test if Rayleigh and Kuiper p-values are higher
<code>rao.on.large.sets.th</code>	Threshold indicating the maximum size of a set to perform Rao test on
<code>iter.rao</code>	Number of iterations for Rao test
<code>force.rao.th</code>	Significance level to execute Rao test if Rayleigh, Kuiper and HR p-values are higher
<code>kuiper.on.large.sets.th</code>	Threshold indicating the maximum size of a set to perform Kuiper test on
<code>iter.kuiper</code>	Number of iterations for Kuiper test
<code>force.kuiper.th</code>	Significance level to execute Kuiper test if Rayleigh p-value is higher

Value

A data.frame indicating number of phases used for computation (n); mean, rho, variance, first, second and third quartiles of the circular distribution in hours; and p-values for each of the four tests

`.get_ancestor_for_gene_lists`

Extraction of all ancestors of a given GO terms vector

Description

Function to extract all ancestors of given GO terms

Usage

```
.get_ancestor_for_gene_lists(go_vector)
```

Arguments

<code>go_vector</code>	GO terms vector to be included in the analysis, along with their ancestors. "all" takes the complete set from the annotation package
------------------------	--

Value

A vector containing ancestors of GO terms

`.helper_init_plot`

Helper circular grid creation

Description

Function to create axis and initialize plot

Usage

```
.helper_init_plot()
```

Value

Axis

```
.HermansRasson2PGroupedRad
```

HR test's p-value calculation for grouped data

Description

Helper function that calculates the p-value of an adaptation of the Hermans-Rasson test for grouped data (discrete phases) as implemented in the article "Grouped circular data in biology: advice for effectively implementing statistical procedures" <https://doi.org/10.1007/s00265-020-02881-6>.

Usage

```
.HermansRasson2PGroupedRad(sample, m, iter = 9999)
```

Arguments

sample	Data points in radians
m	Number of bins
iter	Number of iterations

Value

The p-value for the test

```
.HermansRasson2T
```

Critical value calculation for HR test

Description

A helper function that calculates the critical value for the HR test, as implemented in the article "Grouped circular data in biology: advice for effectively implementing statistical procedures" <https://doi.org/10.1007/s00265-020-02881-6>.

Usage

```
.HermansRasson2T(sample)
```

Arguments

sample	Data points in radians
--------	------------------------

Value

The critical value for the test

`.input_complete_circular_table`

Input checking for complete_circular_table and complete_circular_table_grouped

Description

Helper function checking correctness of input for `complete_circular_table` and `complete_circular_table_grouped`

Usage

`.input_complete_circular_table(phase.list)`

Arguments

`phase.list` Phases list in the same format as the output of `gene_list_to_phases`

Value

An error if input is not adequate

`.input_create_circular_table`

Input checking for .create_circular_table and .create_circular_table_grouped

Description

Helper function checking correctness of input for `.create_circular_table` and `.create_circular_table_grouped`

Usage

`.input_create_circular_table(circa_vector)`

Arguments

`circa_vector` Phases vector in hours

Value

An error if input is not adequate

```
.input_create_gene_list_go
```

Input checking for create_gene_list_go

Description

Helper function checking correctness of input for `create_gene_list_go` and generating a `data.frame` with all the GO terms per gene

Usage

```
.input_create_gene_list_go(go_vector, org.package, go_column, id_column)
```

Arguments

<code>go_vector</code>	GO terms vector to be included in the analysis, along with their ancestors. "all" takes the complete set from the annotation package
<code>org.package</code>	OrgDb object from species annotation package (*.db)
<code>go_column</code>	GO term column name in the annotation package
<code>id_column</code>	Gene ID column name in the annotation package

Value

A `data.frame` showing GO terms per gene as appear in the annotation package

```
.input_create_gene_list_kegg
```

Input checking for create_gene_list_kegg

Description

Helper function checking correctness of input for `create_gene_list_kegg` and generating a `data.frame` with all the KOs per gene

Usage

```
.input_create_gene_list_kegg(
  ko_vector = c("all"),
  org.package,
  ko_column,
  id_column,
  ko_prefix = "map",
  species = "all"
)
```

Arguments

<code>ko_vector</code>	KO vector to be included in the analysis, along with their pathways ancestors. "all" takes the complete set from the annotation package
<code>org.package</code>	OrgDb object from species annotation package (*.db)
<code>ko_column</code>	KO column name in the annotation package
<code>id_column</code>	Gene ID column name in the annotation package
<code>ko_prefix</code>	Prefix used to denote KEGG pathways, either "map" or "ko", depending on package version
<code>species</code>	Species group to filter final pathways ("all" for no filtering), either "bacteria", "archaea", "animals", "plants", "fungi" or "protists"

Value

A data.frame showing KO terms per gene as appear in the annotation package

`.input_gene_contribution`*Input checking for gene_contribution_to_set*

Description

Helper function checking correctness of input for `gene_contribution_to_set` and returning a clean phase list

Usage

```
.input_gene_contribution(circa_result, phase_list)
```

Arguments

<code>circa_result</code>	Table with circular distribution results, as the output from <code>complete_circular_table</code> or <code>complete_circular_table_grouped</code>
<code>phase_list</code>	Phases list in the same format as the output of <code>gene_list_to_phases</code> , containing the row names of <code>circa_result</code> as names of each element

Value

A phase list containing only elements that appear in `circa_result` rows

<code>.KuiperPGroupedRad</code>	<i>Kuiper test's p-value calculation for grouped data</i>
---------------------------------	---

Description

Helper function that calculates the p-value of an adaptation of the Kuiper test for grouped data (discrete phases) as implemented in the article "Grouped circular data in biology: advice for effectively implementing statistical procedures" <https://doi.org/10.1007/s00265-020-02881-6>.

Usage

```
.KuiperPGroupedRad(sample, m, iter = 9999)
```

Arguments

<code>sample</code>	Data points in radians
<code>m</code>	Number of bins
<code>iter</code>	Number of iterations

Value

The p-value for the test

<code>.legend_builder</code>	<i>Helper legend builder for dotplot and histogram</i>
------------------------------	--

Description

Function to create tracks for dotplot

Usage

```
.legend_builder(phases.list, my_color)
```

Arguments

<code>phases.list</code>	A named phase tables list in the same format as the output of <code>gene_list_to_phases</code>
<code>my_color</code>	Color vector to use

Value

Legend of the plots

.legend_builder_boxplot

Helper legend builder for boxplot

Description

Function to create tracks for boxplot

Usage

```
.legend_builder_boxplot(quan.table, my_color)
```

Arguments

quan.table	A circular results table with sem info
my_color	Color vector to use

Value

Legend of the plots

.multimodal_modes_test

Identification of different modes

Description

Function to determine the extract the Rayleigh p-value in the presence of f-fold symmetry (1, 2 or 3)

Usage

```
.multimodal_modes_test(phase_table)
```

Arguments

phase_table	A phase table, as the individual elements of the list derived from gene_list_to_phases
-------------	--

Value

A vector containing the p-values for 1,2 and 3 f-fold symmetry

<code>.pathways_from_kos</code>	<i>Extraction of KEGG pathways from KO terms</i>
---------------------------------	--

Description

Function to extract KEGG pathways associated to a vector of KO terms and create a list containing this information

Usage

```
.pathways_from_kos(ko_vector, ko_prefix)
```

Arguments

<code>ko_vector</code>	KO vector to extract pathways
<code>ko_prefix</code>	Prefix used to denote KEGG pathways, either "map" or "ko", depending on package version

Value

A list of pathways associated to the KO terms input

<code>.RaoPGroupedRad</code>	<i>Rao's spacing test's p-value calculation for grouped data</i>
------------------------------	--

Description

Helper function that calculates the p-value of an adaptation of the Rao's spacing test for grouped data (discrete phases) as implemented in the article "Grouped circular data in biology: advice for effectively implementing statistical procedures" <https://doi.org/10.1007/s00265-020-02881-6>.

Usage

```
.RaoPGroupedRad(sample, m, iter = 9999)
```

Arguments

<code>sample</code>	Data points in radians
<code>m</code>	Number of bins
<code>iter</code>	Number of iterations

Value

The p-value for the test

.RaoTestUngroupedRad *Rao's spacing test's p-value calculation for ungrouped data*

Description

Helper function that calculates the p-value of Rao's spacing test for ungrouped data (continuous phases) as implemented in the article "Circular statistics meets practical limitations: a simulation-based Rao's spacing test for non-continuous data" <https://doi.org/10.1186/s40462-019-0160-x>.

Usage

```
.RaoTestUngroupedRad(sample, iter = 9999)
```

Arguments

sample	Data points in radians
iter	Number of iterations

Value

The p-value for the test

.RaoTestValue *Critical value calculation for Rao's spacing test*

Description

Helper function that calculates the critical value for Rao's spacing test as implemented in the article "Grouped circular data in biology: advice for effectively implementing statistical procedures" <https://doi.org/10.1007/s00265-020-02881-6>.

Usage

```
.RaoTestValue(sample)
```

Arguments

sample	Data points in radians
--------	------------------------

Value

The critical value for the test

.table_builder	<i>Helper function to construct table from circular data</i>
----------------	--

Description

Function that takes the calculated circular summary, variance and p-values of Rayleigh, Kuiper, HR and Rao's tests and construct the result table.

Usage

```
.table_builder(
  circa_summary,
  circa_var,
  circa_kuiper,
  circa_ray,
  circa_hr,
  circa_rao
)
```

Arguments

circa_summary	Process or gene sets list (as the output of create_gene_list_go and create_gene_list_kegg)
circa_var	Table showing acrophases for the complete gene set under study, stated in hours
circa_kuiper	P-value estimated by Kuiper test
circa_ray	P-value estimated by Rayleigh test
circa_hr	P-value estimated by HR test
circa_rao	P-value estimated by Rao test

Value

A table indicating number of phases used for computation (n); mean, rho, variance, first, second and third quantiles of the circular distribution in hours; and p-values for each of the four tests

.track_box_box_cut	<i>Helper track creation boxplot for box cutting the origin</i>
--------------------	---

Description

Function to create tracks for boxplots whose boxes intersect the origin. In this case, the segment must be cut from min to 24 and from 0 to max. Also, the box must be unfolded from first to 24 and from 0 to third quartiles.

Usage

```
.track_box_box_cut(tr.height, quan.table, i, my_color)
```

Arguments

<code>tr.height</code>	Height of each track. For more than 8 tracks, reduce this parameter
<code>quan.table</code>	A circular results table with sem info
<code>i</code>	Track counter
<code>my_color</code>	Color vector to use

Value

Plotted track

<code>.track_box_no_cut</code>	<i>Helper track creation boxplot without cut</i>
--------------------------------	--

Description

Function to create tracks for boxplots that do not intersect the origin. If neither of the segments cut the sunrise,

Usage

```
.track_box_no_cut(tr.height, quan.table, i, my_color)
```

Arguments

<code>tr.height</code>	Height of each track. For more than 8 tracks, reduce this parameter
<code>quan.table</code>	A circular results table with sem info
<code>i</code>	Track counter
<code>my_color</code>	Color vector to use

Value

Plotted track

`.track_box_segments_cut`

Helper track creation boxplot for segments cutting the origin

Description

Function to create tracks for boxplots whose segments intersect the origin.

Usage

```
.track_box_segments_cut(tr.height, quan.table, i, my_color)
```

Arguments

<code>tr.height</code>	Height of each track. For more than 8 tracks, reduce this parameter
<code>quan.table</code>	A circular results table with sem info
<code>i</code>	Track counter
<code>my_color</code>	Color vector to use

Value

Plotted track

`.track_dot`

Helper track creation dotplot

Description

Function to create tracks for dotplot

Usage

```
.track_dot(tr.height, phases.list, i, my_color, my_transparent)
```

Arguments

<code>tr.height</code>	Height of each track. For more than 8 tracks, reduce this parameter
<code>phases.list</code>	A named phase tables list in the same format as the output of <code>gene_list_to_phases</code>
<code>i</code>	Track counter
<code>my_color</code>	Color vector to use
<code>my_transparent</code>	Transparent color vector to use

Value

Axis

circa_table_genescarab	<i>circa_table_genescarab</i>
------------------------	-------------------------------

Description

Dataset example with phase information in two different conditions for *Ostreococcus tauri* from "Multiomics integration unveils photoperiodic plasticity in the molecular rhythms of marine phytoplankton" (<https://doi.org/10.1093/plcell/koaf033>)

Usage

```
data(circa_table_genescarab)
```

Format

Data frame

circular_boxplot	<i>Circular boxplot</i>
------------------	-------------------------

Description

Function to plot a circular boxplot showing the circular distribution of different processes or gene sets

Usage

```
circular_boxplot(res.table, color.palette = "Tam", tr.height = 0.1)
```

Arguments

- res.table A circular results table, as the output from complete_circular_table or complete_circular_table_grouped
- color.palette Color palette to use. Available palettes are those from MetBrewer
- tr.height Height of each track. For more than 8 tracks, reduce this parameter

Value

A PNG file containing the plot

Examples

```

org_Otaurireduced_eg_db <- load_example_annot()
data("circa_table_genesrab")
total_phases_table_sd <- data.frame(
  names = rownames(circa_table_genesrab),
  phase = as.numeric(
    circa_table_genesrab[["sd.peak.time.hours"]]
  )
)

functional_data <- AnnotationDbi::select(org_Otaurireduced_eg_db,
  keys = AnnotationDbi::keys(org_Otaurireduced_eg_db, keytype = "GID"),
  columns = c("GID", "GO")
)
complete_gos <- unique(functional_data$GO)
complete_gos <- complete_gos[!is.na(complete_gos)]

set.seed(2345)
subset_gos <- complete_gos[sample(
  seq_len(length(complete_gos)), 50,
  replace = FALSE
)]
go.list.test <- create_gene_list_go(
  go_vector = subset_gos,
  org.package = org_Otaurireduced_eg_db,
  go_column = "GO", id_column = "GID"
)
phases.list.sd <- gene_list_to_phases(
  go.list.test,
  total_phases_table_sd
)
phases.list.sd.clean <- phases.list.sd[which(
  sapply(phases.list.sd, nrow) != 0
)][seq_len(3)]
go.circa.table.sd <- complete_circular_table(
  phase.list = phases.list.sd.clean,
  hr.on.large.sets.th = 200,
  force.hr.th = 0.04,
  rao.on.large.sets.th = 200,
  force.rao.th = 0.04,
  iter.rao = 999, iter.hr = 999
)
circular_boxplot(go.circa.table.sd[seq_len(2), ],
  color.palette = "Tam"
)

```

Description

Function to plot a circular dotplot showing the circular distribution of different processes or gene sets

Usage

```
circular_dotplot(phases.list, color.palette = "Tam", tr.height = 0.1)
```

Arguments

<code>phases.list</code>	A named phase tables list in the same format as the output of <code>gene_list_to_phases</code>
<code>color.palette</code>	Color palette to use. Available palettes are those from MetBrewer
<code>tr.height</code>	Height of each track. For more than 8 tracks, reduce this parameter

Value

A PNG file containing the plot

Examples

```
org_Otaurireduced_eg_db <- load_example_annot()
data("circa_table_genescarab")
total_phases_table_sd <- data.frame(
  names = rownames(circa_table_genescarab),
  phase = as.numeric(
    circa_table_genescarab[["sd.peak.time.hours"]]
  )
)

functional_data <- AnnotationDbi::select(org_Otaurireduced_eg_db,
  keys = AnnotationDbi::keys(org_Otaurireduced_eg_db, keytype = "GID"),
  columns = c("GID", "GO")
)
complete_gos <- unique(functional_data$GO)
complete_gos <- complete_gos[!is.na(complete_gos)]

set.seed(2345)
subset_gos <- complete_gos[sample(
  seq_len(length(complete_gos)), 50,
  replace = FALSE
)]
go.list.test <- create_gene_list_go(
  go_vector = subset_gos,
  org.package = org_Otaurireduced_eg_db,
  go_column = "GO", id_column = "GID"
)
phases.list.sd <- gene_list_to_phases(
  go.list.test, total_phases_table_sd
)
phases.list.sd.clean <- phases.list.sd[which(
  sapply(phases.list.sd, nrow) != 0
```

```

)][seq_len(3)]
circular_dotplot(phases.list.sd.clean, color.palette = "Tam")

```

circular_histogram *Circular histogram*

Description

Function to plot a circular histogram showing the circular distribution of different processes or gene sets

Usage

```

circular_histogram(
  phases.list,
  color.palette = "Tam",
  tr.height = 0.08,
  nbins = 24
)

```

Arguments

<code>phases.list</code>	A phase tables list in the same format as the output of <code>gene_list_to_phases</code>
<code>color.palette</code>	Color palette to use. Available palettes are those from MetBrewer
<code>tr.height</code>	Height of each track. For more than 8 tracks, reduce this parameter
<code>nbins</code>	Number of bins (breaks) to split the histogram

Value

A PNG file containing the plot

Examples

```

org_Otaurireduced_eg_db <- load_example_annot()
data("circa_table_genescaab")
total_phases_table_sd <- data.frame(
  names = rownames(circa_table_genescaab),
  phase = as.numeric(
    circa_table_genescaab[["sd.peak.time.hours"]]
  )
)

functional_data <- AnnotationDbi::select(org_Otaurireduced_eg_db,
  keys = AnnotationDbi::keys(org_Otaurireduced_eg_db, keytype = "GID"),
  columns = c("GID", "GO")
)
complete_gos <- unique(functional_data$GO)

```

```

complete_gos <- complete_gos[!is.na(complete_gos)]

set.seed(2345)
subset_gos <- complete_gos[sample(
  seq_len(length(complete_gos)), 50,
  replace = FALSE
)]
go.list.test <- create_gene_list_go(
  go_vector = subset_gos,
  org.package = org_Otaurireduced_eg_db,
  go_column = "GO", id_column = "GID"
)
phases.list.sd <- gene_list_to_phases(
  go.list.test, total_phases_table_sd
)
phases.list.sd.clean <- phases.list.sd[which(
  sapply(phases.list.sd, nrow) != 0
)][seq_len(3)]
circular_histogram(phases.list.sd.clean,
  color.palette = "Tam", nbins = 48
)

```

complete_circular_table

Creation of the circular table and deviation from the uniform distribution for a phases list

Description

Function that takes a list of continuous phases associated with each process or gene set (output of the `gene_list_to_phases` function) and calculates the table of statistical measures and deviations from circular uniformity for each one using the Rayleigh, Kuiper, Hermans-Rasson, and Rao tests.

Usage

```

complete_circular_table(
  phase.list,
  hr.on.large.sets.th = 400,
  iter.hr = 999,
  force.hr.th = 0.05,
  rao.on.large.sets.th = 400,
  iter.rao = 999,
  force.rao.th = 0.05
)

```

Arguments

<code>phase.list</code>	Phases list in the same format as the output of <code>gene_list_to_phases</code>
<code>hr.on.large.sets.th</code>	Threshold indicating the maximum size of a set to perform HR test on
<code>iter.hr</code>	Number of iterations for HR test
<code>force.hr.th</code>	Significance level to execute HR test if Rayleigh and Kuiper p-values are higher
<code>rao.on.large.sets.th</code>	Threshold indicating the maximum size of a set to perform Rao test on
<code>iter.rao</code>	Number of iterations for Rao test
<code>force.rao.th</code>	Significance level to execute Rao test if Rayleigh, Kuiper and HR p-values are higher

Value

A data.frame indicating, for each process or gene set: number of phases used for computation (n); mean, rho, variance, first, second and third quantiles of the circular distribution in hours; p-values and adjusted p-values for each of the four tests

Examples

```
org_Otaurireduced_eg_db <- load_example_annot()
data("circa_table_genesArab")
total_phases_table_sd <- data.frame(
  names = rownames(circa_table_genesArab),
  phase = as.numeric(circa_table_genesArab[["sd.peak.time.hours"]])
)
functional_data <- AnnotationDbi::select(org_Otaurireduced_eg_db,
  keys = AnnotationDbi::keys(org_Otaurireduced_eg_db, keytype = "GID"),
  columns = c("GID", "GO")
)
complete_gos <- unique(functional_data$GO)
complete_gos <- complete_gos[!is.na(complete_gos)]

set.seed(2345)
subset_gos <- complete_gos[sample(
  seq_len(length(complete_gos)), 50,
  replace = FALSE
)]
go.list.test <- create_gene_list_go(
  go_vector = subset_gos,
  org.package = org_Otaurireduced_eg_db,
  go_column = "GO", id_column = "GID"
)
phases.list.sd <- gene_list_to_phases(
  go.list.test, total_phases_table_sd
)
phases.list.sd.clean <- phases.list.sd[which(
  sapply(phases.list.sd, nrow) != 0
)][seq_len(3)]
```

```

go.circa.table.sd <- complete_circular_table(
  phase.list = phases.list.sd.clean,
  hr.on.large.sets.th = 200,
  force.hr.th = 0.04,
  rao.on.large.sets.th = 200,
  force.rao.th = 0.04,
  iter.rao = 999, iter.hr = 999
)

```

complete_circular_table_grouped

Creation of the circular table and deviation from the uniform distribution for a discrete (grouped) phases list

Description

Function that takes a list of discrete phases associated with each process or gene set (output of the `gene_list_to_phases` function) and calculates the table of statistical measures and deviations from circular uniformity for each one using the adapted Rayleigh, Kuiper, Hermans-Rasson, and Rao tests.

Usage

```

complete_circular_table_grouped(
  phase.list,
  n_bins,
  hr.on.large.sets.th = 400,
  iter.hr = 999,
  force.hr.th = 0.05,
  rao.on.large.sets.th = 400,
  iter.rao = 999,
  force.rao.th = 0.05,
  kuiper.on.large.sets.th = 400,
  iter.kuiper = 999,
  force.kuiper.th = 0.05
)

```

Arguments

<code>phase.list</code>	Phases list in the same format as the output of <code>gene_list_to_phases</code>
<code>n_bins</code>	Number of bins
<code>hr.on.large.sets.th</code>	Threshold indicating the maximum size of a set to perform HR test on
<code>iter.hr</code>	Number of iterations for HR test
<code>force.hr.th</code>	Significance level to execute HR test if Rayleigh and Kuiper p-values are higher

```

rao.on.large.sets.th      Threshold indicating the maximum size of a set to perform Rao test on
iter.rao                  Number of iterations for Rao test
force.rao.th              Significance level to execute Rao test if Rayleigh, Kuiper and HR p-values are
                           higher
kuiper.on.large.sets.th   Threshold indicating the maximum size of a set to perform kuiper test on
iter.kuiper               Number of iterations for kuiper test
force.kuiper.th           Significance level to execute Kuiper test if Rayleigh p-value is higher

```

Value

A data.frame indicating, for each process or gene set: number of phases used for computation (n); mean, rho, variance, first, second and third quantiles of the circular distribution in hours; p-values and adjusted p-values for each of the four tests

Examples

```

org_Otaurireduced_eg_db <- load_example_annot()
data("circa_table_genesrab")
total_phases_table_sd <- data.frame(
  names = rownames(circa_table_genesrab),
  phase = as.numeric(
    circa_table_genesrab[["sd.peak.time.hours"]]
  )
)
total_phases_table_sd_grouped <- data.frame(
  names = total_phases_table_sd$names,
  phase = trunc(total_phases_table_sd$phase)
)

functional_data <- AnnotationDbi::select(org_Otaurireduced_eg_db,
  keys = AnnotationDbi::keys(org_Otaurireduced_eg_db, keytype = "GID"),
  columns = c("GID", "GO")
)
complete_gos <- unique(functional_data$GO)
complete_gos <- complete_gos[!is.na(complete_gos)]

set.seed(2345)
subset_gos <- complete_gos[sample(
  seq_len(length(complete_gos)), 50,
  replace = FALSE
)]
go.list.test <- create_gene_list_go(
  go_vector = subset_gos,
  org.package = org_Otaurireduced_eg_db,
  go_column = "GO", id_column = "GID"
)
phases.list.sd <- gene_list_to_phases(

```

```

    go.list.test, total_phases_table_sd_grouped
  )
  phases.list.sd.clean <- phases.list.sd[which(
    sapply(phases.list.sd, nrow) != 0
  )][seq_len(3)]
  go.circa.table.sd <- complete_circular_table_grouped(
    phase.list = phases.list.sd.clean, n_bins = 24,
    hr.on.large.sets.th = 200,
    force.hr.th = 0.04,
    rao.on.large.sets.th = 200,
    force.rao.th = 0.04,
    iter.rao = 999, iter.hr = 999
  )

```

create_gene_list_go *Creation of input lists for GO terms from an annotation package*

Description

Function to create a list of genes associated with each GO term based on a species annotation package. This includes all ancestor GO terms of the selected ones.

Usage

```
create_gene_list_go(go_vector = c("all"), org.package, go_column, id_column)
```

Arguments

go_vector	GO terms vector to be included in the analysis, along with their ancestors. "all" takes the complete set from the annotation package
org.package	OrgDb object from species annotation package (*.db)
go_column	GO term column name in the annotation package
id_column	Gene ID column name in the annotation package

Value

A list linking each GO term with the genes that share that annotation

Examples

```

org_Otaurireduced_eg_db <- load_example_annot()
go.list.test <- create_gene_list_go(
  go_vector = "GO:0005515",
  org.package = org_Otaurireduced_eg_db, go_column = "GO",
  id_column = "GID"
)

```

`create_gene_list_kegg` *Creation of input lists for KEGG pathways from an annotation package*

Description

Function to create a list of genes associated with each KEGG pathway based on a species annotation package with KO terms derived from KEGGREST, KAAS or similar tools. It also allows filtering by species group.

Usage

```
create_gene_list_kegg(
  ko_vector = c("all"),
  org.package,
  ko_column,
  id_column,
  ko_prefix = "map",
  species = "all"
)
```

Arguments

<code>ko_vector</code>	KO vector to be included in the analysis, along with their pathways ancestors. "all" takes the complete set from the annotation package
<code>org.package</code>	OrgDb object from species annotation package (*.db)
<code>ko_column</code>	KO column name in the annotation package
<code>id_column</code>	Gene ID column name in the annotation package
<code>ko_prefix</code>	Prefix used to denote KEGG pathways, either "map" or "ko", depending on package version
<code>species</code>	Species group to filter final pathways ("all" for no filtering), either "bacteria", "archaea", "animals", "plants", "fungi" or "protists"

Value

A list linking each KEGG pathway with the genes that share that annotation

Examples

```
org_Otaurireduced_eg_db <- load_example_annot()
create_gene_list_kegg(
  ko_vector = "K10666", org.package = org_Otaurireduced_eg_db,
  ko_column = "KO", id_column = "GID", ko_prefix = "map",
  species = "plants"
)
```

gene_contribution_to_set

Contribution of each gene in a set to the temporal cohesion of the whole set

Description

Function that measures the contribution to the mean and rho of the circular distribution of a process or gene set for each of its constituent genes by removing them one by one. For sets consisting on more than 100 genes, it takes a random sample of 100 gene phases to normalize and ensure noticeable changes.

Usage

```
gene_contribution_to_set(circa_result, phase_list)
```

Arguments

circa_result	Table with circular distribution results, as the output from complete_circular_table or complete_circular_table_grouped
phase_list	Phases list in the same format as the output of gene_list_to_phases, containing the row names of circa_result as names of each element

Value

A list whose elements are tables indicating the difference in circular mean and rho due to removing each gene for each process or gene set, and the rank of each difference

Examples

```
org_Otaurireduced_eg_db <- load_example_annot()
data("circa_table_genescarab")
total_phases_table_sd <- data.frame(
  names = rownames(
    circa_table_genescarab
  ),
  phase = as.numeric(
    circa_table_genescarab[["sd.peak.time.hours"]]
  )
)

functional_data <- AnnotationDbi::select(org_Otaurireduced_eg_db,
  keys = AnnotationDbi::keys(org_Otaurireduced_eg_db, keytype = "GID"),
  columns = c("GID", "GO")
)
complete_gos <- unique(functional_data$GO)
complete_gos <- complete_gos[!is.na(complete_gos)]

set.seed(2345)
```

```

subset_gos <- complete_gos[sample(
  seq_len(length(complete_gos)), 50,
  replace = FALSE
)]
go.list.test <- create_gene_list_go(
  go_vector = subset_gos,
  org.package = org_Otaurireduced_eg_db,
  go_column = "GO", id_column = "GID"
)
phases.list.sd <- gene_list_to_phases(
  go.list.test, total_phases_table_sd
)
phases.list.sd.clean <- phases.list.sd[which(
  sapply(phases.list.sd, nrow) != 0
)][seq_len(3)]
go.circa.table.sd <- complete_circular_table(
  phase.list = phases.list.sd.clean,
  hr.on.large.sets.th = 200,
  force.hr.th = 0.04,
  rao.on.large.sets.th = 200,
  force.rao.th = 0.04,
  iter.rao = 999, iter.hr = 999
)
gene_contributions <- gene_contribution_to_set(
  circa_result =
    go.circa.table.sd[1, ],
  phase_list = phases.list.sd.clean[rownames(
    go.circa.table.sd
  )][1]]
)

```

gene_list_to_phases *Creation of phase table lists*

Description

Function that takes a list of genes associated with specific biological processes (GO, KEGG) or gene sets and a general phase table, and returns a list consisting of the phase tables specific to each set.

Usage

```
gene_list_to_phases(gene.list, phase.table)
```

Arguments

gene.list	Process or gene sets list (as the output of create_gene_list_go and create_gene_list_kegg)
phase.table	Table showing acrophases for the complete gene set under study, stated in hours. It must have two columns, named "names" and "phase"

Value

A list containing phase tables for each process or gene set

Examples

```
org_Otaurireduced_eg_db <- load_example_annot()
data("circa_table_genescarab")

total_phases_table_sd <- data.frame(
  names = rownames(circa_table_genescarab),
  phase = as.numeric(
    circa_table_genescarab[["sd.peak.time.hours"]]
  )
)
go.list.test <- create_gene_list_go(
  go_vector = "G0:0005515",
  org.package = org_Otaurireduced_eg_db, go_column = "GO",
  id_column = "GID"
)
gene_list_to_phases(go.list.test, total_phases_table_sd)
```

GeneSCARAB

GeneSCARAB: Gene Set Circular Analysis and Rhythms Ascertainment in Bioprocesses

Description

GeneSCARAB provides methods for analyzing the cohesion and clustering toward specific points in the cycle of biological sets (genes, metabolites, ...) for chronobiology studies following the proposed best practices in circular statistics.

Details**Main workflow:**

The typical workflow is:

1. Load element phases as a table and species annotation package.
2. Construct the phases list for the desired sets or processes.
3. Compute circular statistics and departure from uniformity for each set.
4. For non-uniform sets, compare against global phase distribution.
5. In the case of multiple conditions, determine differentially timed sets between conditions.
6. For multimodal sets, analyze the number of modes and cluster its elements.
7. Calculate contribution of each element to its set's statistics.
8. Plot results as circular boxplot, dotplot or histogram.

Main functions:

- `load_example_annot()` loads example annotation data.

- `create_gene_list_go()` creates a list of elements associated with each GO term based on a species annotation package.
- `create_gene_list_kegg()` create a list of elements associated with each KEGG pathway based on a species annotation package.
- `gene_list_to_phases()` takes a list of elements associated with specific sets and a complete phase table, and returns a list consisting of the individual phase tables of each set.
- `complete_circular_table()` calculates the table of statistical measures and deviations from circular uniformity of each set.
- `complete_circular_table_grouped()` calculates the table of statistical measures and deviations from circular uniformity of each set (for discrete phases).
- `test_against_gen_dist()` compares whether the distributions of specific sets significantly differ from the overall phase distribution of the entire dataset
- `test_two_dist()` compares whether the distributions of specific sets differ significantly between two different conditions or genotypes.
- `gene_contribution_to_set()` measures the contribution of each element of the set to the mean and rho of the entire set.
- `multimodal_analysis()` determines the most likely number of modes in the presence of f-fold symmetry (1, 2 or 3), cluster elements in f clusters and determine the circular statistic summary for each cluster
- `circular_boxplot()` creates circular boxplot.
- `circular_dotplot()` creates circular dotplot.
- `circular_histogram()` creates circular histogram.

See the package vignette for a complete tutorial.

Author(s)

Maintainer: Marcos Ramos-González <mramos5@us.es> ([ORCID](#))

Authors:

- Marcos Ramos-González <mramos5@us.es> ([ORCID](#))

Other contributors:

- Emma Serrano-Pérez <enma.serrano@ibvf.csic.es> ([ORCID](#)) [contributor]
- Víctor Ramos-González <vramos1@us.es> ([ORCID](#)) [contributor]
- Mercedes García-González <mgonza@us.es> ([ORCID](#)) [contributor]
- Francisco J. Romero-Campero <fran@us.es> ([ORCID](#)) [contributor]

See Also

Useful links:

- <https://github.com/ramosgonzmarc/GeneSCARAB>
- Report bugs at <https://github.com/ramosgonzmarc/GeneSCARAB>

load_example_annot	<i>Loading annotation for examples and vignettes</i>
--------------------	--

Description

Loading SQLite containing GO and KO annotation info for *Ostreococcus tauri* in Annotation-Forge's output format. Original genome and annotation package were download from <https://github.com/franromero-campero/AlgaeFUN> and KO information was generated via KAAS (https://www.genome.jp/kaas-bin/kaas_main).

Usage

```
load_example_annot()
```

Value

OrgDb object

Examples

```
org_Otaurireduced_eg_db <- load_example_annot()
```

multimodal_analysis	<i>Multimodality analysis</i>
---------------------	-------------------------------

Description

Function to determine the most likely number of modes in the presence of f-fold symmetry (1, 2 or 3), cluster genes in f clusters and determine the statistic summary for each cluster

Usage

```
multimodal_analysis(phase_table)
```

Arguments

phase_table A phase table, as the individual elements of the list derived from gene_list_to_phases

Value

A list containing the most likely mode number, the p-value for that number, the p-values corresponding to each mode number, the circular summary table for each cluster and the genes contained in each cluster

Examples

```

org_Otaurireduced_eg_db <- load_example_annot()
data("circa_table_genesArab")
total_phases_table_sd <- data.frame(
  names = rownames(circa_table_genesArab),
  phase = as.numeric(
    circa_table_genesArab[["sd.peak.time.hours"]]
  )
)
functional_data <- AnnotationDbi::select(org_Otaurireduced_eg_db,
  keys = AnnotationDbi::keys(org_Otaurireduced_eg_db, keytype = "GID"),
  columns = c("GID", "GO")
)
complete_gos <- unique(functional_data$GO)
complete_gos <- complete_gos[!is.na(complete_gos)]
set.seed(2345)
subset_gos <- complete_gos[sample(seq_len(
  length(complete_gos)
), 50, replace = FALSE)]
go.list.test <- create_gene_list_go(
  go_vector = subset_gos,
  org.package = org_Otaurireduced_eg_db,
  go_column = "GO", id_column = "GID"
)
phases.list.sd <- gene_list_to_phases(
  go.list.test,
  total_phases_table_sd
)
phases.list.sd.clean <- phases.list.sd[which(
  sapply(phases.list.sd, nrow) != 0
)]
go.circa.table.sd <- complete_circular_table(
  phase.list = phases.list.sd.clean,
  hr.on.large.sets.th = 200,
  force.hr.th = 0.04,
  rao.on.large.sets.th = 200,
  force.rao.th = 0.04,
  iter.rao = 999, iter.hr = 999
)
multi_table <- go.circa.table.sd[which(
  go.circa.table.sd$hr_p_value < 0.05 &
  go.circa.table.sd$n > 2
), ]
multi_go <- rownames(multi_table)[1]
phase_table_multi_go <- phases.list.sd.clean[[multi_go]]
multimodal_list <- multimodal_analysis(phase_table_multi_go)

```

Description

Function that compares whether the distributions of specific processes or gene sets differ significantly from the overall phase distribution of the entire study. Significance is calculated based on a MANOVA analysis as described in <https://doi.org/10.1186/s40462-022-00323-8>.

Usage

```
test_against_gen_dist(phase.list, total.phase.table)
```

Arguments

`phase.list` Phases list in the same format as the output of `gene_list_to_phases`
`total.phase.table` Table showing acrophases for the complete gene set under study, stated in hours

Value

A data.frame with p-value and adjusted p-value (null hypothesis: same distribution as the general one) for each process or gene set

Examples

```
org_Otaurireduced_eg_db <- load_example_annot()
data("circa_table_genescarab")
total_phases_table_sd <- data.frame(
  names = rownames(circa_table_genescarab),
  phase = as.numeric(
    circa_table_genescarab[["sd.peak.time.hours"]]
  )
)

functional_data <- AnnotationDbi::select(org_Otaurireduced_eg_db,
  keys = AnnotationDbi::keys(org_Otaurireduced_eg_db, keytype = "GID"),
  columns = c("GID", "GO")
)
complete_gos <- unique(functional_data$GO)
complete_gos <- complete_gos[!is.na(complete_gos)]

set.seed(2345)
subset_gos <- complete_gos[sample(
  seq_len(length(complete_gos)), 50,
  replace = FALSE
)]
go.list.test <- create_gene_list_go(
  go_vector = subset_gos,
  org.package = org_Otaurireduced_eg_db,
  go_column = "GO", id_column = "GID"
)
phases.list.sd <- gene_list_to_phases(
  go.list.test, total_phases_table_sd
)
```

```

phases.list.sd.clean <- phases.list.sd[which(
  sapply(phases.list.sd, nrow) != 0
)][seq_len(3)]
not_experimentally_distributed_gos <- test_against_gen_dist(
  phases.list.sd.clean,
  total_phases_table_sd
)

```

test_two_dist

*Comparison of circular distributions between two conditions***Description**

Function that compares whether the distributions of specific processes or gene sets differ significantly between two different conditions or genotypes. The two lists must have the same number of elements, although the number of genes associated with each process or gene set may differ. Significance is calculated based on a MANOVA analysis as described in <https://doi.org/10.1186/s40462-022-00323-8>.

Usage

```
test_two_dist(phase.list.1, phase.list.2)
```

Arguments

phase.list.1 Phases list for condition 1, in the same format as the output of gene_list_to_phases
 phase.list.2 Phases list for condition 2, in the same format as the output of gene_list_to_phases

Value

A data.frame with p-value and adjusted p-value (null hypothesis: same distribution in both conditions) for each process or gene set

Examples

```

org_Otaurireduced_eg_db <- load_example_annot()
data("circa_table_genesArab")
total_phases_table_sd <- data.frame(
  names = rownames(circa_table_genesArab),
  phase = as.numeric(
    circa_table_genesArab[["sd.peak.time.hours"]]
  )
)
total_phases_table_ld <- data.frame(
  names = rownames(circa_table_genesArab),
  phase = as.numeric(
    circa_table_genesArab[["ld.peak.time.hours"]]
  )
)

```

```

)

functional_data <- AnnotationDbi::select(org_Otaurireduced_eg_db,
  keys = AnnotationDbi::keys(org_Otaurireduced_eg_db, keytype = "GID"),
  columns = c("GID", "GO"))
)
complete_gos <- unique(functional_data$GO)
complete_gos <- complete_gos[!is.na(complete_gos)]

set.seed(2345)
subset_gos <- complete_gos[sample(
  seq_len(length(complete_gos)), 50,
  replace = FALSE
)]
go.list.test <- create_gene_list_go(
  go_vector = subset_gos,
  org.package = org_Otaurireduced_eg_db,
  go_column = "GO", id_column = "GID"
)
phases.list.sd <- gene_list_to_phases(
  go.list.test, total_phases_table_sd
)
phases.list.sd.clean <- phases.list.sd[which(
  sapply(phases.list.sd, nrow) != 0
)][seq_len(3)]

phases.list.ld <- gene_list_to_phases(
  go.list.test, total_phases_table_ld
)
phases.list.ld.clean <- phases.list.ld[which(
  sapply(phases.list.ld, nrow) != 0
)][names(phases.list.sd.clean)]

diff_distributed_gos <- test_two_dist(
  phases.list.ld.clean, phases.list.sd.clean
)

```

Index

- * **circadian**,
 - GeneSCARAB, 29
- * **datasets**
 - circa_table_genescarab, 17
- * **enrichment**,
 - GeneSCARAB, 29
- * **omics**,
 - GeneSCARAB, 29
- * **phases**
 - GeneSCARAB, 29
- * **rhythms**,
 - GeneSCARAB, 29
- .HermansRasson2PGroupedRad, 6
- .HermansRasson2T, 6
- .KuiperPGroupedRad, 10
- .RaoPGroupedRad, 12
- .RaoTestUngroupedRad, 13
- .RaoTestValue, 13
- .create_circular_table, 3
- .create_circular_table_grouped, 4
- .get_ancestor_for_gene_lists, 5
- .helper_init_plot, 5
- .input_complete_circular_table, 7
- .input_create_circular_table, 7
- .input_create_gene_list_go, 8
- .input_create_gene_list_kegg, 8
- .input_gene_contribution, 9
- .legend_builder, 10
- .legend_builder_boxplot, 11
- .multimodal_modes_test, 11
- .pathways_from_kos, 12
- .table_builder, 14
- .track_box_box_cut, 14
- .track_box_no_cut, 15
- .track_box_segments_cut, 16
- .track_dot, 16
-
- circa_table_genescarab, 17
- circular_boxplot, 17
- circular_boxplot(), 30
-
- circular_dotplot, 18
- circular_dotplot(), 30
- circular_histogram, 20
- circular_histogram(), 30
- complete_circular_table, 21
- complete_circular_table(), 30
- complete_circular_table_grouped, 23
- complete_circular_table_grouped(), 30
- create_gene_list_go, 25
- create_gene_list_go(), 30
- create_gene_list_kegg, 26
- create_gene_list_kegg(), 30
-
- gene_contribution_to_set, 27
- gene_contribution_to_set(), 30
- gene_list_to_phases, 28
- gene_list_to_phases(), 30
- GeneSCARAB, 29
- GeneSCARAB-package (GeneSCARAB), 29
-
- load_example_annot, 31
- load_example_annot(), 29
-
- multimodal_analysis, 31
- multimodal_analysis(), 30
-
- test_against_gen_dist, 32
- test_against_gen_dist(), 30
- test_two_dist, 34
- test_two_dist(), 30