Package: greatR (via r-universe)

March 19, 2025

Title Gene Registration from Expression and Time-Courses in R

Version 2.0.0

Description A tool for registering (aligning) gene expression profiles between reference and query data.

License GPL (>= 3)

URL https://ruthkr.github.io/greatR/, https://github.com/ruthkr/greatR/

BugReports https://github.com/ruthkr/greatR/issues/

Depends R (>= 4.1.0)

Imports cli, data.table, furrr, future, ggplot2, neldermead, optimization, patchwork, scales, stats

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.3.1

Config/testthat/edition 3

NeedsCompilation no

Author Ruth Kristianingsih [aut, cre] (<https://orcid.org/0000-0003-1873-6203>)

Maintainer Ruth Kristianingsih <ruth.kristianingsih30@gmail.com>

Date/Publication 2024-04-09 22:40:07 UTC

Additional_repositories https://cranhaven.r-universe.dev

Repository https://cranhaven.r-universe.dev

RemoteUrl https://github.com/cranhaven/cranhaven.r-universe.dev

RemoteRef package/greatR

RemoteSha 0ce026246291e2a35042b8daf808297403889e0c

RemoteSubdir greatR

Contents

calculate_distance 2
get_approximate_stretch 3
plot 3
register 5
summary 7

`calculate_distance` *Calculate distance between sample data before and after registration*

Description

`calculate_distance()` is a function that allows users to calculate pairwise distances between samples from different time points to investigate the similarity of progression before or after registration.

Usage

```
calculate_distance(results, type = c("registered", "all"), genes_list = NULL)
```

Arguments

`results`	Result of registration process using `register()`.
`type`	Whether to calculate distance considering only "registered" genes (default) or "all" genes.
`genes_list`	Optional vector indicating the `gene_id` values to be considered.

Value

This function returns a `dist_greatR` object containing two data frames:

`registered`	pairwise distance between scaled reference and query expressions using registered time points.
`original`	pairwise distance between scaled reference and query expressions using original time points.

get_approximate_stretch — *Get approximate stretch factor*

Description

`get_approximate_stretch()` is a function to get a stretch factor estimation given input data. This function will take the time point ranges of both reference and query data and compare them to estimate the stretch factor.

Usage

```
get_approximate_stretch(data, reference = "ref", query = "query")
```

Arguments

`data`	Input data frame, either containing all replicates of gene expression or not.
`reference`	Accession name of reference data.
`query`	Accession name of query data.

Value

This function returns an estimation of a stretch factor for registering the data.

plot — *Visualise registration results*

Description

Visualise registration results

Usage

```
## S3 method for class 'res_greatR'
plot(
  x,
  type = c("result", "original"),
  genes_list = NULL,
  show_rep_mean = FALSE,
  ncol = NULL,
  title = NULL,
  ...
)

## S3 method for class 'dist_greatR'
```

```
plot(
  x,
  type = c("result", "original"),
  match_timepoints = TRUE,
  title = NULL,
  ...
)

## S3 method for class 'summary.res_greatR'
plot(
  x,
  type = c("all", "registered"),
  type_dist = c("histogram", "density"),
  genes_list = NULL,
  bins = 30,
  alpha = NA,
  scatterplot_size = c(4, 3),
  title = NULL,
  ...
)
```

Arguments

`x`	Input object. • For `plot.res_greatR()`: registration results, output of the `register()` registration process. • For `plot.summary.res_greatR()`: registration results summary, output of `summary()`. • For `plot.dist_greatR()`: pairwise distances between reference and query time points, output of `calculate_distance()`.
`type`	Type of plot. • For both `plot.res_greatR()` and `plot.dist_greatR()`: whether to use registration "result" (default) or "original" time points. • For `plot.summary.res_greatR()`: whether to show "all" genes (default) or only "registered" ones.
`genes_list`	Optional vector indicating the `gene_id` values to be plotted.
`show_rep_mean`	Whether to show `replicate` mean values.
`ncol`	Number of columns in the plot grid. By default this is calculated automatically.
`title`	Optional plot title.
`...`	Arguments to be passed to methods (ignored).
`match_timepoints`	If `TRUE`, will match query time points to reference time points.
`type_dist`	Type of marginal distribution. Can be either "histogram" (default), or "density".
`bins`	Number of bins to use when `type_dist` = "histogram". By default, 30.
`alpha`	Optional opacity of the points in the scatterplot.

`scatterplot_size`	Vector `c(width, height)` specifying the ratio of width and height of the scatterplot with respect to stretch and shift distribution plots.

Value

- For `plot.res_greatR()`: plot of genes of interest after registration process (`type = "result"`) or showing original time points (`type = "original"`).
- For `plot.dist_greatR()`: distance heatmap of gene expression profiles over time between reference and query.
- For `plot.summary.res_greatR()`: TODO.

`register` *Register or synchronize different expression profiles*

Description

`register()` is a function to register expression profiles a user wishes to compare.

Usage

```
register(
  input,
  stretches = NA,
  shifts = NA,
  reference,
  query,
  scaling_method = c("none", "z-score", "min-max"),
  overlapping_percent = 50,
  use_optimisation = TRUE,
  optimisation_method = c("lbfgsb", "nm", "sa"),
  optimisation_config = NULL,
  exp_sd = NA,
  num_cores = NA
)
```

Arguments

`input`	Input data frame containing all replicates of gene expression in each genotype at each time point.
`stretches`	Candidate registration stretch factors to apply to query data, only required if `use_optimisation = FALSE`.
`shifts`	Candidate registration shift values to apply to query data, only required if `use_optimisation = FALSE`.
`reference`	Accession name of reference data.
`query`	Accession name of query data.

`scaling_method`	Scaling method applied to data prior to registration process. Either none (default), `z-score`, or `min-max`.
`overlapping_percent`	Minimum percentage of overlapping time point range of the reference data. Shifts will be only considered if it leaves at least this percentage of overlapping time point range after applying the registration.
`use_optimisation`	Whether to optimise registration parameters. By default, `TRUE`.
`optimisation_method`	Optimisation method to use. Either `"lbfgsb"` for L-BFGS-B (default), `"nm"` for Nelder-Mead, or `"sa"` for Simulated Annealing.
`optimisation_config`	Optional list with arguments to override the default optimisation configuration.
`exp_sd`	Optional experimental standard deviation on the expression replicates.
`num_cores`	Number of cores to use if the user wants to register genes asynchronously (in parallel) in the background on the same machine. By default, `NA`, the registration will be run without parallelisation.

Value

This function returns a `res_greatR` object containing:

`data`	a table containing the scaled input data and an additional `timepoint_reg` column after applying registration parameters to the query data.
`model_comparison`	a table comparing the optimal registration function for each gene (based on `all_shifts_df` scores) to model with no registration applied.
`fun_args`	a list of arguments used when calling the function.

Examples

```
## Not run:
# Load a data frame from the sample data
data_path <- system.file("extdata/brapa_arabidopsis_data.csv", package = "greatR")
all_data <- utils::read.csv(data_path)

# Running the registration
registration_results <- register(
  input = all_data,
  reference = "Ro18",
  query = "Col0"
)

## End(Not run)
```

`summary` *Summarise registration results*

Description

Summarise registration results

Usage

```
## S3 method for class 'res_greatR'
summary(object, ...)
```

Arguments

`object`	Registration results, output of the `register()` registration process.
`...`	Arguments to be passed to methods (ignored).

Value

This function returns a list containing:

`summary`	table containing the summary of the registration results.
`registered_genes`	vector of gene accessions which were successfully registered.
`non_registered_genes`	vector of non-registered gene accessions.
`reg_params`	table containing distribution of registration parameters.