Package 'progeny'

November 12, 2026

Title Pathway RespOnsive GENes for activity inference from gene expression

Version 1.34.0

Description PROGENy is resource that leverages a large compendium of publicly available signaling perturbation experiments to yield a common core of pathway responsive genes for human and mouse. These, coupled with any statistical method, can be used to infer pathway activities from bulk or single-cell transcriptomics.

URL https://github.com/saezlab/progeny

BugReports https://github.com/saezlab/progeny/issues

Depends R (>= 3.6.0)

Imports Biobase, stats, dplyr, tidyr, ggplot2, ggrepel, gridExtra, decoupleR, reshape2

biocViews SystemsBiology, GeneExpression, FunctionalPrediction, GeneRegulation

License Apache License (== 2.0) | file LICENSE

LazyData true

LazyDataCompression bzip2

Encoding UTF-8

Suggests airway, biomaRt, BiocFileCache, broom, Seurat, SingleCellExperiment, DESeq2, BiocStyle, knitr, readr, readxl, pheatmap, tibble, rmarkdown, testthat (>= 2.1.0)

VignetteBuilder knitr

RoxygenNote 7.1.2

git_url https://git.bioconductor.org/packages/progeny

git_branch RELEASE_3_23

git_last_commit cca15a9

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Author Michael Schubert [aut],
Alberto Valdeolivas [ctb] (ORCID:
<https://orcid.org/0000-0001-5482-9023>),
Christian H. Holland [ctb] (ORCID:
<https://orcid.org/0000-0002-3060-5786>),
Igor Bulanov [ctb],
Aurélien Dugourd [cre, ctb]

Maintainer Aurélien Dugourd <aurelien.dugourd@bioquant.uni-heidelberg.de>

Contents

getModel 2
model_human_full 3
model_mouse_full 3
progeny 4
progenyPerm 6
progenyScatter 7
saveProgenyPlots 8
vignette_data 9

Index **11**

`getModel` *Returns the Progeny Model*

Description

This function is designed for getting a model matrix with top significant genes for each pathway

Usage

```
getModel(organism = "Human", top = 100, decoupleR = F)
```

Arguments

`organism`	"Human" or "Mouse" taken from the main function's argument. Default to "Human"
`top`	Desired top number of genes for each pathway according to their significance(p.value). Default to 100
`decoupleR`	if TRUE, the model matrix is goign to be generated with a format makes it directlz compatible with the decoupleR package

Value

This function returns model matrix according to the top n significant

Examples

```
#getting a model matrix according to the desired top n significant
model <- getModel("Human", top=100)
```

`model_human_full` *The full human linear model underlying PROGENy*

Description

HGNC gene symbols in rows, pathways in columns. Pathway activity inference works by matrix multiplication of gene expression with the model.

Format

The full human model contains 22479 genes, associated pathways, weight and the p-value.

gene gene names in HGNC symbols

pathway names of PROGENy pathways

weight z-scores for a given gene

p.value significance of gene in pathway

Source

https://www.nature.com/articles/s41467-017-02391-6

Examples

```
get("model_human_full", envir = .GlobalEnv)
```

`model_mouse_full` *The full mouse linear model underlying PROGENy*

Description

MGI gene symbols in rows, pathways in columns. Pathway activity inference works by matrix multiplication of gene expression with the model.

Format

The full mouse model contains 17426 genes, associated pathways, weight and the p-value.

gene gene names in HGNC symbols

pathway names of PROGENy pathways

weight z-scores for a given gene

p.value significance of gene in a pathway

Source

https://www.ncbi.nlm.nih.gov/pubmed/31525460

Examples

```
get("model_mouse_full", envir = .GlobalEnv)
```

---

progeny *Calculate PROGENy pathway scores from gene expression*

Description

This function uses the linear model of pathway-responsive genes underlying the PROGENy method. It transforms a gene expression matrix with HGNC/MGI gene symbols in rows and sample names in columns into a pathway score matrix with samples in rows and pathways in columns.

Usage

```
progeny(
  expr,
  scale = TRUE,
  organism = "Human",
  top = 100,
  perm = 1,
  verbose = FALSE,
  z_scores = FALSE,
  get_nulldist = FALSE,
  assay_name = "RNA",
  return_assay = FALSE,
  ...
)
```

Arguments

expr	A gene expression object with HGNC/MGI symbols in rows and samples in columns. In order to run PROGENy in single-cell RNAseq data, it also accepts Seurat and SingleCellExperiment object, taking the normalized counts for the computation.
scale	A logical value indicating whether to scale the scores of each pathway to have a mean of zero and a standard deviation of one. It does not apply if we use permutations.
organism	The model organism - "Human" or "Mouse"
top	The top n genes for generating the model matrix according to significance (p-value)

perm	An interger detailing the number of permutations. No permutations by default (1). When Permutations larger than 1, we compute progeny pathway scores and assesses their significance using a gene sampling-based permutation strategy, for a series of experimental samples/contrasts.
verbose	A logical value indicating whether to display a message about the number of genes used per pathway to compute progeny scores (i.e. number of genes present in the progeny model and in the expression dataset)
z_scores	Only applies if the number of permutations is greater than 1. A logical value. TRUE: the z-scores will be returned for the pathway activity estimations. FALSE: the function returns a normalized z-score value between -1 and 1.
get_nulldist	Only applies if the number of permutations is greater than 1. A logical value. TRUE: the null distributions generated to assess the signifance of the pathways scores is also returned.
assay_name	Only applies if the input is a Seurat object. It selects the name of the assay on which Progeny will be run. Default to: RNA, i.e. normalized expression values.
return_assay	Only applies if the input is a Seurat object. A logical value indicating whether to return progeny results as a new assay called Progeny in the Seurat object used as input. Default to FALSE.
...	Additional arguments to be passed to the functions.

Details

The publication of the method is available at: https://www.nature.com/articles/s41467-017-02391-6

The supplied expression object has to contain HGNC/MGI symbols in rows. This will, in most cases (and how we originally used it), be either normalized gene expression of a microarray experiment or log-transformed (and possible variance-stabilized) counts from an RNA-seq experiment.

The human and mouse model matrices consists of 14 pathways and large set of genes with an associated p-value (p-value per gene and pathway) that accounts for the importance of each gene on each pathway upon perturbation. Its coefficients are non-zero if the gene-pathway pair corresponds to the top N genes (100 by default) that were up-regulated upon stimulation of the pathway in a wide range of experiments. The value corresponds to the fitted z-score across experiments in our model fit. Only rows with at least one non-zero coefficient were included, as the rest is not used to infer pathway activity.

Value

A matrix with samples in rows and pathways in columns. In case we run the method with permutations and the option get_nulldist to TRUE, we will get a list with two elements. The first element is the matrix with the pathway activity as before. The second elements is the null distributions that we generate to assess the signifance of the pathways scores.

See Also

progenyPerm

Examples

```
# use example gene expression matrix here, this is just for illustration
gene_expression <- as.matrix(read.csv(system.file("extdata",
"human_input.csv", package = "progeny"), row.names = 1))

# calculate pathway activities
pathways <- progeny(gene_expression, scale=TRUE,
    organism="Human", top = 100, perm = 1)
```

`progenyPerm` *Compute progeny pathway scores and assesses significance based on permutations*

Description

Compute progeny pathway scores and assesses significance based on permutations

Usage

```
progenyPerm(
  df,
  weight_matrix,
  k = 10000,
  z_scores = TRUE,
  get_nulldist = FALSE
)
```

Arguments

`df`	A data.frame of n*m+1 dimension, where n is the number of omic features to be considered and m is the number of samples/contrasts. The first column should be the identifiers of the omic features. These identifiers must be coherent with the identifiers of the weight matrix.
`weight_matrix`	A progeny coefficient matrix. the first column should be the identifiers of the omic features and should be coherent with the identifiers provided in df.
`k`	The number of permutations to be performed to generate the null-distribution used to estimate the significance of progeny scores. The default value is 10000.
`z_scores`	if true, the z-scores will be returned for the pathway activity estimations. Else, the function returns a normalized z-score value between -1 and 1.
`get_nulldist`	if true, the null score distribution used for normalization will be returned along with the actual normalized score data frame.

Value

This function returns a list of two elements. The first element is a data frame of p*m+1 dimensions, where p is the number of progeny pathways, and m is the number of samples/contrasts. Each cell represents the significance of a progeny pathway score for one sample/contrast. The significance ranges between -1 and 1. The significance is equal to x*2-1, x being the quantile of the progeny pathway score with respect to the null distribution. Thus, this significance can be interpreted as the equivalent of 1-p.value two-sided test over an empirical distribution) with the sign indicating the direction of the regulation. The second element is the null distribution list (a null distribution is generated for each sample/contrast).

Examples

```
# use example gene expression matrix
gene_expression <- as.matrix(read.csv(system.file("extdata",
"human_input.csv", package = "progeny"), row.names = 1))

# calculate pathway activities
progeny(gene_expression, scale=TRUE, organism="Human", top=100, perm=10000)
```

---

progenyScatter *Calculate Progeny Scores with Permutations*

Description

This function generate a series of scatter plot with marginal distribution (in the form of an arrangeGrob object), for each progeny pathway and sample/contrast. Each scatter plot has progeny weights as x-axis and the gene level stat used to compute progeny score as the y-axis. The marginal distribution of the gene level stats is displayed on the right of the plot to give visual support of the significance of each gene contributing to the progeny pathway score. The green and red colors represent the positive and negative contribution of genes to the progeny pathway, respectively. For each gene contribution, 4 cases are possible, as the combinations of the sign of the gene level stat and the sign of the gene level weight. Positive weight will lead to a positive(green)/negative(red) gene contribution if the gene level stat is positive/negative. Negative weight will lead to a negative(red)/positive(green) gene contribution if the gene level stat is positive/negative.

Usage

```
progenyScatter(
  df,
  weight_matrix,
  dfID = 1,
  weightID = 1,
  statName = "gene stats",
  verbose = FALSE
)
```

Arguments

df	an n*m data frame, where n is the number of omic features (genes). m isn't really important, as long as at least one column corresponds to a sample or contrast statistic. One of the columns should correspond to the gene symbols.
weight_matrix	A progeny coefficient matrix. the first column should be the identifiers of the omic features, and should be coherent with the identifiers provided in df.
dfID	an integer corresponding to the column number of the gene identifiers of df.
weightID	an integer corresponding to the column number of the gene identifiers of the weight matrix.
statName	The name of the stat used, to be displayed on the plot
verbose	Logical indicating whether we want to have the messages indicating the different computed weights.

Value

The function returns a list of list of arrangeGrob objects. The first level list elements correspond to samples/contrasts. The second level correspond to pathways. The plots can be saved in a pdf format using the saveProgenyPlots function.

Examples

```
# use example gene expression matrix

gene_expression <- read.csv(system.file("extdata",
"human_input.csv", package = "progeny"))

# getting a model matrix with 100 top significant genes and converting to df
weight_matrix <- getModel("Human", top=100)
weight_matrix <- data.frame(names = row.names(weight_matrix),
  row.names = NULL, weight_matrix)

#use progenyScatter function
plots <- progenyScatter(gene_expression, weight_matrix)
```

---

saveProgenyPlots *Function to save Progeny plots*

Description

This function is designed to save the plots (in pdf format) of a nested (2 level) list of arrangeGrob objects, such as the one returned by the progenyScatter function.

Usage

```
saveProgenyPlots(plots, contrast_names, dirpath)
```

Arguments

plots	a list of list of arrangeGrob object (such as the one returned by the progenyScatter function.).The first level list elements correspond to samples/contrasts. The second level corresponds to pathways. The plots can be saved in a pdf format using the saveProgenyPlots function.
contrast_names	a vector of the same length as the first level of the plot list corresponding to the names of each sample/contrast
dirpath	the path to the directory where the plots should be saved

Value

This function produces the pdf files of plots taken from the progenyScatter function

Examples

```
#create plots using progneyScatter function
gene_expression <- read.csv(system.file("extdata",
"human_input.csv", package = "progeny"))

# getting a weight_matrix
weight_matrix <- getModel("Human", top=100)
weight_matrix <- data.frame(names = row.names(weight_matrix),
  row.names = NULL, weight_matrix)
plots <- progenyScatter(gene_expression, weight_matrix)

#create a list with contrast names
contrast_names <- names(gene_expression[2:ncol(gene_expression)])

#assign a path to store your plots
dirpath <- "./progeny_plots/"

# save it
# saveProgenyPlots(plots, contrast_names, dirpath)
```

---

vignette_data *The RNA data used in the progeny vignette*

Description

List with three elements: the gene counts, the experimental design and the result of limma differential analysis

Format

List with three elements: the gene counts, the experimental design and the result of limma differential analysis

counts gene counts

design experiemental design

limma_ttop differential analysis result using limma

Source

https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE119931

Examples

```
data("vignette_data")
```

Index

∗ **datasets**
model_human_full, 3
model_mouse_full, 3
vignette_data, 9

getModel, 2

model_human_full, 3
model_mouse_full, 3

progeny, 4
progenyPerm, *5*, 6
progenyScatter, 7

saveProgenyPlots, 8

vignette_data, 9