

# Package ‘perturbatr’

March 30, 2021

**Type** Package

**Title** Statistical Analysis of High-Throughput Genetic Perturbation Screens

**Version** 1.10.0

**Date** 2018-03-19

**Maintainer** Simon Dirmeier <simon.dirmeier@web.de>

**URL** <https://github.com/cbg-ethz/perturbatr>

**BugReports** <https://github.com/cbg-ethz/perturbatr/issues>

**Description** perturbatr does stage-wise analysis of large-scale genetic perturbation screens for integrated data sets consisting of multiple screens. For multiple integrated perturbation screens a hierarchical model that considers the variance between different biological conditions is fitted. The resulting list of gene effects is then further extended using a network propagation algorithm to correct for false negatives.

**License** GPL-3

**LazyData** TRUE

**RoxygenNote** 6.0.1

**Depends** R (>= 3.5), methods, stats

**Imports** dplyr, ggplot2, tidyr, assertthat, lme4, splines, igraph, foreach, parallel, doParallel, diffusr, lazyeval, tibble, grid, utils, graphics, scales, magrittr, formula.tools, rlang

**biocViews** ImmunoOncology, Regression, CellBasedAssays, Network

**Suggests** testthat, lintr, knitr, rmarkdown, BiocStyle

**VignetteBuilder** knitr

**Collate** 'methods\_getters.R' 'util\_enums.R' 'class\_analysed.R' 'class\_data.R' 'data.R' 'inference\_diffuse.R' 'inference\_diffuse\_mrwr.R' 'inference\_lmm\_locfdr.R' 'inference\_lmm\_fdr.R' 'inference\_lmm.R' 'inference\_lmm\_model\_data.R' 'methods\_combine.R' 'methods\_filter.R' 'methods\_show.R' 'perturbatr-package.R' 'plot\_data.R' 'plot\_diffusion.R' 'plot\_hm.R' 'util\_as.R' 'util\_effect\_matrix.R' 'util\_functions.R' 'util\_graph.R' 'util\_sampler.R'

**git\_url** <https://git.bioconductor.org/packages/perturbatr>

**git\_branch** RELEASE\_3\_12

**git\_last\_commit** 267b2b6

**git\_last\_commit\_date** 2020-10-27

**Date/Publication** 2021-03-29

**Author** Simon Dirmeier [aut, cre]

## R topics documented:

perturbatr-package . . . . .	2
bootstrap . . . . .	3
dataSet . . . . .	4
diffuse . . . . .	4
filter . . . . .	5
geneEffects . . . . .	6
graph . . . . .	7
hm . . . . .	7
HMANalysedPerturbationData-class . . . . .	8
inference . . . . .	9
isBootstrapped . . . . .	9
modelFit . . . . .	10
nestedGeneEffects . . . . .	10
NetworkAnalysedPerturbationData-class . . . . .	11
params . . . . .	11
PerturbationData-class . . . . .	12
plot.NetworkAnalysedPerturbationData . . . . .	12
plot.PerturbationData . . . . .	13
rbind.PerturbationData . . . . .	14
rnaiscreen . . . . .	14
setModelData . . . . .	16
<b>Index</b>	<b>17</b>

---

perturbatr-package     *perturbatr*

---

### Description

Statistical analysis of high-throughput genetic perturbation screens

### Author(s)

Simon Dirmeier | Computational Biology Group, ETH ZURICH | <simon.dirmeier@web.de>

## References

de Wilde, Adriaan H., et al. (2015), A kinome-wide small interfering RNA screen identifies proviral and antiviral host factors in severe acute respiratory syndrome coronavirus replication, including double-stranded RNA-activated protein kinase and early secretory pathway proteins.

*Journal of virology*

Reiss, Simon, et al. (2011), Recruitment and activation of a lipid kinase by hepatitis C virus NS5A is essential for integrity of the membranous replication compartment.

*Cell Host & Microbe*

---

bootstrap

*Create a bootstrap sample from a data-set*

---

## Description

Create a bootstrap sample from a data-set

## Usage

```
bootstrap(obj, ...)
```

## Arguments

obj	the object which data should be bootstrapped
...	groups on which you be bootstrapped. If you want to create a normal bootstrap sample, you would ignore this argument. If you want to separate your data into groups and bootstrap from every group, you would give the unquoted name of the columns in your obj to group on. For instance, if you provide 'gene' as an argument, then your data set would be grouped into separate 'gene' groups and bootstrapping would be conducted on every of those groups. Afterwards genes are aggregated

## Value

returns an object with bootstrapped data

## Examples

```
data(rnaiscreen)
bootstrap(rnaiscreen)
bootstrap(rnaiscreen, Condition, Perturbation)
```

---

dataSet	<i>Getter for a data set</i>
---------	------------------------------

---

### Description

Returns the data set that underlies an S4 wrapper class as tibble.

### Usage

```
dataSet(obj)

## S4 method for signature 'AbstractAnalysedPerturbationData'
dataSet(obj)

## S4 method for signature 'PerturbationData'
dataSet(obj)
```

### Arguments

obj                    the object for which you want to extract the underlying data

### Value

returns a tibble.

### Examples

```
data(rnaiscreen)
dataSet(rnaiscreen)
```

---

diffuse	<i>Network diffusion</i>
---------	--------------------------

---

### Description

Propagate the estimated gene effects from a previous analysis over a network using network diffusion. First the estimated effects are normalized and mapped to a given genetic network, for instance a PPI or co-expression network. Then the normalized effects are propagated across the edges of the network using a Markov random walk with restarts. By that the initial ranking of genes (as given by their absolute effect sizes) is re-evaluated and the genes are reordered. Thus network diffusion potentially reduced false negative hits.

### Usage

```
diffuse(obj, graph = NULL, r = 0.5, delete.nodes.on.degree = 0,
        do.bootstrap = FALSE, take.largest.component = TRUE,
        correct.for.hubs = TRUE)

## S4 method for signature 'HMANalysedPerturbationData'
diffuse(obj, graph = NULL, r = 0.5,
        delete.nodes.on.degree = 0, do.bootstrap = FALSE,
        take.largest.component = TRUE, correct.for.hubs = TRUE)
```

**Arguments**

<code>obj</code>	HMANalysedPerturbationData object
<code>graph</code>	a <code>data.frame</code> or <code>tibble</code> with three columns representing a symbolic edge list. The first two columns contain node ids. The third column has to be called <i>weight</i> and is the <i>non-negative</i> weight of the edge between the two nodes.
<code>r</code>	restart probability of the random walk
<code>delete.nodes.on.degree</code>	delete nodes from the graph with a degree of less or equal than <code>delete.nodes.on.degree</code>
<code>do.bootstrap</code>	run a diffusion on every bootstrap sample in case bootstrap samples are available
<code>take.largest.component</code>	if <code>true</code> takes only the largest connected component of the graph and discards all nodes that are not in the largest component. If <code>false</code> takes the complete graph.
<code>correct.for.hubs</code>	if <code>true</code> corrects for the fact that the stationary distribution of the random walk is biased towards hubs.

**Value**

returns a `NetworkAnalysedPerturbationData` object

**Examples**

```
data(rnaiscreen)
hm.fit <- hm(rnaiscreen)

graph <- readRDS(system.file(
  "extdata", "graph_small.rds", package = "perturbatr"))
res <- diffuse(hm.fit, graph=graph, r=1)
```

---

`filter`

*Filter the rows of a perturbation data set*

---

**Description**

Takes a perturbation data set and filters the rows by some criterion. The filtered object will have the same type as the previous object.

**Usage**

```
filter(obj, ...)
```

**Arguments**

<code>obj</code>	the object to be filtered
<code>...</code>	variable number of logical predicates in terms of the column names in <code>obj</code> . Multiple predicates will be combined with a logical 'and'. Rows where all conditions are met will be kept. The column names do not need to be quoted. Wraps around <code>dplyr::filter</code> .

**Value**

returns an object of the same type filtered by some criteria

**Examples**

```
data(rnaiscreen)
flt.dat <- filter(rnaiscreen, Condition=="V1")
flt.dat <- filter(rnaiscreen, Condition=="V1", RowIdx==1)
```

---

geneEffects

*Getter for the complete list of genes and their effect sizes*

---

**Description**

Returns a tibble containing the list of genes that have been used in the study along with their estimated effect sizes.

**Usage**

```
geneEffects(obj)

## S4 method for signature 'AbstractAnalysedPerturbationData'
geneEffects(obj)
```

**Arguments**

obj                    the object for which you want to extract the underlying gene

**Value**

returns a list.

**Examples**

```
data(rnaiscreen)
ft <- hm(rnaiscreen)
geneEffects(ft)
```

---

graph	<i>Getter for graph used for network diffusion</i>
-------	--

---

**Description**

Returns the graph that has been used for analysis using network diffusion.

**Usage**

```
graph(obj)
```

```
## S4 method for signature 'NetworkAnalysedPerturbationData'
graph(obj)
```

**Arguments**

obj                    the object for which you want to extract the underlying graph

**Value**

returns a igraph object

**Examples**

```
data(rnaiscreen)
ft <- hm(rnaiscreen)

gr <- readRDS(system.file(
  "extdata", "graph_small.rds", package = "perturbatr"))
diffu <- diffuse(ft, gr, r=1)

graph(diffu)
```

---

hm	<i>Jointly analyse multiple genetic perturbation screens using a hierarchical model</i>
----	---

---

**Description**

Analyse multiple different genetic perturbation screens at once using a hierarchical model. The model estimates general relative effect sizes for genes across all experiments. This could for instance be a pan-pathogenic host factor, i.e. a gene that decisively impacts the life-cycle of multiple pathogens.

**Usage**

```
hm(obj, formula = Readout ~ Condition + (1 | GeneSymbol) + (1 |
  Condition:GeneSymbol), drop = TRUE, weights = 1, bootstrap.cnt = 0)
```

```
## S4 method for signature 'PerturbationData'
hm(obj, formula = Readout ~ Condition + (1 |
  GeneSymbol) + (1 | Condition:GeneSymbol), drop = TRUE, weights = 1,
  bootstrap.cnt = 0)
```

**Arguments**

obj	an PerturbationData object
formula	a formula object that is used to model the readout of your data set. If no formula is provided, the formula 'Readout ~ Condition + (1 GeneSymbol) + (1 Condition:GeneSymbol)' is used. For other data sets with more variables, it might makes sense to use other fixed and random effects
drop	boolean if genes that are not found in every Condition should be dropped
weights	a numeric vector used as weights for the single perturbations
bootstrap.cnt	the number of bootstrap runs you want to do in order to estimate a significance level for the gene effects

**Value**

returns a HMANalysedPerturbationData object

**Examples**

```
data(rnaiscreen)
res <- hm(rnaiscreen)
```

---

HMANalysedPerturbationData-class

*Data wrapper for analysed perturbation data using a hierarchical model*

---

**Description**

Class HMANalysedPerturbationData is a wrapper for various objects of an analysis of a perturbation experiment done using a hierarchical model. Class HMANalysedPerturbationData exposes getters for its members of the same name, but no setters, because the data should be treated as constant once set. Objects of class HMANalysedPerturbationData do not need to be constructed manually but are returned from calling `hm` (see the examples).

**Slots**

nestedGeneEffects the estimated effect sizes for genes on a viral level  
 modelFit the fitted model

**Examples**

```
data(rnaiscreen)
res <- hm(rnaiscreen)
class(res)
```



---

inference	<i>Getter for inference used for analysis of perturbation data</i>
-----------	--

---

**Description**

Returns the inference used in the analysis of a perturbation analysis. This can for instance be a standard t-test or a hierarchical model.

**Usage**

```
inference(obj)
```

```
## S4 method for signature 'AbstractAnalysedPerturbationData'  
inference(obj)
```

**Arguments**

obj                    the object for which you want to extract the underlying inference

**Value**

returns a character.

**Examples**

```
data(rnaiscreen)  
ft <- hm(rnaiscreen)  
inference(ft)
```

---

isBootstrapped	<i>Getter for boolean if bootstrapping was used</i>
----------------	---

---

**Description**

Returns a boolean if for the analysed object bootstrapping was used to create confidence intervals.

**Usage**

```
isBootstrapped(obj)
```

```
## S4 method for signature 'AbstractAnalysedPerturbationData'  
isBootstrapped(obj)
```

**Arguments**

obj                    the object for which you want to extract the boolean

**Value**

returns a boolean.

**Examples**

```
data(rnaiscreen)
ft <- hm(rnaiscreen)
isBootstrapped(ft)
```

---

modelFit	<i>Getter for model fit</i>
----------	-----------------------------

---

**Description**

Returns the mixed effects model fit from the analysis using a hierarchical model.

**Usage**

```
modelFit(obj)

## S4 method for signature 'HMANalysedPerturbationData'
modelFit(obj)
```

**Arguments**

obj                    the object for which you want to extract the underlying fit

**Value**

returns a list

**Examples**

```
data(rnaiscreen)
hm.fit <- hm(rnaiscreen)
ft <- modelFit(hm.fit)
```

---

nestedGeneEffects	<i>Getter for the completed list of nested gene effects</i>
-------------------	---

---

**Description**

Returns a tibble containing the list of nested genes effects, i.e. a table of nested cluster effects.

**Usage**

```
nestedGeneEffects(obj)

## S4 method for signature 'HMANalysedPerturbationData'
nestedGeneEffects(obj)
```

**Arguments**

obj                    the object for which you want to extract the underlying effects

**Value**

returns a tibble.

**Examples**

```
data(rnaiscreen)
ft <- hm(rnaiscreen)
nestedGeneEffects(ft)
```

---

NetworkAnalysedPerturbationData-class

*Data wrapper for analysed perturbation data using network diffusion*

---

**Description**

Class NetworkAnalysedPerturbationData is a wrapper for various objects of an analysis of a perturbation experiment done using network diffusion. Class NetworkAnalysedPerturbationData exposes getters for its members of the same name, but no setters, because the data should be treated as constant once set. Objects of class NetworkAnalysedPerturbationData do not need to be constructed manually but are returned from calling `diffuse` (see the examples).

**Slots**

`initialModel` the model that was provided for analysis  
`graph` an igraph object that served for the diffusion process

**Examples**

```
data(rnaiscreen)
hm.fit <- hm(rnaiscreen)

graph <- readRDS(system.file(
  "extdata", "graph_small.rds", package = "perturbatr"))
res <- diffuse(hm.fit, graph=graph, r=1)
```

---

`params`

*Getter for parameters used for analysis of perturbation data*

---

**Description**

Returns the parameters used in the analysis of a perturbation analysis. Parameters are for instance the significance level, the effect size or restart probability of a random walk.

**Usage**

```
params(obj)

## S4 method for signature 'AbstractAnalysedPerturbationData'
params(obj)
```

**Arguments**

obj                    the object for which you want to extract the underlying params

**Value**

returns a list.

**Examples**

```
data(rnaiscreen)
ft <- hm(rnaiscreen)
params(ft)
```

PerturbationData-class

*Data wrapper for a data set of a perturbation screen.*

**Description**

Class PerturbationData wraps a data set derived from a genetic perturbation screen, e.g. using RNA interference or CRISPR. Class PerturbationData exposes getters for its members of the same name, but no setters, because the data should be treated as constant once set. The easiest way to construct a PerturbationData object is by first creating a data.frame and then calling as. See the examples to construct an object.

**Slots**

dataSet the data set as a tibble

**Examples**

```
df <- data.frame(Condition = c("V1", "V2", "V3"),
                 Replicate = c(1, 1, 1),
                 GeneSymbol = c("TP52", "NegCtrl", "PosCtrl"),
                 Perturbation = c("P1", "P2", "P3"),
                 Readout = c(123, 121, 12),
                 Control = c(0, -1, 1))
methods::as(df, "PerturbationData")
```

plot.NetworkAnalysedPerturbationData

*Plot a NetworkAnalysedPerturbationData object*

**Description**

Creates a table of the gene ranking of a NetworkAnalysedPerturbationData object

**Usage**

```
## S3 method for class 'NetworkAnalysedPerturbationData'  
plot(x, size = 10, main = "", ...)
```

**Arguments**

x	a NetworkAnalysedPerturbationData object
size	size of letters
main	title of the plot
...	additional parameters

**Value**

returns a table if the first cnt highest ranked genes

---

*plot.PerturbationData* *Plot perturbation data*

---

**Description**

Creates a barplot of replicate and gene counts of a PerturbationData object.

**Usage**

```
## S3 method for class 'PerturbationData'  
plot(x, size = 10, ...)
```

**Arguments**

x	the object to plot
size	size of letters
...	additional parameters

**Value**

returns a plot object

---

`rbind.PerturbationData`*Bind multiple perturbation data sets together by row*

---

**Description**

Binds multiple `PerturbationData` objects together by row.

**Usage**

```
## S3 method for class 'PerturbationData'  
rbind(...)
```

**Arguments**

... variable number of `PerturbationData` objects

**Value**

returns a combined object of class `PerturbationData`

**Examples**

```
data(rnaiscreen)  
rbind(rnaiscreen, rnaiscreen)
```

---

`rnaiscreen`*A sample pan-pathogenic perturbation dataset*

---

**Description**

Example `PerturbationData` object of two integrated viral RNAi screens that consists of a HCV kinome screen and a SARS kinome screen (see references).

**Usage**

```
data(rnaiscreen)
```

**Format**

A `PerturbationData` object containing a tibble with 18 columns, each describing a feature.

**Author(s)**

Simon Dirmeier | Computational Biology Group, ETH ZURICH | <simon.dirmeier@web.de>

## References

de Wilde, Adriaan H., et al. (2015), A kinome-wide small interfering RNA screen identifies proviral and antiviral host factors in severe acute respiratory syndrome coronavirus replication, including double-stranded RNA-activated protein kinase and early secretory pathway proteins.

*Journal of virology*

Reiss, Simon, et al. (2011), Recruitment and activation of a lipid kinase by hepatitis C virus NS5A is essential for integrity of the membranous replication compartment.

*Cell Host & Microbe*

Friedman J., Hastie T., Hoefling H. and Tibshirani R. (2007), Pathwise coordinate optimization.

*The Annals of Applied Statistics*

- Condition character names of the viruses
- Replicate integer replicate number
- Plate integer plate index
- RowIdx integer row index of the well on the plate
- ColIdx integer column index of the well on the plate
- GeneSymbol character HugoSymbol of a gene
- Entrez integer entrez ID
- ReadoutType character readout type, such as 'GFP' or 'Luciferase'
- Control integer coding of controls. '-1' for negative control, '1' for positive control, '0' for regular sample
- Library character library type, such as 'Ambion'
- Perturbation character sirna identifier
- Screen character identifier, for example 'Kinome' or 'Genome'
- Cell character cell type, such as 'Huh7.5'
- ScreenType character screen type, such as 'E/R' for entry/replication
- Design character design of the library, e.g. 'pooled'
- Readout numeric readout value, e.g. GFP measurement or read count
- ReadoutClass character class of the readout, such as 'Readout' or 'Viability'
- NumCells integer number of measured cells per well

## Examples

```
data(rnaiscreen)
fit <- hm(rnaiscreen)
pls <- plot(fit)
pls[[1]]
```

---

setModelData	<i>Create model data for an hierarchical model</i>
--------------	--

---

**Description**

Create model data for an hierarchical model

**Usage**

```
setModelData(obj, drop = TRUE, weights = 1)

## S4 method for signature 'PerturbationData'
setModelData(obj, drop = TRUE, weights = 1)
```

**Arguments**

obj	an data set
drop	boolean if genes that are not found in every Condition should be dropped
weights	a numeric vector used as weights for the single perturbations

**Value**

returns an PerturbationData object



# Index

- \* **datasets**
  - rnaiscreen, [14](#)
- \* **data**
  - rnaiscreen, [14](#)
- \* **package**
  - perturbatr-package, [2](#)
- bootstrap, [3](#)
- dataSet, [4](#)
- dataSet, AbstractAnalysedPerturbationData-method (dataSet), [4](#)
- dataSet, PerturbationData-method (dataSet), [4](#)
- diffuse, [4](#), [11](#)
- diffuse, HMANalysedPerturbationData-method (diffuse), [4](#)
- filter, [5](#)
- geneEffects, [6](#)
- geneEffects, AbstractAnalysedPerturbationData-method (geneEffects), [6](#)
- graph, [7](#)
- graph, NetworkAnalysedPerturbationData-method (graph), [7](#)
- hm, [7](#), [8](#)
- hm, PerturbationData-method (hm), [7](#)
- HMANalysedPerturbationData-class, [8](#)
- inference, [9](#)
- inference, AbstractAnalysedPerturbationData-method (inference), [9](#)
- isBootstrapped, [9](#)
- isBootstrapped, AbstractAnalysedPerturbationData-method (isBootstrapped), [9](#)
- modelFit, [10](#)
- modelFit, HMANalysedPerturbationData-method (modelFit), [10](#)
- nestedGeneEffects, [10](#)
- nestedGeneEffects, HMANalysedPerturbationData-method (nestedGeneEffects), [10](#)
- NetworkAnalysedPerturbationData-class, [11](#)
- params, [11](#)
- params, AbstractAnalysedPerturbationData-method (params), [11](#)
- PerturbationData-class, [12](#)
- perturbatr-package, [2](#)
- plot.NetworkAnalysedPerturbationData, [12](#)
- plot.PerturbationData, [13](#)
- rbind.PerturbationData, [14](#)
- rnaiscreen, [14](#)
- setModelData, [16](#)
- setModelData, PerturbationData-method (setModelData), [16](#)