Package ‘sceptre’

September 28, 2020

Type Package

Title Conditional resampling differential expression analysis for pooled CRISPR screens

Version 0.1.0

Author Gene Katsevich, Tim Barry, Kathryn Roeder

Maintainer Tim Barry <tbarry2@andrew.cmu.edu>

Description Functions for performing differential expression analyses in single-cell pooled CRISPR screens based on conditional resampling.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Depends sn,
MASS,
VGAM,
tidyverse,
R (>= 3.10)

Suggests knitr,
rmarkdown

VignetteBuilder knitr

R topics documented:

run_gene_precomputation 2
run_gRNA_precomputation 3
run_sceptre_gRNA_gene_pair 3
run_sceptre_using_precomp 5
simulate_crispr_screen_data 6

Index **8**

`run_gene_precomputation` *Run gene precomputation*

Description

This function runs the precomputation for a given gene. In particlar, it fits an NB regression of expression against covariates. The estimate of theta (i.e., the NB dispersion parameter) is obtained from glm.nb function. This is sensible as, under the null hypothesis, the NB model without the gRNA indicator is true. Offsets are obtained by log-transforming the fitted values.

Usage

```
run_gene_precomputation(
  expressions,
  covariate_matrix,
  gene_precomp_dispersion,
  gene_precomp_offsets
)
```

Arguments

`expressions` the vector of gene expressions

`covariate_matrix`
the cell-specific covariate matrix

`gene_precomp_dispersion`
the pre-computed dispersion parameter (NULL if none)

`gene_precomp_offsets`
the pre-computed gene offsets (NULL of none)

Value

a named list containing two items: offsets and dispersion.

Examples

```
sim_dat <- simulate_crispr_screen_data(num_cells = 1000,
grna_mean_prob = 0.2,
covariate_sampler = list(cell_size = rnorm, cell_cycle = runif),
mRNA_mean_expression = 40,
gRNA_effect = 0,
covariate_effects = c(0.5, 1),
zero_inflation = 0,
neg_binom_size = 2)
expressions <- sim_dat$Y
covariate_matrix <- sim_dat$covariate_df
gene_precomp <- run_gene_precomputation(expressions, covariate_matrix, NULL, NULL)
```

run_gRNA_precomputation

Run gRNA precomputation

Description

This function runs the precomputation for a given gRNA.

Usage

```
run_gRNA_precomputation(gRNA_indicators, covariate_matrix)
```

Arguments

`gRNA_indicators`
a vector of gRNA indicators

`covariate_matrix`
the cell-specific covariate matrix

Value

the fitted probabilities

Examples

```
sim_dat <- simulate_crispr_screen_data(num_cells = 1000,
grna_mean_prob = 0.2,
covariate_sampler = list(cell_size = rnorm, cell_cycle = runif),
mRNA_mean_expression = 40,
gRNA_effect = -4,
covariate_effects = c(0.5, 1),
zero_inflation = 0,
neg_binom_size = 2)
gRNA_indicators <- sim_dat$X
covariate_matrix <-sim_dat$covariate_df
fitted_probs <- run_gRNA_precomputation(gRNA_indicators, covariate_matrix)
```

run_sceptre_gRNA_gene_pair

Run sceptre on a gRNA-gene pair

Description

This function runs the sceptre algorithm on a single gRNA-gene pair. It requires as arguments the gene expression vector, the gRNA indicator vector, and the covariate matrix. Users optionally can pass the gRNA precomputation or gene precomputation as arguments.

Usage

```
run_sceptre_gRNA_gene_pair(
  expressions,
  gRNA_indicators,
  covariate_matrix,
  gRNA_precomp = NULL,
  gene_precomp_dispersion = NULL,
  gene_precomp_offsets = NULL,
  B = 500,
  seed = NULL
)
```

Arguments

`expressions` a vector a gene expressions

`gRNA_indicators` a vector of gRNA inicators

`covariate_matrix` the matrix of cell-specific covariates (e.g., library size, batch effect, cell cycle, etc.)

`gRNA_precomp` (optional) the gRNA precomputation (a vector of gRNA presence conditional probabilities)

`gene_precomp_dispersion` (optional) the pre-computed gene dispersion

`gene_precomp_offsets` (optional) the pre-computed gene offsets

`B` number of resamples (default 500)

`seed` (optional) seed to the random number generator

Value

a p-value of the null hypothesis of no gRNA effect on gene expression

Examples

```
# An example in which the alternative is true.
sim_dat <- simulate_crispr_screen_data(num_cells = 1000,
grna_mean_prob = 0.2,
covariate_sampler = list(cell_size = rnorm, cell_cycle = runif),
mRNA_mean_expression = 40,
gRNA_effect = -4,
covariate_effects = c(0.5, 1),
zero_inflation = 0,
neg_binom_size = 2)
expressions <- sim_dat$Y
gRNA_indicators <- sim_dat$X
covariate_matrix <-sim_dat$covariate_df
run_sceptre_gRNA_gene_pair(expressions = expressions,
gRNA_indicators = gRNA_indicators,
covariate_matrix = covariate_matrix)

# An example in which the null is true.
```

```
sim_dat <- simulate_crispr_screen_data(num_cells = 1000,
grna_mean_prob = 0.2,
covariate_sampler = list(cell_size = rnorm, cell_cycle = runif),
mRNA_mean_expression = 40,
gRNA_effect = 0,
covariate_effects = c(0.5, 1),
zero_inflation = 0,
neg_binom_size = 2)
expressions <- sim_dat$Y
gRNA_indicators <- sim_dat$X
covariate_matrix <-sim_dat$covariate_df
run_sceptre_gRNA_gene_pair(expressions = expressions,
gRNA_indicators = gRNA_indicators,
covariate_matrix = covariate_matrix)
```

run_sceptre_using_precomp

Run sceptre using precomputations for gRNAs and genes.

Description

This function is the workhorse function of the sceptre package. It runs a distilled CRT using a negative binomial test statistic based on an expression vector, a gRNA indicator vector, an offset vector (from the distillation step), gRNA conditional probabilities, an estimate of the negative binomial dispersion parameter, and the number of resampling replicates.

Usage

```
run_sceptre_using_precomp(
  expressions,
  gRNA_indicators,
  gRNA_precomp,
  gene_precomp_dispersion,
  gene_precomp_offsets,
  B,
  seed
)
```

Arguments

`expressions`	a vector of gene expressions (in UMI counts)
`gRNA_indicators`	a vector of gRNA indicators
`gRNA_precomp`	a vector of conditional probabilities for gRNA assignments
`gene_precomp_dispersion`	the pre-computed dispersion
`gene_precomp_offsets`	the pre-computed distillation offsets
`B`	the number of resamples to make (default 500)
`seed`	an arguement to set.seed; if null, no seed is set

Details

This currently is a one-tailed, left-sided test. Thus, it is suitable for up-regulatory elements like enhancers and promoters but not down-regulatory elements like silencers.

Value

a p-value of the null hypothesis of no gRNA effect on gene expression

`simulate_crispr_screen_data` *Simulate CRISPR screen data.*

Description

A function to simulate pooled CRISPR screen data.

Usage

```
simulate_crispr_screen_data(
  num_cells,
  grna_mean_prob,
  covariate_sampler,
  mRNA_mean_expression,
  gRNA_effect,
  covariate_effects,
  zero_inflation,
  neg_binom_size,
  seed = NULL
)
```

Arguments

`num_cells`	number of cells in experiment
`grna_mean_prob`	mean number of cells perturbed (absent covariates)
`covariate_sampler`	a list of functions. Each function in this list takes as an argument num_cells and returns num_cells random variates.
`mRNA_mean_expression`	mean mRNA expression of the cell (absent covariates)
`gRNA_effect`	effect of the gRNA perturbation on mRNA expression
`covariate_effects`	a numeric vector indicating the covariate effects on mRNA expression
`zero_inflation`	a numeric scalar between 0 and 1 (inclusive) giving the mean fraction of cell expressions set to zero
`neg_binom_size`	size parameter for the negative binomial model
`seed`	a seed to the random number generator

Value

a list containing (i) the expression vector Y, (ii) the data frame of covariates, and (iii) the gRNA presence indicator vector.

Examples

```
num_cells <- 1000
grna_mean_prob <- 0.2
covariate_sampler <- list(
 cell_size = runif,
 cell_cycle = function(x) {runif(n = x, min = 0, max = 1)}
)
mRNA_mean_expression <- 40
gRNA_effect <- 4
covariate_effects <- c(2, 1)
zero_inflation <- 0
neg_binom_size <- 2
simulated_data <- simulate_crispr_screen_data(num_cells,
 grna_mean_prob,
 covariate_sampler,
 mRNA_mean_expression,
 gRNA_effect,
 covariate_effects,
 zero_inflation,
 neg_binom_size)
```

Index

run_gene_precomputation, 2
run_gRNA_precomputation, 3
run_sceptre_gRNA_gene_pair, 3
run_sceptre_using_precomp, 5

simulate_crispr_screen_data, 6