Package 'sctransform'

November 18, 2018

Type Package

Title Variance Stabilizing Transformations for Single Cell UMI Data

Version 0.1.0

Description A normalization method for single-cell UMI count data using a variance stabilizing transformation. The transformation is based on a negative binomial regression model with regularized parameters. As part of the same regression framework, this package also provides functions for batch correction, and data correction/denoising.

URL https://github.com/ChristophH/sctransform

BugReports https://github.com/ChristophH/sctransform/issues

License GPL-3 | file LICENSE

Encoding UTF-8

LazyData true

Depends R (>= 2.10)

Imports MASS, Matrix, parallel, methods

Suggests matrixStats, testthat

RoxygenNote 6.1.0

NeedsCompilation no

Author Christoph Hafemeister [aut, cre] (<https://orcid.org/0000-0001-6365-8254>)

Maintainer Christoph Hafemeister <chafemeister@nygenome.org>

Repository CRAN

Date/Publication 2018-11-18 16:30:03 UTC

R topics documented:

`denoise` *Denoise data by setting all latent factors to their median values and reversing the regression model*

Description

Denoise data by setting all latent factors to their median values and reversing the regression model

Usage

```
denoise(x, data = "y", cell_attr = x$cell_attr, do_round = TRUE,
  do_pos = TRUE, show_progress = TRUE)
```

Arguments

`x`	A list that provides model parameters and optionally meta data; use output of vst function
`data`	The name of the entry in x that holds the data
`cell_attr`	Provide cell meta data holding latent data info
`do_round`	Round the result to integers
`do_pos`	Set negative values in the result to zero
`show_progress`	Whether to print progress bar

Value

De-noised data as UMI counts

Examples

```
vst_out <- vst(pbmc, return_cell_attr = TRUE)
umi_denoised <- denoise(vst_out)
```

get_deviance_residuals *Return deviance residuals of regularized models*

Description

Return deviance residuals of regularized models

Usage

```
get_deviance_residuals(vst_out, umi, cell_attr = vst_out$cell_attr,
  bin_size = 256, show_progress = TRUE)
```

Arguments

vst_out	The output of a vst run
umi	The UMI count matrix that will be converted
cell_attr	Data frame of cell meta data
bin_size	Number of genes to put in each bin (to show progress)
show_progress	Whether to print progress bar

Value

A matrix of deviance residuals

Examples

```
## Not run:
vst_out <- vst(pbmc, return_gene_attr = TRUE)
dev_res <- get_deviance_residuals(vst_out, pbmc)

## End(Not run)
```

is_outlier *Identify outliers*

Description

Identify outliers

Usage

```
is_outlier(y, x, th = 10)
```

Arguments

y	Dependent variable
x	Independent variable
th	Outlier score threshold

Value

Boolean vector

---

pbmc *Peripheral Blood Mononuclear Cells (PBMCs)*

Description

UMI counts for a subset of cells freely available from 10X Genomics

Usage

```
pbmc
```

Format

A sparse matrix (dgCMatrix, see Matrix package) of molecule counts. There are 914 rows (genes) and 283 columns (cells). A downsampled version of the 3K PBMC dataset available from 10x Genomics

Source

https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.1.0/pbmc3k

---

robust_scale *Robust scale using median and mad*

Description

Robust scale using median and mad

Usage

```
robust_scale(x)
```

Arguments

x	Numeric

Value

Numeric

robust_scale_binned *Robust scale using median and mad per bin*

Description

Put the values in y in bins based on the values in x and the breaks defined in breaks. Apply robust scaling to y per bin.

Usage

```
robust_scale_binned(y, x, breaks)
```

Arguments

y	Numeric vector
x	Numeric vector
breaks	Numeric vector of breaks

Value

Numeric vector of scaled values

vst *Variance stabilizing transformation for UMI count data*

Description

Apply variance stabilizing transformation to UMI count data using a regularized Negative Binomial regression model. This will remove unwanted effects from UMI data and return Pearson residuals. Uses mclapply; you can set the number of cores it will use to n with command options(mc.cores = n). If n_genes is set, only a (somewhat-random) subset of genes is used for estimating the initial model parameters.

Usage

```
vst(umi, cell_attr = NULL, latent_var = c("log_umi"),
  batch_var = NULL, latent_var_nonreg = NULL, n_genes = 2000,
  n_cells = NULL, method = "poisson", do_regularize = TRUE,
  res_clip_range = c(-sqrt(ncol(umi)), sqrt(ncol(umi))),
  bin_size = 256, min_cells = 5, return_cell_attr = FALSE,
  return_gene_attr = FALSE, return_dev_residuals = FALSE,
  return_corrected_umi = FALSE, bw_adjust = 3, theta_given = NULL,
  show_progress = TRUE)
```

Arguments

umi	A matrix of UMI counts with genes as rows and cells as columns
cell_attr	A data frame containing the dependent variables; if omitted a data frame with umi and gene will be generated
latent_var	The dependent variables to regress out as a character vector; must match column names in cell_attr; default is c("log_umi_per_gene")
batch_var	The dependent variables indicating which batch a cell belongs to; no batch interaction terms used if omiited
latent_var_nonreg	The non-regularized dependent variables to regress out as a character vector; must match column names in cell_attr; default is NULL
n_genes	Number of genes to use when estimating parameters (default uses 2000 genes, set to NULL to use all genes)
n_cells	Number of cells to use when estimating parameters (default uses all cells)
method	Method to use for initial parameter estimation; one of 'poisson', 'nb_fast', 'nb'
do_regularize	Boolean that, if set to FALSE, will bypass parameter regularization
res_clip_range	Numeric of length two specifying the min and max values the results will be clipped to; default is c(-sqrt(ncol(umi)), sqrt(ncol(umi)))
bin_size	Number of genes to put in each bin (to show progress)
min_cells	Only use genes that have been detected in at least this many cells
return_cell_attr	Make cell attributes part of the output
return_gene_attr	Calculate gene attributes and make part of output
return_dev_residuals	If set to TRUE output will be deviance residuals, NOT Pearson residuals; default is FALSE
return_corrected_umi	If set to TRUE output will contain corrected UMI matrix; see denoise function
bw_adjust	Kernel bandwidth adjustment factor used during regurlarization; factor will be applied to output of bw.SJ; default is 3
theta_given	Named numeric vector of fixed theta values for the genes; will only be used if method is set to nb_theta_given; default is NULL
show_progress	Whether to print progress bar

Value

A list with components

y	Matrix of transformed data, i.e. Pearson residuals
umi_corrected	Matrix of corrected UMI counts (optional)
model_str	Character representation of the model formula

`model_pars` Matrix of estimated model parameters per gene (theta and regression coefficients)

`model_pars_outliers` Vector indicating whether a gene was considered to be an outlier

`model_pars_fit` Matrix of fitted / regularized model parameters

`model_str_nonreg` Character representation of model for non-regularized variables

`model_pars_nonreg` Model parameters for non-regularized variables

`genes_log_mean_step1` log-mean of genes used in initial step of parameter estimation

`cells_step1` Cells used in initial step of parameter estimation

`arguments` List of function call arguments

`cell_attr` Data frame of cell meta data (optional)

`gene_attr` Data frame with gene attributes such as mean, detection rate, etc. (optional)

Details

In the first step of the algorithm, per-gene glm model parameters are learned. This step can be done on a subset of genes and/or cells to speed things up. If `method` is set to 'poisson', glm will be called with `family = poisson` and the negative binomial theta parameter will be estimated using the response residuals in `MASS::theta.ml`. If `method` is set to 'nb_fast', glm coefficients and theta are estimated as in the 'poisson' method, but coefficients are then re-estimated using a proper negative binomial model in a second call to glm with `family = MASS::negative.binomial(theta = theta)`. If `method` is set to 'nb', coefficients and theta are estimated by a single call to `MASS::glm.nb`.

Examples

```
vst_out <- vst(pbmc)
```

Index