Package ‘trendsceek’

June 11, 2017

Title Identify Genes with Spatial Expression Trends in Single-Cell Gene Expression Data

Version 0.9.0

Description Identify genes with spatial expression trends in single-cell gene expression data using marked point processes.

URL https://github.com/edsgard/trendsceek

BugReports https://github.com/edsgard/trendsceek/issues

Depends R (>= 3.3.1)

License GPL-3

Encoding UTF-8

LazyData true

Imports BiocParallel,
genefilter,
statmod,
MASS,
matrixStats,
reshape2,
ggplot2,
dplyr,
tidyr,
grid,
tsne,
RColorBrewer,
KernSmooth,
spatstat,
sm,
tibble,
DESeq2,
data.table

Suggests testthat,
knitr,
rmarkdown

VignetteBuilder knitr

RoxygenNote 5.0.1

R topics documented:

---

add_markdist_hotspot *Add hot-spot mark distributions to a point pattern*

Description

`add_markdist_hotspot` adds squared hot-spot mark distributions to a point pattern.

Usage

```
add_markdist_hotspot(pp, low_marks, high_marks, hotspot_size = 0.2)
```

Arguments

`pp`	A point-pattern.
`low_marks`	A numeric or numeric vector specifying the lower value of the mark distribution. If a vector is given 'n' mark distributions are added where 'n' is the length of the vector.
`high_marks`	A numeric or numeric vector specifying the upper value of the mark distribution. If a vector is given 'n' mark distributions are added where 'n' is the length of the vector.
`hotspot_size`	A numeric specifying the relative size of the point-pattern window where the values of the marks will change from low to high.

Value

A point-pattern with added mark distributions.

Examples

```
low_expr = c(10, 10)
high_expr = c(15, 20)
pp = sim_pois(100)
pp = add_markdist_hotspot(pp, low_expr, high_expr)
```

---

add_markdist_step *Add step-gradient mark distributions to a point pattern*

Description

add_markdist_step adds step-gradient mark distributions to a point pattern.

Usage

```
add_markdist_step(pp, low_marks, high_marks, step_border = 0.5)
```

Arguments

pp	A point-pattern.
low_marks	A numeric or numeric vector specifying the lower value of the step mark distribution. If a vector is given 'n' mark distributions are added where 'n' is the length of the vector.
high_marks	A numeric or numeric vector specifying the upper value of the step mark distribution. If a vector is given 'n' mark distributions are added where 'n' is the length of the vector.
step_border	A numeric specifying the relative x-position within the point-pattern window where the values of the marks will change from low to high.

Value

A point-pattern with added mark distributions.

Examples

```
low_expr = c(10, 10)
high_expr = c(15, 20)
pp = sim_pois(100)
pp = add_markdist_step(pp, low_expr, high_expr)
```

add_markdist_streak *Add streak mark distributions to a point pattern*

Description

add_markdist_streak adds rectangular streak mark distributions to a point pattern.

Usage

```
add_markdist_streak(pp, low_marks, high_marks, streak_len_frac = 0.9,
  streak_height_frac = 0.05)
```

Arguments

pp	A point-pattern.
low_marks	A numeric or numeric vector specifying the lower value of the mark distribution. If a vector is given 'n' mark distributions are added where 'n' is the length of the vector.
high_marks	A numeric or numeric vector specifying the upper value of the mark distribution. If a vector is given 'n' mark distributions are added where 'n' is the length of the vector.
streak_len_frac	A numeric specifying the relative length of the point-pattern window where the values of the marks will change from low to high.
streak_height_frac	A numeric specifying the relative height of the point-pattern window where the values of the marks will change from low to high.

Value

A point-pattern with added mark distributions.

Examples

```
low_expr = c(10, 10)
high_expr = c(15, 20)
pp = sim_pois(100)
pp = add_markdist_streak(pp, low_expr, high_expr)
```

calc_varstats *Calculate gene variability stats*

Description

calc_varstats calculates the variability of each gene, conditioned on their expression level

Usage

```
calc_varstats(counts, ercc.raw, n.ercc.min = 2, min.count = 2,
  min.prop = 0.5, min.cv2 = 0.3, quant.cutoff = 0.9,
  min.biol.cv2 = 0.5^2, n.win = 2, method = "glm", min.mean.limit = 0,
  counts.pseudo.count = 1, ercc.pseudo.count = 0)
```

Arguments

`counts` A numeric matrix with read count expression values (genes x cells)

`ercc.raw` A numeric matrix with read count expression values of ERCC spike-in transcripts. If not available then input 'counts' also for this argument.

`n.ercc.min` An integer specifying the minimum number of ERCC transcripts with any read mapped to it. Samples with less ERCC transcripts are removed from the fitting against the ERCC data.

`min.count` An integer specifying the minimum read count for which a transcript is considered to be expressed

`min.prop` A numeric of the proportion of ERCC transcripts that needs to be expressed (above 'min.count'). Samples with lower proportion of expressed ERCC transcripts are removed from the ERCC fitting.

`min.cv2` A numeric specifying the minimum squared coefficient of variation (CV^2) of the ERCC transcript expression. The quantile of the expression distribution of transcripts with higher CV^2 is used to set a minumum expression threshold for genes to be used for ERCC fitting (see argument 'quant.cutoff' below).

`quant.cutoff` A numeric specifying the quantile of the expression distribution of ERCC transcripts. Transcripts with mean expression below the expression at this quantile are excluded from ERCC fitting.

`min.biol.cv2` A numeric specifying the minimum squared coefficient of variation of real biological data.

`n.win` An integer specifying the number of the extremest expression values in the expression distribution of a sample to be set to the 'n.win' highest value (Winsorizing).

`method` A character specifying if generalized linear model ('glm') or robust linear regression ('rlm') should be used.

`min.mean.limit` An integer, transcripts with mean expression below this limit are excluded from ERCC fitting.

`counts.pseudo.count`
An integer with the pseudo-count to be added before taking the logarithm

`ercc.pseudo.count`
An integer with the pseudo-count to be added to the ERCC data before taking the logarithm

Value

A list with variability test statistics for all genes and results from fitting a glm or rlm regression model to the ERCC expression data

Examples

```
data('scialdone')
counts = scialdone[['counts']]
counts_filt = genefilter_exprmat(counts, min.expr = 5, min.ncells.expr = 3)

quantile.cutoff = 0.9 ##filter out the most lowly expressed genes from the fitting
vargenes_stats = calc_varstats(counts_filt, counts_filt, quant.cutoff = quantile.cutoff)
```

---

cellsceek_test *Identify cells located in regions exceeding random background expression level*

Description

cellsceek_test identifies cells located in regions with higher expression level than expected by random. The spatial distribution is presumed to be fixed and conditioned on that, the test assesses whether cells are in a region with high expression levels. The background expression 2-dimensional null-distribution is generated by random resampling of the mark distribution followed by 2-dimensional kernel density estimate for each resampling.

Usage

```
cellsceek_test(pp, nrand = 100, cell_alpha = 0.05, h = NA)
```

Arguments

pp	A point pattern with one or more mark distributions.
nrand	An integer specifying the number of random resamplings of the mark distribution as to create the null-distribution.
cell_alpha	A numeric specifying a signficance level which is used to flag if a cell has significantly higher expression than expected by random for a particular gene or not.
h	A numeric vector of length 2 specifying the bandwidth of the two-dimensional Gaussian kernel (x, y).

Value

A list containing statistics for all cells for each gene.

Examples

```
pp = sim_pois(100)
low_expr = c(10, 10)
high_expr = c(15, 20)
pp = add_markdist_streak(pp, low_expr, high_expr)
cellpeaks = cellsceek_test(pp)
```

deseq_norm *Normalize read counts using DESeq2*

Description

deseq_norm normalizes read counts using size-factors estimated by DESeq2

Usage

```
deseq_norm(counts, min.count)
```

Arguments

counts	A numeric matrix with read count expression values (genes x cells)
min.count	A numeric specifying the minimum total read count across all cells that a transcript need to have. Transcripts with lower expression level are removed from size-factor estimation.

Value

A numeric matrix with normalized expression values (genes x cells)

Examples

```
data('scialdone')
counts = scialdone[['counts']]
counts_norm = deseq_norm(counts, min.count = 1)
```

extract_sig_genes *Extract significant genes from trendsceek results*

Description

extract_sig_genes extracts significant genes from trendsceek results.

Usage

```
extract_sig_genes(trendstat_list, alpha = 0.05)
```

Arguments

trendstat_list	A list containing results generated by calling trendsceek_test.
alpha	A numeric specifying a Benjamini-Hochberg signficance level used to extract significant genes.

Value

A list containing significant genes for each test.

Examples

```
pp = sim_pois(100)
low_expr = c(10, 10)
high_expr = c(15, 20)
pp = add_markdist_hotspot(pp, low_expr, high_expr)
trendstat_list = trendsceek_test(pp, nrand = 100, ncores = 1)
sig_list = extract_sig_genes(trendstat_list, alpha = 0.05)
```

---

genefilter_exprmat *Filter genes on being expressed*

Description

genefilter_exprmat filter genes on being expressed in a number of cells.

Usage

```
genefilter_exprmat(exprmat, min.expr, min.ncells.expr)
```

Arguments

exprmat A numeric matrix (genes x cells)

min.expr A numeric specifying the minimum expression required in a cell

min.ncells.expr An integer specifying the minimum number of cells a gene needs to be expressed in with an expression level above 'min.expr'

Value

A matrix where genes not passing the expression criteria have been removed

Examples

```
a_mat = matrix(rnorm(100, 0, 1), nrow = 10, dimnames = list(1:10, 1:10))

filt_mat = genefilter_exprmat(a_mat, min.expr = 0, min.ncells.expr = 3)
print(nrow(filt_mat))
```

get_sigcells *Extract cells located in regions exceeding random background expression level*

Description

get_sigcells extracts an indicator matrix (cells x genes) specifying which cells are located in regions with higher expression than can be expected by chance, given the spatial location of the cells. It takes the output from cellsceek_test as input.

Usage

```
get_sigcells(cellpeak_stats)
```

Arguments

cellpeak_stats	A list with with cell-peak statistics for each cell and gene as output by cellsceek_test.

Value

An indicator matrix (cells x genes) containing 0's and 1's specifying which cells are located in regions which significantly higher expression.

Examples

```
pp = sim_pois(100)
low_expr = c(10, 10)
high_expr = c(15, 20)
pp = add_markdist_streak(pp, low_expr, high_expr)
cellpeak_stats = cellsceek_test(pp)
cell_ind_mat = get_sigcells(cellpeak_stats)
```

plot_cv2vsmean *Plot squared coefficient of variation against mean expression*

Description

plot_cv2vsmean plots the squared coefficient of variation of all genes against the mean expression using the output from calc_varstats as input

Usage

```
plot_cv2vsmean(vargenes_stats, sel.genes, sel.highlight = TRUE,
  plot.ercc.points = TRUE, method = "glm")
```

Arguments

vargenes_stats A list with variability test statistics output by calc_varstats

sel.genes A character vector with names of genes to be highlighed in the plot as points

sel.highlight A logical specifying whether 'sel.genes' should be highlighed

plot.ercc.points A logical specifying whether the transcripts used for fitting should be shown as points

method A character specifying if a generalized linear model ('glm') or robust linear regression ('rlm') was used.

Examples

```
data('scialdone')
counts = scialdone[['counts']]
counts_filt = genefilter_exprmat(counts, min.expr = 5, min.ncells.expr = 3)

##Calculate gene variability stats
quantile.cutoff = 0.9 ##filter out the most lowly expressed genes from the fitting
vargenes_stats = calc_varstats(counts_filt, counts_filt, quant.cutoff = quantile.cutoff)

##Select subset of top most variable genes
topvar.genes = rownames(vargenes_stats[['real.stats']])[1:500]

##Plot
plot_cv2vsmean(vargenes_stats, topvar.genes, plot.ercc.points = FALSE)
```

plot_pp_density *Plot density-plot of point pattern*

Description

plot_pp_density plots a density-plot of a point-pattern

Usage

```
plot_pp_density(pp, pointsize.dens2d.factor = 7.5/20, log_marks = FALSE,
  cells2highlight = NA)
```

Arguments

pp A point pattern

pointsize.dens2d.factor A numeric determining the size of the plotted points

log_marks A logical specifying if log10 should be applied to the marks.

cells2highlight An indicator matrix (cells x genes) specifying which cells to highlight as red points for every gene. The output from get_sigcells can be used as input.

Examples

```
pp = sim_pois(100)
low_expr = c(10, 10)
high_expr = c(15, 20)
pp = add_markdist_hotspot(pp, low_expr, high_expr)
plot_pp_density(pp)
```

`plot_pp_scatter` *Plot scatter-plot of point pattern*

Description

`plot_pp_scatter` plots a scatter-plot of a point-pattern

Usage

```
plot_pp_scatter(pp, pointsize.factor = 7.5, palette = "Spectral",
  pal.direction = 1, log_marks = TRUE, scale_marks = TRUE)
```

Arguments

`pp`	A point pattern
`pointsize.factor`	A numeric determining the size of the plotted points
`palette`	If a string, will use that named palette. If a number, will index into the list of palettes of appropriate ‘type’ (see ggplot2::scale_colour_distiller).
`pal.direction`	Sets the order of colors in the scale. If 1, the default, colors are as output by ‘brewer.pal’. If -1, the order of colors is reversed.
`log_marks`	A logical specifying if log10 should be applied to the marks.
`scale_marks`	A logical specifying if marks shoule be scaled to [0-1].

Examples

```
pp = sim_pois(100)
low_expr = c(10, 10)
high_expr = c(15, 20)
pp = add_markdist_hotspot(pp, low_expr, high_expr)
plot_pp_scatter(pp)
```

plot_trendstats *Plot trendsceek test-statistics*

Description

plot_trendstats plots the trend-statistics by radius for a set of genes

Usage

```
plot_trendstats(trendstat_list, sel_genes, order_method = "markcorr")
```

Arguments

trendstat_list	A list containing results generated by calling trendsceek_test.
sel_genes	A character vector specifying the names of the mark distributions (genes) to be plotted
order_method	A character vector specifying how the genes should be sorted in the plot. Options include: 'markcorr', 'markvario', 'Emark' and 'Vmark'.

Examples

```
pp = sim_pois(300)
low_expr = c(10, 10)
high_expr = c(20, 50)
pp = add_markdist_hotspot(pp, low_expr, high_expr)
trendstat_list = trendsceek_test(pp, nrand = 100, ncores = 1)
sig_list = extract_sig_genes(trendstat_list, alpha = 0.1)
sig_genes = sig_list[['markcorr']][, 'gene']
plot_trendstats(trendstat_list, sig_genes)
```

pos2pp *Convert positions to point-pattern*

Description

pos2pp converts 2-dimensional positions to a point-pattern object

Usage

```
pos2pp(pos_mat)
```

Arguments

pos_mat	A numeric matrix where each row corresponds to a point with x positions in the first column and y-positions in the second column.

Value

A point pattern

Examples

```
a_mat = matrix(rnorm(100, 0, 1), ncol = 2)

pp = pos2pp(a_mat)
```

pp_select *Subset the mark distributions of a point pattern*

Description

`pp_select` subsets the mark distributions of a point pattern

Usage

```
pp_select(pp, sel.genes)
```

Arguments

`pp`	A point pattern
`sel.genes`	A character vector of mark distributions (genes) to be kept

Value

A point pattern retaining only the marks of the input sel.genes

Examples

```
a_mat = matrix(rnorm(100, 0, 1), ncol = 10)
marx = matrix(rnorm(20, 0, 1), ncol = 10)

pp = pos2pp(a_mat)
pp = set_marks(pp, marx)
pp = pp_select(pp, 1)
```

scialdone *Single-cell RNA-seq data from the Scialdone et al paper*

Description

A dataset containing read counts for the subset of 481 cells annotated as "cluster 3" in the paper by Scialdone et al.

Usage

```
data(scialdone)
```

Format

A list with two elements, "counts" and "meta". Counts is a data-frame containing read counts for 41,388 genes and 481 cells. Meta is a data-frame containing annotations about all 481 cells.

Source

- RNA-seq read counts `http://gastrulation.stemcells.cam.ac.uk/data/counts.gz`
- RNA-seq meta-info`http://gastrulation.stemcells.cam.ac.uk/data/metadata.txt`

`set_marks` *Set the mark distribution of a point pattern*

Description

`set_marks` sets the mark distribution of a point pattern

Usage

```
set_marks(pp, gene.marks, log.fcn = NA, pseudo.count = 1)
```

Arguments

`pp`	A point pattern
`gene.marks`	A matrix (genes x cells) with as many columns as points (cells) in the point-pattern.
`log.fcn`	A log-function with the gene.marks is to be transformed
`pseudo.count`	A numeric specifying a pseudo-count to be added if a log-fcn is supplied.

Value

A point pattern with marks having been set

Examples

```
a_mat = matrix(rnorm(100, 0, 1), ncol = 10)
marx = matrix(rnorm(20, 0, 1), ncol = 10)

pp = pos2pp(a_mat)
pp = set_marks(pp, marx)
```

`sim_pois` *Generate Poisson Point Pattern*

Description

`sim_pois` generates a random point pattern using the Poisson process.

Usage

```
sim_pois(lambda_int, win_len = 1)
```

Arguments

`lambda_int`	An integer specifying the intensity of the Poisson process; the expected number of points *per unit area*. The total number of points in the simulated pattern will be random with expectation value 'mu = lambda * a' where 'a' is the area of the window in which the pattern is simulated.
`win_len`	A numeric specifying the window side-length of a quadratic window in which the pattern is simulated.

Value

A point-pattern

Examples

```
pp = sim_pois(100)
```

`trend.tsne` *Perform dimensionality reduction using t-SNE*

Description

`trend.tsne` runs t-SNE, including a PCA pre-processing step

Usage

```
trend.tsne(counts, tsne.k = 2, init.dims = 100, perp.frac = 0.2,
  max.iter = 500, epoch = 50, pseudo.count = 1)
```

Arguments

`counts`	A numeric matrix (genes x cells).
`tsne.k`	An integer with the dimension of the resulting embedding.
`init.dims`	An integer with the number of dimensions to be reduced to by the initial PCA before t-SNE is run.
`perp.frac`	A numeric specifying the fraction of samples to be used as neighbors by t-SNE (perplexity parameter).

`max.iter` An integer with the maximum number of iterations to perform.

`epoch` An integer with the number of iterations between printing update messages.

`pseudo.count` A numeric with the pseudo.count to be used before logging (log10) the read counts.

Value

A matrix with the positions of the cells in the t-SNE embedding

Examples

```
data('scialdone')
counts = scialdone[['counts']]
counts_norm = deseq_norm(counts, min.count = 1)
tsne_res = trend.tsne(counts_norm[1:500, ], tsne.k = 2, init.dims = 100, perp.frac = 0.2,
max.iter = 400, epoch = 50)
```

`trendsceek_test` *Test for the presence of spatial expression patterns*

Description

`trendsceek_test` tests for the presence of spatial expression patterns. The spatial distribution is presumed to be fixed and conditioned on that, the test assesses whether the mark distribution is non-random.

Usage

```
trendsceek_test(pp, nrand = 1e+05, ncores = 1,
  alpha_env = 0.1/ifelse(is.numeric(pp[["marks"]]), length(pp[["marks"]]),
  ncol(pp[["marks"]])), alpha_bh = 0.05, alpha_nom_early = 0.5)
```

Arguments

`pp` A point pattern with one or more mark distributions.

`nrand` An integer specifying the number of random resamplings of the mark distribution as to create the null-distribution.

`ncores` An integer specifying the number of cores to be used by BiocParallel

`alpha_env` A numeric specifying a Bonferroni-significance level used to create a test-statistic threshold for each radius, an "envelope". Note that this is solely used for plotting purposes.

`alpha_bh` A numeric specifying a Benjamini-Hochberg signficance level used to extract significant genes (note that this can also be done for an arbitrary alpha after trendsceek_test has been called, see `extract_sig_genes`).

`alpha_nom_early`
A numeric specifying an early-stopping threshold for the p-value. If the nominal p-value exceeds this value no more permutations are done for that gene.

Value

A list containing trendsceek-statistics for every gene.

Examples

```
pp = sim_pois(100)
low_expr = c(10, 10)
high_expr = c(15, 20)
pp = add_markdist_hotspot(pp, low_expr, high_expr)
trendstat_list = trendsceek_test(pp, nrand = 100, ncores = 1)
```

Index