Package ‘fishpond’

November 14, 2019

Title Fishpond: differential transcript and gene expression with inferential replicates

Version 1.3.0

Author Anqi Zhu, Avi Srivastava, Joseph Ibrahim, Rob Patro, Michael Love

Maintainer Michael Love <michaelisaiahlove@gmail.com>

Description Fishpond contains methods for differential transcript and gene expression analysis of RNA-seq data using inferential replicates for uncertainty of abundance quantification, as generated by Gibbs sampling or bootstrap sampling. Also the package contains utilities for working with Salmon and Alevin quantification files.

Imports graphics, stats, utils, methods, abind, gtools, qvalue, S4Vectors, SummarizedExperiment, matrixStats, svMisc, Rcpp

Suggests testthat, knitr, rmarkdown, macrophage, tximeta, org.Hs.eg.db, samr, DESeq2, apeglm

LinkingTo Rcpp

SystemRequirements C++11

License GPL-2

Encoding UTF-8

URL https://github.com/mikelove/fishpond

biocViews Sequencing, RNASeq, GeneExpression, Transcription, Normalization, Regression, MultipleComparison, BatchEffect, Visualization, DifferentialExpression, DifferentialSplicing, AlternativeSplicing, SingleCell

VignetteBuilder knitr

LazyData true

RoxygenNote 6.1.1

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

git_branch master

git_last_commit 3749249

git_last_commit_date 2019-10-29

Date/Publication 2019-11-13

R topics documented:

computeInfRV *Compute inferential relative variance (InfRV)*

Description

`InfRV` is used the Swish publication for visualization. This function provides computation of the mean InfRV, a simple statistic that measures inferential uncertainty. Note that InfRV is not used in the `swish` statistical method at all, it is just for visualization. See function code for details.

Usage

```
computeInfRV(y, pc = 5, shift = 0.01)
```

Arguments

`y`	a SummarizedExperiment
`pc`	a pseudocount parameter for the denominator
`shift`	a final shift parameter

Value

a SummarizedExperiment with `meanInfRV` in the metadata columns

deswish *deswish: DESeq2-apeglm With Inferential Samples Helps*

Description

The DESeq2-apeglm With Inferential Samples implementation supposes a hierarchical distribution of log2 fold changes. The final posterior standard deviation is calculated by adding the posterior variance from modeling biological replicates computed by `apeglm`, and the observed variance on the posterior mode over inferential replicates. This function requires the DESeq2 and apeglm packages to be installed and will print an error if they are not found.

Usage

```
deswish(y, x, coef)
```

Arguments

y	a SummarizedExperiment containing the inferential replicate matrices, as output by tximeta, and then with labelKeep applied. One does not need to run scaleInfReps as scaling is done internally via DESeq2.
x	the design matrix
coef	the coefficient to test (see lfcShrink)

Value

a SummarizedExperiment with metadata columns added: the log2 fold change and posterior SD using inferential replicates, and the original log2 fold change (apeglm) and its posterior SD

References

The DESeq and lfcShrink function in the DESeq2 package:

Zhu, Ibrahim, Love "Heavy-tailed prior distributions for sequence count data: removing the noise and preserving large differences" Bioinformatics (2018).

Love, Huber, Anders "Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2" Genome Biology (2014).

Examples

```
# a small example... 500 genes, 10 inf reps
y <- makeSimSwishData(m=500, numReps=10)
y <- labelKeep(y)
y <- deswish(y, ~condition, "condition_2_vs_1")
```

---

labelKeep *Label rows to keep based on minimal count*

Description

Adds a column keep to mcols(y) that specifies which rows of the SummarizedExperiment will be included in statistical testing. Rows are not removed, just marked with the logical keep.

Usage

```
labelKeep(y, minCount = 10, minN = 3, x)
```

Arguments

y	a SummarizedExperiment
minCount	the minimum count
minN	the minimum sample size at minCount
x	the name of the condition variable, will use the smaller of the two groups to set minN. Similar to edgeR's filterByExpr, as the smaller group grows past 10, minN grows only by 0.7 increments of sample size

Value

a SummarizedExperiment with a new column `keep` in `mcols(y)`

Examples

```
y <- makeSimSwishData()
y <- scaleInfReps(y)
y <- labelKeep(y)
```

`makeSimSwishData` *Make simulated data for swish for examples/testing*

Description

Makes a small swish dataset for examples and testing. The first six genes have some differential expression evidence in the counts, with varying degree of inferential variance across inferential replicates (1-2: minor, 3-4: some, 5-6: substantial). The 7th and 8th genes have all zeros to demonstrate `labelKeep`.

Usage

```
makeSimSwishData(m = 1000, n = 10, numReps = 20, null = FALSE)
```

Arguments

`m`	number of genes
`n`	number of samples
`numReps`	how many inferential replicates
`null`	logical, whether to make an all null dataset

Value

a SummarizedExperiment

Examples

```
library(SummarizedExperiment)
y <- makeSimSwishData()
assayNames(y)
```

`plotInfReps` *Plot inferential replicates for a gene or transcript*

Description

Plot inferential replicates for a gene or transcript

Usage

```
plotInfReps(y, idx, x, cov = NULL, cols.drk = c("dodgerblue",
  "goldenrod4"), cols.lgt = c("lightblue1", "goldenrod1"), xaxis)
```

Arguments

`y`	a SummarizedExperiment (see `swish`)
`idx`	the name or row number of the gene or transcript
`x`	the name of the condition variable
`cov`	the name of the covariate for adjustment
`cols.drk`	dark colors for the lines of the boxes
`cols.lgt`	light colors for the inside of the boxes
`xaxis`	logical, whether to label the sample numbers. default is `TRUE` if there are less than 30 samples

Value

nothing, a plot is displayed

Examples

```

y <- makeSimSwishData()
plotInfReps(y, 3, "condition")

y <- makeSimSwishData(n=40)
y$batch <- factor(rep(c(1,2,3,1,2,3),c(5,10,5,5,10,5)))
plotInfReps(y, 3, "condition", "batch")
```

`plotMASwish` *MA plot*

Description

MA plot

Usage

```
plotMASwish(y, alpha = 0.05, sigcolor = "blue", ...)
```

Arguments

`y`	a SummarizedExperiment (see `swish`)
`alpha`	the FDR threshold for coloring points
`sigcolor`	the color for the significant points
`...`	passed to plot

Value

nothing, a plot is displayed

Examples

```
y <- makeSimSwishData()
y <- scaleInfReps(y)
y <- labelKeep(y)
y <- swish(y, x="condition")
plotMASwish(y)
```

`readEDS` *readEDS - a utility function for quickly reading in Alevin's EDS format*

Description

readEDS - a utility function for quickly reading in Alevin's EDS format

Usage

```
readEDS(numOfGenes, numOfOriginalCells, countMatFilename)
```

Arguments

`numOfGenes`	number of genes
`numOfOriginalCells`	number of cells
`countMatFilename`	pointer to the EDS file, `quants_mat.gz`

Value

a genes x cells sparse matrix, of the class `dgCMatrix`

`scaleInfReps` *Scale inferential replicate counts*

Description

A helper function to scale the inferential replicates to the mean sequencing depth. The scaling takes into account a robust estimator of size factor (median ratio method is used). First, counts are corrected per row using the effective lengths (for gene counts, the average transcript lengths), then scaled per column to the geometric mean sequence depth, and finally are adjusted per-column up or down by the median ratio size factor to minimize systematic differences across samples.

Usage

```
scaleInfReps(y, lengthCorrect = TRUE, meanDepth = NULL, sfFun = NULL,
  minCount = 10, minN = 3, quiet = FALSE)
```

Arguments

`y`	a SummarizedExperiment with: `infReps` a list of inferential replicate count matrices, `counts` the estimated counts matrix, and `length` the effective lengths matrix
`lengthCorrect`	whether to use effective length correction (default is TRUE)
`meanDepth`	(optional) user can specify a different mean sequencing depth. By default the geometric mean sequencing depth is computed
`sfFun`	(optional) size factors function. An alternative to the median ratio can be provided here to adjust the scaledTPM so as to remove remaining library size differences
`minCount`	for internal filtering, the minimum count
`minN`	for internal filtering, the minimum sample size at `minCount`
`quiet`	display no messages

Value

a SummarizedExperiment with the inferential replicates as scaledTPM with library size already corrected (no need for further normalization)

Examples

```
y <- makeSimSwishData()
y <- scaleInfReps(y)
```

swish — *swish: SAMseq With Inferential Samples Helps*

Description

swish: SAMseq With Inferential Samples Helps

Usage

```
swish(y, x, cov = NULL, pair = NULL, interaction = FALSE,
  nperms = 30, estPi0 = FALSE, qvaluePkg = "qvalue", pc = 5,
  nRandomPairs = 30, fast = 1, quiet = FALSE)
```

Arguments

`y` a SummarizedExperiment containing the inferential replicate matrices of median-ratio-scaled TPM as assays ’infRep1’, ’infRep2’, etc.

`x` the name of the condition variable. A factor with two levels for a two group analysis (possible to adjust for covariate or matched samples, see next two arguments)

`cov` the name of the covariate for adjustment. If provided a stratified Wilcoxon in performed. Cannot be used with `pair`

`pair` the name of the pair variable, which should be the number of the pair. Can be an integer or factor. If specified, a signed rank test is used to build the statistic. All samples across `x` must be pairs if this is specified. Cannot be used with `cov`.

`interaction` logical, whether to perform a test of an interaction between `x` and `cov`. These are different than the other tests produced by the software, in that they focus on a difference in the log2 fold change across levels of `x` when comparing the two levels in `cov`. If `pair` is specified, this will perform a Wilcoxon rank sum test on the two groups of matched sample LFCs. If `pair` is not included, multiple random pairs of samples within the two groups are chosen, and again a Wilcoxon rank sum test compared the LFCs across groups.

`nperms` the number of permutations

`estPi0` logical, whether to estimate pi0

`qvaluePkg` character, which package to use for q-value estimation, `samr` or `qvalue`

`pc` pseudocount for finite estimation of `log2FC`, not used in calculation of test statistics, `locfdr` or `qvalue`

`nRandomPairs` the number of random pseudo-pairs (only used with `interaction=TRUE` and un-matched samples) to use to calculate the test statistic

`fast` an integer, toggles different methods based on speed (`fast=1` is default). ’0’ involves recomputing ranks of the inferential replicates for each permutation, ’1’ is roughly 10x faster by avoiding re-computing ranks for each permutation. The `fast` argument is only used/relevant for the following three experimental designs: (1) two group Wilcoxon, (2) stratified Wilcoxon, e.g. `cov` is specified, and (3) the paired interaction test, e.g. `pair` and `cov` are specified. For paired design and general interaction test, there are not fast/slow alternatives.

`quiet` display no messages

Value

a SummarizedExperiment with metadata columns added: the statistic (either a centered Wilcoxon Mann-Whitney or a signed rank statistic, aggregated over inferential replicates), a log2 fold change (the median over inferential replicates, and averaged over pairs or groups (if groups, weighted by sample size), the local FDR and q-value, as estimated by the samr package.

References

The citation for swish method is:

Anqi Zhu, Avi Srivastava, Joseph G Ibrahim, Rob Patro, Michael I Love "Nonparametric expression analysis using inferential replicate counts" Nucleic Acids Research (2019). https://doi.org/10.1093/nar/gkz622

The swish method builds upon the SAMseq method, and extends it by incorporating inferential uncertainty, as well as providing methods for additional experimental designs (see vignette).

For reference, the publication describing the SAMseq method is:

Jun Li and Robert Tibshirani "Finding consistent patterns: A nonparametric approach for identifying differential expression in RNA-Seq data" Stat Methods Med Res (2013). https://doi.org/10.1177/0962280211428386

Examples

```
library(SummarizedExperiment)
set.seed(1)
y <- makeSimSwishData()
y <- scaleInfReps(y)
y <- labelKeep(y)
y <- swish(y, x="condition")

# histogram of the swish statistics
hist(mcols(y)$stat, breaks=40, col="grey")
cols = rep(c("blue","purple","red"),each=2)
for (i in 1:6) {
  arrows(mcols(y)$stat[i], 20,
         mcols(y)$stat[i], 10,
         col=cols[i], length=.1, lwd=2)
}

# plot inferential replicates
plotInfReps(y, 1, "condition")
plotInfReps(y, 3, "condition")
plotInfReps(y, 5, "condition")
```

Index