Package 'SanityR'

May 26, 2026

Type Package

Title R/Bioconductor interface to the Sanity model gene expression analysis

Version 1.3.0

Description a Bayesian normalization procedure derived from first principles. Sanity estimates expression values and associated error bars directly from raw unique molecular identifier (UMI) counts without any tunable parameters.

License GPL (>= 3)

Encoding UTF-8

LazyData true

biocViews Software, GeneExpression, SingleCell, Normalization, Bayesian

BiocType Software

URL https://github.com/TeoSakel/SanityR

BugReports https://github.com/TeoSakel/SanityR/issues

Imports Rcpp, BiocGenerics, BiocParallel, MatrixGenerics, methods, S4Vectors, scuttle, SingleCellExperiment, SummarizedExperiment

LinkingTo Rcpp

RoxygenNote 7.3.2

Roxygen list(markdown = TRUE)

Suggests BiocStyle, knitr, rmarkdown, testthat (>= 3.0.0), scater, Rtsne

Config/testthat/edition 3

VignetteBuilder knitr

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

git_branch devel

git_last_commit a38d686

git_last_commit_date 2026-04-28

Repository Bioconductor 3.24

Date/Publication 2026-05-25

Author Teo Sakel [aut, cre] (ORCID: <https://orcid.org/0000-0001-9946-9498>), MCIU/AEI [fnd] (ROR: <https://ror.org/05r0vyz12>, DOI: 10.13039/501100011033)

Maintainer Teo Sakel <teo@intelligentbiodata.com>

Contents

`SanityR-package` *SanityR: R/Bioconductor interface to the Sanity model gene expression analysis*

Description

a Bayesian normalization procedure derived from first principles. Sanity estimates expression values and associated error bars directly from raw unique molecular identifier (UMI) counts without any tunable parameters.

Author(s)

Maintainer: Teo Sakel <teo@intelligentbiodata.com> (ORCID)

Other contributors:

- MCIU/AEI (ROR, DOI) [funder]

See Also

Useful links:

- https://github.com/TeoSakel/SanityR
- Report bugs at https://github.com/TeoSakel/SanityR/issues

`calculateSanityDistance` *Calculate the Sanity distance between samples*

Description

Calculates the expected squared Euclidean distance between two cells using a hierarchical model that shrinks noisy gene differences toward zero.

Usage

```
calculateSanityDistance(
  x,
  assay = "logcounts",
  assay.sd = "logcounts_sd",
  gene_sd = "sanity_activity_sd",
  gene_mu = "sanity_log_activity_mean",
  mu_sd = "sanity_log_activity_mean_sd",
  snr_cutoff = 1,
  nbin = 400L,
  subset.row = NULL,
  BPPARAM = bpparam()
)
```

Arguments

x	A SingleCellExperiment or SummarizedExperiment object which stores the results of the Sanity analysis.
assay	The name of the assay containing the log normalized counts matrix.
assay.sd	The name of the assay containing the standard deviation of the log-normalized counts
gene_sd	The name of the column in the rowData(x) that contains the standard deviation of the gene log-fold change.
gene_mu	The name of the column in the rowData(x) that contains the mean log activity of the genes.
mu_sd	The name of the column in the rowData(x) that contains the standard deviation of the mean log activity of the genes.
snr_cutoff	A numeric value indicating the minimum signal-to-noise ratio (SNR) to consider a gene.
nbin	Number of bins to use when calculating prior variance of the true distance.
subset.row	A vector of row indices or logical vector indicating which rows to use.
BPPARAM	A BiocParallelParam object specifying the parallelization strategy.

Details

Distance Calculation:

The method calculates the expected squared Euclidean distance between two cells, adjusting for uncertainty in gene expression estimates. For each gene g, the contribution to the squared distance between cells c and c' is:

$$\langle \Delta_g^2 \rangle = x_g^2 f_g^2(\alpha) + \eta_g^2 f_g(\alpha)$$

where:

- $x_g = \delta_{gc} - \delta_{gc'}$ (observed difference in Sanity's estimates)
- $\eta_g^2 = \epsilon_{gc}^2 + \epsilon_{gc'}^2$ (combined error variance)
- $f_g(\alpha) = \alpha v_g / (\alpha v_g + \eta_g^2)$ (shrinkage factor)

The shrinkage factor balances the observed gene expression differences x_g against their measurement uncertainty η_g. For genes with high-confidence estimates ($\eta_g \to 0$), it preserves the observed differences while for noisy genes ($\eta_g \gg 0$), it shrinks the result towards the common expected biological variation inferred from the data (αv_g).

The function returns the square root of the expected squared distance

$$\langle d \rangle = \sqrt{\sum_g \langle \Delta_g^2 \rangle}$$

Hyperparameter α:

The key hyperparameter α controls the prior distribution of Δ_g:

$$\Delta_g \sim N(0, \alpha v_g)$$

Thus:

- $\alpha = 0$: the 2 cells have identical expression states.
- $\alpha = 2$: the 2 cells have independent expression states.

The function implements numerical integration over α using a grid of `nbin` values to compute the expected value of the squared distance across all possible α.

Single to Noise Ratio (SNR):

Signal-to-Noise Ratio (SNR) is defined as the ratio of the variance of log-normalized counts across cells versus the mean variance (i.e. error bars) for each genes.

Value

A dist object containing the expected pairwise distances between cells.

Examples

```
sce <- simulate_branched_random_walk(N_gene = 500, N_path = 10, length_path = 10)
sce <- Sanity(sce)  # necessary step before computing distances
d <- calculateSanityDistance(sce)

# Downstream analysis and visualization
hc <- hclust(d, method = "ward.D2")
plot(hc)
```

`Sanity` *Estimate gene-level expression using the Sanity model*

Description

This function provides a user-friendly interface to the Sanity model for gene expression analysis.

Usage

```
Sanity(x, ...)

## S4 method for signature 'ANY'
Sanity(
  x,
  size.factors = NULL,
  vmin = 0.001,
  vmax = 50,
  nbin = 160L,
  a = 1,
  b = 0,
  BPPARAM = bpparam()
)

## S4 method for signature 'SummarizedExperiment'
Sanity(x, ..., assay.type = "counts", name = "logcounts", subset.row = NULL)

## S4 method for signature 'SingleCellExperiment'
Sanity(x, size.factors = sizeFactors(x), ...)
```

Arguments

x	A numeric matrix of counts where features are rows and columns are cells. Alternatively, a SummarizedExperiment or a SingleCellExperiment containing such counts.
...	For the generic, further arguments to pass to each method. For the SummarizedExperiment method, further arguments to pass to the ANY method. For the SingleCellExperiment method, further arguments to pass to the SummarizedExperiment method.
size.factors	A numeric vector of cell-specific size factors. Alternatively NULL, in which case the size factors are computed from x.
vmin	The minimum value for the gene-level variance (must be > 0).
vmax	The maximum value for the gene-level variance.
nbin	Number of variance bins to use.
a, b	Gamma prior parameter (see Details).
BPPARAM	A BiocParallelParam object specifying whether the calculations should be parallelized.
assay.type	A string specifying the assay of x containing the count matrix.
name	String containing an assay name for storing the output normalized values.
subset.row	A vector specifying the subset of rows of x to process.

Details

The method models gene activity using a Bayesian framework, assuming a Gamma prior on expression and integrating over cell-level variability. It returns posterior estimates for mean expression (mu), cell-specific deviations (delta), and their variances, as well as expression variance (var).

Expected log-normalized counts are computed by combining mean expression and cell-specific log-fold changes. The *standard deviation* of log-counts is computed by summing the variances of the components.

If no `size.factors` are provided, they are assumed all equal so that all cells have the same library size `mean(colSums(x))`.

Gamma Prior::

The model adopts a Bayesian framework by placing a Gamma prior `Gamma(a, b)` over the gene activity, where `a` is the shape and `b` the rate parameter, respectively. This allows for flexible regularization and uncertainty modeling. The posterior likelihood is estimated by integrating over possible values of the variance in expression.

Intuitively:

- `a` acts as a pseudo-count added to the total count of the gene.
- `b` acts as a pseudo-count penalizing deviations from the average. expression — i.e., it regularizes the total number of UMIs that differ from the expected value.

Setting `a = 1` and `b = 0` corresponds to an uninformative (uniform) prior, which was used in the original Sanity model publication.

Value

For `matrix`-like object it returns a named list with the following elements (symbols as defined in the Supplementary Text of the publication):

mu Posterior mean of log expression across cells μ_g.

var_mu Posterior variance of the mean expression $(\delta\mu_g)^2$.

var Posterior variance of expression across cells $\langle v_g \rangle$.

delta Vector of log fold-changes for each cell relative to δ_{gc}.

var_delta Posterior variance of the cell-level fold-changes ϵ^2_{gc}.

lik Normalized likelihood across the evaluated variance grid $P\left(v_g \mid n_g\right)$ for diagnostics.

If called on a SingleCellExperiment or SummarizedExperiment it appends the following columns to the `rowData` slot:

sanity_log_activity_mean `mu`

sanity_log_activity_mean_sd `sqrt(var_mu)`

sanity_activity_sd `sqrt(var)`

and appends the following assays (assuming `name = "logcounts"`):

assay(x, "logcounts") `mu + delta`

assay(x, "logcounts_sd") `sqrt(var_mu + var_delta)`

References

Breda, J., Zavolan, M., & van Nimwegen, E. (2021). Bayesian inference of gene expression states from single-cell RNA-seq data. *Nature Biotechnology*, 39, 1008–1016. https://doi.org/10.1038/s41587-021-00875-x

Examples

```
library(SingleCellExperiment)

sce <- simulate_independent_cells(N_cell = 500, N_gene = 100)

# Standard Sanity normalization
sce_norm <- Sanity(sce)
logcounts(sce_norm)[1:5,1:5]

# Using size factors
sf <- colSums(counts(sce))
sizeFactors(sce) <- sf / mean(sf)
sce_norm2 <- Sanity(sce)
logcounts(sce_norm2)[1:5,1:5]
```

`simulate_sce` *Simulate SingleCellExperiment Datasets with Independent or Branched Gene Expression Patterns*

Description

These functions generate synthetic single-cell RNA-seq datasets based the methods described in original Sanity publication for benchmarking.

Usage

```
simulate_independent_cells(
  cell_size = NULL,
  gene_size = NULL,
  N_cell = NULL,
  N_gene = NULL,
  ltq_var_rate = 0.5
)

simulate_branched_random_walk(
  cell_size = NULL,
  gene_size = NULL,
  N_gene = NULL,
  ltq_var_rate = 0.5,
  N_path = 149L,
  length_path = 13L
)
```

Arguments

`cell_size`	Optional vector of real or simulated total UMI counts per cell. If `NULL`, defaults to values from the *Baron et al.* study.
`gene_size`	Optional vector of real or simulated total UMI counts per gene. If `NULL`, defaults to values from the *Baron et al.* study.
`N_cell`	Integer. Number of cells to simulate. (For `simulate_branched_random_walk` is equal to `N_path * length_path`). If `NULL` inferred from `cell_size`.

N_gene	Integer. Number of genes to simulate. If NULL, inferred from gene_size.
ltq_var_rate	Rate parameter for the exponential distribution used to simulate per-gene variance (default: 0.5).
N_path	(Only for simulate_branched_random_walk) Number of branching paths (default: 149).
length_path	(Only for simulate_branched_random_walk) Number of steps (cells) per path (default: 13).

Details

- simulate_independent_cells: gene expression values are generated independently for each cell. This results in uncorrelated expression patterns across the dataset.
- simulate_branched_random_walk: cells follow a **branched random walk** through gene expression space, producing correlated gene expression patterns that reflect pseudo-temporal differentiation trajectories.

Value

A SingleCellExperiment object containing:

- assays$counts: Simulated UMI count matrix.
- assays$logFC: Simulated log fold-changes for each gene-cell pair.
- rowData: Gene-level metadata including ltq_mean and ltq_var.
- colData: Cell-level metadata including predecessor for simulated_branched_random_walk.

References

A Single-Cell Transcriptomic Map of the Human and Mouse Pancreas Reveals Inter- and Intracell Population Structure Baron, Maayan et al. *Cell Systems*, Volume 3, Issue 4, 346 - 360.e4 https://doi.org/10.1016/j.cels.2016.08.011

Examples

```
# Simulate dataset with independent gene expression
sce_indep <- simulate_independent_cells(N_cell = 100, N_gene = 50)

# Simulate dataset with a branched random walk trajectory
sce_branch <- simulate_branched_random_walk(N_path = 20, length_path = 5, N_gene = 50)
```

Index