

Package ‘scFastDE’

July 21, 2026

Type Package

Title Fast Donor-Weighted Pseudo-Bulk Differential Expression for
scRNA-seq

Version 0.99.3

Description scFastDE provides fast, donor-weighted pseudo-bulk differential expression analysis for multi-donor single-cell RNA-seq experiments. Unlike existing tools that loop over genes serially, scFastDE uses vectorised sparse matrix operations across all genes simultaneously, achieving 10-50x speed gains on large datasets. Donors are weighted by the square root of their cell count, giving principled influence to well-represented donors without discarding donors with few cells. Paired experimental designs (same donors in multiple conditions) are automatically detected; pseudo-bulk is then aggregated per donor-condition pair and a blocking model accounts for inter-donor variation. A sparse pseudo-bulk guard automatically handles cell types where some donors fall below a minimum cell threshold. All functions operate natively on SingleCellExperiment objects.

License MIT + file LICENSE

URL <https://github.com/SubhadipJana1409/scFastDE>

BugReports <https://github.com/SubhadipJana1409/scFastDE/issues>

biocViews SingleCell, DifferentialExpression, StatisticalMethod,
Transcriptomics, GeneExpression, Sequencing, ATACSeq,
CellBasedAssays, WorkflowStep

Encoding UTF-8

Depends R (>= 4.6.0)

Imports SingleCellExperiment, SummarizedExperiment, S4Vectors,
BiocParallel, Matrix, limma, methods, stats, ggplot2, rlang,
utils

Suggests BiocStyle, knitr, rmarkdown, testthat (>= 3.0.0), scuttle,
withr

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

git_url <https://git.bioconductor.org/packages/scFastDE>

git_branch devel

git_last_commit 543071e
git_last_commit_date 2026-06-04
Repository Bioconductor 3.24
Date/Publication 2026-07-20
Author Subhadip Jana [aut, cre] (ORCID:
 <<https://orcid.org/0009-0003-7860-2853>>)
Maintainer Subhadip Jana <subhadipjana1409@gmail.com>

Contents

scFastDE-package	2
deTable	3
donorWeights	4
fastDE	5
fastPseudobulk	7
FDEResult	9
FDEResult-class	10
filterSparseDonors	10
params,FDEResult-method	11
plotDEResults	12
pseudobulk	13
show,FDEResult-method	14
Index	15

scFastDE-package	<i>scFastDE: Fast Donor-Weighted Pseudo-Bulk DE for scRNA-seq</i>
------------------	---

Description

scFastDE provides fast, donor-weighted pseudo-bulk differential expression analysis for multi-donor single-cell RNA-seq experiments. Unlike existing tools that test genes serially in a loop, scFastDE uses vectorised sparse matrix operations across all genes simultaneously, giving 10-50x speed gains on large datasets (30k+ genes, 100k+ cells).

Key innovations

- Vectorised gene testing** All genes are tested simultaneously via matrix algebra — no per-gene loop.
- Donor cell-count weighting** Each sample’s pseudo-bulk profile is weighted by `sqrt(n_cells)`, giving principled influence to well-represented samples.
- Paired design auto-detection** When the same donors appear in multiple conditions, fastDE automatically aggregates per donor-condition pair and uses a blocking model ($\sim \emptyset + \text{condition} + \text{donor}$) to account for inter-donor variation.
- Sparse pseudo-bulk guard** Donors with fewer than `min_cells` cells per cell type are flagged and optionally removed before aggregation.

Main functions

- `filterSparseDonors` Remove donors below minimum cell threshold per cell type.
- `fastPseudobulk` Build donor-weighted pseudo-bulk matrix from a `SingleCellExperiment`. Supports per-donor or per-donor-condition aggregation.
- `fastDE` Run vectorised DE across all genes. Auto-detects paired vs unpaired designs.
- `plotDEResults` Volcano plot of DE results.

Author(s)

Maintainer: Subhadip Jana <subhadipjana1409@gmail.com> ([ORCID](#))

Authors:

- Subhadip Jana <subhadipjana1409@gmail.com> ([ORCID](#))

References

Crowell HL et al. (2020). muscat detects subpopulation-specific state transitions from multi-sample multi-condition single-cell transcriptomics data. *Nature Communications*, 11, 6077.

See Also

Useful links:

- <https://github.com/SubhadipJana1409/scFastDE>
- Report bugs at <https://github.com/SubhadipJana1409/scFastDE/issues>

deTable

Accessor for DE table in a FDEResult

Description

Returns the per-gene differential expression statistics `DataFrame` from a `FDEResult` object.

Usage

```
deTable(x, ...)

## S4 method for signature 'FDEResult'
deTable(x, ...)
```

Arguments

`x` A `FDEResult` object.
`...` Additional arguments (not used).

Value

A `DataFrame` with columns `logFC`, `AveExpr`, `t`, `P.Value`, `adj.P.Val`, and `B`.

Examples

```
library(S4Vectors)
de <- DataFrame(logFC = c(1.2, -0.5), P.Value = c(0.001, 0.5),
               adj.P.Val = c(0.01, 0.8))
pb <- matrix(rpois(20, 10), nrow = 2, ncol = 10)
rownames(pb) <- paste0("Gene", 1:2)
colnames(pb) <- paste0("Donor", 1:10)
obj <- FDEResult(de, pb, setNames(sqrt(1:10), paste0("Donor", 1:10)))
deTable(obj)
```

donorWeights

*Accessor for donor weights in a FDEResult***Description**

Returns the per-donor weight vector from a FDEResult object.

Usage

```
donorWeights(x, ...)

## S4 method for signature 'FDEResult'
donorWeights(x, ...)
```

Arguments

x A FDEResult object.

... Additional arguments (not used).

Value

A named numeric vector of donor weights.

Examples

```
library(S4Vectors)
de <- DataFrame(logFC = c(1.2, -0.5), P.Value = c(0.001, 0.5),
               adj.P.Val = c(0.01, 0.8))
pb <- matrix(rpois(20, 10), nrow = 2, ncol = 10)
rownames(pb) <- paste0("Gene", 1:2)
colnames(pb) <- paste0("Donor", 1:10)
obj <- FDEResult(de, pb, setNames(sqrt(1:10), paste0("Donor", 1:10)))
donorWeights(obj)
```

fastDE

*Fast Vectorised Differential Expression***Description**

Runs differential expression analysis across all genes simultaneously using a weighted limma-voom linear model. Unlike tools that loop over genes serially, fastDE passes the entire pseudo-bulk matrix to limma, which uses LAPACK routines to fit all gene models in one vectorised call.

The function automatically detects whether the experimental design is **paired** (same donors in multiple conditions) or **unpaired** (each donor in one condition only) and builds the appropriate linear model:

- **Paired:** pseudo-bulk is aggregated per donor-condition pair, and a $\sim \theta + \text{condition} + \text{donor}$ model accounts for donor-level variation.
- **Unpaired:** pseudo-bulk is aggregated per donor, and a $\sim \theta + \text{condition}$ model is used.

Usage

```
fastDE(
  sce,
  donor = NULL,
  cell_type = NULL,
  condition = NULL,
  target_type = NULL,
  contrast = NULL,
  min_cpm = 1,
  min_donors = 2L,
  min_cells = 10L,
  BPPARAM = SerialParam()
)
```

Arguments

sce	A SingleCellExperiment object with raw counts in assay(sce, "counts").
donor	A character(1) naming the donor column in colData(sce).
cell_type	A character(1) naming the cell type column in colData(sce).
condition	A character(1) naming the condition column in colData(sce) (e.g. "treatment", "disease"). Must have exactly two unique values.
target_type	A character(1) specifying which cell type to test. Must be present in colData(sce)[[cell_type]].
contrast	A character(1) specifying the contrast in limma syntax, e.g. "treat - ctrl". If NULL, the second level minus the first level is used. Default: NULL.
min_cpm	A numeric(1) minimum CPM for a gene to be considered expressed. Default: 1.
min_donors	An integer(1) minimum number of samples in which a gene must exceed min_cpm. Default: 2.
min_cells	An integer(1) minimum cells per donor before pseudo-bulk aggregation. Passed to filterSparseDonors . Default: 10.
BPPARAM	A BiocParallelParam object. Default: SerialParam().

Details

The analysis pipeline is:

1. Filter lowly-expressed genes (CPM > min_cpm in at least min_donors samples).
2. Apply `limma::voom` with sample cell-count weights.
3. Fit a weighted linear model with the appropriate design.
4. Extract the contrast of interest with `limma::makeContrasts`.
5. Apply empirical Bayes moderation with `limma::eBayes`.
6. Return all results as a [FDEResult](#) object.

Value

A [FDEResult](#) object containing:

- `deTable`: per-gene statistics (logFC, AveExpr, t, P.Value, adj.P.Val, B).
- `pseudobulk`: the aggregated count matrix.
- `donorWeights`: the per-sample `sqrt(n_cells)` weights.
- `params`: the analysis parameters.

See Also

[fastPseudobulk](#), [filterSparseDonors](#), [plotDEResults](#)

Examples

```
library(SingleCellExperiment)

set.seed(42)
n_genes <- 200
counts <- matrix(rpois(n_genes * 60, 8), nrow = n_genes, ncol = 60)
rownames(counts) <- paste0("Gene", seq_len(n_genes))
colnames(counts) <- paste0("Cell", seq_len(60))

# Inject DE signal into first 10 genes for treatment group
counts[1:10, 31:60] <- counts[1:10, 31:60] * 3L

sce <- SingleCellExperiment(assays = list(counts = counts))
sce$donor <- rep(paste0("D", 1:6), each = 10)
sce$cell_type <- "Tcell"
sce$condition <- rep(c("ctrl", "treat"), each = 30)

result <- fastDE(sce,
  donor      = "donor",
  cell_type  = "cell_type",
  condition  = "condition",
  target_type = "Tcell",
  min_cells  = 5)

result
head(deTable(result))
```

fastPseudobulk

*Build Donor-Weighted Pseudo-Bulk Profiles***Description**

Aggregates single-cell counts into pseudo-bulk profiles for a specified cell type, using vectorised sparse matrix operations.

When condition is provided, aggregation is performed per donor-condition pair (one column per sample, e.g. D1_ctrl, D1_stim). This is the correct approach for paired experimental designs where the same donors contribute cells under multiple conditions.

When condition is NULL (default), aggregation is performed per donor only (backward-compatible behaviour for unpaired designs).

Each sample's weight equals $\sqrt{n_{\text{cells}}}$, giving more influence to well-represented samples while not discarding those with fewer cells.

Usage

```
fastPseudobulk(
  sce,
  donor = NULL,
  cell_type = NULL,
  target_type = NULL,
  condition = NULL,
  assay_name = "counts"
)
```

Arguments

sce	A SingleCellExperiment object with raw counts in assay(sce, "counts").
donor	A character(1) naming the donor column in colData(sce).
cell_type	A character(1) naming the cell type column in colData(sce).
target_type	A character(1) specifying which cell type to aggregate. Must be a value present in colData(sce)[[cell_type]].
condition	A character(1) naming the condition column in colData(sce). When provided, aggregation is done per donor-condition pair (required for paired designs). Default: NULL (aggregate per donor only).
assay_name	A character(1) name of the assay to aggregate. Default: "counts".

Details

Pseudo-bulk aggregation sums the raw counts for all cells belonging to each donor (or donor-condition pair) within a given cell type:

$$PB_{g,s} = \sum_{c \in \text{sample}_s, \text{type}_t} X_{g,c}$$

Sample weights are then computed as:

$$w_s = \sqrt{n_{s,t}}$$

where $n_{s,t}$ is the number of cells for sample s in cell type t .

The aggregation uses sparse matrix column-sums grouped by sample, avoiding a slow R-level loop.

Value

A list with elements:

`pseudobulk` A matrix of aggregated counts, rows = genes, columns = samples.

`sample_weights` A named numeric vector of per-sample weights ($\sqrt{n_cells}$).

`sample_ncells` A named integer vector giving the raw cell count per sample.

`sample_info` A data.frame with columns `sample_id`, `donor`, and (if applicable) `condition` — one row per pseudo-bulk sample.

For backward compatibility, `donor_weights` and `donor_ncells` are also provided as aliases when `condition` is NULL.

See Also

[filterSparseDonors](#), [fastDE](#)

Examples

```
library(SingleCellExperiment)

set.seed(42)
counts <- matrix(rpois(500 * 60, 5), nrow = 500, ncol = 60)
rownames(counts) <- paste0("Gene", seq_len(500))
colnames(counts) <- paste0("Cell", seq_len(60))
sce <- SingleCellExperiment(assays = list(counts = counts))
sce$donor <- rep(paste0("D", 1:6), each = 10)
sce$cell_type <- "Tcell"
sce$condition <- rep(c("ctrl", "treat"), each = 30)

# Unpaired (aggregate by donor only)
pb <- fastPseudobulk(sce, donor = "donor",
                    cell_type = "cell_type",
                    target_type = "Tcell")

dim(pb$pseudobulk)

# Paired (aggregate by donor x condition)
pb2 <- fastPseudobulk(sce, donor = "donor",
                    cell_type = "cell_type",
                    target_type = "Tcell",
                    condition = "condition")

dim(pb2$pseudobulk)
pb2$sample_info
```

FDEResult	<i>Constructor for FDEResult</i>
-----------	----------------------------------

Description

Create a new FDEResult object.

Usage

```
FDEResult(deTable, pseudobulk, donorWeights, params = list())
```

Arguments

deTable	A DataFrame of per-gene DE statistics.
pseudobulk	A matrix of pseudo-bulk counts.
donorWeights	A named numeric vector of donor weights.
params	A list of analysis parameters.

Value

A FDEResult object.

Examples

```
library(S4Vectors)
de <- DataFrame(
  logFC      = c(1.2, -0.5, 0.8),
  P.Value    = c(0.001, 0.5, 0.02),
  adj.P.Val  = c(0.01, 0.8, 0.05)
)
pb <- matrix(rpois(30, 10), nrow = 3, ncol = 10)
rownames(pb) <- paste0("Gene", 1:3)
colnames(pb) <- paste0("Donor", 1:10)
obj <- FDEResult(
  deTable      = de,
  pseudobulk   = pb,
  donorWeights = setNames(sqrt(1:10), paste0("Donor", 1:10)),
  params       = list(cell_type = "T_cells", condition = "group")
)
obj
```

FDEResult-class	<i>FDEResult: Fast DE Result Container</i>
-----------------	--

Description

An S4 class to store the output of [fastDE](#). Slots hold per-gene DE statistics, the pseudo-bulk matrix used, donor weights, and the analysis parameters.

Slots

`deTable` A `DataFrame` with one row per gene containing columns `logFC`, `AveExpr`, `t`, `P.Value`, `adj.P.Val`, and `B`.

`pseudobulk` A matrix of pseudo-bulk counts (genes x donors) used as input to the DE model.

`donorWeights` A named numeric vector of per-donor weights (sqrt of cell count) used in the linear model.

`params` A list of analysis parameters including `cell_type`, `condition`, `donor`, `min_cells`.

filterSparseDonors	<i>Filter Donors with Too Few Cells per Cell Type</i>
--------------------	---

Description

Removes or flags donors that have fewer than `min_cells` cells for a given cell type. When forming pseudo-bulk profiles, donors with very few cells produce highly variable, unreliable aggregated counts that can inflate false-positive DE calls. This function provides a principled pre-filtering step before calling [fastPseudobulk](#).

Usage

```
filterSparseDonors(
  sce,
  donor = NULL,
  cell_type = NULL,
  min_cells = 10L,
  action = c("remove", "flag")
)
```

Arguments

<code>sce</code>	A SingleCellExperiment object.
<code>donor</code>	A character(1) naming the donor column in <code>colData(sce)</code> .
<code>cell_type</code>	A character(1) naming the cell type column in <code>colData(sce)</code> .
<code>min_cells</code>	A integer(1) minimum number of cells a donor must have for a given cell type to be retained. Default: 10.
<code>action</code>	A character(1), either "remove" (drop sparse cells from the SCE) or "flag" (add a logical column <code>scFastDE_sparse</code> to <code>colData</code>). Default: "remove".

Details

For each combination of cell_type level and donor level, the function counts the number of cells in the sce and compares it to min_cells. Donors below the threshold are either removed (when action = "remove") or flagged in a new colData column (when action = "flag").

As a rule of thumb, min_cells = 10 is a reasonable default for common cell types. For rare cell types (< 1% frequency), consider lowering to min_cells = 5.

Value

The input sce with sparse donor-cell type combinations either removed or flagged depending on action. When action = "flag", a column scFastDE_sparse is added to colData: TRUE means the cell belongs to a sparse donor-cell type combination.

See Also

[fastPseudobulk](#), [fastDE](#)

Examples

```
library(SingleCellExperiment)

set.seed(42)
counts <- matrix(rpois(500 * 80, 5), nrow = 500, ncol = 80)
rownames(counts) <- paste0("Gene", seq_len(500))
colnames(counts) <- paste0("Cell", seq_len(80))
sce <- SingleCellExperiment(assays = list(counts = counts))
sce$donor <- rep(paste0("D", 1:8), each = 10)
sce$cell_type <- rep(c("Tcell", "Bcell"), times = 40)

# Remove donor-cell type combos with fewer than 5 cells
sce_filtered <- filterSparseDonors(sce, donor = "donor",
                                   cell_type = "cell_type",
                                   min_cells = 5)

ncol(sce_filtered)
```

params,FDEResult-method

Accessor for analysis parameters in a FDEResult

Description

Returns the analysis parameter list from a FDEResult object.

Usage

```
## S4 method for signature 'FDEResult'
params(x, ...)
```

Arguments

x A FDEResult object.
... Additional arguments (not used).

Value

A list of analysis parameters.

Examples

```
library(S4Vectors)
de <- DataFrame(logFC = c(1.2, -0.5), P.Value = c(0.001, 0.5),
               adj.P.Val = c(0.01, 0.8))
pb <- matrix(rpois(20, 10), nrow = 2, ncol = 10)
rownames(pb) <- paste0("Gene", 1:2)
colnames(pb) <- paste0("Donor", 1:10)
obj <- FDEResult(
  de,
  pb,
  setNames(sqrt(1:10), paste0("Donor", 1:10)),
  params = list(cell_type = "T_cells", condition = "group")
)
params(obj)
```

plotDEResults

Volcano Plot of Differential Expression Results

Description

Produces a volcano plot from a [FDEResult](#) object, with genes coloured by significance and optionally labelled. Points are coloured by whether they exceed the `lfc_thresh` and `fdr_thresh` thresholds.

Usage

```
plotDEResults(
  result,
  fdr_thresh = 0.05,
  lfc_thresh = 1,
  top_n = 10L,
  point_size = 1,
  point_alpha = 0.6,
  colours = c(up = "#E24B4A", down = "#378ADD", ns = "#888780")
)
```

Arguments

<code>result</code>	A FDEResult object from fastDE .
<code>fdr_thresh</code>	A <code>numeric(1)</code> FDR threshold for significance. Default: 0.05.
<code>lfc_thresh</code>	A <code>numeric(1)</code> absolute log2 fold-change threshold. Default: 1.
<code>top_n</code>	An <code>integer(1)</code> number of top significant genes to label by name. Default: 10.
<code>point_size</code>	A <code>numeric(1)</code> point size. Default: 1.
<code>point_alpha</code>	A <code>numeric(1)</code> point transparency. Default: 0.6.
<code>colours</code>	A named character vector of colours for "up", "down", and "ns" (not significant). Default: <code>c(up = "#E24B4A", down = "#378ADD", ns = "#888780")</code> .

Value

A ggplot2 object.

See Also

[fastDE](#), [FDEResult](#)

Examples

```
library(SingleCellExperiment)

set.seed(42)
n_genes <- 200
counts <- matrix(rpois(n_genes * 60, 8), nrow = n_genes, ncol = 60)
rownames(counts) <- paste0("Gene", seq_len(n_genes))
colnames(counts) <- paste0("Cell", seq_len(60))
counts[1:10, 31:60] <- counts[1:10, 31:60] * 3L

sce <- SingleCellExperiment(assays = list(counts = counts))
sce$donor <- rep(paste0("D", 1:6), each = 10)
sce$cell_type <- "Tcell"
sce$condition <- rep(c("ctrl", "treat"), each = 30)

result <- fastDE(sce, donor = "donor", cell_type = "cell_type",
                 condition = "condition", target_type = "Tcell",
                 min_cells = 5)
plotDEResults(result)
```

pseudobulk

Accessor for pseudo-bulk matrix in a FDEResult

Description

Returns the pseudo-bulk count matrix from a `FDEResult` object.

Usage

```
pseudobulk(x, ...)

## S4 method for signature 'FDEResult'
pseudobulk(x, ...)
```

Arguments

`x` A `FDEResult` object.
`...` Additional arguments (not used).

Value

A matrix of pseudo-bulk counts (genes x donors).

Examples

```
library(S4Vectors)
de <- DataFrame(logFC = c(1.2, -0.5), P.Value = c(0.001, 0.5),
               adj.P.Val = c(0.01, 0.8))
pb <- matrix(rpois(20, 10), nrow = 2, ncol = 10)
rownames(pb) <- paste0("Gene", 1:2)
colnames(pb) <- paste0("Donor", 1:10)
obj <- FDEResult(de, pb, setNames(sqrt(1:10), paste0("Donor", 1:10)))
pseudobulk(obj)
```

show,FDEResult-method *Show method for FDEResult*

Description

Prints a compact summary of a FDEResult object.

Usage

```
## S4 method for signature 'FDEResult'
show(object)
```

Arguments

object A FDEResult object.

Value

Invisibly returns object.

Index

BiocParallelParam, [5](#)

deTable, [3](#)
deTable, FDEResult-method (deTable), [3](#)
donorWeights, [4](#)
donorWeights, FDEResult-method
(donorWeights), [4](#)

fastDE, [3](#), [5](#), [8](#), [10–13](#)
fastPseudobulk, [3](#), [6](#), [7](#), [10](#), [11](#)
FDEResult, [6](#), [9](#), [12](#), [13](#)
FDEResult-class, [10](#)
filterSparseDonors, [3](#), [5](#), [6](#), [8](#), [10](#)

params, FDEResult-method, [11](#)
plotDEResults, [3](#), [6](#), [12](#)
pseudobulk, [13](#)
pseudobulk, FDEResult-method
(pseudobulk), [13](#)

scFastDE (scFastDE-package), [2](#)
scFastDE-package, [2](#)
show, FDEResult-method, [14](#)
SingleCellExperiment, [5](#), [7](#), [10](#)