Package 'spillR'

May 21, 2024

Type Package

Title Spillover Compensation in Mass Cytometry Data

Version 1.1.0

Description Channel interference in mass cytometry can cause spillover and may result in miscounting of protein markers. We develop a nonparametric finite mixture model and use the mixture components to estimate the probability of spillover. We implement our method using expectation-maximization to fit the mixture model.

biocViews FlowCytometry, ImmunoOncology, MassSpectrometry, Preprocessing, SingleCell, Software, StatisticalMethod, Visualization, Regression

License LGPL-3

Encoding UTF-8

LazyData false

Config/testthat/edition 3

RoxygenNote 7.2.3

Imports dplyr, tibble, tidyselect, stats, ggplot2, tidyr, spatstat.geom, S4Vectors, parallel

Depends R (>= 4.3.0), SummarizedExperiment, CATALYST

Suggests knitr, rmarkdown, cowplot, testthat (>= 3.0.0), BiocStyle, hexbin

VignetteBuilder knitr

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

git_branch devel

git_last_commit d4dea65

git_last_commit_date 2024-04-30

Repository Bioconductor 3.20

Date/Publication 2024-05-20

Author Marco Guazzini [aut, cre] (<https://orcid.org/0009-0007-8111-5772>),
Alexander G. Reisach [aut],
Sebastian Weichwald [aut] (<https://orcid.org/0000-0003-0169-7244>),
Christof Seiler [aut] (<https://orcid.org/0000-0001-8802-3642>)

Maintainer Marco Guazzini <m.guazzini@student.maastrichtuniversity.nl>

Contents

`compCytof` *Compute spillover probability and correct for spillover*

Description

Compute spillover probability and correct for spillover

Usage

```
compCytof(
  sce,
  sce_bead,
  marker_to_barc,
  impute_value,
  overwrite = FALSE,
  n_cores = 1,
  naive = FALSE
)
```

Arguments

`sce`	`SingleCellExperiment` for the real cells
`sce_bead`	`SingleCellExperiment` for the bead experiment
`marker_to_barc`	Table that maps the marker to the barcode in the beads experiment
`impute_value`	Imputed value for counts that are declared as spillover
`overwrite`	logical; if TRUE data are overwritten if FALSE data are saved in new columns
`n_cores`	Number of computing cores
`naive`	logical; if TRUE use the naive version

Value

A SingleCellExperiment object

Examples

```
library(CATALYST)
library(dplyr)
bc_key <- c(139, 141:156, 158:176)
sce_bead <- prepData(ss_exp)
sce_bead <- assignPrelim(sce_bead, bc_key, verbose = FALSE)
sce_bead <- applyCutoffs(estCutoffs(sce_bead))
sce_bead <- computeSpillmat(sce_bead)
data(mp_cells, package = "CATALYST")
sce <- prepData(mp_cells)
marker_to_barc <- rowData(sce_bead)[, c("channel_name", "is_bc")] |>
    as_tibble() |>
    filter(is_bc == TRUE) |>
    mutate(barcode = bc_key) |>
    select(marker = channel_name, barcode)
spillR::compCytof(sce, sce_bead, marker_to_barc, impute_value = NA)
```

---

compensate *Compute spillover probability and correct for spillover*

Description

Compute spillover probability and correct for spillover

Usage

```
compensate(
  tb_real,
  tb_bead,
  target_marker,
  spillover_markers,
  impute_value = NA,
  n_iter = 1000
)
```

Arguments

tb_real	Data frame or tibble with proteins counts of real experiment
tb_bead	Data frame or tibble with proteins counts of bead experiment
target_marker	Marker name in real experiment
spillover_markers	Marker names in bead experiment
impute_value	Value for counts that are declared as spillover
n_iter	Maximum number of EM steps

Value

A list of class `spillr` containing

`tb_compensate`	corrected real cells
`tb_spill_prob`	probability curve
`convergence`	covergence table of EM algorithm
`tb_real`	input real cells
`tb_bead`	input bead cells
`target_marker`	input marker in real experiment
`spillover_markers`	input markers in bead experiment

`compensate_naive` *Compute spillover probability and correct for spillover from beads only*

Description

Compute spillover probability and correct for spillover from beads only

Usage

```
compensate_naive(
  tb_real,
  tb_bead,
  target_marker,
  spillover_markers,
  impute_value = NA
)
```

Arguments

`tb_real`	Data frame or tibble with proteins counts of real experiment
`tb_bead`	Data frame or tibble with proteins counts of bead experiment
`target_marker`	Marker name in real experiment
`spillover_markers`	Marker names in bead experiment
`impute_value`	Value for counts that are declared as spillover

Value

A list of class `spillr` containing

`tb_compensate`	corrected real cells
`tb_spill_prob`	probability curve
`convergence`	covergence table of EM algorithm
`tb_real`	input real cells
`tb_bead`	input bead cells
`target_marker`	input marker in real experiment
`spillover_markers`	input markers in bead experiment

`generate_bead` *Generate dataset for vignettes and simulation studies*

Description

Generate dataset for vignettes and simulation studies

Usage

```
generate_bead()
```

Value

`tibble` data frame

Examples

```
set.seed(23)
generate_bead()
```

`generate_real` *Generate dataset for vignettes and simulation studies*

Description

Generate dataset for vignettes and simulation studies

Usage

```
generate_real()
```

Value

tibble data frame

Examples

```
set.seed(23)
generate_real()
```

---

plotDiagnostics *Compute spillover probability and correct for spillover*

Description

Compute spillover probability and correct for spillover

Usage

```
plotDiagnostics(sce, ch)
```

Arguments

sce	A SingleCellExperiment object
ch	Character string specifying the channel to plot

Value

A list of ggplot2 plots

Examples

```
library(CATALYST)
library(dplyr)
bc_key <- c(139, 141:156, 158:176)
sce_bead <- prepData(ss_exp)
sce_bead <- assignPrelim(sce_bead, bc_key, verbose = FALSE)
sce_bead <- applyCutoffs(estCutoffs(sce_bead))
sce_bead <- computeSpillmat(sce_bead)
data(mp_cells, package = "CATALYST")
sce <- prepData(mp_cells)
marker_to_barc <- rowData(sce_bead)[, c("channel_name", "is_bc")] |>
    as_tibble() |>
    filter(is_bc == TRUE) |>
    mutate(barcode = bc_key) |>
    select(marker = channel_name, barcode)
sce <- spillR::compCytof(sce, sce_bead, marker_to_barc, impute_value = NA)
plotDiagnostics(sce, "Yb173Di")
```

`tfm` *Variance stabilizing transform of counts*

Description

Variance stabilizing transform of counts

Usage

```
tfm(x)
```

Arguments

`x`	Raw count

Value

A transformed count

Index