Package ‘sccomp’

October 18, 2022

Title Robust Outlier-aware Estimation of Composition and Heterogeneity for Single-cell Data

Version 1.0.0

Description A robust and outlier-aware method for testing differential tissue composition from single-cell data. This model can infer changes in tissue composition and heterogeneity, and can produce realistic data simulations based on any existing dataset. This model can also transfer knowledge from a large set of integrated datasets to increase accuracy further.

License GPL-3

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.2

Biarch true

Depends R (>= 4.1.0)

Imports methods, Rcpp (>= 0.12.0), RcppParallel (>= 5.0.1), rstantools (>= 2.1.1), rstan (>= 2.18.1), SeuratObject, SingleCellExperiment, parallel, dplyr, tidyr, purrr, magrittr, rlang, tibble, boot, lifecycle, stats, tidyselect, utils, ggplot2, ggrepel, patchwork, forcats, readr, scales, stringr

Suggests BiocStyle, testthat (>= 3.0.0), markdown, ggplot2, knitr, tidyseurat, tidySingleCellExperiment

Enhances furrr, extraDistr

LinkingTo BH (>= 1.66.0), Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0), RcppParallel (>= 5.0.1), rstan (>= 2.18.1), StanHeaders (>= 2.18.0)

SystemRequirements GNU make

VignetteBuilder knitr

RdMacros lifecycle

biocViews ImmunoOncology, Normalization, Sequencing, RNASeq, Software, GeneExpression, Transcriptomics, SingleCell, Clustering

LazyDataCompression xz

Config/testthat/edition 3

URL https://github.com/stemangiola/sccomp

BugReports https://github.com/stemangiola/sccomp/issues

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

git_branch RELEASE_3_15

git_last_commit dcc1b26

git_last_commit_date 2022-04-26

Date/Publication 2022-10-18

Author Stefano Mangiola [aut, cre]

Maintainer Stefano Mangiola <mangiolastefano@gmail.com>

R topics documented:

sccomp-package 2
counts_obj 3
multi_beta_glm 3
plot_summary 4
replicate_data 5
sccomp_glm 6
sce_obj 9
seurat_obj 9
simulate_data 10

Index **12**

---

sccomp-package *The 'sccomp' package.*

Description

A DESCRIPTION OF THE PACKAGE

References

Stan Development Team (2020). RStan: the R interface to Stan. R package version 2.21.2. https://mc-stan.org

counts_obj *counts_obj*

Description

Example data set containing cell counts per cell cluster

Usage

```
data(counts_obj)
```

Format

A tidy data frame.

multi_beta_glm *multi_beta_glm main*

Description

This function runs the data modelling and statistical test for the hypothesis that a cell_type includes outlier biological replicate.

Usage

```
multi_beta_glm(
  .data,
  formula = ~1,
  .sample,
  check_outliers = FALSE,
  approximate_posterior_inference = TRUE,
  cores = detect_cores(),
  seed = sample(1e+05, 1)
)
```

Arguments

`.data`	A tibble including a cell_type name column | sample name column | read counts column | covariate columns | Pvaue column | a significance column
`formula`	A formula. The sample formula used to perform the differential cell_type abundance analysis
`.sample`	A column name as symbol. The sample identifier
`check_outliers`	A boolean. Whether to check for outliers before the fit.

approximate_posterior_inference
: A boolean. Whether the inference of the joint posterior distribution should be approximated with variational Bayes. It confers execution time advantage.

cores
: An integer. How many cored to be used with parallel calculations.

seed
: An integer. Used for development and testing purposes

Value

A nested tibble `tbl` with cell_type-wise information: `sample wise data` | plot | `ppc samples failed` | `exposure deleterious outliers`

`plot_summary` *plot_summary*

Description

This function plots a summary of the results of the model.

Usage

```
plot_summary(.data, significance_threshold = 0.025)
```

Arguments

.data
: A tibble including a cell_group name column | sample name column | read counts column | covariate columns | Pvalue column | a significance column

significance_threshold
: A real. FDR threshold for labelling significant cell-groups.

Value

A `ggplot`

Examples

```
data("counts_obj")
library(dplyr)

estimate =
  sccomp_glm(
  counts_obj ,
   ~ type, ~1, sample, cell_group, count,
    approximate_posterior_inference = "all",
    check_outliers = FALSE,
    cores = 1
  )

estimate |> plot_summary()
```

replicate_data *replicate_data*

Description

This function replicates counts from a real-world dataset.

Usage

```
replicate_data(.data, number_of_draws = 1, mcmc_seed = sample(1e+05, 1))
```

Arguments

`.data`	A tibble including a cell_group name column | sample name column | read counts column | covariate columns | Pvalue column | a significance column
`number_of_draws`	An integer. How may copies of the data you want to draw from the model joint posterior distribution.
`mcmc_seed`	An integer. Used for Markov-chain Monte Carlo reproducibility. By default a random number is sampled from 1 to 999999. This itself can be controlled by set.seed()

Value

A nested tibble `tbl` with cell_group-wise statistics

Examples

```
data("counts_obj")

  sccomp_glm(
  counts_obj ,
   ~ type, ~1,  sample, cell_group, count,
    approximate_posterior_inference = "all",
    check_outliers = FALSE,
    cores = 1
  ) |>

  replicate_data()
```

sccomp_glm *sccomp_glm main*

Description

The function for linear modelling takes as input a table of cell counts with three columns containing a cell-group identifier, sample identifier, integer count and the covariates (continuous or discrete). The user can define a linear model with an input R formula, where the first covariate is the factor of interest. Alternatively, sccomp accepts single-cell data containers (Seurat, SingleCellExperiment44, cell metadata or group-size). In this case, sccomp derives the count data from cell metadata.

Usage

```
sccomp_glm(
  .data,
  formula_composition = ~1,
  formula_variability = ~1,
  .sample,
  .cell_group,
  .count = NULL,
  prior_mean_variable_association = list(intercept = c(5, 2), slope = c(0, 0.6),
    standard_deviation = c(5, 8)),
  check_outliers = TRUE,
  bimodal_mean_variability_association = FALSE,
  cores = detectCores(),
  percent_false_positive = 5,
  approximate_posterior_inference = "outlier_detection",
  test_composition_above_logit_fold_change = 0.2,
  verbose = FALSE,
  noise_model = "multi_beta_binomial",
  exclude_priors = FALSE,
  use_data = TRUE,
  mcmc_seed = sample(1e+05, 1),
  max_sampling_iterations = 20000,
  pass_fit = TRUE
)
```

Arguments

`.data` A tibble including a cell_group name column | sample name column | read counts column (optional depending on the input class) | covariate columns.

`formula_composition` A formula. The formula describing the model for differential abundance, for example ~treatment.

`formula_variability` A formula. The formula describing the model for differential variability, for example ~treatment. In most cases, if differentially variability is of interest, the

formula should only include the factor of interest as a large anount of data is needed to define variability depending to each covariates.

`.sample` A column name as symbol. The sample identifier

`.cell_group` A column name as symbol. The cell_group identifier

`.count` A column name as symbol. The cell_group abundance (read count). Used only for data frame count output. The variable in this column should be of class integer.

`prior_mean_variable_association`
A list of the form list(intercept = c(5, 2), slope = c(0, 0.6), standard_deviation = c(5,8)). Where for intercept and slope parameters, we specify mean and standard deviation, while for standard deviation, we specify shape and rate. This is used to incorporate prior knowledge about the mean/variability association of cell-type proportions.

`check_outliers` A boolean. Whether to check for outliers before the fit.

`bimodal_mean_variability_association`
A boolean. Whether to model the mean-variability as bimodal, as often needed in the case of single-cell RNA sequencing data, and not usually for CyTOF and microbiome data. The plot summary_plot()$credible_intervals_2D can be used to assess whether the bimodality should be modelled.

`cores` An integer. How many cored to be used with parallel calculations.

`percent_false_positive`
A real between 0 and 100 non included. This used to identify outliers with a specific false positive rate.

`approximate_posterior_inference`
A boolean. Whether the inference of the joint posterior distribution should be approximated with variational Bayes. It confers execution time advantage.

`test_composition_above_logit_fold_change`
A positive integer. It is the effect threshold used for the hypothesis test. A value of 0.2 correspond to a change in cell proportion of 10% for a cell type with baseline proportion of 50%. That is, a cell type goes from 45% to 50%. When the baseline proportion is closer to 0 or 1 this effect thrshold has consistent value in the logit uncontrained scale.

`verbose` A boolean. Prints progression.

`noise_model` A character string. The two noise models available are multi_beta_binomial (default) and dirichlet_multinomial.

`exclude_priors` A boolean. Whether to run a prior-free model, for benchmarking purposes.

`use_data` A booelan. Whether to sun the model data free. This can be used for prior predictive check.

`mcmc_seed` An integer. Used for Markov-chain Monte Carlo reproducibility. By default a random number is sampled from 1 to 999999. This itself can be controlled by set.seed()

`max_sampling_iterations`
An integer. This limit the maximum number of iterations in case a large dataset is used, for limiting the computation time.

`pass_fit` A boolean. Whether to pass the Stan fit as attribute in the output. Because the Stan fit can be very large, setting this to FALSE can be used to lower the memory imprint to save the output.

Value

A nested tibble `tbl`, with the following columns

- cell_group - column including the cell groups being tested
- parameter - The parameter being estimated, from the design matrix dscribed with the input formula_composition and formula_variability
- c_lower - lower (2.5%) quantile of the posterior distribution for a composition (c) parameter.
- c_effect - mean of the posterior distribution for a composition (c) parameter.
- c_upper - upper (97.5%) quantile of the posterior distribution fo a composition (c) parameter.
- c_pH0 - Probability of the null hypothesis (no difference) for a composition (c). This is not a p-value.
- c_FDR - False-discovery rate of the null hypothesis (no difference) for a composition (c).
- v_lower - (optional, present if variability is modelled dependent on covariates) lower (2.5%) quantile of the posterior distribution for a variability (v) parameter
- v_effect - (optional, present if variability is modelled dependent on covariates) mean of the posterior distribution for a variability (v) parameter
- v_upper - (optional, present if variability is modelled dependent on covariates) upper (97.5%) quantile of the posterior distribution for a variability (v) parameter
- v_pH0 - (optional, present if variability is modelled dependent on covariates) Probability of the null hypothesis (no difference) for a variability (v). This is not a p-value.
- v_FDR - (optional, present if variability is modelled dependent on covariates) False-discovery rate of the null hypothesis (no difference), for a variability (v).

Examples

```
data("counts_obj")

estimate =
  sccomp_glm(
  counts_obj ,
   ~ type,
   ~1,
   sample,
   cell_group,
   count,
    approximate_posterior_inference = "all",
    check_outliers = FALSE,
    cores = 1
  )
```

sce_obj *sce_obj*

Description

Example SingleCellExperiment data set. SingleCellExperiment data objects can be directly used with sccomp_glm function.

Usage

```
data(sce_obj)
```

Format

A SingeCellExperiment object. SingeCellExperiment data objects can be directly used with sccomp_glm function.

seurat_obj *seurat_obj*

Description

Example Seurat data set. Seurat data objects can be directly used with sccomp_glm function.

Usage

```
data(seurat_obj)
```

Format

A Seurat object

simulate_data *simulate_data*

Description

This function simulates counts from a linear model.

Usage

```
simulate_data(
  .data,
  .estimate_object,
  formula_composition,
  .sample = NULL,
  .cell_group = NULL,
  .coefficients = NULL,
  variability_multiplier = 5,
  number_of_draws = 1,
  mcmc_seed = sample(1e+05, 1)
)
```

Arguments

`.data` A tibble including a cell_group name column | sample name column | read counts column | covariate columns | Pvalue column | a significance column

`.estimate_object` The result of sccomp_glm execution. This is used for sampling from real-data properties.

`formula_composition` A formula. The sample formula used to perform the differential cell_group abundance analysis

`.sample` A column name as symbol. The sample identifier

`.cell_group` A column name as symbol. The cell_group identifier

`.coefficients` The column names for coefficients, for example, c(b_0, b_1)

`variability_multiplier` A real scalar. This can be used for artificially increasing the variability of the simulation for benchmarking purposes.

`number_of_draws` An integer. How may copies of the data you want to draw from the model joint posterior distribution.

`mcmc_seed` An integer. Used for Markov-chain Monte Carlo reproducibility. By default a random number is sampled from 1 to 999999. This itself can be controlled by set.seed()

Value

A nested tibble `tbl` with cell_group-wise statistics

Examples

```
data("counts_obj")
library(dplyr)

estimate =
  sccomp_glm(
  counts_obj ,
   ~ type, ~1,  sample, cell_group, count,
    approximate_posterior_inference = "all",
    check_outliers = FALSE,
    cores = 1
  )

# Set coefficients for cell_groups. In this case all coefficients are 0 for simplicity.
counts_obj = counts_obj |> mutate(b_0 = 0, b_1 = 0)

# Simulate data
simulate_data(counts_obj, estimate, ~type, sample, cell_group, c(b_0, b_1))
```

Index

∗ **datasets**
    counts_obj, 3
    sce_obj, 9
    seurat_obj, 9

counts_obj, 3

multi_beta_glm, 3

plot_summary, 4

replicate_data, 5

sccomp *(sccomp-package)*, 2
sccomp-package, 2
sccomp_glm, 6
sce_obj, 9
seurat_obj, 9
simulate_data, 10