Package 'Tweedieverse'

August 8, 2024

Title Differential analysis of omics data based on the Tweedie distribution

Version 0.0.1

Date 2021-03-03

Description A toolkit for differential analysis of omics data. Implements a range of statistical methodology based on the Tweedie distribution. Unlike traditional single-omics tools, Tweedieverse is technology-agnostic and can be applied to both count and continuous measurements arising from diverse high-throughput technologies (e.g. transcript abundances from bulk and single-cell RNA-Seq studies in the form of UMI counts or non-UMI counts, microbiome taxonomic and functional profiles in the form of counts or relative abundances, and compound abundance levels or peak intensities from metabolomics and other mass spectrometry-based experiments, among others). The software includes multiple analysis methods (e.g. self-adaptive, zero-inflated, and non-zero-inflated statistical models) as well as multiple customization options such as the inclusion of random effects and multiple covariates along with several data exploration capabilities and visualization modules in a unified estimation umbrella.

Depends R (>= 3.6)

Imports cplm, statmod, glmmTMB, pbapply, logging, parallel, dplyr, tweedie, ggplot2, grid, pheatmap, bbmle, parameters, cowplot

Suggests data.table, knitr

License MIT + file LICENSE

LazyData TRUE

RoxygenNote 7.3.2

VignetteBuilder knitr

Contents

`Tweedieverse` *Differential analysis of multi-omics data using Tweedie GLMs*

Description

Fit a per-feature Tweedie generalized linear model to omics features.

Usage

```
Tweedieverse(
  input_features,
  input_metadata = NULL,
  output,
  assay_name = "counts",
  abd_threshold = 0,
  prev_threshold = 0.1,
  var_threshold = 0,
  entropy_threshold = 0,
  base_model = "CPLM",
  link = "log",
  fixed_effects = NULL,
  random_effects = NULL,
  cutoff_ZSCP = 0.3,
  criteria_ZACP = "BIC",
  adjust_offset = TRUE,
  scale_factor = NULL,
  max_significance = 0.05,
  correction = "BH",
  standardize = TRUE,
  cores = 1,
  optimizer = "nlminb",
  na.action = na.exclude,
  plot_heatmap = FALSE,
  plot_scatter = FALSE,
  heatmap_first_n = 50,
  reference = NULL
)
```

Arguments

input_features	A tab-delimited input file or an R data frame of features (can be in rows/columns) and samples (or cells). Samples are expected to have matching names with input_metadata. input_features can also be an object of class SummarizedExperiment or SingleCellExperiment that contains the expression or abundance matrix and other metadata; the assays slot contains the expression or abundance matrix and is named "counts". This matrix should have one row for each feature and one sample for each column. The colData slot should contain a data frame with one row per sample and columns that contain metadata for each sample. Additional information about the experiment can be contained in the metadata slot as a list.
input_metadata	A tab-delimited input file or an R data frame of metadata (rows/columns). Samples are expected to have matching sample names with input_features. This file is ignored when input_features is a SummarizedExperiment or a SingleCellExperiment object with colData containing the same information.
output	The output folder to write results.
assay_name	If the input is provided as one of the accepted Bioconductor objects (e.g., SummarizedExperiment, RangedSummarizedExperiment, SingleCellExperiment, or TreeSummarizedExperiment), this argument selects the name of the assay slot in the input object that contains the omics measurements.

abd_threshold If prevalence-abundance filtering is desired, only features that are present (or detected) in at least prev_threshold percent of samples at abd_threshold minimum abundance (read count or proportion) are retained. Default value for abd_threshold is 0.0. To disable prevalence-abundance filtering, set abd_threshold = -Inf.

prev_threshold If prevalence-abundance filtering is desired, only features that are present (or detected) in at least prev_threshold percent of samples at abd_threshold minimum abundance (read count or proportion) are retained. Default value for prev_threshold is 0.1.

var_threshold If variance filtering is desired, only features that have variances greater than var_threshold are retained. This step is done after the prevalence-abundance filtering. Default value for var_threshold is 0.0 (i.e. no variance filtering).

entropy_threshold
If entropy-based filtering is desired for metadata, only features that have entropy greater than entropy_threshold are retained. Default value for entropy_threshold is 0.0 (i.e. no entropy filtering).

base_model The per-feature base model. Default is "CPLM". Must be one of "CPLM", "ZICP", "ZSCP", or "ZACP".

link A specification of the GLM link function. Default is "log". Must be one of "log", "identity", "sqrt", or "inverse".

fixed_effects Metadata variable(s) describing the fixed effects coefficients.

random_effects Metadata variable(s) describing the random effects part of the model.

cutoff_ZSCP For base_model = "ZSCP", the cutoff to stratify features for adaptive ZI modeling based on sparsity (zero-inflation proportion). Default is 0.3. Must be between 0 and 1.

criteria_ZACP For base_model = "ZACP", the criteria to select the best fitting model per feature. The possible options are 'AIC' and BIC' (default). More criteria will be supported in a future release.

adjust_offset If TRUE (default), an offset term will be included as the logarithm of scale_factor.

scale_factor Name of the numerical variable containing library size (for non-normalized data) or scale factor (for normalized data) across samples to be included as an offset in the base model (when adjust_offset = TRUE). If not found in metadata, defaults to the sample-wise total sums, unless adjust_offset = FALSE.

max_significance
The q-value threshold for significance. Default is 0.05.

correction The correction method for computing the q-value (see p.adjust for options, default is 'BH').

standardize Should continuous metadata be standardized? Default is TRUE. Bypassed for categorical variables.

cores An integer that indicates the number of R processes to run in parallel. Default is 1.

optimizer The optimization routine to be used for estimating the parameters of the Tweedie model. Possible choices are "nlminb" (the default, see nlminb), "bobyqa" (bobyqa), and "L-BFGS-B" (optim). Ignored for random effects modeling which uses an alternative Template Model Builder (TMB) approach (glmmTMB).

na.action How to handle missing values? See na.action. Default is na.exclude.

plot_heatmap Logical. If TRUE (default is FALSE), generate a heatmap of the (top heatmap_first_n) significant associations.

plot_scatter Logical. If TRUE (default is FALSE), generate scatter/box plots of individual associations.

heatmap_first_n In heatmap, plot top N features with significant associations (default is 50).

reference The factor to use as a reference for a variable with more than two levels provided as a string of 'variable,reference' semi-colon delimited for multiple variables (default is NULL).

Value

A data frame containing coefficient estimates, p-values, and q-values (multiplicity-adjusted p-values) are returned.

Author(s)

Himel Mallick, <him4004@med.cornell.edu>

Examples

```
## Not run:

#############################################################################
# Example 1 - Differential Abundance Analysis of Synthetic Microbiome Counts #
#############################################################################

#######################################
# Install and Load Required Libraries #
#######################################

library(devtools)
devtools::install_github('biobakery/sparseDOSSA@varyLibSize')
library(sparseDOSSA)
library(stringi)

######################
# Specify Parameters #
######################

n.microbes <- 200 # Number of Features
n.samples <- 100 # Number of Samples
spike.perc <- 0.02 # Percentage of Spiked-in Bugs
spikeStrength<-"20" # Effect Size

############################
# Specify Binary Metadata #
############################

n.metadata <- 1
UserMetadata<-as.matrix(rep(c(0,1), each=n.samples/2))
UserMetadata<-t(UserMetadata) # Transpose

#################################################
# Spiked-in Metadata (Which Metadata to Spike-in) #
#################################################

Metadatafrozenidx<-1
```

```
spikeCount<-as.character(length(Metadatafrozenidx))
significant_metadata<-paste('Metadata', Metadatafrozenidx, sep='')

##########################################
# Generate SparseDOSSA Synthetic Abundances #
##########################################

DD<-sparseDOSSA::sparseDOSSA(number_features = n.microbes,
number_samples = n.samples,
UserMetadata=UserMetadata,
Metadatafrozenidx=Metadatafrozenidx,
datasetCount = 1,
spikeCount = spikeCount,
spikeStrength = spikeStrength,
noZeroInflate=TRUE,
percent_spiked=spike.perc,
seed = 1234)

##############################
# Gather SparseDOSSA Outputs #
##############################

sparsedossa_results <- as.data.frame(DD$OTU_count)
rownames(sparsedossa_results)<-sparsedossa_results$X1
sparsedossa_results<-sparsedossa_results[-1,-1]
colnames(sparsedossa_results)<-paste('Sample', 1:ncol(sparsedossa_results), sep='')
data<-as.matrix(sparsedossa_results[-c((n.metadata+1):(2*n.microbes+n.metadata)),])
data<-data.matrix(data)
class(data) <- "numeric"
truth<-c(unlist(DD$truth))
truth<-truth[!stri_detect_fixed(truth,":")]
truth<-truth[(5+n.metadata):length(truth)]
truth<-as.data.frame(truth)
significant_features<-truth[seq(1,
(as.numeric(spikeCount)+1)*(n.microbes*spike.perc), (as.numeric(spikeCount)+1)),]
significant_features<-as.vector(significant_features)

###################
# Extract Features #
###################

features<-as.data.frame(t(data[-c(1:n.metadata),]))

###################
# Extract Metadata #
###################

metadata<-as.data.frame(data[1,])
colnames(metadata)<-rownames(data)[1]

##############################
# Mark True Positive Features #
##############################

wh.TP = colnames(features) %in% significant_features
colnames(features)<-paste("Feature", 1:n.microbes, sep = "")
newname = paste0(colnames(features)[wh.TP], "_TP")
```

```
colnames(features)[wh.TP] <- newname;
colnames(features)[grep('TP', colnames(features))]

####################
# Run Tweedieverse #
####################

###################
# Default options #
###################

CPLM <-Tweedieverse(
features,
metadata,
output = './demo_output/CPLM') # Assuming demo_output exists

################################################
# User-defined prevalence-abundance filtering #
################################################

ZICP<-Tweedieverse(
features,
metadata,
output = './demo_output/ZICP', # Assuming demo_output exists
base_model = 'ZICP',
abd_threshold = 0.0,
prev_threshold = 0.2)

###################################
# User-defined variance filtering  #
###################################

sds<-apply(features, 2, sd)
var_threshold = median(sds)/2
ZSCP<-Tweedieverse(
features,
metadata,
output = './demo_output/ZSCP', # Assuming demo_output exists
base_model = 'ZSCP',
var_threshold = var_threshold)

##################
# Multiple cores #
##################

ZACP<-Tweedieverse(
features,
metadata,
output = './demo_output/ZACP', # Assuming demo_output exists
base_model = 'ZACP',
cores = 4)

#############################################################################
# Example 2 - Multivariable Association on HMP2 Longitudinal Microbiomes #
#############################################################################

######################
```

```
# HMP2 input_features Analysis #
########################

#############
# Load input_features #
#############

library(data.table)
input_features <- fread("https://raw.githubusercontent.com/biobakery/Maaslin2/master/inst/extdata/HMP2_taxo
input_metadata <-fread("https://raw.githubusercontent.com/biobakery/Maaslin2/master/inst/extdata/HMP2_metac

###############
# Format data #
###############

library(tibble)
features<- column_to_rownames(input_features, 'ID')
metadata<- column_to_rownames(input_metadata, 'ID')

#############
# Fit Model #
#############

library(Tweedieverse)
HMP2 <- Tweedieverse(
features,
metadata,
output = './demo_output/HMP2', # Assuming demo_output exists
fixed_effects = c('diagnosis', 'dysbiosisnonIBD','dysbiosisUC','dysbiosisCD', 'antibiotics', 'age'),
random_effects = c('site', 'subject'),
base_model = 'CPLM',
adjust_offset = FALSE, # No offset as the values are relative abundances
cores = 8, # Make sure your computer has the capability
standardize = FALSE,
reference = c('diagnosis,nonIBD'))


## End(Not run)
```

Index