Package ‘zinck’

December 16, 2024

Title Zero-Inflated Compositional Knockoff Filter

Version 0.0.0.9000

Description Zinck exploits a zero-inflated variant of the Latent Dirichlet Allocation (LDA) model
to generate valid knockoffs that capture the key characteristics of microbiome data -
mainly its compositional nature and high sparsity. It exhibits the properties of
simultaneous variable selection and FDR control to identify microbial biomarkers.
This package provides an implementation of zinck, which is trained either using the
Automatic Differentiation Variational Inference (ADVI) algorithm or using a collapsed
Gibbs sampler, facilitating variable selection for both continuous as well as binary outcomes.

License MIT

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Imports dplyr,
reshape2,
knockoff,
glmnet,
randomForest,
caret,
rstan (>= 2.21.7),
stats,
fitdistrplus,
ggplot2,
MLmetrics,
phyloseq,
GUniFrac,
kosel,
gridExtra,
zinLDA

Suggests knitr,
rmarkdown

VignetteBuilder knitr

LazyData true

NeedsCompilation no

Author Soham Ghosh [aut, cre],
Zhengzheng Tang [ctb],
Guanhua Chen [ctb]

Maintainer Soham Ghosh <sghosh39@wisc.edu>

Depends R (>= 3.5.0)

R topics documented:

`zinck-package` *zinck: A Package for Using Zero-Inflated Compositional Knockoffs for Microbial Variable Selection*

Description

Description: zinck is a novel knockoff-based framework tailored for microbiome data exploiting a flexible generative model. It can properly capture the zero-inflation and complex dependence structure among microbes, enjoying the property of simultaneous variable selection and FDR control.

Details

Main Functions:

`fit.zinck` Fits the zero-inflated hierarchical model to a count matrix using ADVI or Gibbs sampling.

`generateKnockoff` Generates a knockoff copy of the original matrix once the posterior estimates of the latent parameters are obtained.

`optimal_k` Finds the optimal number of clusters minimizing the Jensen-Shannon Divergence.

`log_normalize` Performs log-normalization of a given matrix.

`draw_heatmap` Creates a heatmap for a microbial sample taxa matrix.

`zinck.filter` Performs FDR-controlled variable selection by fitting a glmnet or a Random Forest model.

Datasets:

`count.genus` Genus level CRC data.

`count` Species level CRC data.

For a complete list of functions and datasets, see the `INDEX` section.

Author(s)

Maintainer: Soham Ghosh <`sghosh39@wisc.edu`>

Other contributors:

- Zhengzheng Tang <`ztang2@wisc.edu`> [contributor]
- Guanhua Chen <`gchen25@wisc.edu`> [contributor]

`count` *CRC data (species level)*

Description

This dataset contains data from a meta-analysis of five geographically and technically diverse fecal shotgun metagenomic studies of colorectal cancer (CRC, n = 574). All raw sequencing data across studies were reprocessed using the same bioinformatics pipeline for taxonomic profiling. The 5 studies involve subjects from 5 different countries. The sample size of the studies is 109, 127, 120, 114, and 104; and the number of cases and controls are roughly balanced in each study. This data-set contains p = 849 species found among all the investigated metagenomic CRC studies.

Usage

```
data(count)
```

Format

An integer matrix with 574 rows and 849 columns.

Source

Zeller Lab

`count.genus` *CRC data (genus level)*

Description

This dataset contains data from a meta-analysis of five geographically and technically diverse fecal shotgun metagenomic studies of colorectal cancer (CRC, n = 574). All raw sequencing data across studies were reprocessed using the same bioinformatics pipeline for taxonomic profiling. The 5 studies involve subjects from 5 different countries. The sample size of the studies is 109, 127, 120, 114, and 104; and the number of cases and controls are roughly balanced in each study. This data-set contains p = 133 distinct genera found among all the investigated metagenomic CRC studies.

Usage

```
data(count.genus)
```

Format

An integer matrix with 574 rows and 133 columns.

Source

Zeller Lab

draw_heatmap *Draw Heatmap for Microbial Sample Taxa Matrix*

Description

This function creates a heatmap for a given microbial sample taxa matrix. It is specifically designed for visualizing abundance patterns across different samples and taxa in microbiome studies. The function applies an arcsinh transformation to the data for normalization and better visualization of abundance patterns, especially useful in handling highly skewed microbiome data.

Usage

```
draw_heatmap(X, title = "")
```

Arguments

X	A numeric matrix representing microbial sample taxa data, where rows represent samples and columns represent taxa.
title	An optional title for the heatmap

Details

The heatmap is generated using `ggplot2` and `reshape2` packages, with taxa on the x-axis and samples on the y-axis. The color intensity in the heatmap represents the arcsinh-transformed abundance of each taxa in each sample.

Value

A ggplot object representing the heatmap. This can be further customized or directly plotted.

Examples

```
# Create a sample matrix representing microbial sample taxa data
mat <- matrix(runif(20), nrow = 5)

# Draw heatmap
draw_heatmap(mat)
```

fit.zinck *fit.zinck*

Description

Fit the Zinck model to the data using either ADVI or Gibbs sampling methods.

Usage

```
fit.zinck(
  X,
  num_clusters,
  method = c("ADVI", "Gibbs"),
  seed = NULL,
  init_values = NULL,
  alpha_param = 0.1,
  boundary_correction = FALSE,
  importance_resampling = FALSE,
  elbo_samples = 500,
  prior_ZIGD = FALSE
)
```

Arguments

X	An OTU matrix with dimensions $D \times p$.
num_clusters	An integer specifying the number of clusters.
method	A character string, either "ADVI" or "Gibbs", specifying the method to fit the model.
seed	An integer used to set the seed for reproducibility.
init_values	A list of initial values for the ADVI algorithm. This parameter is optional and should be used only with the ADVI method. If NULL, the algorithm uses default initialization.
alpha_param	A positive real. The symmetric smoothed Dirichlet parameter for the cluster distributions, default=0.1
boundary_correction	A logical value. If TRUE, it adds and subtracts small numbers so that the log-likelihood doesn't blow up at the boundary points 0 or 1, default = FALSE.
importance_resampling	Logical scalar (defaulting to FALSE) indicating whether to do importance resampling to adjust the draws at the optimum to be more like draws from the posterior distribution.
elbo_samples	A positive integer. The number of samples for Monte Carlo estimate of ELBO (objective function), defaulting to 100. (ELBO stands for "the evidence lower bound".)name description
prior_ZIGD	A logical value. If TRUE and method is "ADVI", the model will impose Gamma priors on the ZIGD hyperparameters.

Value

A list containing the posterior estimates of beta and theta.

References

Kucukelbir, A., Tran, D., Ranganath, R., Gelman, A., and Blei, D. (2017). Automatic Differentiation Variational Inference. Journal of Machine Learning Research 18(14).

Deek, R., and Li, H. (2021). A Zero-Inflated Latent Dirichlet Allocation Model for Microbiome Studies. Frontiers in Genetics 11.

Examples

```
# Generate a random OTU matrix
X <- matrix(rpois(20, lambda = 5), nrow = 5)

# Fit the Zinck model using ADVI with default initialization
fit <- fit.zinck(X, num_clusters = 3, method = "ADVI", alpha_param=1)

# Fit the Zinck model using ADVI with custom initial values
init_vals <- list(theta = array(runif(15), dim = c(5, 3)),
                  zeta = array(runif(12), dim = c(3, 4)))
fit_with_init <- fit.zinck(X, num_clusters = 3, method = "ADVI", init_values = init_vals, alpha_param=1)
```

generateKnockoff *Generate Knockoff Copy of Microbial Sample Data*

Description

This function generates a knockoff copy of microbial sample data given a matrix `X` and matrices `Theta` and `Beta`. The function adjusts the column structure of `Beta` to match `X`, generates samples based on `Theta` and `Beta`, and then compiles these into a knockoff count matrix.

Usage

```
generateKnockoff(X, Theta, Beta, seed = NULL)
```

Arguments

X	A numeric matrix representing original microbial sample data.
Theta	A numeric matrix representing cluster mixing probabilities.
Beta	A numeric matrix representing feature proportions for each cluster.
seed	An optional integer seed for reproducibility of random generation.

Value

A numeric matrix representing the knockoff copy of the microbial sample data.

Examples

```
X <- matrix(runif(40), nrow = 10)
colnames(X) <- paste("Taxa", 1:ncol(X))
Theta <- matrix(runif(30), nrow = 10)
Beta <- matrix(runif(20), nrow = 5)
knockoff_data <- generateKnockoff(X, Theta, Beta)
```

`log_normalize` *Log-Normalization for Microbiome Compositional Data*

Description

This function performs log-normalization on a given matrix, typically used in microbiome data analysis. In microbiome studies, data are often compositional and contain many zero counts. Log-normalization, with the addition of a pseudo-count, is a standard approach to handle zeros and maintain the compositional nature of the data. This function takes a numeric matrix, adds a pseudo-count to zero values, and then applies a log transformation, preserving the relative proportions in the data.

Usage

```
log_normalize(X)
```

Arguments

`X`	A numeric matrix to log-normalize.

Details

The function first adds a pseudo-count of 0.5 to zero entries to handle zeros in the data. It then divides each count by the total counts in its row (sample) to make the data compositional. Finally, it applies a natural logarithm transformation to the normalized data. #'

Value

A matrix of the same dimensions as `X`, with each element being the log-normalized value of the corresponding element in `X`. The base of the logarithm used for normalization is e (natural logarithm).

Examples

```
# Create a sample matrix
mat <- matrix(1:4, nrow = 2)

# Perform log-normalization
log_normalized_mat <- log_normalize(mat)
```

`optimal_k` *Optimal Number of Clusters based on Jensen-Shannon Divergence*

Description

This function identifies the optimal number of clusters for fitting a zinck model using Jensen-Shannon Divergence. The function fits the model for various values of clusters and calculates the average Jensen-Shannon Divergence for each, returning the number of clusters that maximizes this value.

Usage

```
optimal_k(X, kmin, kmax, seed_list = NULL)
```

Arguments

X	A OTU matrix with dimensions $D \times p$.
kmin	Numeric; the minimum number of clusters to be considered.
kmax	Numeric; the maximum number of clusters to be considered.
seed_list	List of numeric values; seeds for reproducibility for each k value, default is NULL.

Value

The optimal number of clusters, K, that maximizes the average Jensen-Shannon Divergence.

Examples

```
## Not run:
data("count.genus.RData")
dcount <- count.genus[,order(decreasing=T,colSums(count.genus,na.rm=T))]
X <- dcount
result <- optimal_k(X, kmin=8, kmax=11, seed_list=list(1,11,1,1))
print(result)

## End(Not run)
```

zinck.filter *Zinck Filter for Variable Selection*

Description

Performs variable selection by fitting the augmented set of features to the response, using a glmnet or Random Forest model

Usage

```
zinck.filter(
  X,
  X_tilde,
  Y,
  model,
  fdr = 0.1,
  offset = 1,
  seed = NULL,
  ntrees = 1000,
  tune_mtry = FALSE,
  mtry = NULL,
  metric = NULL,
  rftuning = FALSE,
  conservative = FALSE
)
```

Arguments

`X` An OTU matrix with dimensions $D \times p$.

`X_tilde` A knockoff matrix corresponding to X with the same dimensions.

`Y` The response variable, either continuous or binary.

`model` A string; the model to be used ("glmnet" or "Random Forest").

`fdr` Numeric; the false discovery rate. Default is 0.1

`offset` Numeric; either 0 or 1. Default is 1

`seed` Numeric; the seed for reproducibility.

`ntrees` Numeric; the number of trees for Random Forest. Default is 1000.

`tune_mtry` Logical; whether to tune mtry in Random Forest. Default is FALSE.

`mtry` Numeric; the number of variables randomly sampled as candidates at each split in Random Forest. Default is NULL (auto-tuned).

`metric` A string; the importance metric for the Random Forest, can be ("Accuracy" or "Gini"). Default is "Accuracy".

`rftuning` Logical; whether to fit a fast or a tuned implementation of a Random Forest. Default is FALSE.

`conservative` Logical; whether to use a conservative knockoff threshold or not. Default is FALSE.

Value

A vector of selected variables at a target false discovery rate

Index