# Package 'IFAA'

April 25, 2023

**Title** Robust Inference for Absolute Abundance in Microbiome Analysis

**Version** 1.2.0

**Description** This package offers a robust approach to make inference on the association of covariates with the absolute abundance (AA) of microbiome in an ecosystem. It can be also directly applied to relative abundance (RA) data to make inference on AA because the ratio of two RA is equal to the ratio of their AA. This algorithm can estimate and test the associations of interest while adjusting for potential confounders. The estimates of this method have easy interpretation like a typical regression analysis. High-dimensional covariates are handled with regularization and it is implemented by parallel computing. False discovery rate is automatically controlled by this approach. Zeros do not need to be imputed by a positive value for the analysis. The IFAA package also offers the 'MZILN' function for estimating and testing associations of abundance ratios with covariates.

**License** GPL-2

**Encoding** UTF-8

**URL** https://pubmed.ncbi.nlm.nih.gov/35241863/,
https://pubmed.ncbi.nlm.nih.gov/30923584/,
https://github.com/quranwu/IFAA

**BugReports** https://github.com/quranwu/IFAA/issues

**LazyData** FALSE

**RoxygenNote** 7.2.3

**Depends** R (>= 4.2.0),

**Imports** mathjaxr, doRNG, foreach (>= 1.4.3), Matrix (>= 1.4-0), HDCI (>= 1.0-2), parallel (>= 3.3.0), doParallel (>= 1.0.11), parallelly , glmnet, stats, utils, SummarizedExperiment, stringr, S4Vectors, DescTools, MatrixExtra, methods

**biocViews** Software, Technology, Sequencing, Microbiome, Regression

**RdMacros** mathjaxr

**Suggests** knitr, rmarkdown, RUnit, BiocGenerics, BiocStyle

**VignetteBuilder** knitr

**NeedsCompilation** no

**git_url** https://git.bioconductor.org/packages/IFAA

**git_branch** RELEASE_3_17

**git_last_commit** 0aee780

**git_last_commit_date** 2023-04-25

**Date/Publication** 2023-04-25

# R topics documented:



---

`IFAA` *Robust association identification and inference for absolute abundance in microbiome analyses*

---

## Description

The IFAA function is to make inference on the association of microbiome with covariates

To model the association, the following equation is used:

$$\log(\mathcal{Y}_i^k)|\mathcal{Y}_i^k > 0 = \beta^{0k} + X_i^T\beta^k + W_i^T\gamma^k + Z_i^T b_i + \epsilon_i^k, \quad k = 1, ..., K+1$$

where

- $\mathcal{Y}_i^k$ is the AA of taxa $k$ in subject $i$ in the entire ecosystem.
- $X_i$ is the covariate matrix.
- $W_i$ is the confounder matrix.
- $Z_i$ is the design matrix for random effects.
- $\beta^k$ is the regression coefficients that will be estimated and tested with the `IFAA()` function.

The challenge in microbiome analysis is that $\mathcal{Y}_i^k$ can not be observed. What is observed is its small proportion: $Y_i^k = C_i\mathcal{Y}_i^k$, where $C_i$ is an unknown number between 0 and 1 that denote the observed proportion.

The IFAA method can successfully addressed this challenge. The `IFAA()` will estimate the parameter $\beta^k$ and their 95% confidence intervals. High-dimensional $X_i$ is handled by regularization.

When using this function, most of the time, users just need to feed these three inputs to the function: `experiment_dat`, `testCov` and `ctrlCov`. All other inputs can just take their default values.

## Usage

```
IFAA(
  experiment_dat,
  microbVar = "all",
  testCov = NULL,
  ctrlCov = NULL,
  sampleIDname = NULL,
  testMany = TRUE,
  ctrlMany = FALSE,
  nRef = 40,
  nRefMaxForEsti = 2,
  refTaxa = NULL,
  adjust_method = "BY",
  fdrRate = 0.05,
  paraJobs = NULL,
  bootB = 500,
  standardize = FALSE,
  sequentialRun = FALSE,
  refReadsThresh = 0.2,
  taxDropThresh = 0,
  SDThresh = 0.05,
  SDquantilThresh = 0,
  balanceCut = 0.2,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| `experiment_dat` | A SummarizedExperiment object containing microbiome data and covariates (see example on how to create a SummarizedExperiment object). The microbiome data can be absolute abundance or relative abundance with each column per sample and each row per taxon/OTU/ASV (or any other unit). No imputation is needed for zero-valued data points. The covariates data contains covariates and confounders with each row per sample and each column per variable. The covariates data has to be numeric or binary. Categorical variables should be converted into dummy variables. |
| `microbVar` | This takes a single or vector of microbiome variable names (e.g., taxa, OTU and ASV names) of interest. Default is "all" meaning all microbiome variables will be analyzed. If a subset of microbiome variables is specified, the output will only contain the specified variables, and p-value adjustment for multiple testing will only be applied to the subset. |
| `testCov` | Covariates that are of primary interest for testing and estimating the associations. It corresponds to $X_i$ in the equation. Default is `NULL` which means all covariates are `testCov`. |
| `ctrlCov` | Potential confounders that will be adjusted in the model. It corresponds to $W_i$ in the equation. Default is `NULL` which means all covariates except those in `testCov` are adjusted as confounders. |

| | |
|---|---|
| sampleIDname | Name of the sample ID variable in the data. In the case that the data does not have an ID variable, this can be ignored. Default is NULL. |
| testMany | This takes logical value TRUE or FALSE. If TRUE, the testCov will contain all the variables in CovData provided testCov is set to be NULL. The default value is TRUE which does not do anything if testCov is not NULL. |
| ctrlMany | This takes logical value TRUE or FALSE. If TRUE, all variables except testCov are considered as control covariates provided ctrlCov is set to be NULL. The default value is FALSE. |
| nRef | The number of randomly picked reference taxa used in phase 1. Default number is 40. |
| nRefMaxForEsti | The maximum number of final reference taxa used in phase 2. The default is 2. |
| refTaxa | A vector of taxa or OTU or ASV names. These are reference taxa specified by the user to be used in phase 1. If the number of reference taxa is less than 'nRef', the algorithm will randomly pick extra reference taxa to make up 'nRef'. The default is NULL since the algorithm will pick reference taxa randomly. |
| adjust_method | The adjusting method for p value adjustment. Default is "BY" for dependent FDR adjustment. It can take any adjustment method for p.adjust function in R. |
| fdrRate | The false discovery rate for identifying taxa/OTU/ASV associated with testCov. Default is 0.05. |
| paraJobs | If sequentialRun is FALSE, this specifies the number of parallel jobs that will be registered to run the algorithm. If specified as NULL, it will automatically detect the cores to decide the number of parallel jobs. Default is NULL. |
| bootB | Number of bootstrap samples for obtaining confidence interval of estimates in phase 2 for the high dimensional regression. The default is 500. |
| standardize | This takes a logical value TRUE or FALSE. If TRUE, the design matrix for X will be standardized in the analyses and the results. Default is FALSE. |
| sequentialRun | This takes a logical value TRUE or FALSE. Default is FALSE. This argument could be useful for debug. |
| refReadsThresh | The threshold of proportion of non-zero sequencing reads for choosing the reference taxon in phase 2. The default is 0.2 which means at least 20% non-zero sequencing reads. |
| taxDropThresh | The threshold of number of non-zero sequencing reads for each taxon to be dropped from the analysis. The default is 0 which means taxon without any sequencing reads will be dropped from the analysis. |
| SDThresh | The threshold of standard deviations of sequencing reads for been chosen as the reference taxon in phase 2. The default is 0.05 which means the standard deviation of sequencing reads should be at least 0.05 in order to be chosen as reference taxon. |
| SDquantilThresh | The threshold of the quantile of standard deviation of sequencing reads, above which could be selected as reference taxon. The default is 0. |
| balanceCut | The threshold of the proportion of non-zero sequencing reads in each group of a binary variable for choosing the final reference taxa in phase 2. The default number is 0.2 which means at least 20% non-zero sequencing reads in each group are needed to be eligible for being chosen as a final reference taxon. |
| verbose | Whether the process message is printed out to the console. The default is TRUE. |

## Value

A list containing 2 elements

- `full_results`: The main results for IFAA containing the estimation and testing results for all associations between all taxa and all test covariates in `testCov`. It is a dataframe with each row representing an association, and eight columns named as "taxon", "cov", "estimate", "SE.est", "CI.low", "CI.up", "adj.p.value", and "sig_ind". The columns correspond to taxon name, covariate name, association estimates, standard error estimates, lower bound and upper bound of the 95% confidence interval, adjusted p value, and the indicator showing whether the association is significant after multiple testing adjustment.
- `metadata`: The metadata is a list.
  - `covariatesData`: A dataset containing covariates and confounders used in the analyses.
  - `final_ref_taxon`: The final 2 reference taxon used for analysis.
  - `ref_taxon_count`: The counts of selection for the associations of all taxa with test covariates in Phase 1.
  - `totalTimeMins`: The average magnitude estimates for the associations of all taxa with test covariates in Phase 1.
  - `ref_taxon_est`: Total time used for the entire analysis.
  - `fdrRate`: FDR rate used for the analysis.
  - `adjust_method`: Multiple testing adjust method used for the analysis.

## References

Li et al.(2021) IFAA: Robust association identification and Inference For Absolute Abundance in microbiome analyses. Journal of the American Statistical Association. 116(536):1595-1608

## Examples

```
library(IFAA)
## A makeup example data from Scratch. 10 taxon, 20 subjects, 3 covariates

set.seed(1)

## create an ID variable for the example data
ID=seq_len(20)

## generate three covariates x1, x2, and x3, with x2 binary
x1<-rnorm(20)
x2<-rbinom(20,1,0.5)
x3<-rnorm(20)
dataC<-data.frame(cbind(ID,x1,x2,x3))

## Coefficients for x1, x2, and x3 among 10 taxa.
beta_1<-c(0.1,rep(0,9))
beta_2<-c(0,0.2,rep(0,8))
beta_3<-rnorm(10)
beta_mat<-cbind(beta_1,beta_2,beta_3)

## Generate absolute abundance for 10 taxa in ecosystem.
```

```
dataM_eco<-floor(exp(10+as.matrix(dataC[,-1])%*%t(beta_mat) + rnorm(200,sd=0.05)))

## Generate sequence depth and generate observed abundance
Ci<-runif(20,0.01,0.05)
dataM<-floor(apply(dataM_eco,2,function(x) x*Ci))
colnames(dataM)<-paste0("rawCount",1:10)

## Randomly introduce 0 to make 25% sparsity level.
dataM[sample(seq_len(length(dataM)),length(dataM)/4)]<-0

dataM<-data.frame(cbind(ID,dataM))

## The following steps are to create a SummarizedExperiment object.
## If you already have a SummarizedExperiment format data, you can
## ignore the following steps and directly feed it to the IFAA function.

## Merge two dataset by ID variable
data_merged<-merge(dataM,dataC,by="ID",all=FALSE)

## Seperate microbiome data and covariate data, drop ID variable from microbiome data
dataM_sub<-data_merged[,colnames(dataM)[!colnames(dataM)%in%c("ID")]]
dataC_sub<-data_merged[,colnames(dataC)]

## Create SummarizedExperiment object
test_dat<-SummarizedExperiment::SummarizedExperiment(
assays=list(MicrobData=t(dataM_sub)), colData=dataC_sub)

## Again, if you already have a SummarizedExperiment format data, you can
## ignore the above steps and directly feed it to the IFAA function.

set.seed(123) # For full reproducibility

results <- IFAA(experiment_dat = test_dat,
                testCov = c("x1", "x2"),
                ctrlCov = c("x3"),
                sampleIDname="ID",
                fdrRate = 0.05,
                nRef = 2,
                paraJobs = 2)
## to extract all results:
summary_res<-results$full_results
## to extract significant results:
sig_results=subset(summary_res,sig_ind==TRUE)

## If only interested in certain taxa, say "rawCount1", "rawCount2",
## and "rawCount3", one can do:
results <- IFAA(
experiment_dat = test_dat,
microbVar = c("rawCount1", "rawCount2", "rawCount3"),
testCov = c("x1", "x2"),
ctrlCov = c("x3"),
sampleIDname = "ID",
fdrRate = 0.05,
```

```
nRef = 2,
paraJobs = 2
)
```

---

MZILN *Conditional regression for microbiome analysis based on multivariate zero-inflated logistic normal model*

---

## Description

The MZILN function is for estimating and testing the associations of abundance ratios with covariates.

The regression model for `MZILN()` can be expressed as follows:

$$\log\left(\frac{\mathcal{Y}_i^k}{\mathcal{Y}_i^{K+1}}\right)|\mathcal{Y}_i^k > 0, \mathcal{Y}_i^{K+1} > 0 = \alpha^{0k} + \mathcal{X}_i^T\alpha^k + \epsilon_i^k, \quad k = 1, ..., K$$

where

- $\mathcal{Y}_i^k$ is the AA of taxa $k$ in subject $i$ in the entire ecosystem.
- $\mathcal{Y}_i^{K+1}$ is the reference taxon (specified by user).
- $\mathcal{X}_i$ is the covariate matrix for all covariates including confounders.
- $\alpha^k$ is the regression coefficients along with their 95% confidence intervals that will be estimated by the `MZILN()` function.

High-dimensional $X_i$ is handled by regularization.

When using this function, most of the time, users just need to feed the first four inputs to the function: `experiment_dat`, `microbVar`, `refTaxa` and `allCov`. All other inputs can just take their default values.

## Usage

```
MZILN(
  experiment_dat,
  microbVar = "all",
  refTaxa,
  allCov = NULL,
  sampleIDname = NULL,
  adjust_method = "BY",
  fdrRate = 0.05,
  paraJobs = NULL,
  bootB = 500,
  taxDropThresh = 0,
  standardize = FALSE,
  sequentialRun = FALSE,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| experiment_dat | A SummarizedExperiment object containing microbiome data and covariates (see example on how to create a SummarizedExperiment object). The microbiome data can be absolute abundance or relative abundance with each column per sample and each row per taxon/OTU/ASV (or any other unit). No imputation is needed for zero-valued data points. The covariates data contains covariates and confounders with each row per sample and each column per variable. The covariates data has to be numeric or binary. Categorical variables should be converted into dummy variables. |
| microbVar | This takes a single or vector of microbiome variable names (e.g., taxa, OTU and ASV names) of interest for the numerator of the ratios. Default is "all" meaning all microbiome variables (except denominator) will be analyzed as numerators. If a subset of microbiome variables is specified, the output will only contain the specified ratios, and p-value adjustment for multiple testing will only be applied to the subset. |
| refTaxa | Denominator taxa names specified by the user for the targeted ratios. This could be a vector of names. |
| allCov | All covariates of interest (including confounders) for estimating and testing their associations with the targeted ratios. Default is 'NULL' meaning that all covariates in covData are of interest. |
| sampleIDname | Name of the sample ID variable in the data. In the case that the data does not have an ID variable, this can be ignored. Default is NULL. |
| adjust_method | The adjusting method for p value adjustment. Default is "BY" for dependent FDR adjustment. It can take any adjustment method for p.adjust function in R. |
| fdrRate | The false discovery rate for identifying taxa/OTU/ASV associated with allCov. Default is 0.05. |
| paraJobs | If sequentialRun is FALSE, this specifies the number of parallel jobs that will be registered to run the algorithm. If specified as NULL, it will automatically detect the cores to decide the number of parallel jobs. Default is NULL. |
| bootB | Number of bootstrap samples for obtaining confidence interval of estimates for the high dimensional regression. The default is 500. |
| taxDropThresh | The threshold of number of non-zero sequencing reads for each taxon to be dropped from the analysis. The default is 0 which means taxon without any sequencing reads will be dropped from the analysis. |
| standardize | This takes a logical value TRUE or FALSE. If TRUE, the design matrix for X will be standardized in the analyses and the results. Default is FALSE. |
| sequentialRun | This takes a logical value TRUE or FALSE. Default is TRUE. It can be set to be "FALSE" to increase speed if there are multiple taxa in the argument 'refTaxa'. |
| verbose | Whether the process message is printed out to the console. The default is TRUE. |

## Value

A list with two elements.

- `full_results`: The main results for MZILN containing the estimation and testing results for all associations between all taxa ratios with refTaxan being the denominator and all covariates in `allCov`. It is a dataframe with each row representing an association, and ten columns named as "ref_tax", "taxon", "cov", "estimate", "SE.est", "CI.low", "CI.up", "adj.p.value", "unadj.p.value", and "sig_ind". The columns correspond to the denominator taxon, numerator taxon, covariate name, association estimates, standard error estimates, lower bound and upper bound of the 95% confidence interval, adjusted p value, and the indicator showing whether the association is significant after multiple testing adjustment.
- `metadata`: The metadata is a list containing total time used in minutes, FDR rate, and multiple testing adjustment method used.

## References

Li et al.(2018) Conditional Regression Based on a Multivariate Zero-Inflated Logistic-Normal Model for Microbiome Relative Abundance Data. Statistics in Biosciences 10(3): 587-608

## Examples

```
library(IFAA)
## A makeup example data from Scratch. 60 taxon, 40 subjects, 3 covariates

set.seed(1)

## create an ID variable for the example data
ID=seq_len(40)

## generate three covariates x1, x2, and x3, with x2 binary
x1<-rnorm(40)
x2<-rbinom(40,1,0.5)
x3<-rnorm(40)
dataC<-data.frame(cbind(ID,x1,x2,x3))

## Coefficients for x1, x2, and x3 among 60 taxa.
beta_1<-c(0.1,rep(0,59))
beta_2<-c(0,0.2,rep(0,58))
beta_3<-rnorm(60)
beta_mat<-cbind(beta_1,beta_2,beta_3)

## Generate absolute abundance for 60 taxa in ecosystem.
dataM_eco<-floor(exp(10+as.matrix(dataC[,-1])%*%t(beta_mat) + rnorm(2400,sd=0.05)))

## Generate sequence depth and generate observed abundance
Ci<-runif(40,0.01,0.05)
dataM<-floor(apply(dataM_eco,2,function(x) x*Ci))
colnames(dataM)<-paste0("rawCount",1:60)

## Randomly introduce 0 to make 25% sparsity level.
dataM[sample(seq_len(length(dataM)),length(dataM)/4)]<-0

dataM<-data.frame(cbind(ID,dataM))
```

```
## The following steps are to create a SummarizedExperiment object.
## If you already have a SummarizedExperiment format data, you can
## ignore the following steps and directly feed it to the IFAA function.

## Merge two dataset by ID variable
data_merged<-merge(dataM,dataC,by="ID",all=FALSE)

## Seperate microbiome data and covariate data, drop ID variable from microbiome data
dataM_sub<-data_merged[,colnames(dataM)[!colnames(dataM)%in%c("ID")]]
dataC_sub<-data_merged[,colnames(dataC)]

## Create SummarizedExperiment object
test_dat<-SummarizedExperiment::SummarizedExperiment(
assays=list(MicrobData=t(dataM_sub)), colData=dataC_sub)

## Again, if you already have a SummarizedExperiment format data, you can
## ignore the above steps and directly feed it to the IFAA function.

## Run MZILN function
set.seed(123) # For full reproducibility

## If interested in the taxa ratios, say `"rawCount1", "rawCount2",
## and "rawCount3"` over "rawCount10", one can do:

results <- MZILN(
experiment_dat = test_dat,
microbVar = c("rawCount1", "rawCount2", "rawCount3"),
refTaxa = c("rawCount10"),
allCov = c("x1", "x2", "x3"),
sampleIDname = "ID",
fdrRate = 0.05
)

## results can be extracted as follows:

summary_res_ratio <- results$full_results
summary_res_ratio

## To explore all ratios with "rawCount10" being the denominator,
## one can do:

results <- MZILN(experiment_dat = test_dat,
                 refTaxa=c("rawCount10"),
                 allCov=c("x1","x2","x3"),
                 sampleIDname="ID",
                 fdrRate=0.05)
## to extract the results for all ratios with rawCount10 as the denominator:
summary_res<-results$full_results
## to extract results for the ratio of a specific taxon (e.g., rawCount45) over rawCount10:
target_ratio=summary_res[summary_res$taxon=="rawCount45",]
## to extract all of the ratios having significant associations:
sig_ratios=subset(summary_res,sig_ind==TRUE)
```

# Index