Package ‘BRACoD.R’

May 17, 2021

Title  BRACoD: Bayesian Regression Analysis of Compositional Data

Version  0.0.1.2

Description  The goal of this method is to identify associations between bacteria and an environmental variable in 16S or other compositional data. The environmental variable is any variable which is measure for each microbiome sample, for example, a butyrate measurement paired with every sample in the data. Microbiome data is compositional, meaning that the total abundance of each sample sums to 1, and this introduces severe statistical distortions. This method takes a Bayesian approach to correcting for these statistical distortions, in which the total abundance is treated as an unknown variable. This package runs the python implementation using reticulate.

Imports  reticulate

Config/reticulate  list( packages = list( list(package = ``BRACoD'') ) )

License  MIT + file LICENSE

Encoding  UTF-8

LazyData  true

RoxygenNote  7.1.1.9001

Suggests  testthat (>= 3.0.0)

Config/testthat/edition  3

NeedsCompilation  no

Author  Adrian Verster [aut, cre]

Maintainer  Adrian Verster <adrian.verster@canada.ca>

Depends  R (>= 3.5.0)

Repository  CRAN

Date/Publication  2021-05-17 07:00:05 UTC

R topics documented:

  convergence_tests  .........................................................  2
  install_bracod  ...............................................................  2
  obesity  .................................................................  3
**Install BRACoD in python**

**Description**

Uses `pip` to install the latest BRACoD release in python. You might need to specify a python environment with either `reticulate::use_virtualenv` or `reticulate::use_condaenv`.

**Usage**

```r
install_bracod(method = "auto", conda = "auto")
```

---

**Convergence Tests**

*Perform convergence tests on the p and beta variables*

**Description**

You may get errors or divergence of some variables after `pymc3` samples the posterior. We are not overly concerned about some of the variables, such as the variance, rather we are really interested in the inclusion probabilities (p) and contribution coefficients (beta). The convergence tests that are included here focus on evaluating those two variables.

**Usage**

```r
convergence_tests(trace, df_relab)
```

**Arguments**

- `trace`: the output of `run_bracod()`
- `df_relab`: the microbiome relative abundance

**Value**

no return value
Arguments

method passed to reticulate::py_install
conda passed to reticulate::py_install

Value

no return value

Description

This data is mouse stool microbiome data from a study of obesity.

Usage

data(obesity)

df_scfa

Format

a DataFrame of 16S microbiome counts, and a dataframe with corresponding butyrate measurements
An object of class data.frame with 119 rows and 1 columns.

remove_null Remove NULL values in your OTU and environmental variable

Description

This will remove samples that are NULL in the environmental variable, as well as the corresponding samples in your relative abundance data.

Usage

remove_null(df_relab, Y)

Arguments

df_relab microbiome relative abundance data in a dataframe
Y values of the environmental variable

Value

a list containing 1) the relative abundance data and 2) the Y values
Run the main BRACoD algorithm

Description

Uses pymc3 to sample the posterior of the model to determine bacteria that are associated with your environmental variable.

Usage

```r
run_bracod(df_relab, env_var, n_sample = 1000, n_burn = 1000, njobs = 4)
```

Arguments

- **df_relab**: A dataframe of relative microbiome abundances. Samples are rows and bacteria are columns.
- **env_var**: the environmental variable you are evaluating. You need 1 measurement associated with each sample.
- **n_sample**: number of posterior samples.
- **n_burn**: number of burn-in steps before actual sampling stops.
- **njobs**: number of parallel MCMC chains to run.

Value

the pymc trace object which holds the samples of the posterior distribution

Examples

```r
## Not run:
data(obesity)
r <- simulate_microbiome_counts(obesity)
sim_counts <- r[[1]]
sim_y <- r[[2]]
contributions <- r[[3]]
sim_relab <- scale_counts(sim_counts)
trace <- run_bracod(sim_relab, sim_y, n_sample = 1000, n_burn=1000, njobs=4)
## End(Not run)
```
**scale_counts**

Normalize OTU counts and add a pseudo count

**Description**

BRACoD requires relative abundance and cannot handle zeros, so this function adds a small pseudo count (1/10th the smallest non-zero value).

**Usage**

scale_counts(df_counts)

**Arguments**

df_counts

A dataframe of OTU counts. Samples are rows and bacteria are columns.

**Value**

A dataframe of relative abundance data

**score**

Score the results of BRACoD

**Description**

This calculate the precision, recall and F1 of your BRACoD results if you know the ground truth, ie. if this is simulated data.

**Usage**

score(bugs_identified, bugs_actual)

**Arguments**

bugs_identified

A list of integers corresponding to the indicies of the bugs you identified with BRACoD

bugs_actual

A list of integers corresponding to the indicies of the bugs that truely contribute to butyrate levels

**Value**

A list containing 1) the precision 2) the recall 3) the f1 metric
Examples

```r
# Not run:
df_summary <- summarize_trace(trace, colnames(sim_relab))
bugs_identified <- df_summary$bugs
bugs_actual <- which(contributions != 0)

r <- score(bugs_identified, bugs_actual)

precision <- r[[1]]
recall <- r[[2]]
f1 <- r[[3]]

print(sprintf("Precision: %.2f, Recall: %.2f, F1: %.2f", precision, recall, f1))

# End(Not run)
```

simulate_microbiome_counts

*Simulate microbiome counts*

**Description**

Each bacteria’s absolute abundance is simulated from a lognormal distribution. Then, convert each sample to relative abundance, and simulate sequencing counts using a multinominal distribution, based on the desired number of reads and the simulated relative abundances. This also simulates an environmental variable that is produced by some of the bacteria.

**Usage**

```r
simulate_microbiome_counts(
  df,
  n_contributors = 20,
  coeff_contributor = 0,
  min_ab_contributor = -9,
  sd_Y = 1,
  n_reads = 1e+05,
  var_contributor = 5,
  use_uniform = TRUE,
  n_samples_use = NULL,
  corr_value = NULL,
  return_absolute = FALSE,
  seed = NULL
)
```

**Arguments**

- `df` A dataframe of OTU counts that is a model for data simulation. Samples are rows and bacteria are columns.
**summarize_trace**

`summarize_trace` Summarize the results of BRACoD

**Description**

This summarizes the trace object that `run_bracod()` returns. It returns a dataframe that contains two parameters of interest, the average inclusion (p) and the average coefficient (beta), telling you the association between that bacteria and the environmental variable.

**Usage**

```
summarize_trace(trace, bug_names = NULL, cutoff = 0.3)
```

**Arguments**

- `trace`: the pymc3 object that is the output of `run_bracod()`
- `bug_names`: optional, a list of names of the bacteria to include in the results
- `cutoff`: this is the cutoff on the average inclusion for inclusion

**Value**

A list containing:
1. The simulated count data
2. The simulated environmental variable
3. The simulated contribution coefficients

**Example**

```r
# Example usage of summarize_trace
summarize_trace(trace, bug_names = NULL, cutoff = 0.3)
```
Value

a dataframe with information about the bacteria that BRACoD identified

Examples

```r
## Not run:
trace <- run_bracod(sim_relab, sim_y, n_sample = 1000, nburn=1000, njobs=4)
df_summary <- summarize_trace(trace, colnames(sim_relab))
## End(Not run)
```

threshold_count_data  Threshold your microbiome counts data

Description

This function removes samples below a minimum counts and bacteria below a minimum log abundance. Run this before running BRACoD because the algorithm does not perform well when there are many low abundance bacteria that are only present in a few samples.

Usage

```r
threshold_count_data(df_counts, min_counts = 1000, min_ab = 1e-04)
```

Arguments

- `df_counts`: A dataframe of OTU counts. Samples are rows and bacteria are columns.
- `min_counts`: threshold samples with fewer than this many counts
- `min_ab`: threshold bacteria whose average log abundance is below this

Value

a dataframe of microbiome counts
Index

* datasets
  obesity, 3

convergence_tests, 2

df_counts_obesity (obesity), 3
df_counts_obesity,df_scfa (obesity), 3
df_scfa (obesity), 3

install_bracod, 2

obesity, 3

remove_null, 3
run_bracod, 4

scale_counts, 5
score, 5
simulate_microbiome_counts, 6
summarize_trace, 7

threshold_count_data, 8