

Package: codalm (via r-universe)

August 31, 2024

Type Package

Title Transformation-Free Linear Regression for Compositional Outcomes and Predictors

Version 0.1.0

Maintainer Jacob Fiksel <jfiksel@gmail.com>

Description Implements the expectation-maximization (EM) algorithm as described in Fiksel et al. (2020) <[arXiv:2004.07881](https://arxiv.org/abs/2004.07881)> for transformation-free linear regression for compositional outcomes and predictors.

License GPL-2

Imports SQUAREM (>= 2020.3), future, future.apply

Encoding UTF-8

LazyData true

URL <https://github.com/jfiksel/codalm>

BugReports <https://github.com/jfiksel/codalm/issues>

RoxygenNote 7.1.0

Suggests knitr, ggtern, gtools, remotes, testthat, markdown

VignetteBuilder knitr

Repository <https://jfiksel.r-universe.dev>

RemoteUrl <https://github.com/jfiksel/codalm>

RemoteRef HEAD

RemoteSha 8b48c713c48275bd8cd7c439e70147dc0f847818

Contents

codalm	2
codalm_ci	3
codalm_indep_test	4

Index	6
--------------	----------

codalm	<i>Transformation-free Linear Regression for Compositional Outcomes and Predictors</i>
--------	--

Description

Implements the expectation-maximization (EM) algorithm as described in Fiksel et al. (2020) for transformation-free linear regression for compositional outcomes and predictors.

Usage

```
codalm(y, x, accelerate = TRUE)
```

Arguments

y	A matrix of compositional outcomes. Each row is an observation, and must sum to 1. If any rows do not sum to 1, they will be renormalized
x	A matrix of compositional predictors. Each row is an observation, and must sum to 1. If any rows do not sum to 1, they will be renormalized
accelerate	A logical variable, indicating whether or not to use the Squarem algorithm for acceleration of the EM algorithm. Default is TRUE.

Value

A $D_s \times D_r$ compositional coefficient matrix, where D_s and D_r are the dimensions of the compositional predictor and outcome, respectively

References

<https://arxiv.org/abs/2004.07881>

Examples

```
require(ggtern)
data("WhiteCells", package = 'ggtern')
image <- subset(WhiteCells, Experiment == "ImageAnalysis")
image_mat <- as.matrix(image[,c("G", "L", "M")])
microscopic <- subset(WhiteCells, Experiment == "MicroscopicInspection")
microscopic_mat <- as.matrix(microscopic[,c("G", "L", "M")])
x <- image_mat / rowSums(image_mat)
y <- microscopic_mat / rowSums(microscopic_mat)
codalm(y, x)
```

codalm_ci	<i>Bootstrap Confidence Intervals Linear Regression for Compositional Outcomes and Predictors</i>
-----------	---

Description

Implements percentile based bootstrapping to estimate the confidence intervals for the regression coefficients when doing linear regression for compositional outcomes and predictors

Usage

```
codalm_ci(
  y,
  x,
  accelerate = TRUE,
  nboot = 500,
  conf = 0.95,
  parallel = FALSE,
  ncpus = NULL,
  strategy = NULL,
  init.seed = 123
)
```

Arguments

y	A matrix of compositional outcomes. Each row is an observation, and must sum to 1. If any rows do not sum to 1, they will be renormalized
x	A matrix of compositional predictors. Each row is an observation, and must sum to 1. If any rows do not sum to 1, they will be renormalized
accelerate	A logical variable, indicating whether or not to use the Squarem algorithm for acceleration of the EM algorithm. Default is TRUE
nboot	The number of bootstrap repetitions to use. Default is 500
conf	A scalar between 0 and 1 containing the confidence level of the required intervals. Default is .95.
parallel	A logical variable, indicating whether or not to use a parallel operation for computing the permutation statistics
ncpus	Optional argument. When provided, is an integer giving the number of clusters to be used in parallelization. Defaults to the number of cores, minus 1.
strategy	Optional argument. When provided, this will be the evaluation function (or name of it) to use for parallel computation (if parallel = TRUE). Otherwise, if parallel = TRUE, then this will default to multisession. See plan .
init.seed	The initial seed for the permutations. Default is 123.

Value

A list, with ci_L and ci_U, giving the lower and upper bounds of each element of the B matrix

Examples

```
require(ggtern)
data("WhiteCells", package = 'ggtern')
image <- subset(WhiteCells, Experiment == "ImageAnalysis")
image_mat <- as.matrix(image[,c("G", "L", "M")])
microscopic <- subset(WhiteCells, Experiment == "MicroscopicInspection")
microscopic_mat <- as.matrix(microscopic[,c("G", "L", "M")])
x <- image_mat / rowSums(image_mat)
y <- microscopic_mat / rowSums(microscopic_mat)
codalm_ci(y, x, nboot = 50, conf = .95)
```

codalm_indep_test	<i>Permutation Test for Linear Independence Between Compositional Outcomes and Predictors</i>
-------------------	---

Description

Implements the loss function based permutation test as described in Fiksel et al. (2020) for a test of linear independence between compositional outcomes and predictors.

Usage

```
codalm_indep_test(
  y,
  x,
  nperms = 500,
  accelerate = TRUE,
  parallel = FALSE,
  ncpus = NULL,
  strategy = NULL,
  init.seed = 123
)
```

Arguments

y	A matrix of compositional outcomes. Each row is an observation, and must sum to 1. If any rows do not sum to 1, they will be renormalized
x	A matrix of compositional predictors. Each row is an observation, and must sum to 1. If any rows do not sum to 1, they will be renormalized
nperms	The number of permutations. Default is 500.
accelerate	A logical variable, indicating whether or not to use the Squarem algorithm for acceleration of the EM algorithm. Default is TRUE.
parallel	A logical variable, indicating whether or not to use a parallel operation for computing the permutation statistics

ncpus	Optional argument. When provided, is an integer giving the number of clusters to be used in parallelization. Defaults to the number of cores, minus 1.
strategy	Optional argument. When provided, this will be the evaluation function (or name of it) to use for parallel computation (if parallel = TRUE). Otherwise, if parallel = TRUE, then this will default to multisession. See plan .
init.seed	The initial seed for the permutations. Default is 123.

Value

The p-value for the independence test

Examples

```
require(gtools)
x <- rdirichlet(100, c(1, 1, 1))
y <- rdirichlet(100, c(1, 1, 1))
codalm_indep_test(y, x)
```

```
require(ggtern)
data("WhiteCells", package = 'ggtern')
image <- subset(WhiteCells, Experiment == "ImageAnalysis")
image_mat <- as.matrix(image[,c("G", "L", "M")])
microscopic <- subset(WhiteCells, Experiment == "MicroscopicInspection")
microscopic_mat <- as.matrix(microscopic[,c("G", "L", "M")])
x <- image_mat / rowSums(image_mat)
y <- microscopic_mat / rowSums(microscopic_mat)
codalm_indep_test(y, x)
```

Index

[codalm](#), [2](#)
[codalm_ci](#), [3](#)
[codalm_indep_test](#), [4](#)

[plan](#), [3](#), [5](#)