Package ‘Compositional’

March 23, 2022

Type Package
Title Compositional Data Analysis
Version 5.5
Date 2022-03-22
Author Michail Tsagris [aut, cre],
Giorgos Athineou [aut],
Abdulaziz Alenazi [ctb],
Christos Adam [ctb]
Maintainer Michail Tsagris <mtsagris@uoc.gr>
Depends R (>= 4.0)
Imports codalm, Directional, doParallel, emplik, energy, FlexDir,
foreach, glmnet, graphics, grDevices, MASS, mda, mixture,
NlcOptim, nnet, parallel, pchc, RANN, Rfast, Rfast2, sn, stats
Description Regression, classification, contour plots, hypothesis testing and fitting of distributions for compositional data are some of the functions included.
The standard textbook for such data is John Aitchison’s (1986) “The statistical analysis of compositional data”. Relevant papers include:


We further include functions for percentages (or proportions).

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2022-03-23 22:30:02 UTC

R topics documented:

Compositional-package .................................................. 6
All pairwise additive log-ratio transformations ....................... 7
Alpha generalised correlations between two compositional datasets ........................................ 8
ANOVA for the log-contrast regression versus the unconstrained log-contrast regression .......... 9
Bayesian network learning with compositional data ................... 10
Beta regression ................................................................ 12
Choose the number of principal components via reconstruction error ................................. 13
Column-wise MLE of some univariate distributions .................... 14
Constrained linear least squares for compositional responses and predictors ....................... 15
Contour plot of mixtures of Dirichlet distributions in $S^2$ ................ 17
Contour plot of the alpha multivariate normal in $S^2$ ...................... 18
Contour plot of the alpha-folded model in $S^2$ .............................. 19
Contour plot of the Dirichlet distribution in $S^2$ ............................ 20
Contour plot of the Flexible Dirichlet distribution in $S^2$ .................. 21
Contour plot of the Gaussian mixture model in $S^2$ ...................... 23
Contour plot of the Kent distribution in $S^2$ ............................... 24
Contour plot of the kernel density estimate in $S^2$ ......................... 25
Contour plot of the normal distribution in $S^2$ .............................. 26
Contour plot of the skew skew-normal distribution in $S^2$ .................... 27
Contour plot of the t distribution in $S^2$ ................................. 28
Cross validation for some compositional regression models .................. 30
Cross validation for the alpha-k-NN regression with compositional predictor variables ........ 31
Cross validation for the alpha-k-NN regression with compositional response data .................. 32
Cross validation for the alpha-kernel regression with compositional response data ............... 34
Cross validation for the kernel regression with Euclidean response data ....................... 35
Cross validation for the regularised and flexible discriminant analysis with compositional data using the alpha-transformation .................. 37
Cross validation for the ridge regression ................................ 39
Cross validation for the ridge regression with compositional data as predictor using the alpha-transformation .............................................. 40
Cross validation for the transformation-free linear regression for compositional responses and predictors ......................................................... 42
Cross-validation for the constrained linear least squares for compositional responses and predictors .......................................................... 43
Cross-validation for the Dirichlet discriminant analysis .................................................. 44
Cross-validation for the LASSO Kullback-Leibler divergence based regression ............. 46
Cross-validation for the LASSO log-ratio regression with compositional response .......... 47
Cross-validation for the naive Bayes classifiers for compositional data ......................... 48
Cross-validation for the naive Bayes classifiers for compositional data using the alpha-transformation .................................................. 50
Density of compositional data from Gaussian mixture models .................................. 51
Density of the Flexible Dirichlet distribution ......................................................... 52
Density of the folded normal distribution .................................................................. 53
Density values of a Dirichlet distribution ..................................................................... 55
Density values of a mixture of Dirichlet distributions .................................................. 56
Dirichlet discriminant analysis .................................................................................. 57
Dirichlet random values simulation ............................................................................ 58
Dirichlet regression ..................................................................................................... 59
Distance based regression models for proportions ....................................................... 61
Divergence based regression for compositional data ..................................................... 62
Divergence based regression for compositional data with compositional data in the co-
variates side using the alpha-transformation ......................................................... 64
Divergence matrix of compositional data ..................................................................... 66
Empirical likelihood for a one sample mean vector hypothesis testing ....................... 67
Empirical likelihood hypothesis testing for two mean vectors ..................................... 68
Energy test of equality of distributions using the alpha-transformation ....................... 70
Estimating location and scatter parameters for compositional data ......................... 71
Estimation of the probability left outside the simplex when using the alpha-transformation 72
Estimation of the value of alpha in the folded model .................................................. 74
Estimation of the value of alpha via the profile log-likelihood ..................................... 75
Exponential empirical likelihood for a one sample mean vector hypothesis testing ....... 76
Exponential empirical likelihood hypothesis testing for two mean vectors .................. 77
Fast estimation of the value of alpha ....................................................................... 79
Fitting a Flexible Dirichlet distribution ...................................................................... 80
Gaussian mixture models for compositional data ...................................................... 82
Gaussian mixture models for compositional data using the alpha-transformation ......... 84
Generate random folds for cross-validation .................................................................. 86
Greenacre’s power transformation .............................................................................. 87
Helper Frechet mean for compositional data .............................................................. 88
Helper functions for the Kullback-Leibler regression .................................................. 89
Hotelling’s multivariate version of the 1 sample t-test for Euclidean data ................... 91
Hotelling’s multivariate version of the 2 sample t-test for Euclidean data ................... 92
Hypothesis testing for two or more compositional mean vectors .................................. 93
ICE plot for projection pursuit regression with compositional predictor variables ....... 95
ICE plot for univariate kernel regression ..................................................................... 96
Inverse of the alpha-transformation ......................................................................... 98
<table>
<thead>
<tr>
<th>R topics documented</th>
</tr>
</thead>
<tbody>
<tr>
<td>James multivariate version of the t-test</td>
</tr>
<tr>
<td>Kernel regression with a numerical response vector or matrix</td>
</tr>
<tr>
<td>Kullback-Leibler divergence and Bhattacharyya distance between two Dirichlet distributions</td>
</tr>
<tr>
<td>LASSO Kullback-Leibler divergence based regression</td>
</tr>
<tr>
<td>LASSO log-ratio regression with compositional response</td>
</tr>
<tr>
<td>Log-contrast logistic or Poisson regression with compositional predictor variables</td>
</tr>
<tr>
<td>Log-contrast regression with compositional predictor variables</td>
</tr>
<tr>
<td>Log-contrast regression with multiple compositional predictors</td>
</tr>
<tr>
<td>Log-likelihood ratio test for a Dirichlet mean vector</td>
</tr>
<tr>
<td>Log-likelihood ratio test for a symmetric Dirichlet distribution</td>
</tr>
<tr>
<td>Mixture model selection via BIC</td>
</tr>
<tr>
<td>Mixture model selection with the alpha-transformation using BIC</td>
</tr>
<tr>
<td>MLE for the multivariate t distribution</td>
</tr>
<tr>
<td>MLE of distributions defined in the (0, 1) interval</td>
</tr>
<tr>
<td>MLE of the Dirichlet distribution</td>
</tr>
<tr>
<td>MLE of the Dirichlet distribution via Newton-Rapshon</td>
</tr>
<tr>
<td>MLE of the folded model for a given value of alpha</td>
</tr>
<tr>
<td>MLE of the zero adjusted Dirichlet distribution</td>
</tr>
<tr>
<td>Multivariate analysis of variance</td>
</tr>
<tr>
<td>Multivariate analysis of variance (James test)</td>
</tr>
<tr>
<td>Multivariate kernel density estimation</td>
</tr>
<tr>
<td>Multivariate kernel density estimation for compositional data</td>
</tr>
<tr>
<td>Multivariate linear regression</td>
</tr>
<tr>
<td>Multivariate normal random values simulation on the simplex</td>
</tr>
<tr>
<td>Multivariate or univariate regression with compositional data in the covariates side using the alpha-transformation</td>
</tr>
<tr>
<td>Multivariate regression with compositional data</td>
</tr>
<tr>
<td>Multivariate skew normal random values simulation on the simplex</td>
</tr>
<tr>
<td>Multivariate t random values simulation on the simplex</td>
</tr>
<tr>
<td>Naive Bayes classifiers for compositional data</td>
</tr>
<tr>
<td>Naive Bayes classifiers for compositional data using the alpha-transformation</td>
</tr>
<tr>
<td>Non linear least squares regression for compositional data</td>
</tr>
<tr>
<td>Non parametric zero replacement strategies</td>
</tr>
<tr>
<td>Perturbation operation</td>
</tr>
<tr>
<td>Plot of the LASSO coefficients</td>
</tr>
<tr>
<td>Power operation</td>
</tr>
<tr>
<td>Principal component analysis</td>
</tr>
<tr>
<td>Principal component analysis using the alpha-transformation</td>
</tr>
<tr>
<td>Principal component generalised linear models</td>
</tr>
<tr>
<td>Principal coordinate analysis using the alpha-distance</td>
</tr>
<tr>
<td>Principal coordinate analysis using the Jensen-Shannon divergence</td>
</tr>
<tr>
<td>Projection pursuit regression for compositional data</td>
</tr>
<tr>
<td>Projection pursuit regression with compositional predictor variables</td>
</tr>
<tr>
<td>Proportionality correlation coefficient matrix</td>
</tr>
<tr>
<td>Quasi binomial regression for proportions</td>
</tr>
<tr>
<td>Random values generation from some univariate distributions defined on the (0,1) interval</td>
</tr>
<tr>
<td>Regression with compositional data using the alpha-transformation</td>
</tr>
<tr>
<td>R topics documented:</td>
</tr>
<tr>
<td>---------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Regularised and flexible discriminant analysis for compositional data using the alpha-transformation</td>
</tr>
<tr>
<td>Regularised discriminant analysis for Euclidean data</td>
</tr>
<tr>
<td>Ridge regression</td>
</tr>
<tr>
<td>Ridge regression plot</td>
</tr>
<tr>
<td>Ridge regression with compositional data in the covariates side using the alpha-transformation</td>
</tr>
<tr>
<td>Ridge regression with the alpha-transformation plot</td>
</tr>
<tr>
<td>Simulation of compositional data from Gaussian mixture models</td>
</tr>
<tr>
<td>Simulation of compositional data from mixtures of Dirichlet distributions</td>
</tr>
<tr>
<td>Simulation of compositional data from the Flexible Dirichlet distribution</td>
</tr>
<tr>
<td>Simulation of compositional data from the folded normal distribution</td>
</tr>
<tr>
<td>Spatial median regression</td>
</tr>
<tr>
<td>Ternary diagram</td>
</tr>
<tr>
<td>Ternary diagram of regression models</td>
</tr>
<tr>
<td>The additive log-ratio transformation and its inverse</td>
</tr>
<tr>
<td>The alpha-distance</td>
</tr>
<tr>
<td>The alpha-IT transformation</td>
</tr>
<tr>
<td>The alpha-IT-distance</td>
</tr>
<tr>
<td>The alpha-k-NN regression for compositional response data</td>
</tr>
<tr>
<td>The alpha-k-NN regression with compositional predictor variables</td>
</tr>
<tr>
<td>The alpha-kernel regression with compositional response data</td>
</tr>
<tr>
<td>The alpha-transformation</td>
</tr>
<tr>
<td>The Box-Cox transformation applied to ratios of components</td>
</tr>
<tr>
<td>The ESOV-distance</td>
</tr>
<tr>
<td>The folded power transformation</td>
</tr>
<tr>
<td>The Frechet mean for compositional data</td>
</tr>
<tr>
<td>The Helmert sub-matrix</td>
</tr>
<tr>
<td>The k-nearest neighbours using the alpha-distance</td>
</tr>
<tr>
<td>The k-NN algorithm for compositional data</td>
</tr>
<tr>
<td>The pivot coordinate transformation and its inverse</td>
</tr>
<tr>
<td>Total variability</td>
</tr>
<tr>
<td>Transformation-free linear regression for compositional responses and predictors</td>
</tr>
<tr>
<td>Tuning of the alpha generalised correlations between two compositional datasets</td>
</tr>
<tr>
<td>Tuning of the bandwidth h of the kernel using the maximum likelihood cross validation</td>
</tr>
<tr>
<td>Tuning of the divergence based regression for compositional data with compositional data using the</td>
</tr>
<tr>
<td>Transformation-free linear regression for compositional responses and predictors</td>
</tr>
<tr>
<td>Tuning of the alpha generalised correlations between two compositional datasets</td>
</tr>
<tr>
<td>Tuning of the bandwidth h of the kernel using the maximum likelihood cross validation</td>
</tr>
<tr>
<td>Tuning of the divergence based regression for compositional data with compositional data using the</td>
</tr>
<tr>
<td>Tuning of the k-NN algorithm for compositional data</td>
</tr>
<tr>
<td>Tuning of the projection pursuit regression for compositional data</td>
</tr>
<tr>
<td>Tuning of the projection pursuit regression with compositional predictor variables</td>
</tr>
<tr>
<td>Tuning the number of PCs in the PCR with compositional data using the alpha-transformation</td>
</tr>
<tr>
<td>Tuning the parameters of the regularised discriminant analysis</td>
</tr>
<tr>
<td>Tuning the principal components with GLMs</td>
</tr>
<tr>
<td>Tuning the value of alpha in the alpha-regression</td>
</tr>
<tr>
<td>Two-sample test of high-dimensional means for compositional data</td>
</tr>
<tr>
<td>Unconstrained log-contrast logistic or Poisson regression with compositional predictor variables</td>
</tr>
<tr>
<td>Unconstrained log-contrast regression with compositional predictor variables</td>
</tr>
<tr>
<td>Unconstrained log-contrast regression with multiple compositional predictors</td>
</tr>
</tbody>
</table>
Compositional-package

Unit-Weibull regression models for proportions ........................................ 218
Zero adjusted Dirichlet regression .............................................................. 219

Index  .......................................................... 221

Compositional-package  Compositional Data Analysis

Description

A Collection of Functions for Compositional Data Analysis.

Details

Package:  Compositional
Type:  Package
Version:  5.5
Date:  2022-03-22
License:  GPL-2

Maintainers

Michail Tsagris <mtsagris@uoc.gr>

Note

Acknowledgments:

Michail Tsagris would like to express his acknowledgments to Professor Andy Wood and Dr Simon Preston from the university of Nottingham for being his supervisors during his PhD in compositional data analysis. We would also like to express our acknowledgments to Professor Kurt Hornik (and also the rest of the R core team) for his help with this package. Manos Papadakis, undergraduate student in the department of computer science, university of Crete, is also acknowledged for his programming tips. Ermanno Affuso from the university of South Alabama suggested that I have a default value in the "mkde" function. Van Thang Hoang from Hasselt university spotted a bug in the "js.compreg" function and is greatly acknowledged for that. Claudia Wehrhahn Cortes spotted a bug in the "diri.reg" function and she is greatly acknowledged for that. Philipp Kynast from Bruker Daltonik GmbH found a mistake in the function "mkde" which is now fixed. Jasmine Heyse from the university of Ghent spotted a bug in the function "kl.compreg" which is now fixed. Magne Neby suggested to add names in the covariance matrix of the divergence based regression models. John Barry from the Centre for Environment, Fisheries, and Aquaculture Science (UK) suggested that I should add more explanation in the "diri.est" function. I hope it is clearer now.
All pairwise additive log-ratio transformations

Author(s)
Michail Tsagris <mtsagris@uoc.gr>, Giorgos Athineou <gioathineou@gmail.com>, Abdulaziz Alenazi <a.alenazi@nbu.edu.sa> and Christos Adam <pada4m4@gmail.com>.

References

Description
All pairwise additive log-ratio transformations.

Usage
alr.all(x)

Arguments
x A numerical matrix with the compositional data.

Details
The additive log-ratio transformation with the first component being the common divisor is applied. Then all the other pairwise log-ratios are computed and added next to each column. For example, divide by the first component, then divide by the second component and so on. This means that no zeros are allowed.

Value
A matrix with all pairwise alr transformed data.

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

See Also
alr, alfa
Examples

```r
x <- as.matrix(iris[, 2:4])
x <- x / rowSums(x)
y <- alr.all(x)
```

Description

α generalised correlations between two compositional datasets.

Usage

```r
acor(y, x, a, type = "dcor")
```

Arguments

- `y`: A matrix with the compositional data.
- `x`: A matrix with the compositional data.
- `a`: The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If \( \alpha = 0 \) the isometric log-ratio transformation is applied. If more than one valuesare supplied the distance or canonical correlation are computed for all values.
- `type`: The type of correlation to compute, the distance correlation ("edist"), the canonical correlation ("cancor") or "both".

Details

The α-transformation is applied to each composition and then the distance correlation or the canonical correlation is computed. If one value of \( \alpha \) is supplied the type="cancor" will return all eigenvalues. If more than one values of \( \alpha \) are provided then the first eigenvalue only will be returned.

Value

A vector or a matrix depending on the length of the values of \( \alpha \) and the type of the correlation to be computed.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.
ANOVA for the log-contrast regression versus the uncostrained log-contrast regression

References


See Also

acor.tune, aeqdist.etest, alfa, alfa.profile

Examples

```r
y <- rdiri(30, runif(3) )
x <- rdiri(30, runif(4) )
acor(y, x, a = 0.4)
```

Description

ANOVA for the log-contrast regression versus the uncostrained log-contrast regression.

Usage

```r
lcreg.aov(mod0, mod1)
```

Arguments

- `mod0`: The log-contrast regression model. The object returned by `lc.reg`.
- `mod1`: The unconstrained log-contrast regression model. The object returned by `ulc.reg`.

Details

An F-test is performed to test the zero-to-sum constraints of the regression coefficients.

Value

A vector with two values, the F-statistic and its associated p-value.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.
Bayesian network learning with compositional data

See Also
lc.reg, ulc.reg, alfa.pcr, alfa.knn.reg

Examples
y <- iris[, 1]
x <- as.matrix(iris[, 2:4])
x <- x / rowSums(x)
mod0 <- lc.reg(y, x)
mod1 <- ulc.reg(y, x)
lcreg.aov(mod0, mod1)

Bayesian network learning with compositional data

Description
Bayesian network learning with compositional data.

Usage
compbn(x, type = "fedhc", max_k = 3, alpha = 0.05, robust = FALSE,
ini.stat = NULL, R = NULL, restart = 10, tabu = 10, score = "bic-g",
blacklist = NULL, whitelist = NULL)

Arguments

x A numerical matrix with the compositional data. They can be either the logged
compositional or the centred log-ratio transformed compositional data. We leave
this open to the user.

type This can be either "fedhc", "pchc", "mmhc", "fedtabu", "pctabu" or "mmtabu".

max_k The maximum conditioning set to use in the conditional independence test (see
Details). Integer, default value is 3

alpha The significance level for assessing the p-values.

robust Do you want outliers to be removed prior to applying the algorithms? If yes, set
this to TRUE to utilise the MCD.

ini.stat If the initial test statistics (univariate associations) are available, pass them through
this parameter.

R If the correlation matrix is available, pass it here.

restart An integer, the number of random restarts.

tabu An integer, the length of the tabu list used in the tabu function.

score A character string, the label of the network score to be used in the algorithm. If
none is specified, the default score is the Bayesian Information Criterion. Other
available scores are: "bic-g" (default), "loglik-g", "aic-g", "bic-g" or "bge".
Bayesian network learning with compositional data

blacklist A data frame with two columns (optionally labeled "from" and "to"), containing a set of arcs not to be included in the graph.

whitelist A data frame with two columns (optionally labeled "from" and "to"), containing a set of arcs to be included in the graph.

Details

The FEDHC algorithm is implemented. The FBED algorithm (Borboudakis and Tsamardinos, 2019), without the backward phase, is implemented during the skeleton identification phase. Next, the Hill Climbing greedy search or the Tabu search is employed to score the network.

The PC algorithm as proposed by Spirites et al. (2001) is first implemented followed by a scoring phase, such as hill climbing or tabu search. The PCHC was proposed by Tsagris (2021), while the PCTABU algorithm is the same but instead of the hill climbing scoring phase, the tabu search is employed.

The MMHC algorithm is implemented without performing the backward elimination during the skeleton identification phase. The MMHC as described in Tsamardinos et al. (2006) employs the MMPC algorithm during the skeleton construction phase and the Tabu search in the scoring phase. In this package, the mmhc function employs the Hill Climbing greedy search in the scoring phase while the mmtabu employs the Tabu search.

Value

A list including:

ini A list including the output of the mmhc.skel function.
dag A "bn" class output. A list including the outcome of the Hill-Climbing or the Tabu search phase. See the package "bnlearn" for more details.
scoring The score value.
runtime The duration of the algorithm.

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

See Also
acor, alr, alfa

Examples

# simulate a dataset with continuous data
x <- rdiri(100, runif(20))
a <- compbn(log(x))

Description
Beta regression.

Usage
beta.reg(y, x, xnew = NULL)

Arguments
y
  The response variable. It must be a numerical vector with proportions excluding 0 and 1.

x
  The independent variable(s). It can be a vector, a matrix or a dataframe with continuous only variables, a data frame with mixed or only categorical variables.

xnew
  If you have new values for the predictor variables (dataset) whose response values you want to predict insert them here.

Details
Beta regression is fitted.

Value
A list including:

phi
  The estimated precision parameter.

info
  A matrix with the estimated regression parameters, their standard errors, Wald statistics and associated p-values.

loglik
  The log-likelihood of the regression model.

est
  The estimated values if xnew is not NULL.

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.
References

See Also
beta.est, prop.reg, diri.reg

Examples
y <- rbeta(300, 3, 5)
x <- matrix(rnorm(300 * 2), ncol = 2)
beta.reg(y, x)

Description
Choose the number of principal components via reconstruction error.

Usage
choose.pc(x, graph = TRUE)

Arguments
x A numerical matrix with more rows than columns.
graph Should the plot of the PRESS values appear? Default value is TRUE.

Details
The functions allows for selecting the number of eigenvectors via the reconstruction error which is computed for all eigenvectors based on SVD.

Value
A list including:
values The eigenvalues of the covariance matrix.
cumprop The cumulative proportion of the eigenvalues of the covariance matrix.
per The differences in the cumulative proportion of the eigenvalues of the covariance matrix.
press The reconstruction error \( \sqrt{\sum_{ij} (x_{ij} - \hat{x}_{ij})^2} \) for each number of eigenvectors.
runtime The runtime of the algorithm.
Column-wise MLE of some univariate distributions

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

pcr, alfa.pcr, alfapcr.tune

Examples

library(MASS)
  x <- as.matrix(fgl[, 2:9])
  a <- choose.pc(x, graph = FALSE)

Column-wise MLE of some univariate distributions

Description

Column-wise MLE of some univariate distributions.

Usage

colbeta.est(x, tol = 1e-07, maxiters = 100, parallel = FALSE)
collogitnorm.est(x)
colunitweibull.est(x, tol = 1e-07, maxiters = 100, parallel = FALSE)
colsimplex.est(x, tol = 1e-07)

Arguments

x A numerical matrix with data. Each column refers to a different vector of observations of the same distribution. The values must by percentages, excluding 0 and 1.
tol The tolerance value to terminate the Newton-Fisher algorithm.
maxiters The maximum number of iterations to implement.
parallel Do you want to calculations to take place in parallel? The default value is FALSE

Details

For each column, the same distribution is fitted and its parameters and log-likelihood are computed.
Constrained linear least squares for compositional responses and predictors

Value

A matrix with two or three columns. The first one or the first two contain the parameter(s) of the distribution and the second or third column the relevant log-likelihood.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also
censpois.mle, gammapois.mle

Examples

```r
x <- matrix( rbeta(200, 3, 4), ncol = 4 )
a <- colbeta.est(x)
```

---

Constrained linear least squares for compositional responses and predictors

Description

Constrained linear least squares for compositional responses and predictors.

Usage

```r
ols.compcomp(y, x, rs = 5, tol = 1e-4, xnew = NULL)
```
Constrained linear least squares for compositional responses and predictors

Arguments

- **y**: A matrix with the compositional data (dependent variable). Zero values are allowed.
- **x**: A matrix with the compositional predictors. Zero values are allowed.
- **rs**: The number of times to run the constrained optimisation using different random starting values each time.
- **tol**: The threshold upon which to stop the iterations of the constrained optimisation.
- **xnew**: If you have new data use it, otherwise leave it NULL.

Details

The function performs least squares regression where the beta coefficients are constrained to be positive and sum to 1. We were inspired by the transformation-free linear regression for compositional responses and predictors of Fiksel, Zeger and Datta (2020).

Value

A list including:

- **runtime**: The time required by the regression.
- **mse**: The mean squared errors.
- **be**: The beta coefficients.
- **est**: The fitted of xnew if xnew is not NULL.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

cv.olscompcomp, tflr, kl.alfapcr

Examples

```r
library(MASS)
set.seed(1234)
y <- rdiri(214, runif(4, 1, 3))
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
mod <- ols.compcomp(y, x, rs = 1)
mod
```
Contour plot of mixtures of Dirichlet distributions in $S^2$

Description

Contour plot of mixtures of Dirichlet distributions in $S^2$.

Usage

mixdiri.contour(a, prob, n = 100, x = NULL, cont.line = FALSE)

Arguments

a A matrix where each row contains the parameters of each Dirichlet distribution.
prob A vector with the mixing probabilities.
n The number of grid points to consider over which the density is calculated.
x This is either NULL (no data) or contains a 3 column matrix with compositional data.
cont.line Do you want the contour lines to appear? If yes, set this TRUE.

Details

The user can plot only the contour lines of a Dirichlet with a given vector of parameters, or can also add the relevant data should he/she wish to.

Value

A ternary diagram with the points and the Dirichlet contour lines.

Author(s)

Michail Tsagris and Christos Adam.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Christos Adam <pada4m4@gmail.com>.

References


See Also

diri.contour,compnorm.contour,bivt.contour,comp.kerncontour,mix.compnorm.contour,diri.nr,dda
Examples

```r
a <- matrix(c(12, 30, 45, 32, 50, 16), byrow = TRUE, ncol = 3)
prob <- c(0.5, 0.5)
mixdiri.contour(a, prob)
```

Description

Contour plot of the $\alpha$ multivariate normal in $S^2$.

Usage

```r
alfa.contour(m, s, a, n = 100, x = NULL, cont.line = FALSE)
```

Arguments

- **m**: The mean vector of the $\alpha$ multivariate normal model.
- **s**: The covariance matrix of the $\alpha$ multivariate normal model.
- **a**: The value of $a$ for the $\alpha$-transformation.
- **n**: The number of grid points to consider over which the density is calculated.
- **x**: This is either NULL (no data) or contains a 3 column matrix with compositional data.
- **cont.line**: Do you want the contour lines to appear? If yes, set this TRUE.

Details

The $\alpha$-transformation is applied to the compositional data and then for a grid of points within the 2-dimensional simplex, the density of the $\alpha$ multivariate normal is calculated and the contours are plotted.

Value

The contour plot of the $\alpha$ multivariate normal appears.

Author(s)

Michail Tsagris and Christos Adam.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Christos Adam <pada4m4@gmail.com>.
Contour plot of the alpha-folded model in $S^2$

References


See Also

folded.contour, compnorm.contour, diri.contour, mix.compnorm.contour, bivt.contour, skewnorm.contour

Examples

x <- as.matrix(iris[, 1:3])
x <- x / rowSums(x)
a <- a.est(x)$best
m <- colMeans(alfa(x, a)$aff)
s <- cov(alfa(x, a)$aff)
alfa.contour(m, s, a)

---

Contour plot of the alpha-folded model in $S^2$

Contour plot of the $\alpha$-folded model in $S^2$.

Description

Contour plot of the $\alpha$-folded model in $S^2$.

Usage

folded.contour(mu, su, p, a, n = 100, x = NULL, cont.line = FALSE)

Arguments

mu The mean vector of the folded model.
su The covariance matrix of the folded model.
p The probability inside the simplex of the folded model.
a The value of $\alpha$ for the $\alpha$-transformation.
n The number of grid points to consider over which the density is calculated.
x This is either NULL (no data) or contains a 3 column matrix with compositional data.
cont.line Do you want the contour lines to appear? If yes, set this TRUE.

Details

The $\alpha$-transformation is applied to the compositional data and then for a grid of points within the 2-dimensional simplex the folded model’s density is calculated and the contours are plotted.
Value

The contour plot of the folded model appears.

Author(s)

Michail Tsagris and Christos Adam.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Christos Adam <pada4m4@gmail.com>.

References


See Also

alfa.contour, compnorm.contour, diri.contour, mix.compnorm.contour, bivt.contour, skewnorm.contour

Examples

```r
x <- as.matrix(iris[, 1:3])
x <- x / rowSums(x)
a <- a.est(x)$best
mod <- alpha.mle(x, a)
folded.contour(mod$mu, mod$su, mod$p, a)
```

Description

Contour plot of the Dirichlet distribution in $S^2$.

Usage

diri.contour(a, n = 100, x = NULL, cont.line = FALSE)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>A vector with three elements corresponding to the 3 (estimated) parameters.</td>
</tr>
<tr>
<td>n</td>
<td>The number of grid points to consider over which the density is calculated.</td>
</tr>
<tr>
<td>x</td>
<td>This is either NULL (no data) or contains a 3 column matrix with compositional data.</td>
</tr>
<tr>
<td>cont.line</td>
<td>Do you want the contour lines to appear? If yes, set this TRUE.</td>
</tr>
</tbody>
</table>
Details

The user can plot only the contour lines of a Dirichlet with a given vector of parameters, or can also add the relevant data should he/she wish to.

Value

A ternary diagram with the points and the Dirichlet contour lines.

Author(s)

Michail Tsagris and Christos Adam.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Christos Adam <pada4m4@gmail.com>.

References


See Also

mixdiri.contour, compnorm.contour, bivt.contour, comp.kerncontour, mix.compnorm.contour

Examples

x <- as.matrix( iris[, 1:3] )
x <- x / rowSums(x)
diri.contour( a = c(3, 4, 2) )
Contour plot of the Flexible Dirichlet distribution in $S^2$

Arguments

- **alpha**: A vector of the non-negative $\alpha$ parameters.
- **prob**: A vector of the clusters' probabilities. It must sum to one.
- **tau**: The non-negative scalar $\tau$ parameter.
- **n**: The number of grid points to consider over which the density is calculated.
- **x**: This is either NULL (no data) or contains a 3 column matrix with compositional data.
- **cont.line**: Do you want the contour lines to appear? If yes, set this TRUE.

Details

The user can plot only the contour lines of a Dirichlet with a given vector of parameters, or can also add the relevant data should they wish to.

Value

A ternary diagram with the points and the Flexible Dirichlet contour lines.

Author(s)

Michail Tsagris and Christos Adam.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Christos Adam <pada4m4@gmail.com>.

References


See Also

- `fd.est`, `compnorm.contour`, `folded.contour`, `bivt.contour`, `comp.kerncontour`, `mix.compnorm.contour`

Examples

`fd.contour(alpha = c(10, 11, 12), prob = c(0.25, 0.25, 0.5), tau = 4)`
Contour plot of the Gaussian mixture model in $S^2$

Description

Contour plot of the Gaussian mixture model in $S^2$.

Usage

mix.compnorm.contour(mod, type = "alr", n = 100, x = NULL, cont.line = FALSE)

Arguments

- **mod**: An object containing the output of a `mix.compnorm` model.
- **type**: The type of trasformation used, either the additive log-ratio ("alr"), the isometric log-ratio ("ilr") or the pivot coordinate ("pivot") transformation.
- **n**: The number of grid points to consider over which the density is calculated.
- **x**: A matrix with the compositional data.
- **cont.line**: Do you want the contour lines to appear? If yes, set this TRUE.

Details

The contour plot of a Gaussian mixture model is plotted. For this you need the (fitted) model.

Value

A ternary plot with the data and the contour lines of the fitted Gaussian mixture model.

Author(s)

Michail Tsagris and Christos Adam.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Christos Adam <pada4m4@gmail.com>.

References


See Also

`mix.compnorm`, `bic.mixcomppnorm`, `diri.contour`
Contour plot of the Kent distribution in $S^2$

Examples

```r
x <- as.matrix(iris[, 1:3])
x <- x / rowSums(x)
mod <- mix.compnorm(x, 3, model = "EII")
mix.compnorm.contour(mod, "alr")
```

Description

Contour plot of the Kent distribution in $S^2$.

Usage

```r
kent.contour(G, param, n = 100, x = NULL, cont.line = FALSE)
```

Arguments

- **G**: A 3 x 3 matrix whose first column is the mean direction. The second and third columns are the major and minor axes respectively.
- **param**: A vector with the concentration $kappa$ and ovalness $\beta$ parameters (the $\psi$ parameter has been absorbed inside the matrix G).
- **n**: The number of grid points to consider over which the density is calculated.
- **x**: This is either NULL (no data) or contains a 3 column matrix with compositional data.
- **cont.line**: Do you want the contour lines to appear? If yes, set this TRUE.

Details

The user can plot only the contour lines of a Dirichlet with a given vector of parameters, or can also add the relevant data should they wish to.

Value

A ternary diagram with the points and the Dirichlet contour lines.

Author(s)

Michail Tsagris and Christos Adam.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Christos Adam <pada4m4@gmail.com>.
References


See Also

compnorm.contour, bivt.contour, comp.kerncontour, mix.compnorm.contour

Examples

```r
G <- matrix( c(0.05713382, 0.96029716, 0.27306608, -0.98809661,
0.01525976, 0.15307588, 0.1428314, -0.2785615, 0.9497382), ncol = 3 )
param <- c(2361.8401338, 1171.3808172, 0.1435577)
kent.contour(G, param)
```

Description

Contour plot of the kernel density estimate in \( S^2 \).

Usage

```r
comp.kerncontour(x, type = "alr", n = 50, cont.line = FALSE)
```

Arguments

- `x`: A matrix with the compositional data. It has to be a 3 column matrix.
- `type`: This is either "alr" or "ilr", corresponding to the additive and the isometric log-ratio transformation respectively.
- `n`: The number of grid points to consider, over which the density is calculated.
- `cont.line`: Do you want the contour lines to appear? If yes, set this TRUE.

Details

The alr or the ilr transformation are applied to the compositional data. Then, the optimal bandwidth using maximum likelihood cross-validation is chosen. The multivariate normal kernel density is calculated for a grid of points. Those points are the points on the 2-dimensional simplex. Finally the contours are plotted.

Value

A ternary diagram with the points and the kernel contour lines.
Author(s)

Michail Tsagris and Christos Adam.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Christos Adam <pada4m4@gmail.com>.

References


See Also

diri.contour, mix.compnorm.contour, bivt.contour, compnorm.contour

Examples

```r
x <- as.matrix(iris[, 1:3])
x <- x / rowSums(x)
comp.kerncontour(x, type = "alr", n = 20)
comp.kerncontour(x, type = "ilr", n = 20)
```

Description

Contour plot of the normal distribution in $S^2$.

Usage

```r
compnorm.contour(m, s, type = "alr", n = 100, x = NULL, cont.line = FALSE)
```

Arguments

- **m**: The mean vector.
- **s**: The covariance matrix.
- **type**: The type of transformation used, either the additive log-ratio ("alr"), the isometric log-ratio ("ilr") or the pivot coordinate ("pivot") transformation.
- **n**: The number of grid points to consider over which the density is calculated.
- **x**: This is either NULL (no data) or contains a 3 column matrix with compositional data.
- **cont.line**: Do you want the contour lines to appear? If yes, set this TRUE.
Details

The alr or the ilr transformation is applied to the compositional data at first. Then for a grid of points within the 2-dimensional simplex the bivariate normal density is calculated and the contours are plotted along with the points.

Value

A ternary diagram with the points (if appear = TRUE) and the bivariate normal contour lines.

Author(s)

Michail Tsagris and Christos Adam.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Christos Adam <pada4m4@gmail.com>.

See Also

diri.contour,mix.compnorm.contour,bivt.contour,skewnorm.contour

Examples

```r
x <- as.matrix(iris[, 1:3])
x <- x / rowSums(x)
y <- Compositional::alr(x)
m <- colMeans(y)
s <- cov(y)
compnorm.contour(m, s)
```

Description

Contour plot of the skew skew-normal distribution in $S^2$.

Usage

```r
skewnorm.contour(x, type = "alr", n = 100, appear = TRUE, cont.line = FALSE)
```

Arguments

- **x**: A matrix with the compositional data. It has to be a 3 column matrix.
- **type**: This is either "alr" or "ilr", corresponding to the additive and the isometric log-ratio transformation respectively.
- **n**: The number of grid points to consider over which the density is calculated.
- **appear**: Should the available data appear on the ternary plot (TRUE) or not (FALSE)?
- **cont.line**: Do you want the contour lines to appear? If yes, set this TRUE.
Details

The alr or the ilr transformation is applied to the compositional data at first. Then for a grid of points within the 2-dimensional simplex the bivariate skew skew-normal density is calculated and the contours are plotted along with the points.

Value

A ternary diagram with the points (if appear = TRUE) and the bivariate skew skew-normal contour lines.

Author(s)

Michail Tsagris and Christos Adam.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Christos Adam <pada4m4@gmail.com>.

References


See Also

diri.contour, mix.compnorm.contour, bivt.contour, compnorm.contour

Examples

```r
x <- as.matrix(iris[51:100, 1:3])
x <- x / rowSums(x)
skewnorm.contour(x)
```

```r

Contour plot of the t distribution in S^2

---

Description

Contour plot of the t distribution in S^2.

Usage

bivt.contour(x, type = "alr", n = 100, appear = TRUE, cont.line = FALSE)
```
Contour plot of the t distribution in $S^2$

**Arguments**

- **x**: A matrix with compositional data. It has to be a 3 column matrix.
- **type**: This is either "alr" or "ilr", corresponding to the additive and the isometric log-ratio transformation respectively.
- **n**: The number of grid points to consider over which the density is calculated.
- **appear**: Should the available data appear on the ternary plot (TRUE) or not (FALSE)?
- **cont.line**: Do you want the contour lines to appear? If yes, set this TRUE.

**Details**

The alr or the ilr transformation is applied to the compositional data at first and the location, scatter and degrees of freedom of the bivariate t distribution are computed. Then for a grid of points within the 2-dimensional simplex the bivariate t density is calculated and the contours are plotted along with the points.

**Value**

A ternary diagram with the points (if appear = TRUE) and the bivariate t contour lines.

**Author(s)**

Michail Tsagris and Christos Adam.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Christos Adam <pada4m4@gmail.com>.

**References**


**See Also**

diri.contour, mix.compnorm.contour, compnorm.contour, skewnorm.contour

**Examples**

```r
x <- as.matrix( iris[, 1:3] )
x <- x / rowSums(x)
bivt.contour(x)
bivt.contour(x, type = "ilr")
```
Cross validation for some compositional regression models

Description

Cross validation for some compositional regression models.

Usage

```
cv.comp.reg(y, x, type = "comp.reg", nfolds = 10, folds = NULL, seed = NULL)
```

Arguments

- `y`: A matrix with compositional data. Zero values are allowed for some regression models.
- `x`: The predictor variable(s).
- `type`: This can be one of the following: "comp.reg", "robust", "kl.compreg", "js.compreg", "diri.reg" or "zadr".
- `nfolds`: The number of folds to be used. This is taken into consideration only if the folds argument is not supplied.
- `folds`: If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.
- `seed`: If seed is TRUE the results will always be the same.

Details

A k-fold cross validation for a compositional regression model is performed.

Value

A list including:

- `runtime`: The runtime of the cross-validation procedure.
- `kl`: The Kullback-Leibler divergences for all runs.
- `js`: The Jensen-Shannon divergences for all runs.
- `perf`: The average Kullback-Leibler divergence and average Jensen-Shannon divergence.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.
Cross validation for the alpha-k-NN regression with compositional predictor variables

See Also

comp.reg, kl.compreg, compppr.tune, aknnreg.tune

Examples

```r
y <- as.matrix( iris[, 1:3] )
y <- y / rowSums(y)
x <- iris[, 4]
mod <- cv.comp.reg(y, x)
```

Description

Cross validation for the \( \alpha \)-k-NN regression with compositional predictor variables.

Usage

```
alfaknnreg.tune(y, x, a = seq(-1, 1, by = 0.1), k = 2:10, nfolds = 10,
apostasi = "euclidean", method = "average", folds = NULL, seed = NULL, graph = FALSE)
```

Arguments

- `y` The response variable, a numerical vector.
- `x` A matrix with the available compositional data. Zeros are allowed.
- `a` A vector with a grid of values of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If \( \alpha = 0 \) the isometric log-ratio transformation is applied.
- `k` The number of nearest neighbours to consider. It can be a single number or a vector.
- `nfolds` The number of folds. Set to 10 by default.
- `apostasi` The type of distance to use, either "euclidean" or "manhattan".
- `method` If you want to take the average of the responses of the k closest observations, type "average". For the median, type "median" and for the harmonic mean, type "harmonic".
- `folds` If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.
- `seed` If seed is TRUE the results will always be the same.
- `graph` If graph is TRUE (default value) a filled contour plot will appear.
Cross validation for the alpha-k-NN regression with compositional response data

Details
A k-fold cross validation for the $\alpha$-k-NN regression for compositional response data is performed.

Value
A list including:
- mspe: The mean square error of prediction.
- performance: The minimum mean square error of prediction.
- opt_a: The optimal value of $\alpha$.
- opt_k: The optimal value of $k$.
- runtime: The runtime of the cross-validation procedure.

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

See Also
alfa.rda, alfa.fda, rda.tune

Examples
library(MASS)
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
y <- fgl[, 1]
mod <- alfa.knnreg.tune(y, x, a = seq(0.2, 0.4, by = 0.1), k = 2:4, nfolds = 5)

Cross validation for the alpha-k-NN regression with compositional response data

Description
Cross validation for the $\alpha$-k-NN regression with compositional response data.

Usage
aknnreg.tune(y, x, a = seq(0.1, 1, by = 0.1), k = 2:10, apostasi = "euclidean", nfolds = 10, folds = NULL, seed = NULL, rann = FALSE)
Cross validation for the alpha-k-NN regression with compositional response data

Arguments

y A matrix with the compositional response data. Zeros are allowed.
x A matrix with the available predictor variables.
a A vector with a grid of values of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If $\alpha = 0$ the isometric log-ratio transformation is applied.
k The number of nearest neighbours to consider. It can be a single number or a vector.
apostasi The type of distance to use, either "euclidean" or "manhattan".
nfolds The number of folds. Set to 10 by default.
folds If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.
seed You can specify your own seed number here or leave it NULL.
rann If you have large scale datasets and want a faster k-NN search, you can use kd-trees implemented in the R package "RANN". In this case you must set this argument equal to TRUE. Note however, that in this case, the only available distance is by default "euclidean".

Details

A k-fold cross validation for the $\alpha$-k-NN regression for compositional response data is performed.

Value

A list including:

kl The Kullback-Leibler divergence for all combinations of $\alpha$ and $k$.
js The Jensen-Shannon divergence for all combinations of $\alpha$ and $k$.
klmin The minimum Kullback-Leibler divergence.
jsmin The minimum Jensen-Shannon divergence.
kl.alpha The optimal $\alpha$ that leads to the minimum Kullback-Leibler divergence.
kl.k The optimal $k$ that leads to the minimum Kullback-Leibler divergence.
js.alpha The optimal $\alpha$ that leads to the minimum Jensen-Shannon divergence.
js.k The optimal $k$ that leads to the minimum Jensen-Shannon divergence.
runtime The runtime of the cross-validation procedure.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

Cross validation for the alpha-kernel regression with compositional response data

See Also

aknn.reg, akernreg.tune, akern.reg, alfa.rda, alfa.fda, rda.tune

Examples

```r
y <- as.matrix(iris[, 1:3])
y <- y / rowSums(y)
x <- iris[, 4]
mod <- aknnreg.tune(y, x, a = c(0.4, 0.6), k = 2:4, nfolds = 5)
```

Description

Cross validation for the \( \alpha \)-kernel regression with compositional response data.

Usage

```r
akernreg.tune(y, x, a = seq(0.1, 1, by = 0.1), h = seq(0.1, 1, length = 10),
type = "gauss", nfolds = 10, folds = NULL, seed = NULL)
```

Arguments

- `y`: A matrix with the compositional response data. Zeros are allowed.
- `x`: A matrix with the available predictor variables.
- `a`: A vector with a grid of values of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If \( \alpha = 0 \) the isometric log-ratio transformation is applied.
- `h`: A vector with the bandwidth value(s) to consider.
- `type`: The type of kernel to use, "gauss" or "laplace".
- `nfolds`: The number of folds. Set to 10 by default.
- `folds`: If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.
- `seed`: You can specify your own seed number here or leave it NULL.

Details

A k-fold cross validation for the \( \alpha \)-kernel regression for compositional response data is performed.
Cross validation for the kernel regression with Euclidean response data

Value

A list including:

- k1: The Kullback-Leibler divergence for all combinations of \( \alpha \) and \( h \).
- js: The Jensen-Shannon divergence for all combinations of \( \alpha \) and \( h \).
- klmin: The minimum Kullback-Leibler divergence.
- jsmin: The minimum Jensen-Shannon divergence.
- kl.alpha: The optimal \( \alpha \) that leads to the minimum Kullback-Leibler divergence.
- kl.h: The optimal \( h \) that leads to the minimum Kullback-Leibler divergence.
- js.alpha: The optimal \( \alpha \) that leads to the minimum Jensen-Shannon divergence.
- js.h: The optimal \( h \) that leads to the minimum Jensen-Shannon divergence.
- runtime: The runtime of the cross-validation procedure.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

akern.reg, aknnreg.tune, aknn.reg, alfa.rda, alfa.fda, rda.tune

Examples

```r
y <- as.matrix( iris[, 1:3] )
y <- y / rowSums(y)
x <- iris[, 4]
mod <- akernreg.tune(y, x, a = c(0.4, 0.6), h = c(0.1, 0.2), nfolds = 5)
```

Cross validation for the kernel regression with Euclidean response data

Cross validation for the kernel regression with Euclidean response data

Description

Cross validation for the kernel regression with Euclidean response data.
Cross validation for the kernel regression with Euclidean response data

Usage
kernreg.tune(y, x, h = seq(0.1, 1, length = 10), type = "gauss",
            nfolds = 10, folds = NULL, seed = NULL, graph = FALSE, ncores = 1)

Arguments
y A matrix or a vector with the Euclidean response.
x A matrix with the available predictor variables.
h A vector with the bandwidth value(s) \( h \) to consider.
type The type of kernel to use, "gauss" or "laplace".
nfolds The number of folds. Set to 10 by default.
folds If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.
seed You can specify your own seed number here or leave it NULL.
graph If graph is TRUE (default value) a plot will appear.
ncores The number of cores to use. Default value is 1.

Details
A k-fold cross validation for the kernel regression with a euclidean response is performed.

Value
A list including:
mspe The mean squared prediction error (MSPE) for each fold and value of \( h \).
h The optimal \( h \) that leads to the minimum MSPE.
performance The minimum MSPE.
runtime The runtime of the cross-validation procedure.

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

See Also
kern.reg, aknnreg.tune, aknn.reg

Examples
y <- iris[, 1]
x <- iris[, 2:4]
mod <- kernreg.tune(y, x, h = c(0.1, 0.2, 0.3))
Cross validation for the regularised and flexible discriminant analysis with compositional data using the alpha-transformation

Description

Cross validation for the regularised and flexible discriminant analysis with compositional data using the $\alpha$-transformation.

Usage

```r
alfarda.tune(x, ina, a = seq(-1, 1, by = 0.1), nfolds = 10, gam = seq(0, 1, by = 0.1), del = seq(0, 1, by = 0.1), ncores = 1, folds = NULL, stratified = TRUE, seed = NULL)
alfafda.tune(x, ina, a = seq(-1, 1, by = 0.1), nfolds = 10, folds = NULL, stratified = TRUE, seed = NULL, graph = FALSE)
```

Arguments

- `x`: A matrix with the available compositional data. Zeros are allowed.
- `ina`: A group indicator variable for the avaialble data.
- `a`: A vector with a grid of values of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If $\alpha = 0$ the isometric log-ratio transformation is applied.
- `nfolds`: The number of folds. Set to 10 by default.
- `gam`: A vector of values between 0 and 1. It is the weight of the pooled covariance and the diagonal matrix.
- `del`: A vector of values between 0 and 1. It is the weight of the LDA and QDA.
- `ncores`: The number of cores to use. If it is more than 1 parallel computing is performed. It is advisable to use it if you have many observations and or many variables, otherwise it will slow down th process.
- `folds`: If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.
- `stratified`: Do you want the folds to be created in a stratified way? TRUE or FALSE.
- `seed`: You can specify your own seed number here or leave it NULL.
- `graph`: If graph is TRUE (default value) a filled contour plot will appear.

Details

A k-fold cross validation is performed.
Cross validation for the regularised and flexible discriminant analysis with compositional data using the alpha-transformation

Value

For the alfa.rda a list including:

- **res** The estimated optimal rate and the best values of $\alpha$, $\gamma$, and $\delta$.
- **percent** For the best value of $\alpha$ the averaged over all folds best rates of correct classification. It is a matrix, where rows correspond to the $\gamma$ values and columns correspond to $\delta$ values.
- **se** The estimated standard errors of the "percent" matrix.
- **runtime** The runtime of the cross-validation procedure.

For the alfa.fda a list including:

- **per** The performance of the fda in each fold for each value of $\alpha$.
- **performance** The average performance for each value of $\alpha$.
- **opt_a** The optimal value of $\alpha$.
- **runtime** The runtime of the cross-validation procedure.

Author(s)

Michail Tsagris.

R implementation and documentation: Giorgos Athineou <gioathineou@gmail.com> and Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

alfa.rda, alfanb.tune, cv.dda, compknn.tune, rda.tune, cv.compnb

Examples

```r
library(MASS)
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
ina <- fgl[, 10]
moda <- alfarda.tune(x, ina, a = seq(0.7, 1, by = 0.1), nfolds = 10,
gam = seq(0.1, 0.3, by = 0.1), del = seq(0.1, 0.3, by = 0.1) )
```
Cross validation for the ridge regression

Description

Cross validation for the ridge regression is performed. There is an option for the GCV criterion which is automatic.

Usage

```r
ridge.tune(y, x, nfolds = 10, lambda = seq(0, 2, by = 0.1), folds = NULL, ncores = 1, seed = NULL, graph = FALSE)
```

Arguments

- `y`: A numeric vector containing the values of the target variable. If the values are proportions or percentages, i.e. strictly within 0 and 1 they are mapped into R using the logit transformation.
- `x`: A numeric matrix containing the variables.
- `nfolds`: The number of folds in the cross validation.
- `lambda`: A vector with the a grid of values of λ to be used.
- `folds`: If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.
- `ncores`: The number of cores to use. If it is more than 1 parallel computing is performed.
- `seed`: You can specify your own seed number here or leave it NULL.
- `graph`: If graph is set to TRUE the performances for each fold as a function of the λ values will appear.

Details

A k-fold cross validation is performed. This function is used by `alfaridge.tune`.

Value

A list including:

- `msp`: The performance of the ridge regression for every fold.
- `mspe`: The values of the mean prediction error for each value of λ.
- `lambda`: The value of λ which corresponds to the minimum MSPE.
- `performance`: The minimum MSPE.
- `runtime`: The time required by the cross-validation procedure.
Cross validation for the ridge regression with compositional data as predictor using the alpha-transformation

Author(s)
Michail Tsagris.
R implementation and documentation: Giorgos Athineou <gioathineou@gmail.com> and Michail Tsagris <mtsagris@uoc.gr>.

References

See Also
ridge.reg, alfaridge.tune

Examples
```r
y <- as.vector(iris[, 1])
x <- as.matrix(iris[, 2:4])
ridge.tune( y, x, nfolds = 10, lambda = seq(0, 2, by = 0.1), graph = TRUE )
```

Description
Cross validation for the ridge regression is performed. There is an option for the GCV criterion which is automatic. The predictor variables are compositional data and the $\alpha$-transformation is applied first.

Usage
```r
alfaridge.tune(y, x, nfolds = 10, a = seq(-1, 1, by = 0.1),
lambda = seq(0, 2, by = 0.1), folds = NULL, ncores = 1,
graph = TRUE, col.nu = 15, seed = NULL)
```

Arguments
- `y`: A numeric vector containing the values of the target variable. If the values are proportions or percentages, i.e. strictly within 0 and 1 they are mapped into R using the logit transformation.
- `x`: A numeric matrix containing the compositional data, i.e. the predictor variables. Zero values are allowed.
- `nfolds`: The number of folds in the cross validation.
Cross validation for the ridge regression with compositional data as predictor using the alpha-transformation

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>a</code></td>
<td>A vector with the a grid of values of $\alpha$ to be used.</td>
</tr>
<tr>
<td><code>lambda</code></td>
<td>A vector with the a grid of values of $\lambda$ to be used.</td>
</tr>
<tr>
<td><code>folds</code></td>
<td>If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.</td>
</tr>
<tr>
<td><code>ncores</code></td>
<td>The number of cores to use. If it is more than 1 parallel computing is performed. It is advisable to use it if you have many observations and or many variables, otherwise it will slow down the process.</td>
</tr>
<tr>
<td><code>graph</code></td>
<td>If graph is TRUE (default value) a filled contour plot will appear.</td>
</tr>
<tr>
<td><code>col.nu</code></td>
<td>A number parameter for the filled contour plot, taken into account only if graph is TRUE.</td>
</tr>
<tr>
<td><code>seed</code></td>
<td>You can specify your own seed number here or leave it NULL.</td>
</tr>
</tbody>
</table>

**Details**

A k-fold cross validation is performed.

**Value**

If graph is TRUE a filled contour a filled contour will appear. A list including:

- `mspe` : The MSPE where rows correspond to the $\alpha$ values and the columns to the number of principal components.
- `best.par` : The best pair of $\alpha$ and $\lambda$.
- `performance` : The minimum mean squared error of prediction.
- `runtime` : The run time of the cross-validation procedure.

**Author(s)**

Michail Tsagris.

R implementation and documentation: Giorgos Athineou <gioathineou@gmail.com> and Michail Tsagris <mtsagris@uoc.gr>.

**References**


**See Also**

`alfa.ridge`, `ridge.tune`
Cross validation for the transformation-free linear regression for compositional responses and predictors

Examples

```r
library(MASS)
y <- as.vector(fgl[, 1])
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
alfaridge.tune( y, x, nfolds = 10, a = seq(0.1, 1, by = 0.1),
               lambda = seq(0, 1, by = 0.1) )
```

Description

Cross validation for the transformation-free linear regression for compositional responses and predictors.

Usage

```r
cv.tflr(y, x, nfolds = 10, folds = NULL, seed = NULL)
```

Arguments

- `y`: A matrix with compositional response data. Zero values are allowed.
- `x`: A matrix with compositional predictors. Zero values are allowed.
- `nfolds`: The number of folds to be used. This is taken into consideration only if the folds argument is not supplied.
- `folds`: If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.
- `seed`: If seed is TRUE the results will always be the same.

Details

A k-fold cross validation for the transformation-free linear regression for compositional responses and predictors is performed.

Value

A list including:

- `runtime`: The runtime of the cross-validation procedure.
- `kl`: The Kullback-Leibler divergences for all runs.
- `js`: The Jensen-Shannon divergences for all runs.
- `perf`: The average Kullback-Leibler divergence and average Jensen-Shannon divergence.
Cross-validation for the constrained linear least squares for compositional responses and predictors

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

See Also
tflr, cv.olscompcomp, klalfapcr.tune

Examples

library(MASS)
y <- rdiri(214, runif(3, 1, 3))
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
mod <- cv.tflr(y, x)
mod

Description
Cross-validation for the constrained linear least squares for compositional responses and predictors.

Usage

cv.olscompcomp(y, x, rs = 5, tol = 1e-4, nfolds = 10,
folds = NULL, seed = NULL)

Arguments

y A matrix with compositional response data. Zero values are allowed.
x A matrix with compositional predictors. Zero values are allowed.
rs The number of times to run the constrained optimisation using different random starting values each time.
tol The threshold upon which to stop the iterations of the constrained optimisation.
nfolds The number of folds to be used. This is taken into consideration only if the folds argument is not supplied.
folds If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.
seed You can specify your own seed number here or leave it NULL.
Cross-validation for the Dirichlet discriminant analysis

Details
The function performs k-fold cross-validation for the least squares regression where the beta coefficients are constrained to be positive and sum to 1.

Value
A list including:
- runtime: The runtime of the cross-validation procedure.
- kl: The Kullback-Leibler divergences for all runs.
- js: The Jensen-Shannon divergences for all runs.
- perf: The average Kullback-Leibler divergence and average Jensen-Shannon divergence.

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

See Also
ols.compcomp, cv.tflr, klalfapcr.tune

Examples

```r
library(MASS)
set.seed(1234)
y <- rdiri(214, runif(3, 1, 3))
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
mod <- cv.olscompcomp(y, x, rs = 1, tol = 1e-4, nfolds = 5, seed = 12345)
mod
```

Description
Cross-validation for the Dirichlet discriminant analysis.

Usage
```r
cv.dda(x, ina, nfolds = 10, folds = NULL, stratified = TRUE, seed = NULL)
```
Cross-validation for the Dirichlet discriminant analysis

Arguments

x A matrix with the available data, the predictor variables.
ina A vector of data. The response variable, which is categorical (factor is acceptable).
folds A list with the indices of the folds.
nfolds The number of folds to be used. This is taken into consideration only if "folds" is NULL.
stratified Do you want the folds to be selected using stratified random sampling? This preserves the analogy of the samples of each group. Make this TRUE if you wish.
seed If you set this to TRUE, the same folds will be created every time.

Details

This function estimates the performance of the Dirichlet discriminant analysis via k-fold cross-validation.

Value

A list including:

percent The percentage of correct classification
runtime The duration of the cross-validation procedure.

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

dda, alfanb.tune, alfarda.tune, compknn.tune, cv.compnb

Examples

x <- as.matrix(iris[, 1:4])
x <- x / rowSums(x)
mod <- cv.dda(x, ina = iris[, 5] )
Cross-validation for the LASSO Kullback-Leibler divergence based regression

Description

Cross-validation for the LASSO Kullback-Leibler divergence based regression.

Usage

```R
cv.lasso.klcomreg(y, x, alpha = 1, type = "grouped", nfolds = 10,
                  folds = NULL, seed = NULL, graph = FALSE)
```

Arguments

- `y` A numerical matrix with compositional data with or without zeros.
- `x` A matrix with the predictor variables.
- `alpha` The elastic net mixing parameter, with $0 \leq \alpha \leq 1$. The penalty is defined as a weighted combination of the ridge and of the Lasso regression. When $\alpha = 1$ LASSO is applied, while $\alpha = 0$ yields the ridge regression.
- `type` This information is copied from the package glmnet. If "grouped" then a grouped lasso penalty is used on the multinomial coefficients for a variable. This ensures they are all in our out together. The default in our case is "grouped".
- `nfolds` The number of folds for the K-fold cross validation, set to 10 by default.
- `folds` If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.
- `seed` You can specify your own seed number here or leave it NULL.
- `graph` If graph is TRUE (default value) a filled contour plot will appear.

Details

The K-fold cross validation is performed in order to select the optimal value for $\lambda$, the penalty parameter in LASSO.

Value

The outcome is the same as in the R package glmnet. The extra addition is that if "graph = TRUE", then the plot of the cross-validated object is returned. The contains the logarithm of $\lambda$ and the deviance. The numbers on top of the figure show the number of set of coefficients for each component, that are not zero.

Author(s)

Michail Tsagris and Abdulaziz Alenazi.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Abdulaziz Alenazi <a.alenazi@nbu.edu.sa>.
Cross-validation for the LASSO log-ratio regression with compositional response

References

See Also
lasso.klcompreg, lassocoef.plot, lasso.compreg, cv.lasso.compreg, kl.compreg

Examples
library(MASS)
y <- rdiri(214, runif(4, 1, 3))
x <- as.matrix(fgl[, 2:9])
mod <- cv.lasso.klcompreg(y, x)

Description
Cross-validation for the LASSO log-ratio regression with compositional response.

Usage
cv.lasso.compreg(y, x, alpha = 1, nfolds = 10,
folds = NULL, seed = NULL, graph = FALSE)

Arguments

- **y**: A numerical matrix with compositional data. Zero values are not allowed as the additive log-ratio transformation (`alr`) is applied to the compositional response prior to implementing the LASSO algorithm.
- **x**: A matrix with the predictor variables.
- **alpha**: The elastic net mixing parameter, with $0 \leq \alpha \leq 1$. The penalty is defined as a weighted combination of the ridge and of the Lasso regression. When $\alpha = 1$ LASSO is applied, while $\alpha = 0$ yields the ridge regression.
- **nfolds**: The number of folds for the K-fold cross validation, set to 10 by default.
- **folds**: If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.
- **seed**: You can specify your own seed number here or leave it NULL.
- **graph**: If graph is TRUE (default value) a filled contour plot will appear.
Cross-validation for the naive Bayes classifiers for compositional data

Details

The K-fold cross validation is performed in order to select the optimal value for $\lambda$, the penalty parameter in LASSO.

Value

The outcome is the same as in the R package glmnet. The extra addition is that if "graph = TRUE", then the plot of the cross-validated object is returned. The contains the logarithm of $\lambda$ and the mean squared error. The numbers on top of the figure show the number of set of coefficients for each component, that are not zero.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

lasso.compreg, lasso.klcompreg, lasso.coef.plot, cv.lasso.klcompreg, comp.reg

Examples

library(MASS)
y <- rdiri(214, runif(4, 1, 3))
x <- as.matrix(fgl[, 2:9])
mod <- cv.lasso.compreg(y, x)
Cross-validation for the naive Bayes classifiers for compositional data

Arguments

x  A matrix with the available data, the predictor variables.
ina A vector of data. The response variable, which is categorical (factor is acceptable).
type The type of naive Bayes, "beta", "logitnorm", "cauchy", "laplace", "gamma", "normlog" or "weibull". For the last 4 distributions, the negative of the logarithm of the compositional data is applied first.
folds A list with the indices of the folds.
nfolds The number of folds to be used. This is taken into consideration only if "folds" is NULL.
stratified Do you want the folds to be selected using stratified random sampling? This preserves the analogy of the samples of each group. Make this TRUE if you wish.
seed You can specify your own seed number here or leave it NULL.
pred.ret If you want the predicted values returned set this to TRUE.

Value

A list including:
preds If pred.ret is TRUE the predicted values for each fold are returned as elements in a list.
crit A vector whose length is equal to the number of k and is the accuracy metric for each k. For the classification case it is the percentage of correct classification.

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

comp.nb

Examples

x <- as.matrix(iris[, 1:4])
x <- x / rowSums(x)
mod <- cv.compnb(x, ina = iris[, 5] )
Cross-validation for the naive Bayes classifiers for compositional data using the α-transformation

Description

Cross-validation for the naive Bayes classifiers for compositional data using the α-transformation.

Usage

```r
alfanb.tune(x, ina, a = seq(-1, 1, by = 0.1), type = "gaussian", folds = NULL, nfolds = 10, stratified = TRUE, seed = NULL)
```

Arguments

- `x`: A matrix with the available data, the predictor variables.
- `ina`: A vector of data. The response variable, which is categorical (factor is acceptable).
- `a`: This can be a vector of values or a single number.
- `type`: The type of naive Bayes, "gaussian", "cauchy" or "laplace".
- `folds`: A list with the indices of the folds.
- `nfolds`: The number of folds to be used. This is taken into consideration only if "folds" is NULL.
- `stratified`: Do you want the folds to be selected using stratified random sampling? This preserves the analogy of the samples of each group. Make this TRUE if you wish.
- `seed`: You can specify your own seed number here or leave it NULL.

Details

This function estimates the performance of the naive Bayes classifier for each value of α of the α-transformation.

Value

A list including:

- `crit`: A vector whose length is equal to the number of k and is the accuracy metric for each k. For the classification case it is the percentage of correct classification.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.
Density of compositional data from Gaussian mixture models

References

See Also
alfa.nb, alfada.tune, compknn.tune, cv.dia, cv.compnb

Examples
```r
x <- as.matrix(iris[, 1:4])
x <- x / rowSums(x)
mod <- alfanb.tune(x, ina = iris[, 5], a = c(0, 0.1, 0.2) )
```

## Density of compositional data from Gaussian mixture models

Simulation of compositional data from Gaussian mixture models.

### Description
Simulation of compositional data from Gaussian mixture models.

### Usage
```r
dmix.compnorm(x, mu, sigma, prob, type = "alr", logged = TRUE)
```

### Arguments
- `x`: A vector or a matrix with compositional data.
- `prob`: A vector with mixing probabilities. Its length is equal to the number of clusters.
- `mu`: A matrix where each row corresponds to the mean vector of each cluster.
- `sigma`: An array consisting of the covariance matrix of each cluster.
- `type`: The type of transformation used, either the additive log-ratio ("alr"), the isometric log-ratio ("ilr") or the pivot coordinate ("pivot") transformation.
- `logged`: A boolean variable specifying whether the logarithm of the density values to be returned. It is set to TRUE by default.

### Details
A sample from a multivariate Gaussian mixture model is generated.

### Value
A vector with the density values.
Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

See Also
mix.compnorm, bic.mixcompnorm

Examples
```r
p <- c(1/3, 1/3, 1/3)
mu <- matrix(nrow = 3, ncol = 4)
s <- array(dim = c(4, 4, 3))
x <- as.matrix(iris[, 1:4])
ina <- as.numeric(iris[, 5])
mu <- rowsum(x, ina) / 50
s[, , 1] <- cov(x[ina == 1, ])
s[, , 2] <- cov(x[ina == 2, ])
s[, , 3] <- cov(x[ina == 3, ])
y <- rmixcomp(100, p, mu, s, type = "alr")$x
mod <- dmix.compnorm(y, mu, s, p)
```

Description
Density of the Flexible Dirichlet distribution

Usage
dfd(x, alpha, prob, tau)

Arguments

- **x**: A vector or a matrix with compositional data.
- **alpha**: A vector of the non-negative $\alpha$ parameters.
- **prob**: A vector of the clusters’ probabilities. It must sum to one.
- **tau**: The non-negative scalar $\tau$ parameter.
Density of the folded normal distribution

Details

For more information see the references.

Value

The density value(s).

Author(s)

Michail Tsagris ported from the R package FlexDir. <mtsagris@uoc.gr>.

References


See Also

fd.est, rfd

Examples

alpha <- c(12, 11, 10)
prob <- c(0.25, 0.25, 0.5)
tau <- 8
x <- rfd(20, alpha, prob, tau)
dfd(x, alpha, prob, tau)
Density of the folded normal distribution

Arguments

- **x**: A vector or a matrix with compositional data. No zeros are allowed.
- **a**: The value of $\alpha$.
- **p**: The probability inside the simplex of the folded model.
- **mu**: The mean vector.
- **su**: The covariance matrix.
- **logged**: A boolean variable specifying whether the logarithm of the density values to be returned. It is set to TRUE by default.

Details

Density values of the folded model.

Value

The density value(s).

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

rfolded, a.est, folded.contour

Examples

```r
s <- c(0.1490676523, -0.4580818209, 0.0020395316, -0.0047446076, -0.4580818209, 1.5227259250, 0.0002596411, 0.0074836251, 0.0020395316, 0.0002596411, 0.0365384838, -0.0471448849, -0.0047446076, 0.0074836251, -0.0471448849, 0.0611442781)
s <- matrix(s, ncol = 4)
m <- c(1.715, 0.914, 0.115, 0.167)
x <- rfolded(100, m, s, 0.5)
mod <- a.est(x)
den <- dfolded(x, mod$best, mod$p, mod$mu, mod$su)
```
Density values of a Dirichlet distribution

Description
Density values of a Dirichlet distribution.

Usage
ddiri(x, a, logged = TRUE)

Arguments
x A matrix containing compositional data. This can be a vector or a matrix with the data.
a A vector of parameters. Its length must be equal to the number of components, or columns of the matrix with the compositional data and all values must be greater than zero.
logged A boolean variable specifying whether the logarithm of the density values to be returned. It is set to TRUE by default.

Details
The density of the Dirichlet distribution for a vector or a matrix of compositional data is returned.

Value
A vector with the density values.

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

See Also
diri.nr, diri.est, diri.contour, rdiri, dda
Examples

```r
x <- rdiri(100, c(5, 7, 4, 8, 10, 6, 4))
a <- diri.est(x)
f <- ddiri(x, a$param)
sum(f)
a
```

Density values of a mixture of Dirichlet distributions

Description

Density values of a mixture of Dirichlet distributions.

Usage

```r
dmixdiri(x, a, prob, logged = TRUE)
```

Arguments

- `x`: A vector or a matrix with compositional data. Zeros are not allowed.
- `a`: A matrix where each row contains the parameters of each Dirichlet component.
- `prob`: A vector with the mixing probabilities.
- `logged`: A boolean variable specifying whether the logarithm of the density values to be returned. It is set to TRUE by default.

Details

The density of the mixture of Dirichlet distribution for a vector or a matrix of compositional data is returned.

Value

A vector with the density values.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

Dirichlet discriminant analysis

See Also
rmixdiri, mixdiri.contour

Examples

a <- matrix( c(12, 30, 45, 32, 50, 16), byrow = TRUE, ncol = 3)
prob <- c(0.5, 0.5)
x <- rmixdiri(100, a, prob)$x
f <- dmixdiri(x, a, prob)

Description
Dirichlet discriminant analysis.

Usage
dda(xnew, x, ina)

Arguments

xnew A matrix with the new compositional predictor data whose class you want to predict. Zeros are allowed.
x A matrix with the available compositional predictor data. Zeros are allowed.
ing A vector of data. The response variable, which is categorical (factor is acceptable).

Details
The function performs maximum likelihood discriminant analysis using the Dirichlet distribution.

Value
A vector with the estimated group.

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>. 
Dirichlet random values simulation

References


See Also
cv.dda, comp.nb, alfa.rda, alfa.knn, comp.knn, mix.compnorm, diri.reg, zadr

Examples

```r
x <- Compositional::rdiri(100, runif(5) )
ina <- rbinom(100, 1, 0.5) + 1
mod <- dda(x, x, ina )
```

Description

Dirichlet random values simulation.

Usage

`rdiri(n, a)`

Arguments

- `n` The sample size, a numerical value.
- `a` A numerical vector with the parameter values.

Details

The algorithm is straightforward, for each vector, independent gamma values are generated and then divided by their total sum.

Value

A matrix with the simulated data.
**Dirichlet regression**

**Author(s)**
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

**References**

**See Also**
diri.est, diri.nr, diri.contour

**Examples**
```r
x <- rdiri(100, c(5, 7, 1, 3, 10, 2, 4))
diri.est(x)
```

---

**Dirichlet regression**

**Description**
Dirichlet regression.

**Usage**
```r
diri.reg(y, x, plot = FALSE, xnew = NULL)
diri.reg2(y, x, xnew = NULL)
```

**Arguments**
- `y` A matrix with the compositional data (dependent variable). Zero values are not allowed.
- `x` The predictor variable(s), they can be either continuous or categorical or both.
- `plot` A boolean variable specifying whether to plot the leverage values of the observations or not. This is taken into account only when xnew = NULL.
- `xnew` If you have new data use it, otherwise leave it NULL.

**Details**
A Dirichlet distribution is assumed for the regression. This involves numerical optimization. The function `diri.reg2` allows for the covariates to be linked with the precision parameter $\phi$ via the exponential link function $\phi = e^{x^T b}$.
Value

A list including:

- **runtime**: The time required by the regression.
- **loglik**: The value of the log-likelihood.
- **phi**: The precision parameter. If covariates are linked with it (function "diri.reg2"), this will be a vector.
- **phipar**: The coefficients of the phi parameter if it is linked to the covariates.
- **std.phi**: The standard errors of the coefficients of the phi parameter if it is linked to the covariates.
- **log.phi**: The logarithm of the precision parameter.
- **std.logphi**: The standard error of the logarithm of the precision parameter.
- **be**: The beta coefficients.
- **seb**: The standard error of the beta coefficients.
- **sigma**: The covariance matrix of the regression parameters (for the mean vector and the phi parameter) in the function "diri.reg2".
- **lev**: The leverage values.
- **est**: For the "diri.reg" this contains the fitted or the predicted values (if xnew is not NULL). For the "diri.reg2" if xnew is NULL, this is also NULL.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References


See Also

js.compreg, kl.compreg, ols.compreg, comp.reg, alfa.reg, diri.nr, dda
Examples

```r
x <- as.vector(iris[, 4])
y <- as.matrix(iris[, 1:3])
y <- y / rowSums(y)
mod1 <- diri.reg(y, x)
mod2 <- diri.reg2(y, x)
mod3 <- comp.reg(y, x)
```

Description

Distance based regression models for proportions.

Usage

```r
ols.prop.reg(y, x, cov = FALSE, tol = 1e-07, maxiters = 100)
helling.prop.reg(y, x, tol = 1e-07, maxiters = 100)
```

Arguments

- `y`: A numerical vector proportions. 0s and 1s are allowed.
- `x`: A matrix or a data frame with the predictor variables.
- `cov`: Should the covariance matrix be returned? TRUE or FALSE.
- `tol`: The tolerance value to terminate the Newton-Raphson algorithm. This is set to 10^{-9} by default.
- `maxiters`: The maximum number of iterations before the Newton-Raphson is terminated automatically.

Details

We are using the Newton-Raphson, but unlike R's built-in function "glm" we do no checks and no extra calculations, or whatever. Simply the model. The functions accept binary responses as well (0 or 1).

Value

A list including:

- `sse`: The sum of squares of errors for the "ols.prop.reg" function.
- `be`: The estimated regression coefficients.
- `seb`: The standard error of the regression coefficients if "cov" is TRUE.
- `covb`: The covariance matrix of the regression coefficients in "ols.prop.reg" if "cov" is TRUE.
- `H`: The Hellinger distance between the true and the observed proportions in "helling.prop.reg".
- `iters`: The number of iterations required by the Newton-Raphson.
**Author(s)**

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

**References**


**See Also**

`propreg, beta.reg`

**Examples**

```r
y <- rbeta(100, 1, 4)
x <- matrix(rnorm(100 * 2), ncol = 2)
a1 <- ols.prop.reg(y, x)
a2 <- helling.prop.reg(y, x)
```

---

**Description**

Regression for compositional data based on the Kullback-Leibler the Jensen-Shannon divergence and the symmetric Kullback-Leibler divergence.

**Usage**

```r
kl.compreg(y, x, B = 1, ncores = 1, xnew = NULL, tol = 1e-07, maxiters = 50)
js.compreg(y, x, B = 1, ncores = 1, xnew = NULL)
tv.compreg(y, x, B = 1, ncores = 1, xnew = NULL)
symkl.compreg(y, x, B = 1, ncores = 1, xnew = NULL)
```

**Arguments**

- **y**
  - A matrix with the compositional data (dependent variable). Zero values are allowed.

- **x**
  - The predictor variable(s), they can be either continuous or categorical or both.

- **B**
  - If B is greater than 1 bootstrap estimates of the standard error are returned. If B=1, no standard errors are returned.
ncores
If ncores is 2 or more parallel computing is performed. This is to be used for the case of bootstrap. If B=1, this is not taken into consideration.

xnew
If you have new data use it, otherwise leave it NULL.

tol
The tolerance value to terminate the Newton-Raphson procedure.

maxiters
The maximum number of Newton-Raphson iterations.

Details
In the kl.compreg the Kullback-Leibler divergence is adopted as the objective function. In case of problematic convergence the "multinom" function by the "nnet" package is employed. This will obviously be slower. The js.compreg uses the Jensen-Shannon divergence and the symkl.compreg uses the symmetric Kullback-Leibler divergence. The tv.compreg uses the Total Variation divergence. There is no actual log-likelihood for neither regression.

Value
A list including:

runtime
The time required by the regression.

iters
The number of iterations required by the Newton-Raphson in the kl.compreg function.

loglik
The log-likelihood. This is actually a quasi multinomial regression. This is basically minus the half deviance, or $-\sum_{i=1}^{n} y_i \log \hat{y}_i / \hat{y}_i$.

be
The beta coefficients.

covbe
The covariance matrix of the beta coefficients, if bootstrap is chosen, i.e. if B > 1.

est
The fitted values of xnew if xnew is not NULL.

Author(s)
Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References


Divergence based regression for compositional data with compositional data in the covariates side using the alpha-transformation

See Also
diri.reg, ols.compreg, comp.reg

Examples
library(MASS)
x <- as.vector(fgl[, 1])
y <- as.matrix(fgl[, 2:9])
y <- y / rowSums(y)
mod1 <- kl.compreg(y, x, B = 1, ncores = 1)
mod2 <- js.compreg(y, x, B = 1, ncores = 1)

Divergence based regression for compositional data with compositional data in the covariates side using the alpha-transformation

Description
Divergence based regression for compositional data with compositional data in the covariates side using the alpha-transformation.

Usage
kl.alfapcr(y, x, covar = NULL, a, k, xnew = NULL, B = 1, ncores = 1, tol = 1e-07, maxiters = 50)

Arguments
y A numerical matrix with compositional data with or without zeros.
x A matrix with the predictor variables, the compositional data. Zero values are allowed.
covar If you have other covariates as well put them here.
a The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If alpha = 0 the isometric log-ratio transformation is applied.
k A number at least equal to 1. How many principal components to use.
xnew A matrix containing the new compositional data whose response is to be predicted. If you have no new data, leave this NULL as is by default.
B If B is greater than 1 bootstrap estimates of the standard error are returned. If B=1, no standard errors are returned.
ncores If ncores is 2 or more parallel computing is performed. This is to be used for the case of bootstrap. If B=1, this is not taken into consideration.
tol The tolerance value to terminate the Newton-Raphson procedure.
maxiters The maximum number of Newton-Raphson iterations.
Divergence based regression for compositional data with compositional data in the covariates side using the alpha-transformation.

Details

The \( \alpha \)-transformation is applied to the compositional data first, the first \( k \) principal component scores are calculated and used as predictor variables for the Kullback-Leibler divergence based regression model.

Value

A list including:

- runtime: The time required by the regression.
- iters: The number of iterations required by the Newton-Raphson in the kl.compreg function.
- loglik: The log-likelihood. This is actually a quasi multinomial regression. This is basically minus the half deviance, or \(-\sum_{i=1}^{n} y_i \log y_i / \hat{y}_i\).
- be: The beta coefficients.
- seb: The standard error of the beta coefficients, if bootstrap is chosen, i.e. if \( B > 1 \).
- est: The fitted values of \( x_{\text{new}} \) if \( x_{\text{new}} \) is not NULL.

Author(s)

Initial code by Abdulaziz Alenazi. Modifications by Michail Tsagris.

R implementation and documentation: Abdulaziz Alenazi <a.alenazi@nbu.edu.sa> and Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

klalfapcr.tune, tflr, pcr, glm.pcr, alfa.pcr, alfa.pcr.tune

Examples

```r
library(MASS)
y <- rdiri(214, runif(4, 1, 3))
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
mod <- alfa.pcr(y = y, x = x, a = 0.7, k = 1)
mod
```
Divergence matrix of compositional data

Description

Divergence matrix of compositional data.

Usage

divergence(x, type = "kullback_leibler", vector = FALSE)

Arguments

x
A matrix with the compositional data.

type
This is either "kullback_leibler" (Kullback-Leibler, which computes the symmetric Kullback-Leibler divergence) or "jensen_shannon" (Jensen-Shannon) divergence.

vector
For return a vector instead a matrix.

Details

The function produces the distance matrix either using the Kullback-Leibler (distance) or the Jensen-Shannon (metric) divergence. The Kullback-Leibler refers to the symmetric Kullback-Leibler divergence.

Value

if the vector argument is FALSE a symmetric matrix with the divergences, otherwise a vector with the divergences.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

comp.knn, js.compreg
Examples

```r
x <- as.matrix(iris[1:20, 1:4])
x <- x / rowSums(x)
divergence(x)
```

Description

Empirical likelihood for a one sample mean vector hypothesis testing.

Usage

```r
el.test1(x, mu, R = 1, ncores = 1, graph = FALSE)
```

Arguments

- `x`: A matrix containing Euclidean data.
- `mu`: The hypothesized mean vector.
- `R`: If R is 1 no bootstrap calibration is performed and the classical p-value via the \( \chi^2 \) distribution is returned. If R is greater than 1, the bootstrap p-value is returned.
- `ncores`: The number of cores to use, set to 1 by default.
- `graph`: A boolean variable which is taken into consideration only when bootstrap calibration is performed. IF TRUE the histogram of the bootstrap test statistic values is plotted.

Details

Multivariate hypothesis test for a one sample mean vector. This is a non parametric test and it works for univariate and multivariate data.

Value

A list with the outcome of the function `el.test` which includes the -2 log-likelihood ratio, the observed P-value by chi-square approximation, the final value of Lagrange multiplier \( \lambda \), the gradient at the maximum, the Hessian matrix, the weights on the observations (probabilities multiplied by the sample size) and the number of iteration performed. In addition the runtime of the procedure is reported. In the case of bootstrap, the bootstrap p-value is also returned.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.
Empirical likelihood hypothesis testing for two mean vectors

References

See Also
eel.test1, hotel1T2, james, hotel2T2, maov, el.test2, comp.test

Examples
x <- Rfast::rmvnorm(100, numeric(10), diag( rexp(10, 0.5) ) )
el.test1(x, mu = numeric(10) )
eel.test1(x, mu = numeric(10) )

Description
Empirical likelihood hypothesis testing for two mean vectors.

Usage
el.test2(y1, y2, R = 0, ncores = 1, graph = FALSE)

Arguments
y1 A matrix containing the Euclidean data of the first group.
y2 A matrix containing the Euclidean data of the second group.
R If R is 0, the classical chi-square distribution is used, if R = 1, the corrected chi-
square distribution (James, 1954) is used and if R = 2, the modified F distribution
(Krishnamoorthy and Yanping, 2006) is used. If R is greater than 3 bootstrap
calibration is performed.
ncores How many to cores to use.
graph A boolean variable which is taken into consideration only when bootstrap cali-
    bration is performed. IF TRUE the histogram of the bootstrap test statistic values
    is plotted.

Details
Empirical likelihood is a non parametric hypothesis testing procedure for one sample. The general-
ization to two (or more samples) is via searching for the mean vector that minimizes the sum of the
two test statistics.
Empirical likelihood hypothesis testing for two mean vectors

Value

- **A list including:**
  - **test** The empirical likelihood test statistic value.
  - **modif.test** The modified test statistic, either via the chi-square or the F distribution.
  - **dof** The degrees of freedom of the chi-square or the F distribution.
  - **pvalue** The asymptotic or the bootstrap p-value.
  - **mu** The estimated common mean vector.
  - **runtime** The runtime of the bootstrap calibration.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

- `eel.test2`, `maovjames`, `maov`, `hotel2T2`, `james`, `comp.test`

Examples

```r
el.test2( y1 = as.matrix(iris[1:25, 1:4]), y2 = as.matrix(iris[26:50, 1:4]), R = 0 )
el.test2( y1 = as.matrix(iris[1:25, 1:4]), y2 = as.matrix(iris[26:50, 1:4]), R = 1 )
el.test2( y1 = as.matrix(iris[1:25, 1:4]), y2 = as.matrix(iris[26:50, 1:4]), R = 2 )
```
Description

Energy test of equality of distributions using the $\alpha$-transformation.

Usage

```r
aeqdist.etest(x, sizes, a = 1, R = 999)
```

Arguments

- `x`: A matrix with the compositional data with all groups stacked one under the other.
- `sizes`: A numeric vector matrix with the sample sizes.
- `a`: The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If $\alpha = 0$ the isometric log-ratio transformation is applied. If more than one values are supplied the energy distance of equality of distributions is applied for each value of $\alpha$.
- `R`: The number of permutations to apply in order to compute the approximate p-value.

Details

The $\alpha$-transformation is applied to each composition and then the energy distance of equality of distributions is applied for each value of $\alpha$ or for the single value of $\alpha$.

Value

A numerical value or a numerical vector, depending on the length of the values of $\alpha$, with the approximate p-value(s) of the energy test.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


Estimating location and scatter parameters for compositional data


See Also

acor, acor.tune, alfa, alfa.profile

Examples

```r
y <- rdiri(50, c(3, 4, 5))
x <- rdiri(60, c(3, 4, 5))
aeqdist.etest(rbind(x, y), c(dim(x)[1], dim(y)[1]), a = c(-1, 0, 1))
```

Description

Estimating location and scatter parameters for compositional data in a robust and non robust way.

Usage

```r
comp.den(x, type = "alr", dist = "normal", tol = 1e-07)
```

Arguments

- **x**: A matrix containing compositional data. No zero values are allowed.
- **type**: A boolean variable indicating the transformation to be used. Either "alr" or "ilr" corresponding to the additive or the isometric log-ratio transformation respectively.
- **dist**: Takes values "normal", "t", "skewnorm", "rob" and "spatial". They first three options correspond to the parameters of the normal, t and skew normal distribution respectively. If it set to "rob" the MCD estimates are computed and if set to "spatial" the spatial median and spatial sign covariance matrix are computed.
- **tol**: A tolerance level to terminate the process of finding the spatial median when dist = "spatial". This is set to 1e-09 by default.

Details

This function calculates robust and non robust estimates of location and scatter.

Value

A list including: The mean vector and covariance matrix mainly. Other parameters are also returned depending on the value of the argument "dist".
Estimation of the probability left outside the simplex when using the alpha-transformation

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

spatmed.reg,multivt

Examples

```r
library(MASS)
x <- as.matrix(iris[, 1:4])
x <- x / rowSums(x)
comp.den(x)
comp.den(x, type = "alr", dist = "t")
comp.den(x, type = "alr", dist = "spatial")
```

Estimation of the probability left outside the simplex when using the alpha-transformation

Description

Estimation of the probability left outside the simplex when using the alpha-transformation.

Usage

`probout(mu, su, a)`
Estimation of the probability left outside the simplex when using the alpha-transformation

Arguments

\( \mu \)  
The mean vector.

\( \Sigma \)  
The covariance matrix.

\( a \)  
The value of \( \alpha \).

Details

When applying the \( \alpha \)-transformation based on a multivariate normal there might be probability left outside the simplex as the space of this transformation is a subspace of the Euclidean space. The function estimates the missing probability via Monte Carlo simulation using 40 million generated vectors.

Value

The estimated probability left outside the simplex.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

alfa, alpha.mle, a.est, rfolded

Examples

```r
## Not run:
s <- c(0.1490676523, -0.4580818209, 0.0020395316, -0.0047446076, -0.4580818209,
1.5227259250, 0.0002596411, 0.0074836251, 0.0020395316, 0.0002596411,
0.0365384838, -0.0471448849, -0.0047446076, 0.0074836251, -0.0471448849,
0.0611442781)
s <- matrix(s, ncol = 4)
m <- c(1.715, 0.914, 0.115, 0.167)
probout(m, s, 0.5)
## End(Not run)
```
Description

Estimation of the value of $\alpha$ in the folded model.

Usage

a.est(x)

Arguments

x A matrix with the compositional data. No zero values are allowed.

Details

This is a function for choosing or estimating the value of $\alpha$ in the folded model (Tsagris and Stewart, 2020).

Value

A list including:

- runtime: The runtime of the algorithm.
- best: The estimated optimal $\alpha$ of the folded model.
- loglik: The maximised log-likelihood of the folded model.
- p: The estimated probability inside the simplex of the folded model.
- mu: The estimated mean vector of the folded model.
- su: The estimated covariance matrix of the folded model.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


Estimation of the value of alpha via the profile log-likelihood

See Also

alfa.profile, alfa, alfainv, alpha.mle

Examples

```r
x <- as.matrix(iris[, 1:4])
x <- x / rowSums(x)
a <- alfa.tune(x)
a.est(x)
```

Description

Estimation of the value of \( \alpha \) via the alfa profile log-likelihood.

Usage

```r
alfa.profile(x, a = seq(-1, 1, by = 0.01))
```

Arguments

- `x`: A matrix with the compositional data. Zero values are not allowed.
- `a`: A grid of values of \( \alpha \).

Details

For every value of \( \alpha \) the normal likelihood (see the reference) is computed. At the end, the plot of the values is constructed.

Value

A list including:

- `res`: The chosen value of \( \alpha \), the corresponding log-likelihood value and the log-likelihood when \( \alpha = 0 \).
- `ci`: An asymptotic 95% confidence interval computed from the log-likelihood ratio test.

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>. 
Exponential empirical likelihood for a one sample mean vector hypothesis testing

References

See Also
alfa.tune, alfa, alfainv

Examples
x <- as.matrix(iris[, 1:4])
x <- x / rowSums(x)
alfa.tune(x)
alfa.profile(x)

description
Exponential empirical likelihood for a one sample mean vector hypothesis testing.

Usage
eel.test1(x, mu, tol = 1e-06, R = 1)

Arguments
x A matrix containing Euclidean data.
mu The hypothesized mean vector.
tol The tolerance value used to stop the Newton-Raphson algorithm.
R The number of bootstrap samples used to calculate the p-value. If R = 1 (default value), no bootstrap calibration is performed

Details
Multivariate hypothesis test for a one sample mean vector. This is a non parametric test and it works for univariate and multivariate data.
Value

A list including:

- \( p \) The estimated probabilities.
- \( \lambda \) The value of the Lagrangian parameter \( \lambda \).
- \( \text{iter} \) The number of iterations required by the Newton-Raphson algorithm.
- \( \text{info} \) The value of the log-likelihood ratio test statistic along with its corresponding p-value.
- \( \text{runtime} \) The runtime of the process.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

- `eel.test1`, `hotel1T2`, `james`, `hotel2T2`, `maov`, `el.test2`, `comp.test`

Examples

```r
x <- Rfast::rmvnorm(100, numeric(10), diag( rexp(10, 0.5) ) )
eel.test1(x, numeric(10) )
eel.test1(x, numeric(10) )
```

---

Description

Exponential empirical likelihood hypothesis testing for two mean vectors.

Usage

- `eel.test2(y1, y2, tol = 1e-07, R = 0, graph = FALSE)`
Exponential empirical likelihood hypothesis testing for two mean vectors

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>y1</td>
<td>A matrix containing the Euclidean data of the first group.</td>
</tr>
<tr>
<td>y2</td>
<td>A matrix containing the Euclidean data of the second group.</td>
</tr>
<tr>
<td>tol</td>
<td>The tolerance level used to terminate the Newton-Raphson algorithm.</td>
</tr>
<tr>
<td>R</td>
<td>If R is 0, the classical chi-square distribution is used, if R = 1, the corrected chi-square distribution (James, 1954) is used and if R = 2, the modified F distribution (Krishnamoorthy and Yanping, 2006) is used. If R is greater than 3 bootstrap calibration is performed.</td>
</tr>
<tr>
<td>graph</td>
<td>A boolean variable which is taken into consideration only when bootstrap calibration is performed. IF TRUE the histogram of the bootstrap test statistic values is plotted.</td>
</tr>
</tbody>
</table>

Details

Exponential empirical likelihood is a non parametric hypothesis testing procedure for one sample. The generalization to two (or more samples) is via searching for the mean vector that minimises the sum of the two test statistics.

Value

A list including:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>test</td>
<td>The empirical likelihood test statistic value.</td>
</tr>
<tr>
<td>modif.test</td>
<td>The modified test statistic, either via the chi-square or the F distribution.</td>
</tr>
<tr>
<td>dof</td>
<td>The degrees of freedom of the chi-square or the F distribution.</td>
</tr>
<tr>
<td>pvalue</td>
<td>The asymptotic or the bootstrap p-value.</td>
</tr>
<tr>
<td>mu</td>
<td>The estimated common mean vector.</td>
</tr>
<tr>
<td>runtime</td>
<td>The runtime of the bootstrap calibration.</td>
</tr>
</tbody>
</table>

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

.el.test2, maovjames, maov, hotel2T2, james, comp.test

Examples

y1 = as.matrix(iris[1:25, 1:4])
y2 = as.matrix(iris[26:50, 1:4])
el.test2(y1, y2)
el.test2(y1, y2)
el.test2(y1, y2)
el.test2(y1, y2)

---

**Fast estimation of the value of alpha**

**Fast estimation of the value of α**

---

**Description**

Fast estimation of the value of α.

**Usage**

alfa.tune(x, B = 1, ncores = 1)

**Arguments**

- \(x\)  
  A matrix with the compositional data. No zero values are allowed.

- \(B\)  
  If no (bootstrap based) confidence intervals should be returned this should be 1 and more than 1 otherwise.

- \(ncores\)  
  If ncores is greater than 1 parallel computing is performed. It is advisable to use it if you have many observations and or many variables, otherwise it will slow down the process.

**Details**

This is a faster function than \texttt{alfa.profile} for choosing the value of \(α\).
Fitting a Flexible Dirichlet distribution

Value
A vector with the best alpha, the maximised log-likelihood and the log-likelihood at $\alpha = 0$, when $B = 1$ (no bootstrap). If $B > 1$ a list including:

- **param**: The best alpha and the value of the log-likelihood, along with the 95% bootstrap based confidence intervals.
- **message**: A message with some information about the histogram.
- **runtime**: The time (in seconds) of the process.

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References

See Also
alfa.profile, alfa, alfainv

Examples
```r
library(MASS)
x <- as.matrix(iris[, 1:4])
x <- x / rowSums(x)
alfa.tune(x)
alfa.profile(x)
```

---

Fitting a Flexible Dirichlet distribution

Description
Fitting a Flexible Dirichlet distribution.

Usage
```r
fd.est(x, ini.iter = 50, final.iter = 100)
```
Fitting a Flexible Dirichlet distribution

Arguments

- **x**: A matrix or a dataframe containing the compositional data.
- **ini.iter**: Number of iterations for the initial SEM step. Default value is 50.
- **final.iter**: Number of iterations for the final EM step. Default value is 100.

Details

For more information see the references.

Value

A list including:

- **alpha**: Estimated values of the parameter vector \( \alpha \).
- **prob**: Estimated values of the parameter vector \( p \).
- **tau**: Estimated value of the parameter \( \tau \).
- **loglik**: The Log-likelihood value.

Author(s)

Michail Tsagris ported from the R package FlexDir. <mtsagris@uoc.gr>.

References


See Also

- rfd, rfd

Examples

```r
## Not run:
x <- rfd(n = 50, a = c(12, 11, 10), p = c(0.25, 0.25, 0.5), tau = 4)
ela <- fd.est(x, ini.iter = 10, final.iter = 20)
ela
## End(Not run)
```
Description

Gaussian mixture models for compositional data.

Usage

mix.compnorm(x, g, model, type = "alr", veo = FALSE)

Arguments

x    A matrix with the compositional data.
g    How many clusters to create.
model    The type of model to be used.

1. "EII": All groups have the same diagonal covariance matrix, with the same variance for all variables.
2. "VII": Different diagonal covariance matrices, with the same variance for all variables within each group.
3. "EEI": All groups have the same diagonal covariance matrix.
4. "VEI": Different diagonal covariance matrices. If we make all covariance matrices have determinant 1, (divide the matrix with the $p$-th root of its determinant) then all covariance matrices will be the same.
5. "EVI": Different diagonal covariance matrices with the same determinant.
6. "VVI": Different diagonal covariance matrices, with nothing in common.
7. "EEE": All covariance matrices are the same.
8. "EEV": Different covariance matrices, but with the same determinant and in addition, if we make them have determinant 1, they will have the same trace.
9. "VEV": Different covariance matrices but if we make the matrices have determinant 1, then they will have the same trace.
10. "VVV": Different covariance matrices with nothing in common.
11. "EVE": Different covariance matrices, but with the same determinant. In addition, calculate the eigenvectors for each covariance matrix and you will see the extra similarities.
12. "VVE": Different covariance matrices, but they have something in common with their directions. Calculate the eigenvectors of each covariance matrix and you will see the similarities.
13. "VEE": Different covariance matrices, but if we make the matrices have determinant 1, then they will have the same trace. In addition, calculate the eigenvectors for each covariance matrix and you will see the extra similarities.
14. "EVV": Different covariance matrices, but with the same determinant.

**type**
The type of transformation to be used, either the additive log-ratio ("alr"), the isometric log-ratio ("ilr") or the pivot coordinate ("pivot") transformation.

**veo**
Stands for "Variables exceed observations". If TRUE then if the number variables in the model exceeds the number of observations, but the model is still fitted.

**Details**
A log-ratio transformation is applied and then a Gaussian mixture model is constructed.

**Value**
A list including:

- **mu**: A matrix where each row corresponds to the mean vector of each cluster.
- **su**: An array containing the covariance matrix of each cluster.
- **prob**: The estimated mixing probabilities.
- **est**: The estimated cluster membership values.

**Author(s)**
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

**References**

**See Also**
bic.mixcompnorm, rmixcomp, mix.compnorm.contour, alfa.mix.norm, alfa.knn, alfa.rda, comp.nb

**Examples**
```r
## Not run:
x <- as.matrix(iris[, 1:4])
x <- x/rowSums(x)
mod1 <- mix.compnorm(x, 3, model = "EII")
mod2 <- mix.compnorm(x, 4, model = "VII")
## End(Not run)
```
Gaussian mixture models for compositional data using the \( \alpha \)-transformation

**Description**

Gaussian mixture models for compositional data using the \( \alpha \)-transformation.

**Usage**

\[
\text{alfa.mix.norm}(x, g, a, \text{model}, \text{veo} = \text{FALSE})
\]

**Arguments**

- \( x \)  
  A matrix with the compositional data.

- \( g \)  
  How many clusters to create.

- \( a \)  
  The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If \( \alpha = 0 \) the isometric log-ratio transformation is applied.

- \( \text{model} \)  
  The type of model to be used.
  
  1. "EII": All groups have the same diagonal covariance matrix, with the same variance for all variables.
  2. "VII": Different diagonal covariance matrices, with the same variance for all variables within each group.
  3. "EEI": All groups have the same diagonal covariance matrix.
  4. "VEI": Different diagonal covariance matrices. If we make all covariance matrices have determinant 1, (divide the matrix with the \( p \)-th root of its determinant) then all covariance matrices will be the same.
  5. "EVI": Different diagonal covariance matrices with the same determinant.
  6. "VVI": Different diagonal covariance matrices, with nothing in common.
  7. "EEE": All covariance matrices are the same.
  8. "EEV": Different covariance matrices, but with the same determinant and in addition, if we make them have determinant 1, they will have the same trace.
  9. "VEV": Different covariance matrices but if we make the matrices have determinant 1, then they will have the same trace.
  10. "VVV": Different covariance matrices with nothing in common.
  11. "EVE": Different covariance matrices, but with the same determinant. In addition, calculate the eigenvectors for each covariance matrix and you will see the extra similarities.
  12. "VVE": Different covariance matrices, but they have something in common with their directions. Calculate the eigenvectors of each covariance matrix and you will see the similarities.
13. "VEE": Different covariance matrices, but if we make the matrices have determinant 1, then they will have the same trace. In addition, calculate the eigenvectors for each covariance matrix and you will see the extra similarities.

14. "EVV": Different covariance matrices, but with the same determinant.

\texttt{veo}

Stands for "Variables exceed observations". If TRUE then if the number variables in the model exceeds the number of observations, but the model is still fitted.

Details

A log-ratio transformation is applied and then a Gaussian mixture model is constructed.

Value

A list including:

\texttt{mu}

A matrix where each row corresponds to the mean vector of each cluster.

\texttt{su}

An array containing the covariance matrix of each cluster.

\texttt{prob}

The estimated mixing probabilities.

\texttt{est}

The estimated cluster membership values.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

\texttt{bic.alfamixnorm, bic.mixcompnorm, rmixcomp, mix.compnorm.contour, mix.compnorm, alfa, alfa.knn, alfa.rda, comp.nb}

Examples

\begin{verbatim}
## Not run:
x <- as.matrix(iris[, 1:4])
x <- x/ rowSums(x)
mod1 <- alfa.mix.norm(x, 3, 0.4, model = "EII")
mod2 <- alfa.mix.norm(x, 4, 0.7, model = "VII")

## End(Not run)
\end{verbatim}
Generate random folds for cross-validation

Description
Random folds for use in a cross validation are generated. There is the option for stratified splitting as well.

Usage
makefolds(ina, nfolds = 10, stratified = TRUE, seed = NULL)

Arguments
- ina: A variable indicating the groupings.
- nfolds: The number of folds to produce.
- stratified: A boolean variable specifying whether stratified random (TRUE) or simple random (FALSE) sampling is to be used when producing the folds.
- seed: You can specify your own seed number here or leave it NULL.

Details
I was inspired by the command in the package TunePareto in order to do the stratified version.

Value
A list with nfolds elements where each elements is a fold containing the indices of the data.

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

See Also
rda.tune

Examples
a <- makefolds(iris[, 5], nfolds = 5, stratified = TRUE)
table(iris[a[[1]], 5]) # # 10 values from each group
**Greenacre’s power transformation**

**Description**

Greenacre’s power transformation.

**Usage**

```
green(x, theta)
```

**Arguments**

- `x` A matrix with the compositional data.
- `theta` The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If \( \theta = 0 \) the log transformation is applied.

**Details**

Greenacre’s transformation is applied to the compositional data.

**Value**

A matrix with the power transformed data.

**Author(s)**

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

**References**


**See Also**

`alfa`

**Examples**

```r
library(MASS)
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
y1 <- green(x, 0.1)
y2 <- green(x, 0.2)
rbind( colMeans(y1), colMeans(y2) )
```
Helper Frechet mean for compositional data

Description
Helper Frechet mean for compositional data.

Usage
frechet2(x, di, a, k)

Arguments
- x: A matrix with the compositional data.
- di: A matrix with indices as produced by the function "dista" of the package "Rfast" or the function "nn2" of the package "RANN". Better see the details section.
- a: The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If $\alpha = 0$ the isometric log-ratio transformation is applied and the closed geometric mean is calculated.
- k: The number of nearest neighbours used for the computation of the Frechet means.

Details
The power transformation is applied to the compositional data and the mean vector is calculated. Then the inverse of it is calculated and the inverse of the power transformation applied to the last vector is the Frechet mean.

What this helper function do is to speed up the Frechet mean when used in the $\alpha$-k-NN regression. The $\alpha$-k-NN regression computes the Frechet mean of the k nearest neighbours for a value of $\alpha$ and this function does exactly that. Suppose you want to predict the compositional value of some new predictors. For each predictor value you must use the Frechet mean computed at various nearest neighbours. This function performs these computations in a fast way. It is not the fastest way, yet it is a pretty fast way. This function is being called inside the function aknn.reg.

Value
A list where each element contains a matrix. Each matrix contains the Frechet means computed at various nearest neighbours.

Author(s)
Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.
References


See Also

alfa, alfainv, profile

Examples

## Not run:
library(MASS)
library(Rfast)
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
xnew <- x[1:10, ]
x <- x[-c(1:10), ]
k <- 2:5
di <- Rfast::dista( xnew, x, k = max(k), index = TRUE, square = TRUE )
est <- frechet2(x, di, 0.2, k)
## End(Not run)

Helper functions for the Kullback-Leibler regression

### Usage

kl.compreg2(y, x, xnew = NULL, tol = 1e-07, maxiters = 50)
klcompreg.boot(y, x, der, der2, id, b1, n, p, d, tol = 1e-07, maxiters = 50)

### Arguments

- **y**
  - A matrix with the compositional data (dependent variable). Zero values are allowed. For the klcompreg.boot the first column is removed.

- **x**
  - The predictor variable(s), they can be either continuous or categorical or both. In the klcompreg.boot this is the design matrix, with the ones in the first column.

- **xnew**
  - If you have new data use it, otherwise leave it NULL.

- **tol**
  - The tolerance value to terminate the Newton-Raphson procedure.

- **maxiters**
  - The maximum number of Newton-Raphson iterations.
**Helper functions for the Kullback-Leibler regression**

- **der**
  - An vector to put the first derivative there.

- **der2**
  - An empty matrix to put the second derivatives there, the Hessian matrix will be put here.

- **id**
  - A help vector with indices.

- **b1**
  - The matrix with the initial estimated coefficients.

- **n**
  - The sample size

- **p**
  - The number of columns of the design matrix.

- **d**
  - The dimensionality of the simplex, that is the number of columns of the compositional data minus 1.

**Details**

These are help functions for the `kl.compreg` function. They are not to be called directly by the user.

**Value**

For kl.compreg2 a list including:

- **iters**
  - The number of iterations required by the Newton-Raphson.

- **loglik**
  - The loglikelihood.

- **be**
  - The beta coefficients.

- **est**
  - The fitted or the predicted values (if xnew is not NULL).

For klcompreg.boot a list including:

- **loglik**
  - The loglikelihood.

- **be**
  - The beta coefficients.

**Author(s)**

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

**References**


**See Also**

`diri.reg, js.compreg, ols.compreg, comp.reg`
Hotelling’s multivariate version of the 1 sample t-test for Euclidean data

Examples

```r
library(MASS)
x <- as.vector(fgl[, 1])
y <- as.matrix(fgl[, 2:9])
y <- y / rowSums(y)
mod1 <- kl.compreg(y, x, B = 1, ncores = 1)
mod2 <- js.compreg(y, x, B = 1, ncores = 1)
```

Description

Hotelling’s test for testing one Euclidean population mean vector.

Usage

```
hotel1T2(x, M, a = 0.05, R = 999, graph = FALSE)
```

Arguments

- `x`: A matrix containing Euclidean data.
- `a`: The significance level, set to 0.05 by default.
- `M`: The hypothesized mean vector.
- `R`: If R is 1 no bootstrap calibration is performed and the classical p-value via the F distribution is returned. If R is greater than 1, the bootstrap p-value is returned.
- `graph`: A boolean variable which is taken into consideration only when bootstrap calibration is performed. IF TRUE the histogram of the bootstrap test statistic values is plotted.

Details

Multivariate hypothesis test for a one sample mean vector. This is the multivariate analogue of the one sample t-test. The p-value can be calculated either asymptotically or via bootstrap.

Value

A list including:

- `m`: The sample mean vector.
- `info`: The test statistic, the p-value, the critical value and the degrees of freedom of the F distribution (numerator and denominator). This is given if no bootstrap calibration is employed.
- `pvalue`: The bootstrap p-value is bootstrap is employed.
- `runtime`: The runtime of the bootstrap calibration.
Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

See Also
eel.test1, el.test1, james, hotel2T2, maov, el.test2, comp.test

Examples
```
x <- Rfast::rmvnorm(100, numeric(10), diag( rexp(10,0.5) ) )
hotel1T2(x, numeric(10), R = 1)
hotel1T2(x, numeric(10), R = 999, graph = TRUE)
```

Description
Hotelling’s test for testing the equality of two Euclidean population mean vectors.

Usage
```
hotel2T2(x1, x2, a = 0.05, R = 999, graph = FALSE)
```

Arguments
- `x1`: A matrix containing the Euclidean data of the first group.
- `x2`: A matrix containing the Euclidean data of the second group.
- `a`: The significance level, set to 0.05 by default.
- `R`: If `R` is 1 no bootstrap calibration is performed and the classical p-value via the F distribution is returned. If `R` is greater than 1, the bootstrap p-value is returned.
- `graph`: A boolean variable which is taken into consideration only when bootstrap calibration is performed. If TRUE the histogram of the bootstrap test statistic values is plotted.

Details
Multivariate analysis of variance assuming equality of the covariance matrices. The p-value can be calculated either asymptotically or via bootstrap.
Hypothesis testing for two or more compositional mean vectors

Value

A list including:

- **mesoi**: The two mean vectors.
- **info**: The test statistic, the p-value, the critical value and the degrees of freedom of the F distribution (numerator and denominator). This is given if no bootstrap calibration is employed.
- **pvalue**: The bootstrap p-value if bootstrap is employed.
- **note**: A message informing the user that bootstrap calibration has been employed.
- **runtime**: The runtime of the bootstrap calibration.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

- `james`, `maov`, `el.test2`, `eel.test2`, `comp.test`

Examples

```r
hotel2T2( as.matrix(iris[1:25, 1:4]), as.matrix(iris[26:50, 1:4]) )
hotel2T2( as.matrix(iris[1:25, 1:4]), as.matrix(iris[26:50, 1:4]), R = 1 )
james( as.matrix(iris[1:25, 1:4]), as.matrix(iris[26:50, 1:4]) )
james( as.matrix(iris[1:25, 1:4]), as.matrix(iris[26:50, 1:4]), R = 1 )
```

Description

Hypothesis testing for two or more compositional mean vectors.

Usage

```r
comp.test(x, ina, test = "james", R = 0, ncores = 1, graph = FALSE)
```
Hypothesis testing for two or more compositional mean vectors

Arguments

\textbf{x} \hspace{1cm} A matrix containing compositional data.

\textbf{ina} \hspace{1cm} A numerical or factor variable indicating the groups of the data.

\textbf{test} \hspace{1cm} This can take the values of "james" for James’ test, "hotel" for Hotelling’s test, "maov" for multivariate analysis of variance assuming equality of the covariance matrices, "maovjames" for multivariate analysis of variance without assuming equality of the covariance matrices. "el" for empirical likelihood or "eel" for exponential empirical likelihood.

\textbf{R} \hspace{1cm} This depends upon the value of the argument "test". If the test is "maov" or "maovjames", R is not taken into consideration. If test is "hotel", then R denotes the number of bootstrap resamples. If test is "james", then R can be 1 (chi-square distribution), 2 (F distribution), or more for bootstrap calibration. If test is "el", then R can be 0 (chi-square), 1 (corrected chi-square), 2 (F distribution) or more for bootstrap calibration. See the help page of each test for more information.

\textbf{ncores} \hspace{1cm} How many to cores to use. This is taken into consideration only if test is "el" and R is more than 2.

\textbf{graph} \hspace{1cm} A boolean variable which is taken into consideration only when bootstrap calibration is performed. IF TRUE the histogram of the bootstrap test statistic values is plotted. This is taken into account only when R is greater than 2.

Details

The idea is to apply the \( \alpha \)-transformation, with \( \alpha = 1 \), to the compositional data and then use a test to compare their mean vectors. See the help page of each test for more information. The function is visible so you can see exactly what is going on.

Value

A list including:

\textbf{result} \hspace{1cm} The outcome of each test.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References


ICE plot for projection pursuit regression with compositional predictor variables


See Also
hd.meantest2, maovjames, maov, hotel2T2, el.test2, eel.test2

Examples
ina <- rep(1:2, each = 50)
x <- as.matrix(iris[1:100, 1:4])
x <- x/rowSums(x)
comp.test( x, ina, test = "james")
comp.test( x, ina, test = "hotel")
comp.test( x, ina, test = "el")
comp.test( x, ina, test = "eel")

Description
ICE plot for projection pursuit regression with compositional predictor variables.

Usage
ice.pprcomp(model, x, k = 1, frac = 0.1, type = "log")

Arguments
model The ppr model, the outcome of the pprcomp function.
x A matrix with the compositional data. No zero values are allowed.
k Which variable to select?.
frac Fraction of observations to use. The default value is 0.1.
type Either "alr" or "log" corresponding to the additive log-ratio transformation or the simple logarithm applied to the compositional data.
Details

This function implements the Individual Conditional Expecation plots of Goldstein et al. (2015). See the references for more details.

Value

A graph with several curves. The horizontal axis contains the selected variable, whereas the vertical axis contains the centered predicted values. The black curves are the effects for each observation and the blue line is their average effect.

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

pprcomp, pprcomp.tune, ice.kernreg, alfa.pcr, lc.reg, comp.ppr

Examples

```r
x <- as.matrix( iris[, 2:4] )
x <- x/ rowSums(x)
y <- iris[, 1]
model <- pprcomp(y, x)
ice <- ice.pprcomp(model, x, k = 1)
```

Description

ICE plot for univariate kernel regression.

Usage

```r
ice.kernreg(y, x, h, type = "gauss", k = 1, frac = 0.1)
```
**ICE plot for univariate kernel regression**

**Arguments**

- **y**: A numerical vector with the response values.
- **x**: A numerical matrix with the predictor variables.
- **h**: The bandwidth value to consider.
- **type**: The type of kernel to use, "gauss" or "laplace".
- **k**: Which variable to select?
- **frac**: Fraction of observations to use. The default value is 0.1.

**Details**

This function implements the Individual Conditional Expectation plots of Goldstein et al. (2015). See the references for more details.

**Value**

A graph with several curves. The horizontal axis contains the selected variable, whereas the vertical axis contains the centered predicted values. The black curves are the effects for each observation and the blue line is their average effect.

**Author(s)**

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

**References**


**See Also**

ice.pprcomp, kernreg.tune, alfa.pcr, lc.reg

**Examples**

```r
x <- as.matrix( iris[, 2:4] )
y <- iris[, 1]
ice <- ice.kernreg(y, x, h = 0.1, k = 1)
```
Description

The inverse of the $\alpha$-transformation.

Usage

```
 alfainv(x, a, h = TRUE)
```

Arguments

- **x**: A matrix with Euclidean data. However, they must lie within the feasible, acceptable space. See references for more information.
- **a**: The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If $\alpha = 0$, the inverse of the isometric log-ratio transformation is applied.
- **h**: If h = TRUE this means that the multiplication with the Helmer sub-matrix will take place. It is set to TRUE by default.

Details

The inverse of the $\alpha$-transformation is applied to the data. If the data lie outside the $\alpha$-space, NAs will be returned for some values.

Value

A matrix with the pairwise distances.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References


James multivariate version of the t-test

See Also

alfa, alfadist

Examples

library(MASS)
x <- as.matrix(fgl[1:10, 2:9])
x <- x / rowSums(x)
y <- alfa(x, 0.5)$aff
alfainv(y, 0.5)

Description

James test for testing the equality of two population mean vectors without assuming equality of the covariance matrices.

Usage

james(y1, y2, a = 0.05, R = 999, graph = FALSE)

Arguments

y1
A matrix containing the Euclidean data of the first group.
y2
A matrix containing the Euclidean data of the second group.
a
The significance level, set to 0.05 by default.
R
If R is 1 no bootstrap calibration is performed and the classical p-value via the F distribution is returned. If R is greater than 1, the bootstrap p-value is returned.
graph
A boolean variable which is taken into consideration only when bootstrap calibration is performed. IF TRUE the histogram of the bootstrap test statistic values is plotted.

Details

Multivariate analysis of variance without assuming equality of the covariance matrices. The p-value can be calculated either asymptotically or via bootstrap. The James test (1954) or a modification proposed by Krishnamoorthy and Yanping (2006) is implemented. The James test uses a corrected chi-square distribution, whereas the modified version uses an F distribution.
Kernel regression with a numerical response vector or matrix

Value

A list including:

- **note**: A message informing the user about the test used.
- **mesoi**: The two mean vectors.
- **info**: The test statistic, the p-value, the correction factor and the corrected critical value of the chi-square distribution if the James test has been used or, the test statistic, the p-value, the critical value and the degrees of freedom (numerator and denominator) of the F distribution if the modified James test has been used.
- **pvalue**: The bootstrap p-value if bootstrap is employed.
- **runtime**: The runtime of the bootstrap calibration.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References


See Also

- hotel2T2, maovjames, el.test2, eel.test2, comp.test

Examples

```r
james( as.matrix(iris[1:25, 1:4]), as.matrix(iris[26:50, 1:4]), R = 1 )
james( as.matrix(iris[1:25, 1:4]), as.matrix(iris[26:50, 1:4]), R = 2 )
james( as.matrix(iris[1:25, 1:4]), as.matrix(iris[26:50, 1:4]) )
hotel2T2( as.matrix(iris[1:25, 1:4]), as.matrix(iris[26:50, 1:4]) )
```

Description

Kernel regression (Nadaraya-Watson estimator) with a numerical response vector or matrix.

Usage

```r
kern.reg(xnew, y, x, h = seq(0.1, 1, length = 10), type = "gauss")
```
Kernel regression with a numerical response vector or matrix

Arguments

xnew A matrix with the new predictor variables whose compositions are to be predicted.
y A numerical vector or a matrix with the response value.
x A matrix with the available predictor variables.
h The bandwidth value(s) to consider.
type The type of kernel to use, "gauss" or "laplace".

Details

The Nadaraya-Watson estimator regression is applied.

Value

The fitted values. If a single bandwidth is considered then this is a vector or a matrix, depending on the nature of the response. If multiple bandwidth values are considered then this is a matrix, if the response is a vector, or a list, if the response is a matrix.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

kernreg.tune, ice.kernreg, akern.reg, aknn.reg

Examples

y <- iris[, 1]
x <- iris[, 2:4]
est <- kern.reg(x, y, x, h = c(0.1, 0.2))
Kullback-Leibler divergence and Bhattacharyya distance between two Dirichlet distributions

Description

Kullback-Leibler divergence and Bhattacharyya distance between two Dirichlet distributions.

Usage

kl.diri(a, b, type = "KL")

Arguments

a A vector with the parameters of the first Dirichlet distribution.
b A vector with the parameters of the second Dirichlet distribution.
type A variable indicating whether the Kullback-Leibler divergence ("KL") or the Bhattacharyya distance ("bhatt") is to be computed.

Details

Note that the order is important in the Kullback-Leibler divergence, since this is asymmetric, but not in the Bhattacharyya distance, since it is a metric.

Value

The value of the Kullback-Leibler divergence or the Bhattacharyya distance.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also
diri.est, diri.nr
Examples

```r
library(MASS)
a <- runif(10, 0, 20)
b <- runif(10, 1, 10)
kl.diri(a, b)
kl.diri(b, a)
kl.diri(a, b, type = "bhatt")
kl.diri(b, a, type = "bhatt")
```

Description

LASSO Kullback-Leibler divergence based regression.

Usage

```r
lasso.klcompreg(y, x, alpha = 1, lambda = NULL, nlambda = 100, type = "grouped", xnew = NULL)
```

Arguments

- `y`: A numerical matrix with compositional data. Zero values are allowed.
- `x`: A numerical matrix containing the predictor variables.
- `alpha`: The elastic net mixing parameter, with $0 \leq \alpha \leq 1$. The penalty is defined as a weighted combination of the ridge and of the Lasso regression. When $\alpha = 1$ LASSO is applied, while $\alpha = 0$ yields the ridge regression.
- `lambda`: This information is copied from the package glmnet. A user supplied lambda sequence. Typical usage is to have the program compute its own lambda sequence based on `nlambda` and `lambda.min.ratio`. Supplying a value of lambda overrides this. **WARNING**: use with care. Avoid supplying a single value for lambda (for predictions after CV use predict() instead). Supply instead a decreasing sequence of lambda values. glmnet relies on its warms starts for speed, and its often faster to fit a whole path than compute a single fit.
- `nlambda`: This information is copied from the package glmnet. The number of `lambda` values, default is 100.
- `type`: This information is copied from the package glmnet.. If "grouped" then a grouped lasso penalty is used on the multinomial coefficients for a variable. This ensures they are all in our out together. The default in our case is "grouped".
- `xnew`: If you have new data use it, otherwise leave it NULL.

Details

The function uses the glmnet package to perform LASSO penalised regression. For more details see the function in that package.
LASSO log-ratio regression with compositional response

Value

A list including:

- **mod**: We decided to keep the same list that is returned by glmnet. So, see the function in that package for more information.

- **est**: If you supply a matrix in the “xnew” argument this will return an array of many matrices with the fitted values, where each matrix corresponds to each value of \( \lambda \).

Author(s)

Michail Tsagris and Abdulaziz Alenazi.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Abdulaziz Alenazi <a.alenazi@nbu.edu.sa>.

References


See Also

lassocoef.plot, cv.lasso.klcompreg, kl.compreg, lasso.compreg, ols.compreg, alfa.pcr, alfa.knn.reg

Examples

```r
y <- as.matrix(iris[, 1:4])
y <- y / rowSums(y)
x <- matrix( rnorm(150 * 30), ncol = 30 )
a <- lasso.klcompreg(y, x)
```

LASSO log-ratio regression with compositional response

LASSO log-ratio regression with compositional response

Description

LASSO log-ratio regression with compositional response.

Usage

```r
lasso.compreg(y, x, alpha = 1, lambda = NULL, nlambda = 100, xnew = NULL)
```
LASSO log-ratio regression with compositional response

Arguments

- **y**: A numerical matrix with compositional data. Zero values are not allowed as the additive log-ratio transformation (alr) is applied to the compositional response prior to implementing the LASSO algorithm.
- **x**: A numerical matrix containing the predictor variables.
- **alpha**: The elastic net mixing parameter, with \(0 \leq \alpha \leq 1\). The penalty is defined as a weighted combination of the ridge and of the Lasso regression. When \(\alpha = 1\) LASSO is applied, while \(\alpha = 0\) yields the ridge regression.
- **lambda**: **This information is copied from the package glmnet.** A user supplied lambda sequence. Typical usage is to have the program compute its own lambda sequence based on nlambda and lambda.min.ratio. Supplying a value of lambda overrides this. WARNING: use with care. Avoid supplying a single value for lambda (for predictions after CV use predict() instead). Supply instead a decreasing sequence of lambda values. glmnet relies on its warms starts for speed, and its often faster to fit a whole path than compute a single fit.
- **nlambda**: **This information is copied from the package glmnet.** The number of lambda values, default is 100.
- **xnew**: If you have new data use it, otherwise leave it NULL.

Details

The function uses the glmnet package to perform LASSO penalised regression. For more details see the function in that package.

Value

A list including:

- **mod**: We decided to keep the same list that is returned by glmnet. So, see the function in that package for more information.
- **est**: If you supply a matrix in the "xnew" argument this will return an array of many matrices with the fitted values, where each matrix corresponds to each value of \(\lambda\).

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

cv.lasso.compreg, lassocoef.plot, lasso.klcompreg, cv.lasso.klcompreg, comp.reg
Log-contrast logistic or Poisson regression with compositional predictor variables

Examples

```r
y <- as.matrix(iris[, 1:4])
y <- y / rowSums(y)
x <- matrix( rnorm(150 * 30), ncol = 30 )
a <- lasso.compreg(y, x)
```

Description

Log-contrast logistic or Poisson regression with compositional predictor variables.

Usage

```r
lc.glm(y, x, z = NULL, model = "logistic", xnew = NULL, znew = NULL)
```

Arguments

- `y`: A numerical vector containing the response variable values. This must be a continuous variable.
- `x`: A matrix with the predictor variables, the compositional data. No zero values are allowed.
- `z`: A matrix, data.frame, factor or a vector with some other covariate(s).
- `model`: This can be either "logistic" or "poisson".
- `xnew`: A matrix containing the new compositional data whose response is to be predicted. If you have no new data, leave this NULL as is by default.
- `znew`: A matrix, data.frame, factor or a vector with the values of some other covariate(s). If you have no new data, leave this NULL as is by default.

Details

The function performs the log-contrast logistic or Poisson regression model. The logarithm of the compositional predictor variables is used (hence no zero values are allowed). The response variable is linked to the log-transformed data with the constraint that the sum of the regression coefficients equals 0. If you want the regression without the zum-to-zero constraints see `ulc.glm`. Extra predictors variables are allowed as well, for instance categorical or continuous.
Value

A list including:

- runtime: The duration of the algorithm required to minimize the deviance.
- dev: The residual deviance of the logistic or Poisson regression model.
- be: The constrained regression coefficients. Their sum equals 0.
- est: If the arguments "xnew" and znew were given these are the predicted or estimated values, otherwise it is NULL.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

ulc.glm, lc.reg, lc.reg2, alfa.pcr, glm.pcr

Examples

```r
y <- rbinom(150, 1, 0.5)
x <- rdiri(150, runif(3, 1, 4))
x <- x / rowSums(x)
mod1 <- lc.glm(y, x)
```

Description

Log-contrast regression with compositional predictor variables.

Usage

```r
lc.reg(y, x, z = NULL, xnew = NULL, znew = NULL)
```
Log-contrast regression with compositional predictor variables

Arguments

\(y\) A numerical vector containing the response variable values. This must be a continuous variable.

\(x\) A matrix with the predictor variables, the compositional data. No zero values are allowed.

\(z\) A matrix, data.frame, factor or a vector with some other covariate(s).

\(x_{\text{new}}\) A matrix containing the new compositional data whose response is to be predicted. If you have no new data, leave this NULL as is by default.

\(z_{\text{new}}\) A matrix, data.frame, factor or a vector with the values of some other covariate(s). If you have no new data, leave this NULL as is by default.

Details

The function performs the log-contrast regression model as described in Aitchison (2003), pg. 84-85. The logarithm of the compositional predictor variables is used (hence no zero values are allowed). The response variable is linked to the log-transformed data with the constraint that the sum of the regression coefficients equals 0. Hence, we apply constrained least squares, which has a closed form solution. The constrained least squares is described in Chapter 8.2 of Hansen (2019). The idea is to minimise the sum of squares of the residuals under the constraint \(R^T \beta = c\), where \(c = 0\) in our case. If you want the regression without the zum-to-zero contraints see \texttt{ulc.reg}. Extra predictors variables are allowed as well, for instance categorical or continuous.

Value

A list including:

\(\text{be}\) The constrained regression coefficients. Their sum equals 0.

\(\text{covbe}\) If covariance matrix of the constrained regression coefficients.

\(\text{va}\) The estimated regression variance.

\(\text{residuals}\) The vector of residuals.

\(\text{est}\) If the arguments "xnew" and znew were given these are the predicted or estimated values, otherwise it is NULL.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

\texttt{ulc.reg, lcreg.aov, lc.reg2, alfa.pcr, alfa.knn.reg}
Log-contrast regression with multiple compositional predictors

Examples

```r
y <- iris[, 1]
x <- as.matrix(iris[, 2:4])
x <- x / rowSums(x)
mod1 <- lc.reg(y, x)
mod2 <- lc.reg(y, x, z = iris[, 5])
```

Description

Log-contrast regression with multiple compositional predictors.

Usage

```r
lc.reg2(y, x, z = NULL, xnew = NULL, znew = NULL)
```

Arguments

- `y` A numerical vector containing the response variable values. This must be a continuous variable.
- `x` A list with multiple matrices with the predictor variables, the compositional data. No zero values are allowed.
- `z` A matrix, data.frame, factor or a vector with some other covariate(s).
- `xnew` A matrix containing a list with multiple matrices with compositional data whose response is to be predicted. If you have no new data, leave this NULL as is by default.
- `znew` A matrix, data.frame, factor or a vector with the values of some other covariate(s). If you have no new data, leave this NULL as is by default.

Details

The function performs the log-contrast regression model as described in Aitchison (2003), pg. 84-85. The logarithm of the compositional predictor variables is used (hence no zero values are allowed). The response variable is linked to the log-transformed data with the constraint that the sum of the regression coefficients equals 0. Hence, we apply constrained least squares, which has a closed form solution. The constrained least squares is described in Chapter 8.2 of Hansen (2019). The idea is to minimise the sum of squares of the residuals under the constraint $R^T \beta = c$, where $c = 0$ in our case. If you want the regression without the zum-to-zero constraints see `ulc.reg2`. Extra predictors variables are allowed as well, for instance categorical or continuous. The difference with `lc.reg` is that instead of one, there are multiple compositions treated as predictor variables.
Log-contrast regression with multiple compositional predictors

Value

A list including:

be
The constrained regression coefficients. Their sum equals 0.
covbe
If covariance matrix of the constrained regression coefficients.
va
The estimated regression variance.
residuals
The vector of residuals.
est
If the arguments "xnew" and "znew" were given these are the predicted or estimated values, otherwise it is NULL.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

ulc.reg2, lc.reg, ulc.reg, lcreg.aov, alfa.pcr, alfa.knn.reg

Examples

```r
y <- iris[, 1]
x <- list()
x1 <- as.matrix(iris[, 2:4])
x1 <- x1 / rowSums(x1)
x[[ 1 ]] <- x1
x[[ 2 ]] <- rdiri(150, runif(4) )
x[[ 3 ]] <- rdiri(150, runif(5) )
mod <- lc.reg2(y, x)
```
Log-likelihood ratio test for a Dirichlet mean vector

Description

Log-likelihood ratio test for a Dirichlet mean vector.

Usage

dirimean.test(x, a)

Arguments

x A matrix with the compositional data. No zero values are allowed.
a A compositional mean vector. The concentration parameter is estimated at first. If the elements do not sum to 1, it is assumed that the Dirichlet parameters are supplied.

Details

Log-likelihood ratio test is performed for the hypothesis the given vector of parameters "a" describes the compositional data well.

Value

If there are no zeros in the data, a list including:

param A matrix with the estimated parameters under the null and the alternative hypothesis.
loglik The log-likelihood under the alternative and the null hypothesis.
info The value of the test statistic and its relevant p-value.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References


See Also

sym.test, diri.nr, diri.est, rdiri, ddiri
Log-likelihood ratio test for a symmetric Dirichlet distribution

Examples

```r
x <- rdiri(100, c(1, 2, 3))
dirimean.test(x, c(1, 2, 3))
dirimean.test(x, c(1, 2, 3)/6)
```

Description

Log-likelihood ratio test for a symmetric Dirichlet distribution.

Usage

```r
sym.test(x)
```

Arguments

- `x`: A matrix with the compositional data. No zero values are allowed.

Details

Log-likelihood ratio test is performed for the hypothesis that all Dirichlet parameters are equal.

Value

A list including:

- `est.par`: The estimated parameters under the alternative hypothesis.
- `one.par`: The value of the estimated parameter under the null hypothesis.
- `res`: The loglikelihood under the alternative and the null hypothesis, the value of the test statistic, its relevant \( p \)-value and the associated degrees of freedom, which are actually the dimensionality of the simplex, \( D - 1 \), where \( D \) is the number of components.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

**Mixture model selection via BIC**

See Also

`diri.nr, diri.est, rdiri, dirimean.test`

Examples

```r
x <- rdiri(100, c(5, 7, 1, 3, 10, 2, 4))
sym.test(x)
x <- rdiri(100, c(5, 5, 5, 5, 5))
sym.test(x)
```

**Mixture model selection via BIC**

*Description*

Mixture model selection via BIC.

*Usage*

```r
bic.mixcompnorm(x, G, type = "alr", veo = FALSE, graph = TRUE)
```

*Arguments*

- `x`: A matrix with compositional data.
- `G`: A numeric vector with the number of components, clusters, to be considered, e.g. 1:3.
- `type`: The type of transformation to be used, either the additive log-ratio ("alr"), the isometric log-ratio ("ilr") or the pivot coordinate ("pivot") transformation.
- `veo`: Stands for "Variables exceed observations". If TRUE then if the number variables in the model exceeds the number of observations, but the model is still fitted.
- `graph`: A boolean variable, TRUE or FALSE specifying whether a graph should be drawn or not.

*Details*

The alr or the ilr-transformation is applied to the compositional data first and then mixtures of multivariate Gaussian distributions are fitted. BIC is used to decide on the optimal model and number of components.
Mixture model selection with the alpha-transformation using BIC

Value

A plot with the BIC of the best model for each number of components versus the number of components. A list including:

- mod: A message informing the user about the best model.
- BIC: The BIC values for every possible model and number of components.
- optG: The number of components with the highest BIC.
- optmodel: The type of model corresponding to the highest BIC.

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

mix.compnorm, mix.compnorm.contour, rmixcomp, bic.alfamixnorm

Examples

```r
## Not run:
x <- as.matrix( iris[, 1:4] )
x <- x/ rowSums(x)
bic.mixcompmnorm(x, 1:3, type = "alr", graph = FALSE)
bic.mixcompmnorm(x, 1:3, type = "ilr", graph = FALSE)
## End(Not run)
```

Description

Mixture model selection with the α-transformation using BIC.

Usage

`bic.alfamixnorm(x, G, a = seq(-1, 1, by = 0.1), veo = FALSE, graph = TRUE)`
Mixture model selection with the alpha-transformation using BIC

Arguments

- **x**: A matrix with compositional data.
- **G**: A numeric vector with the number of components, clusters, to be considered, e.g. 1:3.
- **a**: A vector with a grid of values of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If $\alpha = 0$ the isometric log-ratio transformation is applied.
- **veo**: Stands for "Variables exceed observations". If TRUE then if the number variables in the model exceeds the number of observations, but the model is still fitted.
- **graph**: A boolean variable, TRUE or FALSE specifying whether a graph should be drawn or not.

Details

The $\alpha$-transformation is applied to the compositional data first and then mixtures of multivariate Gaussian distributions are fitted. BIC is used to decide on the optimal model and number of components.

Value

A list including:

- **abic**: A list that contains the matrices of all BIC values for all values of $\alpha$.
- **optalpha**: The value of $\alpha$ that leads to the highest BIC.
- **optG**: The number of components with the highest BIC.
- **optmodel**: The type of model corresponding to the highest BIC.

If graph is set equal to TRUE a plot with the BIC of the best model for each number of components versus the number of components and a list with the results of the Gaussian mixture model for each value of $\alpha$.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


MLE for the multivariate \( t \) distribution

**See Also**

`alfa.mix.norm, mix.compnorm, mix.compnorm.contour, rmixcomp, alfa, alfa.knn, alfa.rda, comp.nb`

**Examples**

```r
## Not run:
x <- as.matrix( iris[, 1:4] )
x <- x/ rowSums(x)
bic.alfamixnorm(x, 1:3, a = c(0.4, 0.5, 0.6), graph = FALSE)
## End(Not run)
```

### Description

MLE of the parameters of a multivariate \( t \) distribution.

### Usage

```r
multivt(y, plot = FALSE)
```

### Arguments

- `y` A matrix with continuous data.
- `plot` If plot is TRUE the value of the maximum log-likelihood as a function of the degrees of freedom is presented.

### Details

The parameters of a multivariate \( t \) distribution are estimated. This is used by the functions `comp.den` and `bivt.contour`.

### Value

A list including:

- `center` The location estimate.
- `scatter` The scatter matrix estimate.
- `df` The estimated degrees of freedom.
- `loglik` The log-likelihood value.
- `mesos` The classical mean vector.
- `covariance` The classical covariance matrix.
MLE of distributions defined in the \((0, 1)\) interval

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References

See Also
bivt.contour, comp.den, rmvt

Examples
```r
x <- as.matrix(iris[, 1:4])
multivt(x)
```

MLE of distributions defined in the \((0, 1)\) interval

Description
MLE of distributions defined in the \((0, 1)\) interval.

Usage
```r
beta.est(x, tol = 1e-07) logitnorm.est(x) hsecant01.est(x, tol = 1e-07) simplex.est(x, tol = 1e-07) kumar.est(x, tol = 1e-07) unitweibull.est(x, tol = 1e-07, maxiters = 100) ibeta.est(x, tol = 1e-07) zilogitnorm.est(x)
```

Arguments
- `x`: A numerical vector with proportions, i.e. numbers in \((0, 1)\) (zeros and ones are not allowed).
- `tol`: The tolerance level up to which the maximisation stops.
- `maxiters`: The maximum number of iterations the Newton-Raphson algorithm will perform.
MLE of distributions defined in the (0, 1) interval

Details

Maximum likelihood estimation of the parameters of some distributions are performed, some of which use the Newton-Raphson. Some distributions and hence the functions do not accept zeros. "logitnorm.mle" fits the logistic normal, hence no Newton-Raphson is required and the "hypersecant01.mle" and "simplex.est" use the golden ratio search as it is faster than the Newton-Raphson (less computations). The "zilogitnorm.est" stands for the zero inflated logistic normal distribution. The "ibeta.est" fits the zero or the one inflated beta distribution.

Value

A list including:

- iters: The number of iterations required by the Newton-Raphson.
- loglik: The value of the log-likelihood.
- param: The estimated parameters. In the case of "hypersecant01.est" this is called "theta" as there is only one parameter.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


You can also check the relevant wikipedia pages.

See Also

diri.nr2,

Examples

```r
x <- rbeta(1000, 1, 4)
beta.est(x)
ibeta.est(x)

x <- runif(1000)
hsecant01.est(x)
logitnorm.est(x)
ibeta.est(x)
```
MLE of the Dirichlet distribution

```r
x <- rbeta(1000, 2, 5)
x[sample(1:1000, 50)] <- 0
ibeta.est(x)
```

**MLE of the Dirichlet distribution**

**Description**

MLE of the parameters of a Dirichlet distribution.

**Usage**

```r
diri.est(x, type = "mle")
```

**Arguments**

- `x`: A matrix containing compositional data.
- `type`: If you want to estimate the parameters use `type="mle"`. If you want to estimate the mean vector along with the precision parameter, the second parametrisation of the Dirichlet, use `type="prec"`.

**Details**

Maximum likelihood estimation of the parameters of a Dirichlet distribution is performed.

**Value**

A list including:

- `loglik`: The value of the log-likelihood.
- `param`: The estimated parameters.
- `phi`: The estimated precision parameter, if `type = "prec"`.
- `mu`: The estimated mean vector, if `type = "prec"`.
- `runtime`: The run time of the maximisation procedure.

**Author(s)**

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

**References**


See Also

\texttt{diri.nr,diri.contour,rdiri,ddiri,dda,diri.reg}

Examples

\begin{verbatim}
x <- rdiri( 100, c(5, 7, 1, 3, 10, 2, 4) )
diri.est(x)
diri.est(x, type = "prec")
\end{verbatim}

Description

MLE of the Dirichlet distribution via Newton-Raphson.

Usage

\texttt{diri.nr(x, type = 1, tol = 1e-07)}

Arguments

- \texttt{x} A matrix containing compositional data. Zeros are not allowed.
- \texttt{type} Type can either be 1, so that the Newton-Raphson is used for the maximisation of the log-likelihood, as Minka (2002) suggested or it can be 1. In the latter case the Newton-Raphson algorithm is implemented involving matrix inversions. In addition an even faster implementation has been implemented (in C++) in the package \texttt{Rfast} and is used here.
- \texttt{tol} The tolerance level indicating no further increase in the log-likelihood.

Details

Maximum likelihood estimation of the parameters of a Dirichlet distribution is performed via Newton-Raphson. Initial values suggested by Minka (2003) are used. The estimation is super faster than "diri.est" and the difference becomes really apparent when the sample size and or the dimensions increase. In fact this will work with millions of observations. So in general, I trust this one more than "diri.est".

The only problem I have seen with this method is that if the data are concentrated around a point, say the center of the simplex, it will be hard for this and the previous methods to give estimates of the parameters. In this extremely difficult scenario I would suggest the use of the previous function with the precision parametrization "diri.est(x, type = "prec")". It will be extremely fast and accurate.
MLE of the folded model for a given value of alpha

Value
A list including:
- `iter` The number of iterations required. If the argument "type" is set to 2 this is not returned.
- `loglik` The value of the log-likelihood.
- `param` The estimated parameters.
- `runtime` The run time of the procedure.

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

See Also
diri.est, diri.contour rdiri, ddiri, dda

Examples
```r
x <- rdiri( 100, c(5, 7, 5, 8, 10, 6, 4) )
diri.nr(x)
diri.nr(x, type = 2)
diri.est(x)
```

MLE of the folded model for a given value of alpha

Description
MLE of the folded model for a given value of $\alpha$.

Usage
```r
alpha.mle(x, a)
a.mle(a, x)
```

Arguments
- `x` A matrix with the compositional data. No zero values are allowed.
- `a` A value of $\alpha$. 
Details

This is a function for choosing or estimating the value of $\alpha$ in the folded model (Tsagris and Stewart, 2020). It is called by `a.est`.

Value

If "alpha.mle" is called, a list including:

- `iters` The number of iterations the EM algorithm required.
- `loglik` The maximized log-likelihood of the folded model.
- `p` The estimated probability inside the simplex of the folded model.
- `mu` The estimated mean vector of the folded model.
- `su` The estimated covariance matrix of the folded model.

If "a.mle" is called, the log-likelihood is returned only.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

`alfa.profile, alfa, alfavinv, a.est`

Examples

```r
x <- as.matrix(iris[, 1:4])
x <- x / rowSums(x)
mod <- alfa.tune(x)
mod
alpha.mle(x, mod[1])
```
MLE of the zero adjusted Dirichlet distribution

Usage
zad.est(y, tol = 1e-05)

Arguments
y A matrix with the compositional data (dependent variable). The number of observations (vectors) with no zero values should be more than the columns of the predictor variables. Otherwise, the initial values will not be calculated.
tol A tolerance level to terminate the maximisation process.

Details
A zero adjusted Dirichlet distribution is being fitted and its parameters are estimated.

Value
A list including:
loglik The value of the log-likelihood.
phi The precision parameter. If covariates are linked with it (function "diri.reg2"), this will be a vector.
mu The mean vector of the distribution.
runtime The time required by the regression.

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

See Also
zadr, diri.nr, zilogitnorm.est, zereplace
Multivariate analysis of variance

Examples

```r
y <- as.matrix(iris[, 1:3])
y <- y / rowSums(y)
mod1 <- diri.nr(y)
y[sample(1:450, 15)] <- 0
mod2 <- zad.est(y)
```

Description

Multivariate analysis of variance with assuming equality of the covariance matrices.

Usage

```r
maov(x, ina)
```

Arguments

- **x**: A matrix containing Euclidean data.
- **ina**: A numerical or factor variable indicating the groups of the data.

Details

Multivariate analysis of variance assuming equality of the covariance matrices.

Value

A list including:

- **note**: A message stating whether the F or the chi-square approximation has been used.
- **result**: The test statistic and the p-value.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References


Multivariate analysis of variance (James test)

See Also

maovjames, hotel2T2, james, comp.test

Examples

maov( as.matrix(iris[,1:4]), iris[,5] )
maovjames( as.matrix(iris[,1:4]), iris[,5] )

Description

Multivariate analysis of variance without assuming equality of the covariance matrices.

Usage

maovjames(x, ina, a = 0.05)

Arguments

x               A matrix containing Euclidean data.
ina             A numerical or factor variable indicating the groups of the data.
a               The significance level, set to 0.005 by default.

Details

Multivariate analysis of variance without assuming equality of the covariance matrices.

Value

A vector with the next 4 elements:

test            The test statistic.
correction      The value of the correction factor.
corr.critical   The corrected critical value of the chi-square distribution.
p-value          The p-value of the corrected test statistic.

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>. 
References


See Also

maov, hotel2T2, james, comp.test

Examples

maov( as.matrix(iris[,1:4]), iris[,5] )
maovjames( as.matrix(iris[,1:4]), iris[,5] )

Description

Multivariate kernel density estimation.

Usage

mkde(x, h = NULL, thumb = "silverman")

Arguments

x  A matrix with Euclidean (continuous) data.

h  The bandwidth value. It can be a single value, which is turned into a vector and then into a diagonal matrix, or a vector which is turned into a diagonal matrix. If you put this NULL then you need to specify the "thumb" argument below.

thumb  Do you want to use a rule of thumb for the bandwidth parameter? If no, set h equal to NULL and put "estim" for maximum likelihood cross-validation, "scott" or "silverman" for Scott’s and Silverman’s rules of thumb respectively.

Details

The multivariate kernel density estimate is calculated with a (not necessarily given) bandwidth value.

Value

A vector with the density estimates calculated for every vector.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>. 
Multivariate kernel density estimation for compositional data

References

See Also
mkde.tune, comp.kerncontour

Examples

mkde( as.matrix(iris[, 1:4]), thumb = "scott" )
mkde( as.matrix(iris[, 1:4]), thumb = "silverman" )

Description
Multivariate kernel density estimation for compositional data.

Usage

comp.kern(x, type = "alr", h = NULL, thumb = "silverman")

Arguments

x 
A matrix with Euclidean (continuous) data.

type 
The type of trasformation used, either the additive log-ratio ("alr"), the isometric log-ratio ("ilr") or the pivot coordinate ("pivot") transformation.

h 
The bandwidth value. It can be a single value, which is turned into a vector and then into a diagonal matrix, or a vector which is turned into a diagonal matrix. If it is NULL, then you need to specify the "thumb" argument below.

thumb 
Do you want to use a rule of thumb for the bandwidth parameter? If no, leave the "h" NULL and put "estim" for maximum likelihood cross-validation, "scott" or "silverman" for Scott’s and Silverman’s rules of thumb respectively.

Details
The multivariate kernel density estimate is calculated with a (not necessarily given) bandwidth value.

Value
A vector with the density estimates calculated for every vector.
Multivariate linear regression

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References
Arsalane Chouaib Guidoum (2015). Kernel Estimator and Bandwidth Selection for Density and its
Derivatives.
The kedd R package.

See Also
comp.kerncontour, mkde

Examples
x <- as.matrix(iris[, 1:3])
x <- x / rowSums(x)
f <- comp.kern(x)

Multivariate linear regression
Multivariate linear regression

Description
Multivariate linear regression.

Usage
multivreg(y, x, plot = TRUE, xnew = NULL)

Arguments

y A matrix with the Euclidean (continuous) data.
x A matrix with the predictor variable(s), they have to be continuous.
plot Should a plot appear or not?
xnew If you have new data use it, otherwise leave it NULL.

Details
The classical multivariate linear regression model is obtained.
Value

A list including:

- **suma**: A summary as produced by `lm`, which includes the coefficients, their standard error, t-values, p-values.
- **r.squared**: The value of the $R^2$ for each univariate regression.
- **resid.out**: A vector with number indicating which vectors are potential residual outliers.
- **x.leverage**: A vector with number indicating which vectors are potential outliers in the predictor variables space.
- **out**: A vector with number indicating which vectors are potential outliers in the residuals and in the predictor variables space.
- **est**: The predicted values if xnew is not NULL.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References


See Also

diri.reg, js.compreg, kl.compreg, ols.compreg, comp.reg

Examples

```r
library(MASS)
x <- as.matrix(iris[, 1:2])
y <- as.matrix(iris[, 3:4])
multivreg(y, x, plot = TRUE)
```

Description

Multivariate normal random values simulation on the simplex.

Usage

```r
rcompnorm(n, m, s, type = "alr")
```
Arguments

- **n**: The sample size, a numerical value.
- **m**: The mean vector in $\mathbb{R}^d$.
- **s**: The covariance matrix in $\mathbb{R}^d$.
- **type**: The alr (type = "alr") or the ilr (type = "ilr") is to be used for closing the Euclidean data onto the simplex.

Details

The algorithm is straightforward, generate random values from a multivariate normal distribution in $\mathbb{R}^d$ and brings the values to the simplex $S^d$ using the inverse of a log-ratio transformation.

Value

A matrix with the simulated data.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

- `comp.den`, `rdiri`, `rcompt`, `rcompsn`

Examples

```r
x <- as.matrix(iris[, 1:2])
m <- colMeans(x)
s <- var(x)
y <- rcompnorm(100, m, s)
comp.den(y)
ternary(y)
```
Multivariate or univariate regression with compositional data in the covariates side using the alpha-transformation

Description

Multivariate or univariate regression with compositional data in the covariates side using the \( \alpha \)-transformation.

Usage

\texttt{alfa.pcr(y, x, a, k, model = "gaussian", xnew = NULL)}

Arguments

\textbf{y}  
A numerical vector containing the response variable values. They can be continuous, binary, discrete (counts). This can also be a vector with discrete values or a factor for the multinomial regression (model = "multinomial").

\textbf{x}  
A matrix with the predictor variables, the compositional data.

\textbf{a}  
The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If \( \alpha = 0 \) the isometric log-ratio transformation is applied.

\textbf{k}  
A number at least equal to 1. How many principal components to use.

\textbf{model}  
The type of regression model to fit. The possible values are "gaussian", "multinomial", "binomial" and "poisson".

\textbf{xnew}  
A matrix containing the new compositional data whose response is to be predicted. If you have no new data, leave this NULL as is by default.

Details

The \( \alpha \)-transformation is applied to the compositional data first, the first \( k \) principal component scores are calculated and used as predictor variables for a regression model. The family of distributions can be either, "normal" for continuous response and hence normal distribution, "binomial" corresponding to binary response and hence logistic regression or "poisson" for count response and poisson regression.

Value

A list including:

\textbf{be}  
If linear regression was fitted, the regression coefficients of the \( k \) principal component scores on the response variable \( y \).

\textbf{mod}  
If another regression model was fitted its outcome as produced in the package \texttt{Rfast}.
Multivariate regression with compositional data

Description

Multivariate regression with compositional data.

Usage

comp.reg(y, x, type = "classical", xnew = NULL, yb = NULL)

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

pcr, glm.pcr, alfa.pcr.tune

Examples

library(MASS)
y <- as.vector(fgl[, 1])
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
mod <- alfa.pcr(y = y, x = x, 0.7, 1)
mod

Author(s)

The percentage of variance explained by the first k principal components.

vec

The first k principal components, loadings or eigenvectors. These are useful for future prediction in the sense that one needs not fit the whole model again.

est

If the argument "xnew" was given these are the predicted or estimated values, otherwise it is NULL.
Multivariate regression with compositional data

Arguments

\textit{y} \hspace{1cm} A matrix with compositional data. Zero values are not allowed.
\textit{x} \hspace{1cm} The predictor variable(s), they have to be continuous.
\textit{type} \hspace{1cm} The type of regression to be used, "classical" for standard multivariate regression, or "spatial" for the robust spatial median regression. Alternatively you can type "lmfit" for the fast classical multivariate regression that does not return standard errors whatsoever.
\textit{xnew} \hspace{1cm} This is by default set to NULL. If you have new data whose compositional data values you want to predict, put them here.
\textit{yb} \hspace{1cm} If you have already transformed the data using the additive log-ratio transformation, put it here. Otherwise leave it NULL. This is intended to be used in the function \texttt{alfareg.tune} in order to speed up the process.

Details

The additive log-ratio transformation is applied and then the chosen multivariate regression is implemented. The alr is easier to explain than the ilr and that is why the latter is avoided here.

Value

A list including:

\textit{runtime} \hspace{1cm} The time required by the regression.
\textit{be} \hspace{1cm} The beta coefficients.
\textit{seb} \hspace{1cm} The standard error of the beta coefficients.
\textit{est} \hspace{1cm} The fitted values of \textit{xnew} if \textit{xnew} is not NULL.

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris \texttt{mtsagris@uoc.gr} and Giorgos Athineou \texttt{gioathineou@gmail.com}.

References


See Also

\texttt{multivreg}, \texttt{spatmed.reg}, \texttt{js.compreg}, \texttt{diri.reg}
Multivariate skew normal random values simulation on the simplex

Examples

```r
library(MASS)
y <- as.matrix(iris[, 1:3])
y <- y / rowSums(y)
x <- as.vector(iris[, 4])
mod1 <- comp.reg(y, x)
mod2 <- comp.reg(y, x, type = "spatial")
```

Description

Multivariate skew normal random values simulation on the simplex.

Usage

```r
rcompsn(n, xi, Omega, alpha, dp = NULL, type = "alr")
```

Arguments

- `n`: The sample size, a numerical value.
- `xi`: A numeric vector of length \(d\) representing the location parameter of the distribution.
- `Omega`: A \(d \times d\) symmetric positive-definite matrix of dimension.
- `alpha`: A numeric vector which regulates the slant of the density.
- `dp`: A list with three elements, corresponding to xi, Omega and alpha described above. The default value is FALSE. If dp is assigned, individual parameters must not be specified.
- `type`: The alr (type = "alr") or the ilr (type = "ilr") is to be used for closing the Euclidean data onto the simplex.

Details

The algorithm is straightforward, generate random values from a multivariate t distribution in \(R^d\) and brings the values to the simplex \(S^d\) using the inverse of a log-ratio transformation.

Value

A matrix with the simulated data.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.
Multivariate t random values simulation on the simplex

References


See Also

comp.den, rdiri, rcompnorm, rmvt

Examples

x <- as.matrix(iris[, 1:2])
par <- sn::msn.mle(y = x)$dp
y <- rcompsn(100, dp = par)
comp.den(y, dist = "skewnorm")
ternary(y)

Description

Multivariate t random values simulation on the simplex.

Usage

rcompt(n, m, s, dof, type = "alr")

Arguments

n
The sample size, a numerical value.

m
The mean vector in \( R^d \).

s
The covariance matrix in \( R^d \).

dof
The degrees of freedom.

type
The alr (type = "alr") or the ilr (type = "ilr") is to be used for closing the Euclidean data onto the simplex.

Details

The algorithm is straightforward, generate random values from a multivariate t distribution in \( R^d \) and brings the values to the simplex \( S^d \) using the inverse of a log-ratio transformation.
Naive Bayes classifiers for compositional data

Value
A matrix with the simulated data.

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

See Also
comp.den, rdiri, rcompnorm, rmvt

Examples
x <- as.matrix(iris[, 1:2])
m <- Rfast::colmeans(x)
s <- var(x)
y <- rcompt(100, m, s, 10)
comp.den(y, dist = "t")
ternary(y)

Naive Bayes classifiers for compositional data

Description
Naive Bayes classifiers for compositional data.

Usage
comp.nb(xnew = NULL, x, ina, type = "beta")

Arguments
xnew A matrix with the new compositional predictor data whose class you want to predict. Zeros are not allowed
x A matrix with the available compositional predictor data. Zeros are not allowed
ina A vector of data. The response variable, which is categorical (factor is acceptable).
type The type of naive Bayes, "beta", "logitnorm", "cauchy", "laplace", "gamma", "normlog" or "weibull". For the last 4 distributions, the negative of the logarithm of the compositional data is applied first.
Value

Depending on the classifier a list including (the ni and est are common for all classifiers):

- **shape**: A matrix with the shape parameters.
- **scale**: A matrix with the scale parameters.
- **expmu**: A matrix with the mean parameters.
- **sigma**: A matrix with the (MLE, hence biased) variance parameters.
- **location**: A matrix with the location parameters (medians).
- **scale**: A matrix with the scale parameters.
- **mean**: A matrix with the scale parameters.
- **var**: A matrix with the variance parameters.
- **a**: A matrix with the "alpha" parameters.
- **b**: A matrix with the "beta" parameters.
- **ni**: The sample size of each group in the dataset.
- **est**: The estimated group of the xnew observations. It returns a numerical value back regardless of the target variable being numerical as well or factor. Hence, it is suggested that you do \"as.numeric(est)\" in order to see what is the predicted class of the new data.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

cv.compnb, alfa.rda, alfa.knn, comp.knn, mix.compnorm, dda

Examples

```r
x <- Compositional::rdiri(100, runif(5) )
ina <- rbinom(100, 1, 0.5) + 1
a <- comp.nb(x, x, ina, type = "beta")
```
Naive Bayes classifiers for compositional data using the alpha-transformation

Description

Naive Bayes classifiers for compositional data using the $\alpha$-transformation.

Usage

alfa.nb(xnew, x, ina, a, type = "gaussian")

Arguments

- **xnew**: A matrix with the new compositional predictor data whose class you want to predict. Zeros are allowed.
- **x**: A matrix with the available compositional predictor data. Zeros are allowed.
- **ina**: A vector of data. The response variable, which is categorical (factor is acceptable).
- **a**: This can be a vector of values or a single number.
- **type**: The type of naive Bayes, "gaussian", "cauchy" or "laplace".

Details

The $\alpha$-transformation is applied to the compositional and a naive Bayes classifier is employed.

Value

A matrix with the estimated groups. One column for each value of $\alpha$.

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

comp.nb, alfa.rda, alfa.knn, comp.knn, mix.compnorm
Examples

```r
x <- Compositional::rdiri(100, runif(5) )
ina <- rbinom(100, 1, 0.5) + 1
mod <- alfa.nb(x, x, a = c(0, 0.1, 0.2), ina )
```

Description

Non linear least squares regression for compositional data.

Usage

```r
ols.compreg(y, x, B = 1, ncores = 1, xnew = NULL)
```

Arguments

- `y`: A matrix with the compositional data (dependent variable). Zero values are allowed.
- `x`: The predictor variable(s), they have to be continuous.
- `B`: If B is greater than 1 bootstrap estimates of the standard error are returned. If B=1, no standard errors are returned.
- `ncores`: If ncores is 2 or more parallel computing is performed. This is to be used for the case of bootstrap. If B=1, this is not taken into consideration.
- `xnew`: If you have new data use it, otherwise leave it NULL.

Details

The ordinary least squares between the observed and the fitted compositional data is adopted as the objective function. This involves numerical optimization since the relationship is non linear. There is no log-likelihood.

Value

A list including:

- `runtime`: The time required by the regression.
- `beta`: The beta coefficients.
- `covbe`: The covariance matrix of the beta coefficients, if bootstrap is chosen, i.e. if B > 1.
- `est`: The fitted of xnew if xnew is not NULL.
Non parametric zero replacement strategies

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

diri.reg, js.compreg, kl.compreg, comp.reg, comp.reg, alfa.reg

Examples

library(MASS)
x <- as.vector(fgl[, 1])
y <- as.matrix(fgl[, 2:9])
y <- y / rowSums(y)
mod1 <- ols.compreg(y, x, B = 1, ncores = 1)
mod2 <- js.compreg(y, x, B = 1, ncores = 1)

Non parametric zero replacement strategies

Description

Non parametric zero replacement strategies.

Usage

zeroreplace(x, a = 0.65, delta = NULL, type = "multiplicative")

Arguments

x A matrix with the compositional data.
a The replacement value (\(\delta\)) will be "a" times the minimum value observed in the compositional data.
delta Unless you specify the replacement value \(\delta\) here.
type This can be any of "multiplicative", "additive" or "simple". See the references for more details.

Details

The "additive" is the zero replacement strategy suggested in Aitchison (1986, pg. 269). All of the three strategies can be found in Martin-Fernandez et al. (2003).
Perturbation operation

Value

A matrix with the zero replaced compositional data.

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

perturbation, alfa

Examples

```r
x <- as.matrix(iris[1:20, 1:4])
x <- x / rowSums(x)
x[ sample(1:20, 4), sample(1:4, 1) ] <- 0
x <- x / rowSums(x)
zeroreplace(x)
```

Perturbation operation

Perturbation operation

Description

Perturbation operation.

Usage

```r
perturbation(x, y, oper = "+")
```

Arguments

- **x**: A matrix with the compositional data.
- **y**: Either a matrix with compositional data or a vector with compositional data. In either case, the data may not be compositional data, as long as they non negative.
- **oper**: For the summation this must be "*" and for the negation it must be "/". According to Aitchison (1986), multiplication is equal to summation in the log-space, and division is equal to negation.
Details

This is the perturbation operation defined by Aitchison (1986).

Value

A matrix with the perturbed compositional data.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

power

Examples

```r
x <- as.matrix(iris[1:15, 1:4])
y <- as.matrix(iris[21:35, 1:4])
perturbation(x, y)
perturbation(x, y[1, ])
```

Description

Plot of the LASSO coefficients.

Usage

```r
lassocoef.plot(lasso, lambda = TRUE)
```

Arguments

- `lasso`: An object where you have saved the result of the LASSO regression. See the examples for more details.
- `lambda`: If you want the x-axis to contain the logarithm of the penalty parameter \(\log(\lambda)\) set this to TRUE. Otherwise the x-axis will contain the \(L_1\)-norm of the coefficients.
Details

This function plots the $L_2$-norm of the coefficients of each predictor variable versus the $\log(\lambda)$ or the $L_1$-norm of the coefficients. This is the same plot as the one produced by the glmnet package with type.coef = "2norm".

Value

A plot of the $L_2$-norm of the coefficients of each predictor variable (y-axis) versus the $L_1$-norm of all the coefficients (x-axis).

Author(s)

Michail Tsagris and Abdulaziz Alenazi.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Abdulaziz Alenazi <a.alenazi@nbu.edu.sa>.

References


See Also

lasso.klcompreg, cv.lasso.klcompreg, lasso.compreg, cv.lasso.compreg, kl.compreg, comp.reg

Examples

```r
y <- as.matrix(iris[, 1:4])
y <- y / rowSums(y)
x <- matrix( rnorm(150 * 30), ncol = 30 )
a <- lasso.klcompreg(y, x)
lassocoef.plot(a)
b <- lasso.compreg(y, x)
lassocoef.plot(b)
```

Description

Power operation.

Usage

`pow(x, a)`
Principal component analysis

Arguments

x             A matrix with the compositional data.
a             Either a vector with numbers of a single number.

Details

This is the power operation defined by Aitchison (1986). It is also the starting point of the \( \alpha \)-transformation.

Value

A matrix with the power transformed compositional data.

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

perturbation, alfa

Examples

x <- as.matrix(iris[1:15, 1:4])
a <- runif(1)
pow(x, a)

logpca(x, center = TRUE, scale = TRUE, k = NULL, vectors = FALSE)
Principal component analysis

Arguments

x          A matrix with the compositional data. Zero values are not allowed.
center     Do you want your data centered? TRUE or FALSE.
scale      Do you want each of your variables scaled, i.e. to have unit variance? TRUE or FALSE.
k          If you want a specific number of eigenvalues and eigenvectors set it here, otherwise all eigenvalues (and eigenvectors if requested) will be returned.
vectors    Do you want the eigenvectors be returned? By default this is FALSE.

Details

The logarithm is applied to the compositional data and PCA is performed.

Value

A list including:

values     The eigenvalues.
vectors    The eigenvectors.

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

alfa.pca, alfa.pcr, kl.alfapcr

Examples

x <- as.matrix(iris[, 1:4])
x <- x/rowSums(x)
a <- logpca(x)
Principal component analysis using the alpha-transformation

Description

Principal component analysis using the $\alpha$-transformation.

Usage

alfa.pca(x, a, center = TRUE, scale = TRUE, k = NULL, vectors = FALSE)

Arguments

- **x**: A matrix with the compositional data. Zero values are allowed. In that case "a" should be positive.
- **a**: The value of $\alpha$ to use in the $\alpha$-transformation.
- **center**: Do you want your data centered? TRUE or FALSE.
- **scale**: Do you want each of your variables scaled, i.e. to have unit variance? TRUE or FALSE.
- **k**: If you want a specific number of eigenvalues and eigenvectors set it here, otherwise all eigenvalues (and eigenvectors if requested) will be returned.
- **vectors**: Do you want the eigenvectors be returned? By default this is FALSE.

Details

The $\alpha$-transformation is applied to the compositional data and then PCA is performed. Note however, that the right multiplication by the Helmert sub-matrix is not applied in order to be in accordance with Aitchison (1983). When $\alpha = 0$, this results to the PCA proposed by Aitchison (1983).

Value

A list including:

- **values**: The eigenvalues.
- **vectors**: The eigenvectors.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.
References


See Also

logpca, alfa.pcr, kl.alfapcr

Examples

```r
x <- as.matrix(iris[, 1:4])
x <- x/ rowSums(x)
a <- alfa.pca(x, 0.5)
```

Principal component generalised linear models

**Description**

Principal component generalised linear models.

**Usage**

```r
pcr(y, x, k = 1, xnew = NULL)
glm.pcr(y, x, k = 1, xnew = NULL)
```

**Arguments**

- `y` A numerical vector, a real values vector or a numeric vector with 0 and 1 (binary) or a vector with discrete (count) data.
- `x` A matrix with the predictor variable(s), they have to be continuous.
- `k` A number greater than or equal to 1. How many principal components to use. In the case of "pcr" this can be a single number or a vector. In the second case you get results for the sequence of principal components.
- `xnew` If you have new data use it, otherwise leave it NULL.

**Details**

Principal component regression is performed with linear, binary logistic or Poisson regression, depending on the nature of the response variable. The principal components of the cross product of the independent variables are obtained and classical regression is performed. This is used in the function `alfa.pcr`.
**Value**

A list including:

- **be** The beta coefficients of the predictor variables computed via the principal components if "pcr" is used.
- **model** The summary of the logistic or Poisson regression model.
- **per** The percentage of variance of the predictor variables retained by the k principal components.
- **vec** The principal components, the loadings.
- **est** The fitted or the predicted values (if xnew is not NULL).

**Author(s)**

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

**References**


**See Also**

`alfa.pcr`, `alfapcr.tune`

**Examples**

```r
library(MASS)
x <- as.matrix(fgl[2:9])
y <- as.vector(fgl[, 1])
mod1 <- pcr(y, x, 1)
mod2 <- pcr(y, x, 2)
mod <- pcr(y, x, k = 1:4) # many results at once

x <- as.matrix(iris[1:4])
y <- rbinom(150, 1, 0.6)
mod <- glm.pcr(y, x, k = 1)
```
Principal coordinate analysis using the alpha-distance

Description
Principal coordinate analysis using the \( \alpha \)-distance.

Usage
\[
\text{alfa.mds}(x, a, k = 2, \text{eig} = \text{TRUE})
\]

Arguments
- **x**: A matrix with the compositional data. Zero values are allowed.
- **a**: The value of \( \alpha \). In case of zero values in the data it has to be greater than 1.
- **k**: The maximum dimension of the space which the data are to be represented in. This can be a number between 1 and \( D - 1 \), where \( D \) denotes the number of dimensions.
- **eig**: Should eigenvalues be returned? The default value is TRUE.

Details
The function computes the \( \alpha \)-distance matrix and then plugs it into the classical multidimensional scaling function in the "cmdscale" function.

Value
A list with the results of "cmdscale" function.

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References
Principal coordinate analysis using the Jensen-Shannon divergence

See Also
esov.mds, alfa.pca.

Examples

```r
x <- as.matrix(iris[, 1:4])
x <- x/ rowSums(x)
a <- esov.mds(x)
```

Description

Principal coordinate analysis using the Jensen-Shannon divergence.

Usage

```r
esov.mds(x, k = 2, eig = TRUE)
```

Arguments

- `x`: A matrix with the compositional data. Zero values are allowed.
- `k`: The maximum dimension of the space which the data are to be represented in. This can be a number between 1 and \( D - 1 \), where \( D \) denotes the number of dimensions.
- `eig`: Should eigenvalues be returned? The default value is TRUE.

Details

The function computes the Jensen-Shannon divergence matrix and then plugs it into the classical multidimensional scaling function in the \( \text{cmdscale} \) function.

Value

A list with the results of \( \text{cmdscale} \) function.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris \texttt{mtsagris@uoc.gr}. 

Projection pursuit regression for compositional data

References


See Also

alfa.mds, alfa.pca,

Examples

x <- as.matrix(iris[, 1:4])
x <- x/ rowSums(x)
a <- esov.mds(x)

Description

Projection pursuit regression for compositional data.

Usage

comp.ppr(y, x, nterms = 3, type = "alr", xnew = NULL, yb = NULL)

Arguments

y A matrix with the compositional data.

x A matrix with the continuous predictor variables or a data frame including categorical predictor variables.

nterms The number of terms to include in the final model.

type Either "alr" or "ilr" corresponding to the additive or the isometric log-ratio transformation respectively.

xnew If you have new data use it, otherwise leave it NULL.

yb If you have already transformed the data using a log-ratio transformation put it here. Otherwise leave it NULL.

Details

This is the standard projection pursuit. See the built-in function "ppr" for more details.
Projection pursuit regression with compositional predictor variables

Value

A list includign:

- **runtime**: The runtime of the regression.
- **mod**: The produced model as returned by the function "ppr".
- **est**: The fitted values of xnew if xnew is not NULL.

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

comp.ppr.tune, aknn.reg, akern.reg, comp.reg, kl.compreg, alfa.reg

Examples

```r
y <- as.matrix(iris[, 1:3])
y <- y/ rowSums(y)
x <- iris[, 4]
mod <- comp.ppr(y, x)
```

Description

Projection pursuit regression with compositional predictor variables.

Usage

```r
pprcomp(y, x, nterms = 3, type = "log", xnew = NULL)
```

Arguments

- **y**: A numerical vector with the continuous variable.
- **x**: A matrix with the compositional data. No zero values are allowed.
- **nterms**: The number of terms to include in the final model.
- **type**: Either "alr" or "log" corresponding to the additive log-ratio transformation or the simple logarithm applied to the compositional data.
- **xnew**: If you have new data use it, otherwise leave it NULL.
Details

This is the standard projection pursuit. See the built-in function "ppr" for more details. When the data are transformed with the additive log-ratio transformation this is close in spirit to the log-contrast regression.

Value

A list including:

- runtime: The runtime of the regression.
- mod: The produced model as returned by the function "ppr".
- est: The fitted values of xnew if xnew is not NULL.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

pprcomp.tune, ice.pprcomp, alfa.pcr, lc.reg, comp.ppr

Examples

```r
x <- as.matrix( iris[, 2:4] )
x <- x/ rowSums(x)
y <- iris[, 1]
pprcomp(y, x)
```

Description

Proportionality correlation coefficient matrix.

Usage

pcc(x)
Arguments

x  A numerical matrix with the compositional data. Zeros are not allowed as the logarithm is applied.

Details

The function returns the proportionality correlation coefficient matrix. See Lovell et al. (2015) for more information.

Value

A matrix with the alr transformed data (if alr is used) or with the compositional data (if the alrinv is used).

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

acor, alr

Examples

```R
x <- Compositional::rdiri(100, runif(4) )
a <- Compositional::pcc(x)
```

Description

Quasi binomial regression for proportions.

Usage

```R
propreg(y, x, varb = "quasi", tol = 1e-07, maxiters = 100)
propregs(y, x, varb = "quasi", tol = 1e-07, logged = FALSE, maxiters = 100)
```
Quasi binomial regression for proportions

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>A numerical vector proportions. 0s and 1s are allowed.</td>
</tr>
<tr>
<td>x</td>
<td>For the &quot;propreg&quot; a matrix with data, the predictor variables. This can be a matrix or a data frame. For the &quot;propregs&quot; this must be a numerical matrix, where each columns denotes a variable.</td>
</tr>
<tr>
<td>tol</td>
<td>The tolerance value to terminate the Newton-Raphson algorithm. This is set to 10^{-9} by default.</td>
</tr>
<tr>
<td>varb</td>
<td>The type of estimate to be used in order to estimate the covariance matrix of the regression coefficients. There are two options, either &quot;quasi&quot; (default value) or &quot;glm&quot;. See the references for more information.</td>
</tr>
<tr>
<td>logged</td>
<td>Should the p-values be returned (FALSE) or their logarithm (TRUE)?</td>
</tr>
<tr>
<td>maxiters</td>
<td>The maximum number of iterations before the Newton-Raphson is terminated automatically.</td>
</tr>
</tbody>
</table>

Details

We are using the Newton-Raphson, but unlike R’s built-in function "glm" we do no checks and no extra calculations, or whatever. Simply the model. The "proregs" is to be used for very many univariate regressions. The "x" is a matrix in this case and the significance of each variable (column of the matrix) is tested. The function accepts binary responses as well (0 or 1).

Value

For the "proreg" function a list including:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iters</td>
<td>The number of iterations required by the Newton-Raphson.</td>
</tr>
<tr>
<td>varb</td>
<td>The covariance matrix of the regression coefficients.</td>
</tr>
<tr>
<td>phi</td>
<td>The phi parameter is returned if the input argument &quot;varb&quot; was set to &quot;glm&quot;, otherwiser this is NULL.</td>
</tr>
<tr>
<td>info</td>
<td>A table similar to the one produced by &quot;glm&quot; with the estimated regression coefficients, their standard error, Wald test statistic and p-values.</td>
</tr>
</tbody>
</table>

For the "proregs" a two-column matrix with the test statistics (Wald statistic) and the associated p-values (or their logarithm).

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


Random values generation from some univariate distributions defined on the (0,1) interval

See Also

anovapropreg univglms, score.glms, logistic_only

Examples

```r
y <- rbeta(100, 1, 4)
x <- matrix(rnorm(100 * 3), ncol = 3)
a <- propreg(y, x)
y <- rbeta(100, 1, 4)
x <- matrix(rnorm(400 * 100), ncol = 400)
b <- propreg(y, x)
mean(b[, 2] < 0.05)
```

Description

Random values generation from some univariate distributions defined on the (0,1) interval.

Usage

```r
rbeta1(n, a)
runitweibull(n, a, b)
rlogitnorm(n, m, s, fast = FALSE)
```

Arguments

- `n` The sample size, a numerical value.
- `a` The shape parameter of the beta distribution. In the case of the unit Weibull, this is the shape parameter.
- `b` This is the scale parameter for the unit Weibull distribution.
- `m` The mean of the univariate normal in $R$.
- `s` The standard deviation of the univariate normal in $R$.
- `fast` If you want a faster generation set this equal to TRUE. This will use the Rnorm() function from the Rfast package. However, the speed is only observable if you want to simulate at least 500 (this number may vary among computers) observations. The larger the sample size the higher the speed-up.

Details

The function generates random values from the Be(a, 1), the unit Weibull or the univariate logistic normal distribution.
Regression with compositional data using the alpha-transformation

Value
A vector with the simulated data.

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

See Also
beta.est, colbeta.est, rdiri

Examples
x <- rbeta1(100, 3)

Regression with compositional data using the alpha-transformation

Regression with compositional data using the \( \alpha \)-transformation

Description
Regression with compositional data using the \( \alpha \)-transformation.

Usage
alfa.reg(y, x, a, xnew = NULL, yb = NULL, seb = FALSE)

Arguments
y A matrix with the compositional data.
x A matrix with the continuous predictor variables or a data frame including categorical predictor variables.
a The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If \( \alpha = 0 \) the isometric log-ratio transformation is applied and the solution exists in a closed form, since it the classical mutivariate regression.
xnew If you have new data use it, otherwise leave it NULL.
yb If you have already transformed the data using the \( \alpha \)-transformation with the same \( \alpha \) as given in the argument "a", put it here. Otherwise leave it NULL. This is intended to be used in the function alfreg.tune in order to speed up the process. The time difference in that function is small for small samples. But, if you have a few thousands and or a few more components, there will be bigger differences.
seb Do you want the standard error of the coefficients to be returned? In the alfreg.tune function this extra computation that is avoided can save some time.
Details

The $\alpha$-transformation is applied to the compositional data first and then multivariate regression is applied. This involves numerical optimisation.

Value

A list including:

- `runtime` The time required by the regression.
- `be` The beta coefficients.
- `seb` The standard error of the beta coefficients.
- `est` The fitted values for `xnew` if `xnew` is not NULL.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References


See Also

`alfareg.tune, diri.reg, js.compreg, kl.compreg, ols.compreg, comp.reg`

Examples

```r
library(MASS)
x <- as.vector(fgl[1:40, 1])
y <- as.matrix(fgl[1:40, 2:9])
y <- y / rowSums(y)
mod <- alfa.reg(y, x, 0.2)
```
Regularised and flexible discriminant analysis for compositional data using the \(\alpha\)-transformation

Description

Regularised and flexible discriminant analysis for compositional data using the \(\alpha\)-transformation.

Usage

\begin{verbatim}
alfa.rda(xnew, x, ina, a, gam = 1, del = 0)
alfa.fda(xnew, x, ina, a)
\end{verbatim}

Arguments

- **xnew**: A matrix with the new compositional data whose group is to be predicted. Zeros are allowed, but you must be careful to choose strictly positive values of \(\alpha\).
- **x**: A matrix with the available compositional data. Zeros are allowed, but you must be careful to choose strictly positive values of \(\alpha\).
- **ina**: A group indicator variable for the available data.
- **a**: The value of \(\alpha\) for the \(\alpha\)-transformation.
- **gam**: This is a number between 0 and 1. It is the weight of the pooled covariance and the diagonal matrix.
- **del**: This is a number between 0 and 1. It is the weight of the LDA and QDA.

Details

For the alfa.rda, the covariance matrix of each group is calculated and then the pooled covariance matrix. The spherical covariance matrix consists of the average of the pooled variances in its diagonal and zeros in the off-diagonal elements. \(\text{gam}\) is the weight of the pooled covariance matrix and \(1-\text{gam}\) is the weight of the spherical covariance matrix, \(\text{Sa} = \text{gam} \times \text{Sp} + (1-\text{gam}) \times \text{sp}\). Then it is a compromise between LDA and QDA. \(\text{del}\) is the weight of \(\text{Sa}\) and \(1-\text{del}\) the weight of each group covariance group. This function is a wrapper for alfa.rda.

For the alfa.fda a flexible discriminant analysis is performed. See the R package \texttt{fda} for more details.

Value

For the alfa.rda a list including:

- **prob**: The estimated probabilities of the new data of belonging to each group.
- **scores**: The estimated scores of the new data of each group.
- **est**: The estimated group membership of the new data.

For the alfa.fda a list including:
Regularised discriminant analysis for Euclidean data

`mod` A `fda` object as returned by the command `fda` of the R package `mda`.

`est` The estimated group membership of the new data.

**Author(s)**

Michail Tsagris.

R implementation and documentation: Michail Tsagris (<mtsagris@uoc.gr>) and Giorgos Athineou (<gioathineou@gmail.com>.

**References**


**See Also**

`rda, alfa, alfarda.tune, alfa.knn, alfa.nb, comp.nb, mix.compnorm`

**Examples**

```r
x <- as.matrix(iris[, 1:4])
x <- x / rowSums(x)
ina <- iris[, 5]
mod <- alfa.rda(x, x, ina, 0)
table(ina, mod$est)
mod2 <- alfa.fda(x, x, ina, 0)
table(ina, mod2$est)
```

---

Regularised discriminant analysis for Euclidean data

Regularised discriminant analysis for Euclidean data

**Description**

Regularised discriminant analysis for Euclidean data.

**Usage**

`rda(xnew, x, ina, gam = 1, del = 0)`
Regularised discriminant analysis for Euclidean data

Arguments

xnew
A matrix with the new data whose group is to be predicted. They have to be continuous.

x
A matrix with the available data. They have to be continuous.

ina
A group indicator variable for the available data.

gam
This is a number between 0 and 1. It is the weight of the pooled covariance and the diagonal matrix.

del
This is a number between 0 and 1. It is the weight of the LDA and QDA.

Details

The covariance matrix of each group is calculated and then the pooled covariance matrix. The spherical covariance matrix consists of the average of the pooled variances in its diagonal and zeros in the off-diagonal elements. gam is the weight of the pooled covariance matrix and 1-gam is the weight of the spherical covariance matrix, $S_a = \text{gam} \times Sp + (1-\text{gam}) \times sp$. Then it is a compromise between LDA and QDA. del is the weight of $S_a$ and 1-del the weight of each group covariance group. This function is a wrapper for alfa.rda.

Value

A list including:

prob
The estimated probabilities of the new data of belonging to each group.

scores
The estimated socres of the new data of each group.

est
The estimated group membership of the new data.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References


See Also

rda.tune, alfa
Examples

```r
x <- as.matrix(iris[, 1:4])
ina <- iris[, 5]
mod <- rda(x, x, ina)
table(ina, mod$est)
```

Ridge regression

Description

Ridge regression.

Usage

```r
ridge.reg(y, x, lambda, B = 1, xnew = NULL)
```

Arguments

- `y`: A real valued vector. If it contains percentages, the logit transformation is applied.
- `x`: A matrix with the predictor variable(s), they have to be continuous.
- `lambda`: The value of the regularisation parameter $\lambda$.
- `B`: If `B = 1` (default value) no bootstrpa is performed. Otherwise bootstrap standard errors are returned.
- `xnew`: If you have new data whose response value you want to predict put it here, otherwise leave it as is.

Details

This is used in the function `alfa.ridge`. There is also a built-in function available from the MASS library, called `lm.ridge`.

Value

A list including:

- `beta`: The beta coefficients.
- `seb`: The standard error of the coefficients. If `B > 1` the bootstrap standard errors will be returned.
- `est`: The fitted or the predicted values (if `xnew` is not NULL).

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.
Ridge regression plot

References


See Also

ridge.tune, alfa.ridge, ridge.plot

Examples

y <- as.vector(iris[, 1])
x <- as.matrix(iris[, 2:4])
mod1 <- ridge.reg(y, x, lambda = 0.1)
mod2 <- ridge.reg(y, x, lambda = 0)

Ridge regression plot

Description

A plot of the regularised regression coefficients is shown.

Usage

ridge.plot(y, x, lambda = seq(0, 5, by = 0.1) )

Arguments

y
  A numeric vector containing the values of the target variable. If the values are proportions or percentages, i.e. strictly within 0 and 1 they are mapped into R using the logit transformation. In any case, they must be continuous only.

x
  A numeric matrix containing the continuous variables. Rows are samples and columns are features.

lambda
  A grid of values of the regularisation parameter $\lambda$.

Details

For every value of $\lambda$ the coefficients are obtained. They are plotted versus the $\lambda$ values.

Value

A plot with the values of the coefficients as a function of $\lambda$. 
Ridge regression with compositional data in the covariates side using the alpha-transformation

Author(s)

Michail Tsagris.

R implementation and documentation: Giorgos Athineou <gioathineou@gmail.com> and Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

ridge.reg, ridge.tune, alfa.ridge, alfaridge.plot

Examples

```r
y <- as.vector(iris[, 1])
x <- as.matrix(iris[, 2:4])
ridge.plot(y, x, lambda = seq(0, 1, by = 0.1))
```

Description

Ridge regression with compositional data in the covariates side using the α-transformation.

Usage

```r
alfa.ridge(y, x, a, lambda, B = 1, xnew = NULL)
```

Arguments

- `y`: A numerical vector containing the response variable values. If they are percentages, they are mapped onto $R$ using the logit transformation.
- `x`: A matrix with the predictor variables, the compositional data. Zero values are allowed, but you must be careful to choose strictly positive values of $\alpha$.
- `a`: The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If $\alpha = 0$ the isometric log-ratio transformation is applied.
- `lambda`: The value of the regularisation parameter, $\lambda$.
- `B`: If $B > 1$ bootstrap estimation of the standard errors is implemented.
- `xnew`: A matrix containing the new compositional data whose response is to be predicted. If you have no new data, leave this NULL as is by default.
Ridge regression with the alpha-transformation plot

Details

The $\alpha$-transformation is applied to the compositional data first and then ridge components regression is performed.

Value

The output of the `ridge.reg`.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References


See Also

`ridge.reg, alfaridge.tune, alfaridge.plot`

Examples

```r
library(MASS)
y <- as.vector(fgl[, 1])
x <- as.matrix(fgl[, 2:9])
x <- x/ rowSums(x)
mod1 <- alfa.ridge(y, x, a = 0.5, lambda = 0.1, B = 1, xnew = NULL)
mod2 <- alfa.ridge(y, x, a = 0.5, lambda = 1, B = 1, xnew = NULL)
```

Ridge regression with the alpha-transformation plot

A plot of the regularised regression coefficients is shown.

Usage

`alfaridge.plot(y, x, a, lambda = seq(0, 5, by = 0.1) )`
**Ridge regression with the alpha-transformation plot**

**Arguments**

- \( y \) A numeric vector containing the values of the target variable. If the values are proportions or percentages, i.e. strictly within 0 and 1 they are mapped into R using the logit transformation. In any case, they must be continuous only.
- \( x \) A numeric matrix containing the continuous variables.
- \( a \) The value of the \( \alpha \)-transformation. It has to be between -1 and 1. If there are zero values in the data, you must use a strictly positive value.
- \( \lambda \) A grid of values of the regularisation parameter \( \lambda \).

**Details**

For every value of \( \lambda \) the coefficients are obtained. They are plotted versus the \( \lambda \) values.

**Value**

A plot with the values of the coefficients as a function of \( \lambda \).

**Author(s)**

Michail Tsagris.

R implementation and documentation: Giorgos Athineou <gioathineou@gmail.com> and Michail Tsagris <mtsagris@uoc.gr>.

**References**


**See Also**

- `ridge.plot`, `alfa.ridge`

**Examples**

```r
library(MASS)
y <- as.vector(fgl[, 1])
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
alfaridge.plot(y, x, a = 0.5, lambda = seq(0, 5, by = 0.1) )
```
Simulation of compositional data from Gaussian mixture models

Description

Simulation of compositional data from Gaussian mixture models.

Usage

rmixcomp(n, prob, mu, sigma, type = "alr")

Arguments

n
The sample size.

prob
A vector with mixing probabilities. Its length is equal to the number of clusters.

mu
A matrix where each row corresponds to the mean vector of each cluster.

sigma
An array consisting of the covariance matrix of each cluster.

type
Should the additive ("type=alr") or the isometric (type="ilr") log-ratio be used? The default value is for the additive log-ratio transformation.

Details

A sample from a multivariate Gaussian mixture model is generated.

Value

A list including:

id
A numeric variable indicating the cluster of simulated vector.

x
A matrix containing the simulated compositional data. The number of dimensions will be + 1.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

mix.compnorm, bic.mixcompnorm
Examples

```r
p <- c(1/3, 1/3, 1/3)
mu <- matrix(nrow = 3, ncol = 4)
s <- array(dim = c(4, 4, 3))
x <- as.matrix(iris[, 1:4])
ina <- as.numeric(iris[, 5])
mu <- rowsum(x, ina) / 50
s[, , 1] <- cov(x[ina == 1, ])
s[, , 2] <- cov(x[ina == 2, ])
s[, , 3] <- cov(x[ina == 3, ])
y <- rmixcomp(100, p, mu, s, type = "alr")
```

Description

Simulation of compositional data from mixtures of Dirichlet distributions.

Usage

```r
rmixdiri(n, a, prob)
```

Arguments

- **n**: The sample size.
- **a**: A matrix where each row contains the parameters of each Dirichlet component.
- **prob**: A vector with the mixing probabilities.

Details

A sample from a Dirichlet mixture model is generated.

Value

A list including:

- **id**: A numeric variable indicating the cluster of simulated vector.
- **x**: A matrix containing the simulated compositional data.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.
Simulation of compositional data from the Flexible Dirichlet distribution

References


See Also

rmixcomp, mixdiri.contour,

Examples

```r
a <- matrix( c(12, 30, 45, 32, 50, 16), byrow = TRUE, ncol = 3)
prob <- c(0.5, 0.5)
x <- rmixdiri(100, a, prob)
```

Description

Simulation of compositional data from the Flexible Dirichlet distribution.

Usage

```r
rfd(n, alpha, prob, tau)
```

Arguments

- `n` The sample size.
- `alpha` A vector of the non-negative $\alpha$ parameters.
- `prob` A vector of the clusters’ probabilities that must sum to one.
- `tau` The positive scalar $\tau$ parameter.

Details

For more information see the references.

Value

A matrix with compositional data.

Author(s)

Michail Tsagris ported from the R package FlexDir. <mtsagris@uoc.gr>.
Simulation of compositional data from the folded normal distribution

References


See Also

fd.est, dfd

Examples

alpha <- c(12, 11, 10)
prob <- c(0.25, 0.25, 0.5)
x <- rfd(100, alpha, prob, 7)

Description

Simulation of compositional data from the folded model normal distribution.

Usage

rfolded(n, mu, su, a)

Arguments

n The sample size.
mu The mean vector.
su The covariance matrix.
a The value of $\alpha$.

Details

A sample from the folded model is generated.

Value

A matrix with compositional data.
Spatial median regression

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

See Also
alfa, alpha.mle, a.est

Examples
s <- c(0.1490676523, -0.4580818209, 0.0020395316, -0.0047446076, -0.4580818209, 1.522759250, 0.002596411, 0.002393251, 0.002393316, 0.002596411, 0.0365384838, -0.0471448849, -0.0471448849, -0.0471448849, 0.0471448849, 0.0471448849)
s <- matrix(s, ncol = 4)
m <- c(1.715, 0.914, 0.115, 0.167)
x <- rfolded(100, m, s, 0.5)
a.est(x)

Spatial median regression

Description
Spatial median regression with Euclidean data.

Usage
spatmed.reg(y, x, xnew = NULL, tol = 1e-07, ses = FALSE)

Arguments

y A matrix with the compositional data. Zero values are not allowed.
x The predictor variable(s), they have to be continuous.
xnew If you have new data use it, otherwise leave it NULL.
tol The threshold upon which to stop the iterations of the Newton-Rapshon algorithm.

ses If you want to extract the standard errors of the parameters, set this to TRUE. Be careful though as this can slow down the algorithm dramatically. In a run example with 10,000 observations and 10 variables for y and 30 for x, when ses = FALSE the algorithm can take 0.20 seconds, but when ses = TRUE it can go up to 140 seconds.
Details

The objective function is the minimization of the sum of the absolute residuals. It is the multivariate generalization of the median regression. This function is used by `comp.reg`.

Value

A list including:

- `iter`: The number of iterations that were required.
- `runtime`: The time required by the regression.
- `be`: The beta coefficients.
- `seb`: The standard error of the beta coefficients is returned if `ses=TRUE` and NULL otherwise.
- `est`: The fitted of `xnew` if `xnew` is not NULL.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

`multivreg`, `comp.reg`, `alfa.reg`, `js.compreg`, `diri.reg`

Examples

```r
library(MASS)
x <- as.matrix(iris[, 3:4])
y <- as.matrix(iris[, 1:2])
mod1 <- spatmed.reg(y, x)
mod2 <- multivreg(y, x, plot = FALSE)
```

**Description**

Ternary diagram.

Usage

ternary(x, dg = FALSE, hg = FALSE, means = TRUE, pca = FALSE, colour = NULL)
Arguments

- **x**: A matrix with the compositional data.
- **dg**: Do you want diagonal grid lines to appear? If yes, set this TRUE.
- **hg**: Do you want horizontal grid lines to appear? If yes, set this TRUE.
- **means**: A boolean variable. Should the closed geometric mean and the arithmetic mean appear (TRUE) or not (FALSE)?
- **pca**: Should the first PCA calculated by Aitchison (1983) described appear? If yes, then this should be TRUE, or FALSE otherwise.
- **colour**: If you want the points to appear in different colour put a vector with the colour numbers or colours.

Details

There are two ways to create a ternary graph. We used here that one where each edge is equal to 1 and it is what Aitchison (1986) uses. For every given point, the sum of the distances from the edges is equal to 1. Horizontal and or diagonal grid lines can appear, so as the closed geometric and the simple arithmetic mean. The first PCA is calculated using the centred log-ratio transformation as Aitchison (1983, 1986) suggested. If the data contain zero values, the first PCA will not be plotted. Zeros in the data appear with green circles in the triangle and you will also see NaN in the closed geometric mean.

Value

The ternary plot and a 2-row matrix with the means. The closed geometric and the simple arithmetic mean vector and or the first principal component will appear as well if the user has asked for them. Additionally, horizontal or diagonal grid lines can appear as well.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References


See Also

- `comp.den`, `alfa`, `diri.contour`, `comp.kerncontour`

Examples

```r
x <- as.matrix(iris[, 1:3])
x <- x / rowSums(x)
ternary(x, means = TRUE, pca = TRUE)
```
**Ternary diagram of regression models**

**Description**

Ternary diagram of regression models.

**Usage**

```r
ternary.reg(y, est, id, labs)
```

**Arguments**

- `y`: A matrix with the compositional data.
- `est`: A matrix with all fitted compositional data for all regression models, one under the other.
- `id`: A vector indicating the regression model of each fitted compositional data set.
- `labs`: The names of the regression models to appear in the legend.

**Details**

The points first appear on the ternary plot. Then, the fitted compositional data appear with different lines for each regression model.

**Value**

The ternary plot and lines for the fitted values of each regression model.

**Author(s)**

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

**References**


**See Also**

`ternary`, `diri.contour`, `comp.kerncontour`
The additive log-ratio transformation and its inverse

Examples

x <- cbind(1, rnorm(50) )
a <- exp( x %*% matrix( rnorm(6,0, 0.4), ncol = 3) )
y <- matrix(NA, 50, 3)
for (i in 1:50) y[i, ] <- rdiri(1, a[i, ])
est <- comp.reg(y, x[, -1], xnew = x[, -1])$est
ternary.reg(y, est, id = rep(1, 50), labs = “ALR regression”)

Description

The additive log-ratio transformation and its inverse.

Usage

alr(x)
alrinv(y)

Arguments

x A numerical matrix with the compositional data.
y A numerical matrix with data to be closed into the simplex.

Details

The additive log-ratio transformation with the first component being the common divisor is applied. The inverse of this transformation is also available. This means that no zeros are allowed.

Value

A matrix with the alr transformed data (if alr is used) or with the compositional data (if the alrinv is used).

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

bc, pivot, fp, green, alfa, alfainv, alfa.profile, alfa.tune
Examples

library(MASS)
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
y <- alr(x)
x1 <- alrinv(y)

Description

This is the Euclidean (or Manhattan) distance after the $\alpha$-transformation has been applied.

Usage

alfadist(x, a, type = "euclidean", square = FALSE)
alfadista(xnew, x, a, type = "euclidean", square = FALSE)

Arguments

xnew A matrix or a vector with new compositional data.
x A matrix with the compositional data.
a The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If $\alpha = 0$, the isometric log-ratio transformation is applied.
type Which type distance do you want to calculate after the $\alpha$-transformation, "euclidean", or "manhattan".
square In the case of the Euclidean distance, you can choose to return the squared distance by setting this TRUE.

Details

The $\alpha$-transformation is applied to the compositional data first and then the Euclidean or the Manhattan distance is calculated.

Value

For "alfadist" a matrix including the pairwise distances of all observations or the distances between xnew and x. For "alfadista" a matrix including the pairwise distances of all observations or the distances between xnew and x.

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.
The alpha-IT transformation

References


See Also

alfa, alfainv, alfa.reg, esov

Examples

library(MASS)
x <- as.matrix(fgl[1:20, 2:9])
x <- x / rowSums(x)
alfadist(x, 0.1)
alfadist(x, 1)

The alpha-IT transformation

The $\alpha$-IT transformation

Description

The $\alpha$-IT transformation.

Usage

ait(x, a, h = TRUE)

Arguments

x A matrix with the compositional data.
a The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If $\alpha = 0$ the isometric log-ratio transformation is applied.
h A boolean variable. If is TRUE (default value) the multiplication with the Helmert sub-matrix will take place. When $\alpha = 0$ and h = FALSE, the result is the centred log-ratio transformation (Aitchison, 1986). In general, when h = FALSE the resulting transformation maps the data onto a singular space. The sum of the vectors is equal to 0. Hence, from the simplex constraint the data go to another constraint.

Details

The $\alpha$-IT transformation is applied to the compositional data.
Value

A matrix with the $\alpha$-IT transformed data.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

aitdist, ait.knn, alfa, green, alr

Examples

```r
library(MASS)
x <- as.matrix(fgl[, 2:9])x <- x / rowSums(x)
y1 <- ait(x, 0.2)
y2 <- ait(x, 1)
rbind( colMeans(y1), colMeans(y2) )
```

Description

This is the Euclidean (or Manhattan) distance after the $\alpha$-IT-transformation has been applied.

Usage

```r
aitdist(x, a, type = "euclidean", square = FALSE)
aitdista(xnew, x, a, type = "euclidean", square = FALSE)
```

Arguments

- `xnew`: A matrix or a vector with new compositional data.
- `x`: A matrix with the compositional data.
- `a`: The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If $\alpha = 0$, the isometric log-ratio transformation is applied.
The alpha-k-NN regression for compositional response data

<table>
<thead>
<tr>
<th>type</th>
<th>Which type distance do you want to calculate after the $\alpha$-transformation, &quot;euclidean&quot;, or &quot;manhattan&quot;.</th>
</tr>
</thead>
<tbody>
<tr>
<td>square</td>
<td>In the case of the Euclidean distance, you can choose to return the squared distance by setting this TRUE.</td>
</tr>
</tbody>
</table>

Details

The $\alpha$-IT-transformation is applied to the compositional data first and then the Euclidean or the Manhattan distance is calculated.

Value

For "alfadist" a matrix including the pairwise distances of all observations or the distances between xnew and x. For "alfadista" a matrix including the pairwise distances of all observations or the distances between xnew and x.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

ait, alfadist, alfa

Examples

```r
library(MASS)
x <- as.matrix(fgl[1:20, 2:9])
x <- x / rowSums(x)
aitdist(x, 0.1)
aitdist(x, 1)
```

The alpha-k-NN regression for compositional response data

The $\alpha$-k-NN regression for compositional response data

Description

The $\alpha$-k-NN regression for compositional response data.
Usage

aknn.reg(xnew, y, x, a = seq(0.1, 1, by = 0.1), k = 2:10, apostasi = "euclidean", rann = FALSE)

Arguments

xnew A matrix with the new predictor variables whose compositions are to be predicted.

y A matrix with the compositional response data. Zeros are allowed.

x A matrix with the available predictor variables.

a The value(s) of \(\alpha\). Either a single value or a vector of values. As zero values in the compositional data are allowed, you must be careful to choose strictly positive values of \(\alpha\). However, if negative values are passed, the positive ones are used only.

k The number of nearest neighbours to consider. It can be a single number or a vector.

apostasi The type of distance to use, either "euclidean" or "manhattan".

rann If you have large scale datasets and want a faster k-NN search, you can use kdtrees implemented in the R package "RANN". In this case you must set this argument equal to TRUE. Note however, that in this case, the only available distance is by default "euclidean".

Details

The \(\alpha\)-k-NN regression for compositional response variables is applied.

Value

A list with the estimated compositional response data for each value of \(\alpha\) and \(k\).

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

aknnreg.tune, akern.reg, alfa.reg, comp.ppr, comp.reg, kl.compreg
The alpha-k-NN regression with compositional predictor variables

Examples

```r
y <- as.matrix(iris[,1:3])
y <- y / rowSums(y)
x <- iris[,4]
mod <- aknn.reg(x, y, x, a = c(0.4, 0.5), k = 2:3, apostasi = "euclidean")
```

Description

The $\alpha$-k-NN regression with compositional predictor variables.

Usage

```r
alfa.knn.reg(xnew, y, x, a = 1, k = 2:10, apostasi = "euclidean", method = "average")
```

Arguments

- `xnew`: A matrix with the new compositional predictor variables whose response is to be predicted. Zeros are allowed.
- `y`: The response variable, a numerical vector.
- `x`: A matrix with the available compositional predictor variables. Zeros are allowed.
- `a`: A single value of $\alpha$. As zero values in the compositional data are allowed, you must be careful to choose strictly positive values of $\alpha$. If negative values are passed, the positive ones are used only. If the data are already alpha-transformed, you can make this NULL.
- `k`: The number of nearest neighbours to consider. It can be a single number or a vector.
- `apostasi`: The type of distance to use, either "euclidean" or "manhattan".
- `method`: If you want to take the average of the responses of the $k$ closest observations, type "average". For the median, type "median" and for the harmonic mean, type "harmonic".

Details

The $\alpha$-k-NN regression with compositional predictor variables is applied.

Value

A matrix with the estimated response data for each value of $k$. 

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

aknn.reg, alfa.knn, alfa.pcr, alfa.ridge

Examples

library(MASS)
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
y <- fgl[, 1]
mod <- alfa.knn.reg(x, y, x, a = 0.5, k = 2:4)

The alpha-kernel regression with compositional response data

The \( \alpha \)-kernel regression with compositional response data.

Description

The \( \alpha \)-kernel regression with compositional response data.

Usage

akern.reg( xnew, y, x, a = seq(0.1, 1, by = 0.1),
h = seq(0.1, 1, length = 10), type = "gauss" )

Arguments

xnew A matrix with the new predictor variables whose compositions are to be predicted.
y A matrix with the compositional response data. Zeros are allowed.
x A matrix with the available predictor variables.
a The value(s) of \( \alpha \). Either a single value or a vector of values. As zero values in the compositional data are allowed, you must be careful to choose strictly positive values of \( \alpha \). However, if negative values are passed, the positive ones are used only.
h The bandwidth value(s) to consider.
type The type of kernel to use, "gauss" or "laplace".
The alpha-transformation

Details

The $\alpha$-kernel regression for compositional response variables is applied.

Value

A list with the estimated compositional response data for each value of $\alpha$ and $h$.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

akernreg.tune, aknn.reg, aknnreg.tune, alfa.reg, comp.ppr, comp.reg, kl.compreg

Examples

```r
y <- as.matrix( iris[, 1:3] )
y <- y / rowSums(y)
x <- iris[, 4]
mod <- akern.reg( x, y, x, a = c(0.4, 0.5), h = c(0.1, 0.2) )
```

The alpha-transformation

**The $\alpha$-transformation**

Description

The $\alpha$-transformation.

Usage

```r
alfa(x, a, h = TRUE)
alef(x, a)
```
The alpha-transformation

Arguments

- `x`: A matrix with the compositional data.
- `a`: The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If $\alpha = 0$ the isometric log-ratio transformation is applied.
- `h`: A boolean variable. If is TRUE (default value) the multiplication with the Helmert sub-matrix will take place. When $\alpha = 0$ and `h = FALSE`, the result is the centred log-ratio transformation (Aitchison, 1986). In general, when `h = FALSE` the resulting transformation maps the data onto a singular space. The sum of the vectors is equal to 0. Hence, from the simplex constraint the data go to another constraint.

Details

The $\alpha$-transformation is applied to the compositional data. The command "alef" is the same as "alfa(x, a, h = FALSE)"; but reurns a different element as well and is necessary for the functions `a.est`, `a.mle` and `alpha.mle`.

Value

A list including:

- `sa`: The logarithm of the Jacobian determinant of the $\alpha$-transformation. This is used in the "profile" function to speed up the computations.
- `sk`: If the "alef" was called, this will return the sum of the $\alpha$-power transformed data, prior to being normalised to sum to 1. If $\alpha = 0$, this will not be returned.
- `aff`: The $\alpha$-transformed data.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References


See Also

`alfainv`, `pivot`, `alfa.profile`, `alfa.tune a.est`, `alpha.mle`, `alr`, `bc`, `fp`, `green`
The Box-Cox transformation applied to ratios of components

Examples

```r
library(MASS)
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
y1 <- alfa(x, 0.2)$aff
y2 <- alfa(x, 1)$aff
rbind( colMeans(y1), colMeans(y2) )
y3 <- alfa(x, 0.2)$aff
dim(y1) ; dim(y3)
rowSums(y1)
rowSums(y3)
```

Description

The Box-Cox transformation applied to ratios of components.

Usage

```r
bc(x, lambda)
```

Arguments

- `x` A matrix with the compositional data. The first component must be zero values free.
- `lambda` The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If \( \lambda = 0 \) the additive log-ratio transformation (alr) is applied.

Details

The Box-Cox transformation applied to ratios of components, as described in Aitchison (1986) is applied.

Value

A matrix with the transformed data.

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.
The ESOV-distance

References


See Also

alr, fp, green, alfa

Examples

library(MASS)
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
y1 <- bc(x, 0.2)
y2 <- bc(x, 0)
rowSums(y1)
rowSums(y2)

The ESOV-distance

Description

The ESOV-distance.

Usage

esov(x)
esova(xnew, x)
es(x1, x2)

Arguments

x A matrix with compositional data.
xnew A matrix or a vector with new compositional data.
x1 A vector with compositional data.
x2 A vector with compositional data.

Details

The ESOV distance is calculated.

Value

For "esov" a matrix including the pairwise distances of all observations or the distances between xnew and x. For "esova" a matrix including the pairwise distances of all observations or the distances between xnew and x. For "es" a number, the ESOV distance between x1 and x2.
The folded power transformation

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

See Also
alfadist, comp.knn, js.compreg

Examples
library(MASS)
x <- as.matrix(fgl[1:20, 2:9])
x <- x / rowSums(x)
esov(x)

The folded power transformation

Description
The folded power transformation.

Usage
fp(x, lambda)

Arguments
x: A matrix with the compositional data. Zero values are allowed.
lambda: The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If \( \lambda = 0 \) the additive log-ratio transformation (alr) is applied. If zero values are present \( \lambda \) must be strictly positive.

Details
The folded power transformation is applied to the compositional data.
The Frechet mean for compositional data

Value
A matrix with the transformed data.

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

See Also
alr, bc, green, alfa

Examples

```r
library(MASS)
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
y1 <- fp(x, 0.2)
y2 <- fp(x, 0)
rowSums(y1)
rowSums(y2)
```

Description
Mean vector or matrix with mean vectors of compositional data using the \( \alpha \)-transformation.

Usage

`frechet(x, a)`

Arguments

- `x`: A matrix with the compositional data.
- `a`: The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If \( \alpha = 0 \) the isometric log-ratio transformation is applied and the closed geometric mean is calculated. You can also provide a sequence of values of alpha and in this case a matrix of Frechet means will be returned.
The Helmert sub-matrix

Details

The power transformation is applied to the compositional data and the mean vector is calculated. Then the inverse of it is calculated and the inverse of the power transformation applied to the last vector is the Frechet mean.

Value

If \( \alpha \) is a single value, the function will return a vector with the Frechet mean for the given value of \( \alpha \). Otherwise the function will return a matrix with the Frechet means for each value of \( \alpha \).

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References


See Also

alfa, alfainv, profile

Examples

```r
library(MASS)
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
frechet(x, 0.2)
frechet(x, 1)
```

The Helmert sub-matrix

The Helmert sub-matrix

Description

The Helmert sub-matrix.

Usage

helm(n)
The k-nearest neighbours using the alpha-distance

Arguments

\( n \)  
A number grater than or equal to 2.

Details

The Helmert sub-matrix is returned. It is an orthogonal matrix without the first row.

Value

A \((n - 1) \times n\) matrix.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References


See Also

alfa, alfainv

Examples

helm(3)
helm(5)

The k-nearest neighbours using the alpha-distance

The k-nearest neighbours using the \( \alpha \)-distance

Description

The k-nearest neighbours using the \( \alpha \)-distance.

Usage

alfa(xnew, x, a, k = 10, rann = FALSE)
Arguments

- `xnew`: A matrix or a vector with new compositional data.
- `x`: A matrix with the compositional data.
- `a`: The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If $\alpha = 0$, the isometric log-ratio transformation is applied.
- `k`: The number of nearest neighbours to search for.
- `rann`: If you have large scale datasets and want a faster k-NN search, you can use kd-trees implemented in the R package "RANN". In this case you must set this argument equal to TRUE. Note however, that in this case, the only available distance is by default "euclidean".

Details

The $\alpha$-transformation is applied to the compositional data first and the indices of the k-nearest neighbours using the Euclidean distance are returned.

Value

A matrix including the indices of the nearest neighbours of each `xnew` from `x`.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

`alfa.knn`, `comp.nb`, `alfa.rda`, `alfa.nb`, `link{aknn.reg}`, `alfa`, `alfainv`

Examples

```r
library(MASS)
xnew <- as.matrix(fgl[1:20, 2:9])
xnew <- xnew / rowSums(xnew)
x <- as.matrix(fgl[-c(1:20), 2:9])
x <- x / rowSums(x)
b <- alfann(xnew, x, a = 0.1, k = 10)
```
The k-NN algorithm for compositional data

Description

The k-NN algorithm for compositional data with and without using the power transformation.

Usage

```r
comp.knn(xnew, x, ina, a = 1, k = 5, apostasi = "ESOV", mesos = TRUE)

alfa.knn(xnew, x, ina, a = 1, k = 5, mesos = TRUE,
apostasi = "euclidean", rann = FALSE)

ait.knn(xnew, x, ina, a = 1, k = 5, mesos = TRUE,
apostasi = "euclidean", rann = FALSE)
```

Arguments

- `xnew`: A matrix with the new compositional data whose group is to be predicted. Zeros are allowed, but you must be careful to choose strictly positive values of $\alpha$ or not to set `apostasi = "Ait"`.
- `x`: A matrix with the available compositional data. Zeros are allowed, but you must be careful to choose strictly positive values of $\alpha$ or not to set `apostasi = "Ait"`.
- `ina`: A group indicator variable for the available data.
- `a`: The value of $\alpha$. As zero values in the compositional data are allowed, you must be careful to choose strictly positive values of $\alpha$. You have the option to put `a = NULL`. In this case, the `xnew` and `x` are assumed to be the already $\alpha$-transformed data.
- `k`: The number of nearest neighbours to consider. It can be a single number or a vector.
- `apostasi`: The type of distance to use. For the `comp.knn` this can be one of the following: "ESOV", "taxicab", "Ait", "Hellinger", "angular" or "CS". See the references for them. For the `alfa.knn` this can be either "euclidean" or "manhattan".
- `mesos`: This is used in the non standard algorithm. If TRUE, the arithmetic mean of the distances is calculated, otherwise the harmonic mean is used (see details).
- `rann`: If you have large scale datasets and want a faster k-NN search, you can use kd-trees implemented in the R package "RANN". In this case you must set this argument equal to TRUE. Note however, that in this case, the only available distance is by default "euclidean".
The k-NN algorithm for compositional data

Details

The k-NN algorithm is applied for the compositional data. There are many metrics and possibilities to choose from. The algorithm finds the k nearest observations to a new observation and allocates it to the class which appears most times in the neighbours. It then computes the arithmetic or the harmonic mean of the distances. The new point is allocated to the class with the minimum distance.

Value

A vector with the estimated groups.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References


See Also

compknn.tune, alfa.rda, comp.nb, alfa.nb, alfa, esov, mix.compnorm

Examples

```r
x <- as.matrix( iris[, 1:4] )
x <- x/ rowSums(x)
ina <- iris[, 5]
mod <- comp.knn(x, x, ina, a = 1, k = 5)
table(ina, mod)
mod2 <- alfa.knn(x, x, ina, a = 1, k = 5)
table(ina, mod2)
```
The pivot coordinate transformation and its inverse

Description
The pivot coordinate transformation and its inverse.

Usage
pivot(x)
pivotinv(y)

Arguments
x A numerical matrix with the compositional data.
y A numerical matrix with data to be closed into the simplex.

Details
The pivot coordinate transformation and its inverse are computed. This means that no zeros are allowed.

Value
A matrix with the alr transformed data (if pivot is used) or with the compositional data (if the pivotinv is used).

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

See Also
alfa,alfainv,alr,green

Examples
library(MASS)
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
y <- pivot(x)
x1 <- alrinv(y)
Total variability

Description

Total variability.

Usage

\texttt{totvar(x, a = 0)}

Arguments

\begin{itemize}
\item \texttt{x} A numerical matrix with the compositional data.
\item \texttt{a} The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If $\alpha = 0$ the centred log-ratio transformation is used.
\end{itemize}

Details

The $\alpha$-transformation is applied and the sum of the variances of the transformed variables is calculated. This is the total variability. Aitchison (1986) used the centred log-ratio transformation, but we have extended it to cover more geometries, via the $\alpha$-transformation.

Value

The total variability of the data in a given geometry as dictated by the value of $\alpha$.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

\texttt{alfa, link(alfainv,)} \texttt{alfa.profile,alfa.tune}

Examples

\begin{verbatim}
x <- as.matrix(iris[, 1:4])
x <- x / rowSums(x)
totvar(x)
\end{verbatim}
Transformation-free linear regression for compositional responses and predictors

Description

Transformation-free linear regression for compositional responses and predictors.

Usage

tflr(y, x, xnew = NULL)

Arguments

y A matrix with the compositional response. Zero values are allowed.
x A matrix with the compositional predictors. Zero values are allowed.
xnew If you have new data use it, otherwise leave it NULL.

Details

The transformation-free linear regression for compositional responses and predictors is implemented. The function to be minimized is $-\sum_{i=1}^{n} y_i \log y_i / (X_i B)$.

Value

A list including:

runtime The time required by the regression.
loglik The log-likelihood.
be The beta coefficients.
est The fitted values of xnew if xnew is not NULL.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

cv.tflr, ols.compcomp, kl.alfapcr
Tuning of the alpha generalised correlations between two compositional datasets

Examples

```r
library(MASS)
y <- rdiri(214, runif(3, 1, 3))
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
mod <- tflr(y, x, x)
mod
```

Description

Tuning of the alpha generalised correlations between two compositional datasets.

Usage

```r
acor.tune(y, x, a, type = "dcor")
```

Arguments

- `y`: A matrix with the compositional data.
- `x`: A matrix with the compositional data.
- `a`: The range of values of the power transformation to search for the optimal one. If zero values are present it has to be greater than 0.
- `type`: the type of correlation to compute, the distance correlation ("edist"), the canonical correlation type 1 ("cancor1") or the canonical correlation type 2 ("cancor2"). See details for more information.

Details

The $\alpha$-transformation is applied to each composition and then the distance correlation or the canonical correlation is computed. If type = "cancor1" the function returns the value of $\alpha$ that maximizes the product of the eigenvalues. If type = "cancor2" the function returns the value of $\alpha$ that maximizes the the largest eigenvalue.

Value

A list including:

- `alfa`: The optimal value of $\alpha$.
- `acor`: The maximum value of the acor.
- `runtime`: The runtime of the optimization.
Tuning of the bandwidth $h$ of the kernel using the maximum likelihood cross validation

**Author(s)**
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

**See Also**
acor, alfa.profile, alfa, alfainv

**Examples**
```r
y <- rdiri(30, runif(3) )
x <- rdiri(30, runif(4) )
acor(y, x, a = 0.4)
```

---

**Description**
Tuning of the bandwidth $h$ of the kernel using the maximum likelihood cross validation.

**Usage**
```r
mkde.tune( x, low = 0.1, up = 3, s = cov(x) )
```

**Arguments**
- **x**: A matrix with Euclidean (continuous) data.
- **low**: The minimum value to search for the optimal bandwidth value.
- **up**: The maximum value to search for the optimal bandwidth value.
- **s**: A covariance matrix. By default it is equal to the covariance matrix of the data, but can change to a robust covariance matrix, MCD for example.

**Details**
Maximum likelihood cross validation is applied in order to choose the optimal value of the bandwidth parameter. No plot is produced.

**Value**
A list including:
- **hopt**: The optimal bandwidth value.
- **maximum**: The value of the pseudo-log-likelihood at that given bandwidth value.
Tuning of the divergence based regression for compositional data with compositional data in the covariates side using the alpha-transformation

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References


See Also

mkde, comp.kerncontour

Examples

library(MASS)
mkde.tune(as.matrix(iris[, 1:4]), c(0.1, 3) )

Tuning of the divergence based regression for compositional data with compositional data in the covariates side using the \( \alpha \)-transformation

Description

Tuning of the divergence based regression for compositional data with compositional data in the covariates side using the \( \alpha \)-transformation.

Usage

klalfapcr.tune(y, x, covar = NULL, nfolds = 10, maxk = 50, a = seq(-1, 1, by = 0.1), folds = NULL, graph = FALSE, tol = 1e-07, maxiters = 50, seed = NULL)

Arguments

y A numerical matrix with compositional data with or without zeros.
x A matrix with the predictor variables, the compositional data. Zero values are allowed.
covar If you have other continuous covariates put them here.
nfolds The number of folds for the K-fold cross validation, set to 10 by default.
maxk The maximum number of principal components to check.
a The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If \( \alpha = 0 \) the isometric log-ratio transformation is applied.
Tuning of the divergence based regression for compositional data with compositional data in the covariates side using the alpha-transformation

- **folds**: If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.
- **graph**: If graph is TRUE (default value) a plot will appear.
- **tol**: The tolerance value to terminate the Newton-Raphson procedure.
- **maxiters**: The maximum number of Newton-Raphson iterations.
- **seed**: You can specify your own seed number here or leave it NULL.

**Details**

The M-fold cross validation is performed in order to select the optimal values for $\alpha$ and $k$, the number of principal components. The $\alpha$-transformation is applied to the compositional data first, the first $k$ principal component scores are calculated and used as predictor variables for the Kullback-Leibler divergence based regression model. This procedure is performed $M$ times during the M-fold cross validation.

**Value**

A list including:

- **mspe**: A list with the KL divergence for each value of $\alpha$ and $k$ in every fold.
- **performance**: A matrix with the KL divergence for each value of $\alpha$ averaged over all folds. If graph is set to TRUE this matrix is plotted.
- **best.perf**: The minimum KL divergence.
- **params**: The values of $\alpha$ and $k$ corresponding to the minimum KL divergence.

**Author(s)**

Initial code by Abdulaziz Alenazi. Modifications by Michail Tsagris.

R implementation and documentation: Abdulaziz Alenazi <a.alenazi@nbu.edu.sa> and Michail Tsagris <mtsagris@uoc.gr>.

**References**


**See Also**

kl.alfapcr, cv.tflr, pcr, glm.pcr, alfapcr.tune
Examples

library(MASS)
y <- rdiri(214, runif(4, 1, 3))
x <- as.matrix(fgl[,2:9])
x <- x / rowSums(x)
mod <- klalfapcr.tune(y = y, x = x, a = c(0.7, 0.8))
mod

Tuning of the k-NN algorithm for compositional data

Description

Tuning of the k-NN algorithm for compositional data with and without using the power or the \( \alpha \)-transformation. In addition, estimation of the rate of correct classification via K-fold cross-validation.

Usage

compknn.tune(x, ina, nfolds = 10, k = 2:5, mesos = TRUE,
a = seq(-1, 1, by = 0.1), apostasi = "ESOV", folds = NULL,
stratified = TRUE, seed = NULL, graph = FALSE)

alfaknn.tune(x, ina, nfolds = 10, k = 2:5, mesos = TRUE,
a = seq(-1, 1, by = 0.1), apostasi = "euclidean", rann = FALSE,
folds = NULL, stratified = TRUE, seed = NULL, graph = FALSE)

aitknn.tune(x, ina, nfolds = 10, k = 2:5, mesos = TRUE,
a = seq(-1, 1, by = 0.1), apostasi = "euclidean", rann = FALSE,
folds = NULL, stratified = TRUE, seed = NULL, graph = FALSE)

Arguments

x A matrix with the available compositional data. Zeros are allowed, but you must be careful to choose strictly positive values of \( \alpha \) or not to set apostasi= "Ait".

ina A group indicator variable for the available data.
nfolds The number of folds to be used. This is taken into consideration only if the folds argument is not supplied.
k A vector with the nearest neighbours to consider.
mesos This is used in the non standard algorithm. If TRUE, the arithmetic mean of the distances is calculated, otherwise the harmonic mean is used (see details).
a A grid of values of \( \alpha \) to be used only if the distance chosen allows for it.
apostasi The type of distance to use. For the compk.knn this can be one of the following: "ESOV", "taxicab", "Ait", "Hellinger", "angular" or "CS". See the references for them. For the alfa.knn this can be either "euclidean" or "manhattan".
rann

If you have large scale datasets and want a faster k-NN search, you can use kd-trees implemented in the R package "RANN". In this case you must set this argument equal to TRUE. Note however, that in this case, the only available distance is by default "euclidean".

folds

If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.

stratified

Do you want the folds to be created in a stratified way? TRUE or FALSE.

seed

You can specify your own seed number here or leave it NULL.

graph

If set to TRUE a graph with the results will appear.

Details

The k-NN algorithm is applied for the compositional data. There are many metrics and possibilities to choose from. The algorithm finds the k nearest observations to a new observation and allocates it to the class which appears most times in the neighbours.

Value

A list including:

per  A matrix or a vector (depending on the distance chosen) with the averaged over all folds rates of correct classification for all hyper-parameters ($\alpha$ and k).

performance  The estimated rate of correct classification.

best_a  The best value of $\alpha$. This is returned for "ESOV" and "taxicab" only.

best_k  The best number of nearest neighbours.

runtime  The run time of the cross-validation procedure.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References


See Also

comp.knn, alfarda.tune, cv.dda, cv.compnb

Examples

```r
x <- as.matrix(iris[, 1:4])
x <- x/ rowSums(x)
ina <- iris[, 5]
mod1 <- compknn.tune(x, ina, a = seq(1, 1, by = 0.1) )
mod2 <- alfaknn.tune(x, ina, a = seq(-1, 1, by = 0.1) )
```

Description

Tuning of the projection pursuit regression for compositional data.

Usage

```r
compppr.tune(y, x, nfolds = 10, folds = NULL, seed = NULL,
 nterms = 1:10, type = "alr", yb = NULL )
```

Arguments

- **y**: A matrix with the available compositional data, but zeros are not allowed.
- **x**: A matrix with the continuous predictor variables.
- **nfolds**: The number of folds to use.
- **folds**: If you have the list with the folds supply it here.
- **seed**: You can specify your own seed number here or leave it NULL.
- **nterms**: The number of terms to try in the projection pursuit regression.
- **type**: Either "alr" or "ilr" corresponding to the additive or the isometric log-ratio transformation respectively.
- **yb**: If you have already transformed the data using a log-ratio transformation put it here. Otherwise leave it NULL.
Tuning of the projection pursuit regression with compositional predictor variables

Details

The function performs tuning of the projection pursuit regression algorithm.

Value

A list including:

- \( k_l \) The average Kullback-Leibler divergence.
- \( \text{perf} \) The average Kullback-Leibler divergence.
- \( \text{runtime} \) The run time of the cross-validation procedure.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

- comp.ppr
- aknnreg.tune
- akernreg.tune

Examples

```r
y <- as.matrix(iris[, 1:3])
y <- y/ rowSums(y)
x <- iris[, 4]
mod <- compppr.tune(y, x)
```

Tuning of the projection pursuit regression with compositional predictor variables

Description

Tuning of the projection pursuit regression with compositional predictor variables.

Usage

```r
pprcomp.tune(y, x, nfolds = 10, folds = NULL, seed = NULL,
nterms = 1:10, type = "log", graph = FALSE)
```
Tuning of the projection pursuit regression with compositional predictor variables

Arguments

- **y**: A numerical vector with the continuous variable.
- **x**: A matrix with the available compositional data, but zeros are not allowed.
- **nfolds**: The number of folds to use.
- **folds**: If you have the list with the folds supply it here.
- **seed**: You can specify your own seed number here or leave it NULL.
- **nterms**: The number of terms to try in the projection pursuit regression.
- **type**: Either "alr" or "log" corresponding to the additive log-ratio transformation or the logarithm applied to the compositional predictor variables.
- **graph**: If graph is TRUE (default value) a filled contour plot will appear.

Details

The function performs tuning of the projection pursuit regression algorithm with compositional predictor variables.

Value

A list including:

- **runtime**: The run time of the cross-validation procedure.
- **mse**: The mean squared error of prediction for each number of terms.
- **opt.nterms**: The number of terms with the minimum mean squared error of prediction.
- **performance**: The minimum mean squared error of prediction.

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

- pprcomp, ice.pprcomp, alfapcr.tune, compppr.tune

Examples

```r
x <- as.matrix(iris[, 2:4])
x <- x/ rowSums(x)
y <- iris[, 1]
mod <- pprcomp.tune(y, x)
```
Tuning the number of PCs in the PCR with compositional data using the alpha-transformation

Description
This is a cross-validation procedure to decide on the number of principal components when using regression with compositional data (as predictor variables) using the $\alpha$-transformation.

Usage

```r
alfapcr.tune(y, x, model = "gaussian", nfolds = 10, maxk = 50, a = seq(-1, 1, by = 0.1),
  folds = NULL, ncores = 1, graph = TRUE, col.nu = 15, seed = NULL)
```

Arguments

- `y`: A vector with either continuous, binary or count data.
- `x`: A matrix with the predictor variables, the compositional data. Zero values are allowed.
- `model`: The type of regression model to fit. The possible values are "gaussian", "binomial" and "poisson".
- `nfolds`: The number of folds for the K-fold cross validation, set to 10 by default.
- `maxk`: The maximum number of principal components to check.
- `a`: A vector with a grid of values of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If $\alpha = 0$ the isometric log-ratio transformation is applied.
- `folds`: If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.
- `ncores`: How many cores to use. If you have heavy computations or do not want to wait for long time more than 1 core (if available) is suggested. It is advisable to use it if you have many observations and or many variables, otherwise it will slow down the process.
- `graph`: If graph is TRUE (default value) a filled contour plot will appear.
- `col.nu`: A number parameter for the filled contour plot, taken into account only if graph is TRUE.
- `seed`: You can specify your own seed number here or leave it NULL.

Details
The $\alpha$-transformation is applied to the compositional data first and the function "pcr.tune" or "glm-pcr.tune" is called.
Value

If graph is TRUE a filled contour will appear. A list including:

- **mspe**: The MSPE where rows correspond to the \(\alpha\) values and the columns to the number of principal components.
- **best.par**: The best pair of \(\alpha\) and number of principal components.
- **performance**: The minimum mean squared error of prediction.
- **runtime**: The time required by the cross-validation procedure.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

`alfa`, `profile`, `alfa.pcr`, `pcr.tune`, `glmpcr.tune`, `glm`

Examples

```r
library(MASS)
y <- as.vector(fgl[, 1])
x <- as.matrix(fgl[, 2:9])
x <- x / rowSums(x)
mod <- alfapcr.tune(y, x, nfolds = 10, maxk = 50, a = seq(-1, 1, by = 0.1) )
```

Tuning the parameters of the regularised discriminant analysis

Tuning the parameters of the regularised discriminant analysis

Description

Tuning the parameters of the regularised discriminant analysis for Euclidian data.

Usage

```r
rda.tune(x, ina, nfolds = 10, gam = seq(0, 1, by = 0.1), del = seq(0, 1, by = 0.1),
ncores = 1, folds = NULL, stratified = TRUE, seed = NULL)
```
Arguments

- **x**: A matrix with the data.
- **ina**: A group indicator variable for the available data.
- **nfolds**: The number of folds in the cross validation.
- **gam**: A grid of values for the $\gamma$ parameter as defined in Tsagris et al. (2016).
- **del**: A grid of values for the $\delta$ parameter as defined in Tsagris et al. (2016).
- **ncores**: The number of cores to use. If more than 1, parallel computing will take place. It is advisable to use it if you have many observations and or many variables, otherwise it will slow down the process.
- **folds**: If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.
- **stratified**: Do you want the folds to be created in a stratified way? TRUE or FALSE.
- **seed**: You can specify your own seed number here or leave it NULL.

Details

Cross validation is performed to select the optimal parameters for the regularised discriminant analysis and also estimate the rate of accuracy.

The covariance matrix of each group is calculated and then the pooled covariance matrix. The spherical covariance matrix consists of the average of the pooled variances in its diagonal and zeros in the off-diagonal elements. $\text{gam}$ is the weight of the pooled covariance matrix and $1-\text{gam}$ is the weight of the spherical covariance matrix, $\text{Sa} = \text{gam} \times \text{Sp} + (1-\text{gam}) \times \text{sp}$. Then it is a compromise between LDA and QDA. $\text{del}$ is the weight of $\text{Sa}$ and $1-\text{del}$ the weight of each group covariance group. This function is a wrapper for alfa.rda.

Value

A list including: If graph is TRUE a plot of a heatmap of the performance s will appear.

- **per**: An array with the estimate rate of correct classification for every fold. For each of the $M$ matrices, the row values correspond to gam and the columns to the del parameter.
- **percent**: A matrix with the mean estimated rates of correct classification. The row values correspond to gam and the columns to the del parameter.
- **se**: A matrix with the standard error of the mean estimated rates of correct classification. The row values correspond to gam and the columns to the del parameter.
- **result**: The estimated rate of correct classification along with the best gam and del parameters.
- **runtime**: The time required by the cross-validation procedure.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.
Tuning the principal components with GLMs

References


See Also

rda, alfa

Examples

mod <- rda.tune(as.matrix(iris[,1:4]), iris[,5], gam = seq(0, 1, by = 0.2), del = seq(0, 1, by = 0.2) )
mod

Tuning the principal components with GLMs

Description

Tuning the number of principal components in the generalised linear models.

Usage

pcr.tune(y, x, nfolds = 10, maxk = 50, folds = NULL, ncores = 1, seed = NULL, graph = TRUE)

glmpcr.tune(y, x, nfolds = 10, maxk = 10, folds = NULL, ncores = 1, seed = NULL, graph = TRUE)

multinompcr.tune(y, x, nfolds = 10, maxk = 10, folds = NULL, ncores = 1, seed = NULL, graph = TRUE)

Arguments

y A real valued vector for “pcr.tune”. A real valued vector for the “glmpcr.tune” with either two numbers, 0 and 1 for example, for the binomial regression or with positive discrete numbers for the poisson. For the “multinompcr.tune” a vector or a factor with more than just two values. This is a multinomial regression.

x A matrix with the predictor variables, they have to be continuous.
nfolds The number of folds in the cross validation.
maxk The maximum number of principal components to check.
Tuning the principal components with GLMs

- **folds**: If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.
- **ncores**: The number of cores to use. If more than 1, parallel computing will take place. It is advisable to use it if you have many observations and or many variables, otherwise it will slow down the process.
- **seed**: You can specify your own seed number here or leave it NULL.
- **graph**: If graph is TRUE a plot of the performance for each fold along the values of $\alpha$ will appear.

**Details**

Cross validation is performed to select the optimal number of principal components in the GLMs or the multinomial regression. This is used by `alfapcr.tune`.

**Value**

If graph is TRUE a plot of the performance versus the number of principal components will appear. A list including:

- **msp**: A matrix with the mean deviance of prediction or mean accuracy for every fold.
- **mpd**: A vector with the mean deviance of prediction or mean accuracy, each value corresponds to a number of principal components.
- **k**: The number of principal components which minimizes the deviance or maximises the accuracy.
- **performance**: The optimal performance, MSE for the linear regression, minimum deviance for the GLMs and maximum accuracy for the multinomial regression.
- **runtime**: The time required by the cross-validation procedure.

**Author(s)**

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

**References**


**See Also**

`pcr.tune, glm.pcr, alfa.pcr, alfapcr.tune`
Examples

```r
library(MASS)
x <- as.matrix(fgl[, 2:9])
y <- rpois(214, 10)
glm.pcr.tune(y, x, nfolds = 10, maxk = 20, folds = NULL, ncores = 1)
```

Description

Tuning the value of α in the α-regression.

Usage

```r
alfareg.tune(y, x, a = seq(0.1, 1, by = 0.1), nfolds = 10,
folds = NULL, nc = 1, seed = NULL, graph = FALSE)
```

Arguments

- `y`: A matrix with compositional data. Zero values are allowed.
- `x`: A matrix with the continuous predictor variables or a data frame including categorical predictor variables.
- `a`: The value of the power transformation, it has to be between -1 and 1. If zero values are present it has to be greater than 0. If \( \alpha = 0 \) the isometric log-ratio transformation is applied.
- `nfolds`: The number of folds to split the data.
- `folds`: If you have the list with the folds supply it here. You can also leave it NULL and it will create folds.
- `nc`: The number of cores to use. If you have a multicore computer it is advisable to use more than 1. It makes the procedure faster. It is advisable to use it if you have many observations and or many variables, otherwise it will slow down the process.
- `seed`: You can specify your own seed number here or leave it NULL.
- `graph`: If graph is TRUE a plot of the performance for each fold along the values of \( \alpha \) will appear.

Details

The \( \alpha \)-transformation is applied to the compositional data and the numerical optimisation is performed for the regression, unless \( \alpha = 0 \), where the coefficients are available in closed form.
Tuning the value of alpha in the alpha-regression

Value

A plot of the estimated Kullback-Leibler divergences (multiplied by 2) along the values of $\alpha$ (if graph is set to TRUE). A list including:

- **runtime**: The runtime required by the cross-validation.
- **kula**: A matrix with twice the Kullback-Leibler divergence of the observed from the fitted values. Each row corresponds to a fold and each column to a value of $\alpha$. The average over the columns equal the next argument, "kl".
- **kl**: A vector with twice the Kullback-Leibler divergence of the observed from the fitted values. Every value corresponds to a value of $\alpha$.
- **opt**: The optimal value of $\alpha$.
- **value**: The minimum value of twice the Kullback-Leibler.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr> and Giorgos Athineou <gioathineou@gmail.com>.

References


See Also

- alfa.reg, alfa

Examples

library(MASS)
y <- as.matrix(fgl[1:40, 2:4])
y <- y / rowSums(y)
x <- as.vector(fgl[1:40, 1])
mod <- alfareg.tune(y, x, a = seq(0, 1, by = 0.1), nfold = 5)
Two-sample test of high-dimensional means for compositional data

Description

Two-sample test of high-dimensional means for compositional data.

Usage

hd.meantest2(y1, y2, R = 1)

Arguments

y1 A matrix containing the compositional data of the first group.
y2 A matrix containing the compositional data of the second group.
R If R is 1 no bootstrap calibration is performed and the asymptotic p-value is returned. If R is greater than 1, the bootstrap p-value is returned.

Details

A two sample for high dimensional mean vectors of compositional data is implemented. See references for more details.

Value

A vector with the test statistic value and its associated (bootstrap) p-value.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

comp.test, maovjames, maov, hotel2T2, el.test2, eel.test2

Examples

m <- runif(200, 10, 15)
x1 <- rdiri(100, m)
x2 <- rdiri(100, m)
hd.meantest2(x1, x2)
Unconstrained log-contrast logistic or Poisson regression with compositional predictor variables

Description

Unconstrained log-contrast logistic or Poisson regression with compositional predictor variables.

Usage

ulc.glm(y, x, z = NULL, model = "logistic", xnew = NULL, znew = NULL)

Arguments

y A numerical vector containing the response variable values. This must be a continuous variable.

x A matrix with the predictor variables, the compositional data. No zero values are allowed.

z A matrix, data.frame, factor or a vector with some other covariate(s).

model This can be either "logistic" or "poisson".

xnew A matrix containing the new compositional data whose response is to be predicted. If you have no new data, leave this NULL as is by default.

znew A matrix, data.frame, factor or a vector with the values of some other covariate(s). If you have no new data, leave this NULL as is by default.

Details

The function performs the unconstrained log-contrast logistic or Poisson regression model. The logarithm of the compositional predictor variables is used (hence no zero values are allowed). The response variable is linked to the log-transformed data without the constraint that the sum of the regression coefficients equals 0. If you want the regression without the zum-to-zero contraints see lc.glm. Extra predictors variables are allowed as well, for instance categorical or continuous.

Value

A list including:

devi The residual deviance of the logistic or Poisson regression model.

be The constrained regression coefficients. Their sum equals 0.

est If the arguments "xnew" and znew were given these are the predicted or estimated values, otherwise it is NULL.

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>. 
Unconstrained log-contrast regression with compositional predictor variables

References


See Also

lc.glm, lc.reg, lc.reg2, alfa.pcr, glm.pcr

Examples

```r
y <- rbinom(150, 1, 0.5)
x <- rdiri(150, runif(3, 1,3))
mod <- ulc.glm(y, x)
```

Description

Unconstrained log-contrast regression with compositional predictor variables.

Usage

```r
ulc.reg(y, x, z = NULL, xnew = NULL, znew = NULL)
```

Arguments

- `y`: A numerical vector containing the response variable values. This must be a continuous variable.
- `x`: A matrix with the predictor variables, the compositional data. No zero values are allowed.
- `z`: A matrix, data.frame, factor or a vector with some other covariate(s).
- `xnew`: A matrix containing the new compositional data whose response is to be predicted. If you have no new data, leave this NULL as is by default.
- `znew`: A matrix, data.frame, factor or a vector with the values of some other covariate(s). If you have no new data, leave this NULL as is by default.

Details

The function performs the unconstrained log-contrast regression model as opposed to the log-contrast regression described in Aitchison (2003), pg. 84-85. The logarithm of the compositional predictor variables is used (hence no zero values are allowed). The response variable is linked to the log-transformed data without the constraint that the sum of the regression coefficients equals 0. If you want the regression model with the zum-to-zero contraints see `lc.reg`. Extra predictors variables are allowed as well, for instance categorical or continuous.
Unconstrained log-contrast regression with multiple compositional predictors

Value

A list including:

- **be**: The constrained regression coefficients. Their sum equals 0.
- **covbe**: If covariance matrix of the constrained regression coefficients.
- **va**: The estimated regression variance.
- **residuals**: The vector of residuals.
- **est**: If the arguments "xnew" and "znew" were given these are the predicted or estimated values, otherwise it is NULL.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

- lc.reg, lcreg.aov, lc.reg2, ulc.reg2, alfa.pcr, alfa.knn.reg

Examples

```r
y <- iris[, 1]
x <- as.matrix(iris[, 2:4])
x <- x / rowSums(x)
mod1 <- ulc.reg(y, x)
mod2 <- ulc.reg(y, x, z = iris[, 5])
```

Unconstrained log-contrast regression with multiple compositional predictors

Unconstrained log-contrast regression with multiple compositional predictors

Description

Unconstrained log-contrast regression with multiple compositional predictors.

Usage

```r
ulc.reg2(y, x, z = NULL, xnew = NULL, znew = NULL)
```
Unconstrained log-contrast regression with multiple compositional predictors

Arguments

y A numerical vector containing the response variable values. This must be a continuous variable.
x A list with multiple matrices with the predictor variables, the compositional data. No zero values are allowed.
z A matrix, data.frame, factor or a vector with some other covariate(s).
xnew A matrix containing a list with multiple matrices with compositional data whose response is to be predicted. If you have no new data, leave this NULL as is by default.
znew A matrix, data.frame, factor or a vector with the values of some other covariate(s). If you have no new data, leave this NULL as is by default.

Details

The function performs the unconstrained log-contrast regression model as opposed to the log-contrast regression described in Aitchison (2003), pg. 84-85. The logarithm of the compositional predictor variables is used (hence no zero values are allowed). The response variable is linked to the log-transformed data without the constraint that the sum of the regression coefficients equals 0. If you want the regression model with the zem-to-zero contraints see lc.reg2. Extra predictors variables are allowed as well, for instance categorical or continuous. Similarly to lc.reg2 there are multiple compositions treated as predictor variables.

Value

A list including:

be The constrained regression coefficients. Their sum equals 0.
covbe If covariance matrix of the constrained regression coefficients.
va The estimated regression variance.
residuals The vector of residuals.
est If the arguments "xnew" and "znew" were given these are the predicted or estimated values, otherwise it is NULL.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

Unit-Weibull regression models for proportions

See Also

lc.reg2, ulc.reg, lc.reg, alfa.pcr, alfa.knn.reg

Examples

```r
y <- iris[, 1]
x <- list()
x1 <- as.matrix(iris[, 2:4])
x1 <- x1 / rowSums(x1)
x[[ 1 ]] <- x1
x[[ 2 ]] <- rdirc(150, runif(4) )
x[[ 3 ]] <- rdirc(150, runif(5) )
mod <- lc.reg2(y, x)
```

Description

Unit-Weibull regression models for proportions.

Usage

`unitweib.reg(y, x, tau = 0.5)`

Arguments

- `y`: A numerical vector proportions. 0s and 1s are allowed.
- `x`: A matrix or a data frame with the predictor variables.
- `tau`: The quantile to be used for estimation. The default value is 0.5 yielding the median.

Details

See the reference paper.

Value

A list including:

- `loglik`: The loglikelihood of the regression model.
- `info`: A matrix with all estimated parameters, their standard error, their Wald-statistic and its associated p-value.
Zero adjusted Dirichlet regression

Author(s)
Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

See Also
propreg, beta.reg

Examples
y <- exp(-rweibull(100, 1, 1))
x <- matrix(rnorm(100*2), ncol = 2)
a <- unitweib.reg(y, x)

zero adjusted Dirichlet regression

Description
Zero adjusted Dirichlet regression.

Usage
zadr(y, x, xnew = NULL, tol = 1e-05)
mixreg(param, z)

Arguments

y A matrix with the compositional data (dependent variable). The number of observations (vectors) with no zero values should be more than the columns of the predictor variables. Otherwise, the initial values will not be calculated.

x The predictor variable(s), they can be either continuous or categorical or both.

xnew If you have new data use it, otherwise leave it NULL.

tol A tolerance level to terminate the maximisation process.

param Some arguments passed on to the mixreg helper function.

z Some arguments passed on to the mixreg helper function.
Details

A zero adjusted Dirichlet regression is being fitted. The likelihood consists of two components. The contributions of the non zero compositional values and the contributions of the compositional vectors with at least one zero value. The second component may have many different sub-categories, one for each pattern of zeros. The function "mixreg" is a helper function and is not intended to be called directly by the user.

Value

A list including:

- `runtime`: The time required by the regression.
- `loglik`: The value of the log-likelihood.
- `phi`: The precision parameter. If covariates are linked with it (function "diri.reg2"), this will be a vector.
- `be`: The beta coefficients.
- `seb`: The standard error of the beta coefficients.
- `sigma`: The covariance matrix of the regression parameters (for the mean vector and the phi parameter) in the function "diri.reg2".
- `est`: The fitted or the predicted values (if xnew is not NULL).

Author(s)

Michail Tsagris.
R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References


See Also

zad.est, diri.reg, kl.compreg, ols.compreg, alfa.reg

Examples

```R
x <- as.vector(iris[, 4])
y <- as.matrix(iris[, 1:3])
y <- y / rowSums(y)
mod1 <- diri.reg(y, x)
y[sample(1:450, 15) ] <- 0
mod2 <- zadr(y, x)
```
Index

* Additive log-ratio-transformation
  The additive log-ratio transformation and its inverse, 175

* Bhattacharyya distance
  Kullback-Leibler divergence and Bhattacharyya distance between two Dirichlet distributions, 102

* Compositional data
  Compositional-package, 6

* Contour plots
  Compositional-package, 6

* Dirichlet mean vector
  Log-likelihood ratio test for a Dirichlet mean vector, 111

* Dirichlet distribution
  Contour plot of the Dirichlet distribution in S^2, 20
  Contour plot of the Flexible Dirichlet distribution in S^2, 21
  Density values of a Dirichlet distribution, 55
  Dirichlet random values simulation, 58
  Kullback-Leibler divergence and Bhattacharyya distance between two Dirichlet distributions, 102
  MLE of the Dirichlet distribution, 119
  MLE of the Dirichlet distribution via Newton-Raphson, 120

* Dirichlet regression
  Dirichlet regression, 59

* Discriminant analysis
  Compositional-package, 6

* Equality of the covariance matrices
  Hotelling’s multivariate version of the 2 sample t-test for Euclidean data, 92
  Multivariate analysis of variance, 124

* Euclidean distance
  The alpha-distance, 176
  The alpha-IT-distance, 178
  The k-nearest neighbours using the alpha-distance, 190

* Frechet mean
  Helper Frechet mean for compositional data, 88
  The Frechet mean for compositional data, 188

* GLMs
  Quasi binomial regression for proportions, 154

* GLM
  Tuning the principal components with GLMs, 209

* Gaussian mixture models
  Density of compositional data from Gaussian mixture models, 51
  Density of the folded normal distribution, 53
  Simulation of compositional data from Gaussian mixture models, 167
  Simulation of compositional data from the folded normal distribution, 170

* Gaussianmixture model
  Contour plot of the Gaussian mixture model in S^2, 23

* Hypothesis test
  Empirical likelihood for a one sample mean vector hypothesis testing, 67
Exponential empirical likelihood for a one sample mean vector hypothesis testing, 76
Hotelling’s multivariate version of the 1 sample t-test for Euclidean data, 91

* Kullback-Leibler divergence
Helper functions for the Kullback-Leibler regression, 89

* Kullback-Leibler divergence
Kullback-Leibler divergence and Bhattacharyya distance between two Dirichlet distributions, 102

* Log-likelihood ratio test
Log-likelihood ratio test for a Dirichlet mean vector, 111
Log-likelihood ratio test for a symmetric Dirichlet distribution, 112

* MLE
Compositional-package, 6

* Manhattan distance
The alpha-distance, 176
The alpha-IT-distance, 178
The k-nearest neighbours using the alpha-distance, 190

* Multivariate analysis of variance
Hotelling’s multivariate version of the 2 sample t-test for Euclidean data, 92
James multivariate version of the t-test, 99
Multivariate analysis of variance, 124
Multivariate analysis of variance (James test), 125
Two-sample test of high-dimensional means for compositional data, 213

* Multivariate hypothesis testing
Empirical likelihood hypothesis testing for two mean vectors, 68
Exponential empirical likelihood hypothesis testing for two mean vectors, 77
Hypothesis testing for two or more compositional mean vectors, 93

* Newton-Raphson
MLE of the Dirichlet distribution via Newton-Rapshon, 120

* No equality of the covariance matrices
Multivariate analysis of variance (James test), 125

* Non parametric test
Empirical likelihood hypothesis testing for two mean vectors, 68

* Quasi regression
Quasi binomial regression for proportions, 154

* Regression
Compositional-package, 6

* Regularised discriminant analysis
Cross validation for the regularised and flexible discriminant analysis with compositional data using the alpha-transformation, 37
Regularised and flexible discriminant analysis for compositional data using the alpha-transformation, 159
Regularised discriminant analysis for Euclidean data, 160
Tuning of the k-NN algorithm for compositional data, 201
Tuning the parameters of the regularised discriminant analysis, 207

* Ridge regression
Cross validation for the ridge regression, 39

* Tuning of the hyper-parameters
Tuning the parameters of the regularised discriminant analysis, 207

* Unequality of the covariance matrices
James multivariate version of the t-test, 99
Two-sample test of high-dimensional means for compositional data, 213

* bandwidth tuning
Tuning of the bandwidth $h$ of the kernel using the maximum likelihood cross validation, 198

*bivariate normal distribution*
- Contour plot of the alpha multivariate normal in $S^2$, 18
- Contour plot of the alpha-folded model in $S^2$, 19
- Contour plot of the normal distribution in $S^2$, 26

*bivariate skew skewnormal distribution*
- Contour plot of the skew skew-normal distribution in $S^2$, 27

*bivariate $t$ distribution*
- Contour plot of the $t$ distribution in $S^2$, 28

*compositional data*
- Hypothesis testing for two or more compositional mean vectors, 93

*contour plot*
- Contour plot of the alpha multivariate normal in $S^2$, 18
- Contour plot of the alpha-folded model in $S^2$, 19
- Contour plot of the Dirichlet distribution in $S^2$, 20
- Contour plot of the Flexible Dirichlet distribution in $S^2$, 21
- Contour plot of the Gaussian mixture model in $S^2$, 23
- Contour plot of the kernel density estimate in $S^2$, 25
- Contour plot of the normal distribution in $S^2$, 26
- Contour plot of the skew skew-normal distribution in $S^2$, 27
- Contour plot of the $t$ distribution in $S^2$, 28

*cross-validation*
- Cross validation for the regularised and flexible discriminant analysis with compositional data using the alpha-transformation, 37

*density values*
- Density values of a Dirichlet distribution, 55

*fractional response*
- Quasi binomial regression for proportions, 154

*inverse transformation*
- Inverse of the alpha-transformation, 98

*kernel density estimate*
- Multivariate kernel density estimation, 126

*kernel density*
- Contour plot of the kernel density estimate in $S^2$, 25
- Tuning of the bandwidth $h$ of the kernel using the maximum likelihood cross validation, 198

*location and scatter*
- Estimating location and scatter parameters for compositional data, 71

*maximum likelihood cross validation*
- Tuning of the bandwidth $h$ of the kernel using the maximum likelihood cross validation, 198

*maximum likelihood estimation*
- Estimation of the value of alpha in the folded model, 74
- MLE of the Dirichlet distribution, 119
- MLE of the Dirichlet distribution via Newton-Raphson, 120
- MLE of the folded model for a given value of alpha, 121

*maximum log-likelihood estimation*
- Fast estimation of the value of alpha, 79

*mean vector*
- Empirical likelihood for a one sample mean vector hypothesis testing, 67
- Exponential empirical likelihood for a one sample mean vector hypothesis testing, 76
- Hotelling’s multivariate version
INDEX

of the 1 sample t-test for Euclidean data, 91
* mixtures of normal distributions
  Mixture model selection via BIC, 113
* model selection
  Mixture model selection via BIC, 113
* multivariate kernel
  Multivariate kernel density estimation, 126
* multivariate linear regression
  Multivariate linear regression, 128
* multivariate normal distribution
  Multivariate normal random values simulation on the simplex, 129
* multivariate regression
  Dirichlet regression, 59
  Helper functions for the Kullback-Leibler regression, 89
  Non linear least squares regression for compositional data, 139
  Regression with compositional data using the alpha-transformation, 157
  Spatial median regression, 171
* multivariate regression
  Multivariate regression with compositional data, 132
* multivariate skew normal distribution
  Multivariate skew normal random values simulation on the simplex, 134
* multivariate t distribution
  MLE for the multivariate t distribution, 116
  Multivariate t random values simulation on the simplex, 135
* non parametric test
  Exponential empirical likelihood hypothesis testing for two mean vectors, 77
  Hypothesis testing for two or more compositional mean vectors, 93
* ordinary least squares
  Non linear least squares regression for compositional data, 139
* orthogonal matrix
  The Helmert sub-matrix, 189
* parameters tuning
  Cross validation for the ridge regression with compositional data as predictor using the alpha-transformation, 40
* plot
  Ridge regression plot, 163
  Ridge regression with the alpha-transformation plot, 165
* principal components regression
  Multivariate or univariate regression with compositional data in the covariates side using the alpha-transformation, 131
  Tuning the number of PCs in the PCR with compositional data using the alpha-transformation, 206
* principal components
  Tuning the principal components with GLMs, 209
* profile log-likelihood
  Estimation of the value of alpha via the profile log-likelihood, 75
* random values simulation
  Dirichlet random values simulation, 58
  Multivariate normal random values simulation on the simplex, 129
  Multivariate skew normal random values simulation on the simplex, 134
  Multivariate t random values simulation on the simplex, 135
* random vectors simulation
  Density of compositional data from Gaussian mixture models, 51
  Density of the folded normal distribution, 53
  Simulation of compositional data from Gaussian mixture models, 167
Simulation of compositional data from the folded normal distribution, 170

* regularisation
Ridge regression, 162

* ridge regression
Cross validation for the ridge regression with compositional data as predictor using the alpha-transformation, 40
Ridge regression, 162
Ridge regression plot, 163
Ridge regression with compositional data in the covariates side using the alpha-transformation, 164
Ridge regression with the alpha-transformation plot, 165

* robust estimation
Estimating location and scatter parameters for compositional data, 71

* spatial median regression
Multivariate regression with compositional data, 132
Spatial median regression, 171

* symmetric Dirichlet distribution
Log-likelihood ratio test for a symmetric Dirichlet distribution, 112

* ternary plot
Ternary diagram, 172
Ternary diagram of regression models, 174

* tuning
Tuning the number of PCs in the PCR with compositional data using the alpha-transformation, 206

* visualisation
Ternary diagram, 172
Ternary diagram of regression models, 174

a.est, 54, 73, 122, 171, 184
a.est (Estimation of the value of alpha in the folded model), 74
a.mle, 184

a.mle (MLE of the folded model for a given value of alpha), 121
acor, 12, 71, 154, 198
acor (Alpha generalised correlations between two compositional datasets), 8
acor.tune, 9, 71
acor.tune (Tuning of the alpha generalised correlations between two compositional datasets), 197
aeqdist.etest, 9
aeqdist.etest (Energy test of equality of distributions using the alpha-transformation), 70
ait, 179
ait (The alpha-IT transformation), 177
ait.knn, 178
ait.knn (The k-NN algorithm for compositional data), 192
aitdist, 178
aitdist (The alpha-IT-distance), 178
aitdista (The alpha-IT-distance), 178
aitknn.tune (Tuning of the k-NN algorithm for compositional data), 201
akern.reg, 34, 35, 101, 152, 180
akern.reg (The alpha-kernel regression with compositional response data), 182
akernreg.tune, 34, 183, 204
akernreg.tune (Cross validation for the alpha-kernel regression with compositional response data), 34
aknn.reg, 34–36, 88, 101, 152, 182, 183
aknn.reg (The alpha-k-NN regression for compositional response data), 179
aknnreg.tune, 31, 35, 36, 180, 183, 204
aknnreg.tune (Cross validation for the alpha-k-NN regression with compositional response data), 32
alef (The alpha-transformation), 183
alfa, 7, 9, 12, 71, 73, 75, 76, 80, 85, 87, 89, 99, 116, 122, 141, 144, 160, 161, 171, 173, 175, 177–179, 186,
INDEX

alfa (The alpha-transformation), 183
alfa.contour, 20
alfa.contour (Contour plot of the alpha multivariate normal in $S^2$), 18
alfa.fda, 32, 34, 35
alfa.fda (Regularised and flexible discriminant analysis for compositional data using the alpha-transformation), 159
alfa.knn, 58, 83, 85, 116, 137, 160, 182, 191
alfa.knn (The k-NN algorithm for compositional data), 192
alfa.knn.reg, 10, 104, 108, 110, 216, 218
alfa.knn.reg (The alpha-k-NN regression with compositional predictor variables), 181
alfa.mds, 151
alfa.mds (Principal coordinate analysis using the alpha-distance), 149
alfa.mix.norm, 83, 116
alfa.mix.norm (Gaussian mixture models for compositional data using the alpha-transformation), 84
alfa.nb, 51, 160, 191, 193
alfa.nb (Naive Bayes classifiers for compositional data using the alpha-transformation), 138
alfa.pca, 145, 150, 151
alfa.pca (Principal component analysis using the alpha-transformation), 146
alfa.pcr (Multivariate or univariate regression with compositional data in the covariates side using the alpha-transformation), 131
alfa.profile, 9, 71, 75, 79, 80, 122, 175, 184, 195, 198
alfa.profile (Estimation of the value of alpha via the profile log-likelihood), 75
alfa.rda, 32, 34, 35, 38, 83, 85, 116, 137, 138, 159, 161, 191, 193, 208
alfa.rda (Regularised and flexible discriminant analysis for compositional data using the alpha-transformation), 159
alfa.reg, 60, 140, 152, 172, 177, 180, 183, 212, 220
alfa.reg (Regression with compositional data using the alpha-transformation), 157
alfa.ridge, 41, 162–164, 166, 182
alfa.ridge (Ridge regression with compositional data in the covariates side using the alpha-transformation), 164
alfa.tune, 76, 175, 184, 195
alfa.tune (Fast estimation of the value of alpha), 79
alfadist, 99, 179, 187
alfadist (The alpha-distance), 176
alfadista (The alpha-distance), 176
alfafda.tune (Cross validation for the regularised and flexible discriminant analysis with compositional data using the alpha-transformation), 37
alfainv, 75, 76, 80, 89, 122, 175, 177, 184, 189–191, 194, 198
alfainv (Inverse of the alpha-transformation), 98
alfaknn.tune (Tuning of the k-NN algorithm for compositional data), 201
alfaknnreg.tune (Cross validation for the alpha-k-NN regression with compositional predictor variables), 31
alfanb.tune, 38, 45
alfanb.tune (Cross-validation for the naive Bayes classifiers for compositional data using the alpha-transformation), 50
alfann (The k-nearest neighbours using the alpha-distance), 190
alfapcr.tune, 14, 65, 132, 148, 200, 205, 210
alfapcr.tune (Tuning the number of PCs in the PCR with compositional
data using the alpha-transformation), 206
alfarda.tune, 45, 51, 160, 203
alfarda.tune (Cross validation for the regularised and flexible discriminant analysis with compositional data using the alpha-transformation), 37
alfareg.tune, 133, 157, 158
alfareg.tune (Tuning the value of alpha in the alpha-regression), 211
alfaridge.plot, 164, 165
alfaridge.plot (Ridge regression with the alpha-transformation plot), 165
alfaridge.tune, 39, 40, 165
alfaridge.tune (Cross validation for the ridge regression with compositional data as predictor using the alpha-transformation), 40
All pairwise additive log-ratio transformations, 7
Alpha generalised correlations between two compositional datasets, 8
alpha.mle, 73, 75, 171, 184
alpha.mle (MLE of the folded model for a given value of alpha), 121
alr, 7, 12, 47, 105, 154, 178, 184–188, 194
alr (The additive log-ratio transformation and its inverse), 175
alr.all (All pairwise additive log-ratio transformations), 7
alrinv (The additive log-ratio transformation and its inverse), 175
ANOVA for the log-contrast regression versus the uncostrained log-contrast regression, 9
anova_propreg, 156
Bayesian network learning with compositional data, 10
bc, 175, 184, 188
bc (The Box-Cox transformation applied to ratios of components), 185
Beta regression, 12
beta.est, 13, 157
beta.reg, 62, 219
beta.reg (Beta regression), 12
bic.alfamixnorm, 85, 114
bic.alfamixnorm (Mixture model selection with the alpha-transformation using BIC), 114
bic.mixcompnorm, 23, 52, 83, 85, 167
bic.mixcompnorm (Mixture model selection via BIC), 113
bivt.contour, 17, 19–22, 25–28, 116, 117
bivt.contour (Contour plot of the t distribution in S^2), 28
censpois.mle, 15
Choose the number of principal components via reconstruction error, 13
choose.pc (Choose the number of principal components via reconstruction error), 13
colbeta.est, 157
colbeta.est (Column-wise MLE of some univariate distributions), 14
collogitnorm.est (Column-wise MLE of some univariate distributions), 14
colsimplex.est (Column-wise MLE of some univariate distributions), 14
colunitweibull.est (Column-wise MLE of some univariate distributions), 14
comp.den, 116, 117, 130, 135, 136, 173
comp.den (Estimating location and scatter parameters for compositional data), 71
comp.kern (Multivariate kernel density estimation for compositional data), 127
comp.kerncontour, 17, 21, 22, 25, 127, 128, 173, 174, 199
comp.kerncontour (Contour plot of the kernel density estimate in S^2), 25
comp.knn, 58, 66, 137, 138, 187, 203
comp.knn (The k-NN algorithm for compositional data), 192
comp.nb (Naive Bayes classifiers for compositional data), 136
comp.ppr, 96, 153, 180, 183, 204
comp.ppr (Projection pursuit regression for compositional data), 151
comp.reg, 31, 48, 60, 64, 90, 105, 129, 140, 143, 152, 158, 172, 180, 183
comp.reg (Multivariate regression with compositional data), 132
comp.test, 68, 69, 77, 79, 92, 93, 100, 125, 126, 213
comp.test (Hypothesis testing for two or more compositional mean vectors), 93
compbn (Bayesian network learning with compositional data), 10
compknn.tune, 38, 45, 51, 193
compknn.tune (Tuning of the k-NN algorithm for compositional data), 201
comppnorm.contour, 17, 19–22, 25, 26, 28, 29
comppnorm.contour (Contour plot of the normal distribution in S^2), 26
Compositional-package, 6
compppr.tune, 31, 152, 205
compppr.tune (Tuning of the projection pursuit regression for compositional data), 203
Constrained linear least squares for compositional responses and predictors, 15
Contour plot of mixtures of Dirichlet distributions in S^2, 17
Contour plot of the alpha multivariate normal in S^2, 18
Contour plot of the alpha-folded model in S^2, 19
Contour plot of the Dirichlet distribution in S^2, 20
Contour plot of the Flexible Dirichlet distribution in S^2, 21
Contour plot of the Gaussian mixture model in S^2, 23
Contour plot of the Kent distribution in S^2, 24
Contour plot of the kernel density estimate in S^2, 25
Contour plot of the normal distribution in S^2, 26
Contour plot of the skew skew-normal distribution in S^2, 27
Contour plot of the t distribution in S^2, 28
Cross validation for some compositional regression models, 30
Cross validation for the alpha-k-NN regression with compositional predictor variables, 31
Cross validation for the alpha-k-NN regression with compositional response data, 32
Cross validation for the alpha-kernel regression with compositional response data, 34
Cross validation for the kernel regression with Euclidean response data, 35
Cross validation for the regularised and flexible discriminant analysis with compositional data using the alpha-transformation, 37
Cross validation for the ridge regression, 39
Cross validation for the ridge regression with compositional data as predictor using the alpha-transformation, 40
Cross validation for the transformation-free linear regression for compositional responses and predictors, 42
Cross-validation for the constrained linear least squares for compositional responses and predictors, 43
Cross-validation for the Dirichlet discriminant analysis, 44
Cross-validation for the LASSO Kullback-Leibler divergence
based regression, 46
Cross-validation for the LASSO
cross-validation for the naive Bayes
Cross-validation for the naive Bayes
Cross-validation for the naive Bayes
Cross-validation for the naive Bayes
Cross-validation for some
Cross-validation for the
Cross-validation for the
Cross-validation for the
Cross-validation for the
Cross-validation for the
cross-validation for the
Cross-validation for the
Cross-validation for the
Cross-validation for the
Cross-validation for the
Cross-validation for the
Cross-validation for the
Cross-validation for the
Cross-validation for the
Density of compositional data from
Density of the Flexible Dirichlet
density of the Folded normal
density values of a Dirichlet
density values of a mixture of
Density of the Flexible Dirichlet
density of the Folded normal
density values of a Dirichlet
density values of a mixture of
Density of the Flexible Dirichlet
density of the Folded normal
density values of a Dirichlet
density values of a mixture of
ddf, 170
dfd (Density of the Flexible Dirichlet
ddf (density of the folded normal
dir. contour, 17, 19, 20, 23, 26–29, 55, 59, 120, 121, 173, 174
dir. contour (Contour plot of the
Density of the Flexible Dirichlet
density of the folded normal
dir. est, 55, 59, 102, 111, 113, 121
dir. est (MLE of the Dirichlet
dir. lasso, 13, 58, 64, 90, 120, 129, 133, 140, 158, 172, 220
dir. reg (Dirichlet regression), 59
dir. reg2 (dirichlet regression), 59
Dirichlet discriminant analysis, 57
Dirichlet random values simulation, 58
Dirichlet regression, 59
dirimean.test, 113
dirimean.test (Log-likelihood ratio
distance based regression models for
Divergence based regression for
Divergence based regression for
Divergence based regression for
Divergence based regression for
Divergence based regression for
Divergence based regression for
Divergence based regression for
compositional data in the covariates side using the alpha-transformation. 64
Divergence matrix of compositional data. 66
dmix.compnorm (Density of compositional data from Gaussian mixture models). 51
dmixdiri (Density values of a mixture of Dirichlet distributions). 56
eel.test1, 68, 92
eel.test1 (Exponential empirical likelihood for a one sample mean vector hypothesis testing), 76
eel.test2, 69, 93, 95, 100, 213
eel.test2 (Exponential empirical likelihood hypothesis testing for two mean vectors), 77
el.test, 67
el.test1, 77, 92
el.test1 (Empirical likelihood for a one sample mean vector hypothesis testing), 67
eel.test2, 68, 77, 79, 92, 93, 95, 100, 213
eel.test2 (Empirical likelihood hypothesis testing for two mean vectors), 68
Empirical likelihood for a one sample mean vector hypothesis testing. 67
Empirical likelihood hypothesis testing for two mean vectors. 68
Energy test of equality of distributions using the alpha-transformation. 70
es (The ESOV-distance). 186
esov, 177, 193
esov (The ESOV-distance). 186
esov.mds, 150
esov.mds (Principal coordinate analysis using the Jensen-Shannon divergence). 150
esova (The ESOV-distance). 186
Estimating location and scatter parameters for compositional data. 71
Estimation of the probability left outside the simplex when using the alpha-transformation. 72
Estimation of the value of alpha in the folded model. 74
Estimation of the value of alpha via the profile log-likelihood. 75
Exponential empirical likelihood for a one sample mean vector hypothesis testing. 76
Exponential empirical likelihood hypothesis testing for two mean vectors. 77
Fast estimation of the value of alpha. 79
fd.contour (Contour plot of the Flexible Dirichlet distribution in S^2). 21
fd.est, 22, 53, 170
fd.est (Fitting a Flexible Dirichlet distribution). 80
Fitting a Flexible Dirichlet distribution. 80
folded.contour, 19, 22, 54
folded.contour (Contour plot of the alpha-folded model in S^2). 19
fp, 175, 184, 186
fp (The folded power transformation). 187
frechet (The Frechet mean for compositional data). 188
frechet2 (Helper Frechet mean for compositional data). 88
gammapois.mle, 15
Gaussian mixture models for compositional data. 82
Gaussian mixture models for compositional data using the alpha-transformation. 84
Generate random folds for cross-validation. 86
glm, 207
glm.pcr, 65, 107, 132, 200, 210, 215
glm.pcr (Principal component generalised linear models). 147
glm.pcr.tune, 207
INDEX

glmpcr.tune (Tuning the principal components with GLMs), 209
green, 175, 178, 184, 186, 188, 194
Greenacre’s power transformation, 87
Greenacre’s power transformation, 87
hd.meantest2, 95
hd.meantest2 (Two-sample test of high-dimensional means for compositional data), 213
helling.prop.reg (Distance based regression models for proportions), 61
helm (The Helmert sub-matrix), 189
Helper Frechet mean for compositional data, 88
Helper functions for the Kullback-Leibler regression, 89
hotel1T2, 68, 77
hotel1T2 (Hotelling’s multivariate version of the 1 sample t-test for Euclidean data), 91
hotel2T2, 68, 69, 77, 79, 92, 95, 100, 125, 126, 213
hotel2T2 (Hotelling’s multivariate version of the 2 sample t-test for Euclidean data), 92
Hotelling’s multivariate version of the 1 sample t-test for Euclidean data, 91
Hotelling’s multivariate version of the 2 sample t-test for Euclidean data, 92
hsecant01.est (MLE of distributions defined in the (0, 1) interval), 117
Hypothesis testing for two or more compositional mean vectors, 93
ibeta.est (MLE of distributions defined in the (0, 1) interval), 117
ICE plot for projection pursuit regression with compositional predictor variables, 95
ICE plot for univariate kernel regression, 96
ice.kernreg, 96, 101
ice.kernreg (ICE plot for univariate kernel regression), 96
ice.pprcomp, 97, 153, 205
ice.pprcomp (ICE plot for projection pursuit regression with compositional predictor variables), 95
Inverse of the alpha-transformation, 98
james, 68, 69, 77, 79, 92, 93, 125, 126
James multivariate version of the t-test, 99
js.compreg, 60, 66, 90, 129, 133, 140, 158, 172, 187
js.compreg (Divergence based regression for compositional data), 62
kent.contour (Contour plot of the Kent distribution in S^2), 24
kern.reg, 96
kern.reg (Kernel regression with a numerical response vector or matrix), 100
Kernel regression with a numerical response vector or matrix, 100
kernreg.tune, 97, 101
kernreg.tune (Cross validation for the kernel regression with Euclidean response data), 35
kl.alfapcr, 16, 145, 147, 196, 200
kl.alfapcr (Divergence based regression for compositional data with compositional data in the covariates side using the alpha-transformation), 64
kl.compreg, 31, 47, 60, 90, 104, 129, 140, 143, 152, 158, 180, 183, 220
kl.compreg (Divergence based regression for compositional data), 62
kl.compreg2 (Helper functions for the Kullback-Leibler regression), 89
kl.diri (Kullback-Leibler divergence and Bhattacharyya distance)
### INDEX

- between two Dirichlet distributions, 102
- klalfapcr.tune, 43, 44, 65
- klalfapcr.tune (Tuning of the divergence based regression for compositional data with compositional data in the covariates side using the alpha-transformation), 199
- klcompreg.boot (Helper functions for the Kullback-Leibler regression), 89
- Kullback-Leibler divergence and Bhattacharyya distance between two Dirichlet distributions, 102
- kumar.est (MLE of distributions defined in the (0, 1) interval), 117
- LASSO Kullback-Leibler divergence based regression, 103
- LASSO log-ratio regression with compositional response, 104
- lasso.compreg, 47, 48, 104, 143
- lasso.compreg (LASSO log-ratio regression with compositional response), 104
- lasso.klcompreg, 47, 48, 105, 143
- lasso.klcompreg (LASSO Kullback-Leibler divergence based regression), 103
- lassocoef.plot, 47, 48, 104, 105
- lassocoef.plot (Plot of the LASSO coefficients), 142
- lc.glm, 214, 215
- lc.glm (Log-contrast logistic or Poisson regression with compositional predictor variables), 106
- lc.reg, 9, 10, 96, 97, 107, 109, 110, 153, 215, 216, 218
- lc.reg (Log-contrast regression with compositional predictor variables), 107
- lc.reg2 (Log-contrast regression with multiple compositional predictors), 109
- lcreg.aov (ANNOVA for the log-contrast regression versus the uncostrained log-contrast regression), 9
- lm, 129
- lm.ridge, 162
- Log-contrast logistic or Poisson regression with compositional predictor variables, 106
- Log-contrast regression with compositional predictor variables, 107
- Log-contrast regression with multiple compositional predictors, 109
- Log-likelihood ratio test for a Dirichlet mean vector, 111
- Log-likelihood ratio test for a symmetric Dirichlet distribution, 112
- logistic_only, 156
- logitnorm.est (MLE of distributions defined in the (0, 1) interval), 117
- logpca, 147
- logpca (Principal component analysis), 144

- makefolds (Generate random folds for cross-validation), 86
- maov, 68, 69, 77, 79, 92, 93, 95, 126, 213
- maov (Multivariate analysis of variance), 124
- maovjames, 69, 79, 95, 100, 125, 213
- maovjames (Multivariate analysis of variance (James test)), 125
- mix.compnorm (Gaussian mixture models for compositional data), 82
- mix.compnorm.contour (Contour plot of the Gaussian mixture model in S^2), 23
- mixdiri.contour, 21, 57, 169
- mixdiri.contour (Contour plot of mixtures of Dirichlet distributions in S^2), 17
mixreg (Zero adjusted Dirichlet regression), 219
Mixture model selection via BIC, 113
Mixture model selection with the alpha-transformation using BIC, 114
mkde, 128, 199
mkde (Multivariate kernel density estimation), 126
mkde.tune, 127
mkde.tune (Tuning of the bandwidth h of the kernel using the maximum likelihood cross validation), 198
MLE for the multivariate t distribution, 116
MLE of distributions defined in the (0, 1) interval, 117
MLE of the Dirichlet distribution, 119
MLE of the Dirichlet distribution via Newton-Rapshon, 120
MLE of the folded model for a given value of alpha, 121
MLE of the zero adjusted Dirichlet distribution, 123
mmhc.skel, 11
multinompcr.tune (Tuning the principal components with GLMs), 209
Multivariate analysis of variance, 124
Multivariate analysis of variance (James test), 125
Multivariate kernel density estimation, 126
Multivariate kernel density estimation for compositional data, 127
Multivariate linear regression, 128
Multivariate normal random values simulation on the simplex, 129
Multivariate or univariate regression with compositional data in the covariates side using the alpha-transformation, 131
Multivariate regression with compositional data, 132
Multivariate skew normal random values simulation on the simplex, 134
Multivariate t random values simulation on the simplex, 135
multivreg, 133, 172
multivreg (Multivariate linear regression), 128
multivt, 72
multivt (MLE for the multivariate t distribution), 116
Naive Bayes classifiers for compositional data, 136
Naive Bayes classifiers for compositional data using the alpha-transformation, 138
Non linear least squares regression for compositional data, 139
Non parametric zero replacement strategies, 140
ols.compcomp, 44, 196
ols.compcomp (Constrained linear least squares for compositional responses and predictors), 15
ols.compreg, 60, 64, 90, 104, 129, 158, 220
ols.compreg (Non linear least squares regression for compositional data), 139
ols.prop.reg (Distance based regression models for proportions), 61
pcc (Proportionality correlation coefficient matrix), 153
pcr, 14, 65, 132, 200
pcr (Principal component generalised linear models), 147
pcr.tune, 207, 210
pcr.tune (Tuning the principal components with GLMs), 209
perturbation, 141, 144
perturbation (Perturbation operation), 141
Perturbation operation, 141
pivot, 175, 184
pivot (The pivot coordinate transformation and its inverse), 194
pivotinv (The pivot coordinate transformation and its inverse), 194
Plot of the LASSO coefficients, 142
pow (Power operation), 143
power, 142
Power operation, 143
pprcomp, 95, 96, 205
pprcomp (Projection pursuit regression with compositional predictor variables), 152
pprcomp.tune, 96, 153
pprcomp.tune (Tuning of the projection pursuit regression with compositional predictor variables), 204
Principal component analysis, 144
Principal component analysis using the alpha-transformation, 146
Principal component generalised linear models, 147
Principal coordinate analysis using the alpha-distance, 149
Principal coordinate analysis using the Jensen-Shannon divergence, 150
probout (Estimation of the probability left outside the simplex when using the alpha-transformation), 72
profile, 89, 189, 207
Projection pursuit regression for compositional data, 151
Projection pursuit regression with compositional predictor variables, 152
prop.reg, 13
Proportionality correlation coefficient matrix, 153
propreg, 62, 219
propreg (Quasi binomial regression for proportions), 154
propregs (Quasi binomial regression for proportions), 154
Quasi binomial regression for proportions, 154
Random values generation from some univariate distributions defined on the (0,1) interval, 156
rbeta1 (Random values generation from some univariate distributions defined on the (0,1) interval), 156
rcomppnorm, 135, 136
rcomppnorm (Multivariate normal random values simulation on the simplex), 129
rcompsn, 130
rcompsn (Multivariate skew normal random values simulation on the simplex), 134
rcompt, 130
rcompt (Multivariate $t$ random values simulation on the simplex), 135
rda, 160, 209
rda (Regularised discriminant analysis for Euclidean data), 160
rda.tune, 32, 34, 35, 38, 86, 161
rda.tune (Tuning the parameters of the regularised discriminant analysis), 207
rdiri, 55, 111, 113, 120, 121, 130, 135, 136, 157
rdiri (Dirichlet random values simulation), 58
Regression with compositional data using the alpha-transformation, 157
Regularised and flexible discriminant analysis for compositional data using the alpha-transformation, 159
Regularised discriminant analysis for Euclidean data, 160
rfd, 53, 81
rfd (Simulation of compositional data from the Flexible Dirichlet distribution), 169
rfolded, 54, 73
rfolded (Simulation of compositional data from the folded normal distribution), 170
Ridge regression, 162
Ridge regression plot, 163
Ridge regression with compositional data in the covariates side using the
alpha-transformation, 164
Ridge regression with the
alpha-transformation plot, 165
ridge.plot, 163, 166
ridge.plot (Ridge regression plot), 163
ridge.reg, 40, 164, 165
ridge.reg (Ridge regression), 162
ridge.tune, 41, 163, 164
ridge.tune (Cross validation for the
ridge regression), 39
rlogitnorm (Random values generation
from some univariate
distributions defined on the
(0,1) interval), 156
rmixcomp, 83, 85, 114, 116, 169
rmixcomp (Simulation of compositional
data from Gaussian mixture
models), 167
rmixdiri, 57
rmixdiri (Simulation of compositional
data from mixtures of
Dirichlet distributions), 168
rmvt, 117, 135, 136
runItweibull (Random values generation
from some univariate
distributions defined on the
(0,1) interval), 156
score.glms, 156
simplex.est (MLE of distributions
defined in the (0, 1)
interval), 117
Simulation of compositional data from
Gaussian mixture models, 167
Simulation of compositional data from
mixtures of Dirichlet
distributions, 168
Simulation of compositional data from
the Flexible Dirichlet
distribution, 169
Simulation of compositional data from
the folded normal
distribution, 170
skewnorm.contour, 19, 20, 27, 29
skewnorm.contour (Contour plot of the
skew skew-normal distribution
in $S^2$), 27
Spatial median regression, 171
spatmed.reg, 72, 133
spatmed.reg (Spatial median
regression), 171
sym.test, 111
sym.test (Log-likelihood ratio test
for a symmetric Dirichlet
distribution), 112
symkl.compreg (Divergence based
regression for compositional
data), 62
ternary, 174
ternary (Ternary diagram), 172
Ternary diagram, 172
Ternary diagram of regression models,
174
ternary.reg (Ternary diagram of
regression models), 174
tflr, 16, 43, 65
tflr (Transformation-free linear
regression for compositional
responses and predictors), 196
The additive log-ratio transformation
and its inverse, 175
The alpha-distance, 176
The alpha-IT transformation, 177
The alpha-IT-distance, 178
The alpha-k-NN regression for
compositional response data,
179
The alpha-k-NN regression with
compositional predictor
variables, 181
The alpha-kernel regression with
compositional response data,
182
The alpha-transformation, 183
The Box-Cox transformation applied to
ratios of components, 185
The ESOV-distance, 186
The folded power transformation, 187
The Frechet mean for compositional
data, 188
The Helmert sub-matrix, 189
The k-nearest neighbours using the
alpha-distance, 190
The k-NN algorithm for compositional
data, 192
The pivot coordinate transformation
and its inverse, 194
INDEX

Total variability, 195
totvar (Total variability), 195
Transformation-free linear regression for compositional responses and predictors, 196
Tuning of the alpha generalised correlations between two compositional datasets, 197
Tuning of the bandwidth h of the kernel using the maximum likelihood cross validation, 198
Tuning of the divergence based regression for compositional data with compositional data in the covariates side using the alpha-transformation, 199
Tuning of the k-NN algorithm for compositional data, 201
Tuning of the projection pursuit regression for compositional data, 203
Tuning of the projection pursuit regression with compositional predictor variables, 204
Tuning the number of PCs in the PCR with compositional data using the alpha-transformation, 206
Tuning the parameters of the regularised discriminant analysis, 207
Tuning the principal components with GLMs, 209
Tuning the value of alpha in the alpha-regression, 211
tv.compreg (Divergence based regression for compositional data), 62
Two-sample test of high-dimensional means for compositional data, 213

ulc.glm, 106, 107
ulc.glm (Unconstrained log-contrast logistic or Poisson regression with compositional predictor variables), 214
ulc.reg, 9, 10, 108, 110, 218
ulc.reg (Unconstrained log-contrast regression with compositional predictor variables), 215
ulc.reg2, 109, 110, 216
ulc.reg2 (Unconstrained log-contrast regression with multiple compositional predictors), 216
Unconstrained log-contrast logistic or Poisson regression with compositional predictor variables, 214
Unconstrained log-contrast regression with compositional predictor variables, 215
Unconstrained log-contrast regression with multiple compositional predictors, 216
Unit-Weibull regression models for proportions, 218
unitweib.reg (Unit-Weibull regression models for proportions), 218
unitweibull.est (MLE of distributions defined in the (0, 1) interval), 117
univglms, 156
zad.est, 220
zad.est (MLE of the zero adjusted Dirichlet distribution), 123
zadr, 58, 123
zadr (Zero adjusted Dirichlet regression), 219
Zero adjusted Dirichlet regression, 219
zeroreplace, 123
zeroreplace (Non parametric zero replacement strategies), 140
zilogitnorm.est, 123
zilogitnorm.est (MLE of distributions defined in the (0, 1) interval), 117