Package ‘countprop’

May 8, 2026

Type Package

Title Calculate Model-Based Metrics of Proportionality on Count-Based Compositional Data

Version 1.1.1

Maintainer Kevin McGregor <`kevin.mcgregor@umanitoba.ca`>

Description Calculates metrics of proportionality using the logit-normal multinomial model. It can also provide empirical and plugin estimates of these metrics.

License GPL (>= 3)

Encoding UTF-8

LazyData true

Depends R (>= 3.5.0)

Imports glasso, compositions, parallel, zCompositions

RoxygenNote 7.3.3

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Kevin McGregor [aut, cre, cph],
Nneka Okaeme [aut]

Repository CRAN

Date/Publication 2025-12-11 10:10:43 UTC

Contents

`convertSigma` *Convert between CLR and ALR covariance matrices*

Description

Convert between CLR and ALR covariance matrices

Usage

```
convertSigma(S, direction = c("alr2clr", "clr2alr"))
```

Arguments

`S`	Covariance matrix to be converted
`direction`	Which direction to convert between alr and clr.

Value

A covariance matrix on the requested scale.

Examples

```
convertSigma(diag(3), "alr2clr")
```

`ebic` *Extended Bayesian Information Criterion*

Description

Calculates the Extended Bayesian Information Criterion (EBIC) of a model. Used for model selection to asses the fit of the multinomial logit-Normal model which includes a graphical lasso penalty.

Usage

```
ebic(l, n, d, df, gamma)
```

Arguments

`l`	Log-likelihood estimates of the model
`n`	Number of rows of the data set for which the log-likelihood has been calculated
`d`	The size of the (k-1) by (k-1) covariance matrix of a k by k count-compositional data matrix
`df`	Degrees of freedom
`gamma`	A tuning parameter. Larger values means more penalization

Value

The value of the EBIC.

Note

The graphical lasso penalty is the sum of the absolute value of the elements of the covariance matrix `Sigma`. The penalization parameter lambda controls the sparsity of Sigma.

Examples

```
data(singlecell)
mle <- mleLR(singlecell, lambda.gl=0.5)
log.lik_1 <- mle$est[[1]]$log.lik
n <- NROW(singlecell)
k <- NCOL(singlecell)
df_1 <- mle$est[[1]]$df

ebic(log.lik_1, n, k, df_1, 0.1)
```

`ebicPlot` *Extended Bayesian Information Criterion Plot*

Description

Plots the extended Bayesian information criterion (EBIC) of the model fit for various penalization parameters `lambda`.

Usage

```
ebicPlot(fit, xlog = TRUE, col = "darkred")
```

Arguments

`fit`	The model fit object from `mlePath()`
`xlog`	TRUE or FALSE. Renders plot with the x-axis in the log-scale if `TRUE`
`col`	Colour of the plot (character)

Value

Plot of the EBIC (y-axis) against each lambda (x-axis).

Examples

```
data(singlecell)
mle <- mlePath(singlecell, tol=1e-4, tol.nr=1e-4, n.lambda = 2, n.cores = 1)

ebicPlot(mle, xlog = TRUE)
```

`logitNormalVariation` *Logit Normal Variation*

Description

Estimates the variation matrix of count-compositional data based on a multinomial logit-Normal distribution. Estimation is performed using only the parameters of the distribution.

Usage

```
logitNormalVariation(
  mu,
  Sigma,
  type = c("standard", "phi", "phis", "rho"),
  lr = c("alr", "clr"),
  order = c("second", "first")
)
```

Arguments

`mu`	The mle estimate of the mu matrix
`Sigma`	The mle estimate of the Sigma matrix (input Sigma on the ALR scale, even if requesting metrics on the CLR scale)
`type`	Type of variation metric to be calculated: `standard`, `phi`, `phis` (a symmetrical version of `phi`), or `rho`
`lr`	Which scale to calculate the proportionality metric on, either alr or clr.
`order`	Deprecated: The order of the Taylor-series approximation to be used in the estimation

Value

An estimate of the requested metric of proportionality.

Examples

```
data(singlecell)
mle <- mleLR(singlecell)
mu.hat <- mle$mu
Sigma.hat <- mle$Sigma

logitNormalVariation(mu.hat, Sigma.hat)
logitNormalVariation(mu.hat, Sigma.hat, type="phi")
logitNormalVariation(mu.hat, Sigma.hat, type="rho")
```

`logLik` *Log-Likelihood*

Description

Calculates the log-likelihood, under the multinomial logit-Normal model.

Usage

```
logLik(v, y, ni, S, invSigma, lr = c("alr", "clr"))
```

Arguments

`v`	The additive or centered log-ratio transform of y
`y`	Compositional dataset
`ni`	The row sums of y
`S`	Covariance of `v`
`invSigma`	The inverse of the Sigma matrix
`lr`	Which log-ratio transformation to use

Value

The estimated log-likelihood under the Multinomial logit-Normal distribution.

Examples

```
data(singlecell)
mle.sim <- mlePath(singlecell, tol=1e-4, tol.nr=1e-4, n.lambda = 2, n.cores = 1)

n <- NROW(singlecell)


logLik(mle.sim$est.min$v,
      singlecell,
      n,
      cov(mle.sim$est.min$v),
      mle.sim$est.min$Sigma.inv)
```

`logVarTaylorFull` *Full logp Variance-Covariance*

Description

Estimates the variance-covariance of the log of the proportions using a Taylor-series approximation.

Usage

```
logVarTaylorFull(
  mu,
  Sigma,
  transf = c("alr", "clr"),
  order = c("second", "first")
)
```

Arguments

`mu`	The mean vector of the log-ratio-transformed data (ALR or CLR)
`Sigma`	The variance-covariance matrix of the log-ratio-transformed data (ALR or CLR)
`transf`	The desired transformation. If `transf="alr"` the inverse additive log-ratio transformation is applied. If `transf="clr"` the inverse centered log-ratio transformation is applied.
`order`	The desired order of the Taylor Series approximation

Value

The estimated variance-covariance matrix for `log p`.

Examples

```
data(singlecell)
mle <- mleLR(singlecell)
mu <- mle$mu
Sigma <- mle$Sigma

logVarTaylorFull(mu, Sigma)
```

`mleLR` *Maximum Likelihood Estimate for multinomial logit-normal model*

Description

Returns the maximum likelihood estimates of multinomial logit-normal model parameters given a count-compositional dataset. The MLE procedure is based on the multinomial logit-Normal distribution, using the EM algorithm from Hoff (2003).

Usage

```
mleLR(
  y,
  max.iter = 10000,
  max.iter.nr = 100,
  tol = 1e-06,
  tol.nr = 1e-06,
  lambda.gl = 0,
  gamma = 0.1,
  lr.penalty = c("alr", "clr"),
  verbose = FALSE
)
```

Arguments

`y`	Matrix of counts; samples are rows and features are columns.
`max.iter`	Maximum number of iterations
`max.iter.nr`	Maximum number of Newton-Raphson iterations
`tol`	Stopping rule
`tol.nr`	Stopping rule for the Newton-Raphson algorithm
`lambda.gl`	Penalization parameter lambda, for the graphical lasso penalty. Controls the sparsity of Sigma
`gamma`	Gamma value for EBIC calculation of the log-likelihood
`lr.penalty`	Should the precision matrix be penalized on the ALR or CLR scale?
`verbose`	If TRUE, print information as the functions run

Value

The additive log-ratio of `y` (`v`); maximum likelihood estimates of `mu`, `Sigma`, and `Sigma.inv` (e.g. the covariance and precision matrices) on the ALR scale. The CLR scale versions are `Sigma.clr` and `Sigma.inv.clr`, respectively; the log-likelihood (`log.lik`); the EBIC (extended Bayesian information criterion) of the log-likelihood of the multinomial logit-Normal model with the graphical lasso penalty (`ebic`); degrees of freedom of the `Sigma.inv` matrix (`df`).

Note

The graphical lasso penalty is the sum of the absolute value of the elements of the covariance matrix `Sigma`. The penalization parameter lambda controls the sparsity of Sigma.

This function is also used within the `mlePath()` function.

Examples

```
data(singlecell)
mle <- mleLR(singlecell)

mle$mu
mle$Sigma
mle$ebic
```

`mlePath` *Maximum Likelihood Estimator Paths*

Description

Calculates the maximum likelihood estimates of the parameters for the mutlinomial logit-Normal distribution under various values of the penalization parameter `lambda`. Parameter `lambda` controls the sparsity of the covariance matrix `Sigma`, and penalizes the false large correlations that may arise in high-dimensional data.

Usage

```
mlePath(
  y,
  max.iter = 10000,
  max.iter.nr = 100,
  tol = 1e-06,
  tol.nr = 1e-06,
  lambda.gl = NULL,
  lambda.min.ratio = 0.1,
  n.lambda = 1,
  n.cores = 1,
  gamma = 0.1,
  lr.penalty = c("alr", "clr")
)
```

Arguments

`y`	Matrix of counts; samples are rows and features are columns.
`max.iter`	Maximum number of iterations
`max.iter.nr`	Maximum number of Newton-Raphson iterations

`tol`	Stopping rule
`tol.nr`	Stopping rule for the Newton Raphson algorithm
`lambda.gl`	Vector of penalization parameters lambda, for the graphical lasso penalty
`lambda.min.ratio`	Minimum lambda ratio of the maximum lambda, used for the sequence of lambdas
`n.lambda`	Number of lambdas to evaluate the model on
`n.cores`	Number of cores to use (for parallel computation)
`gamma`	Gamma value for EBIC calculation of the log-likelihood
`lr.penalty`	Should the precision matrix be penalized on the ALR or CLR scale?

Value

The MLE estimates of `y` for each element lambda of lambda.gl, (`est`); the value of the estimates which produce the minimum EBIC, (`est.min`); the vector of lambdas used for graphical lasso, (`lambda.gl`); the index of the minimum EBIC (extended Bayesian information criterion), (`min.idx`); vector containing the EBIC for each lambda, (`ebic`).

Note

If using parallel computing, consider setting `n.cores` to be equal to the number of lambdas being evaluated for, `n.lambda`.

The graphical lasso penalty is the sum of the absolute value of the elements of the covariance matrix `Sigma`. The penalization parameter lambda controls the sparsity of Sigma.

Examples

```
data(singlecell)
mle.sim <- mlePath(singlecell, tol=1e-4, tol.nr=1e-4, n.lambda = 2, n.cores = 1)

mu.hat <- mle.sim$est.min$mu
Sigma.hat <- mle.sim$est.min$Sigma
```

`naiveVariation` *Naive (Empirical) Variation*

Description

Naive (empirical) estimates of proportionality metrics using only the observed counts.

Usage

```
naiveVariation(
  counts,
  pseudo.count = 0,
  type = c("standard", "phi", "phis", "rho"),
  lr = c("alr", "clr"),
  impute.zeros = TRUE,
  ...
)
```

Arguments

`counts`	Matrix of counts; samples are rows and features are columns
`pseudo.count`	Positive count to be added to all elements of count matrix.
`type`	Type of variation metric to be calculated: `standard`, `phi`, `phis` (a symmetric version of `phi`), `rho`, or `logp` (the variance-covariance matrix of log-transformed proportions)
`lr`	Which scale to calculate the proportionality metric on, either alr or clr.
`impute.zeros`	If TRUE, then `cmultRepl()` from the `zCompositions` package is used to impute zero values in the counts matrix.
`...`	Optional arguments passed to zero-imputation function `cmultRepl()`

Value

An estimate of the requested metric of proportionality.

Examples

```
#' data(singlecell)

naiveVariation(singlecell)
naiveVariation(singlecell, type="phi")
naiveVariation(singlecell, type="rho")
naiveVariation(singlecell, type="rho", lr="clr")
```

`pluginVariation` *Plugin Variation (Deprecated)*

Description

Estimates the variation matrix of count-compositional data based on a the same approximation used in logitNormalVariation() only for this function it uses empirical estimates of mu and Sigma. Also performs zero-imputation using `cmultRepl()` from the `zCompositions` package.

Usage

```
pluginVariation(
  counts,
  type = c("standard", "phi", "phis", "rho"),
  order = c("second", "first"),
  impute.zeros = TRUE,
  ...
)
```

Arguments

`counts`	Matrix of counts; samples are rows and features are columns.
`type`	Type of variation metric to be calculated: `standard`, `phi`, `phis` (a symmetrical version of `phi`), `rho`, or `logp` (the variance-covariance matrix of log-transformed proportions).
`order`	The order of the Taylor-series approximation to be used in the estimation
`impute.zeros`	If TRUE, then `cmultRepl()` from the `zCompositions` package is used to impute zero values in the counts matrix.
`...`	Optional arguments passed to zero-imputation function `cmultRepl()`

Value

An estimate of the requested metric of proportionality.

Examples

```
data(singlecell)

pluginVariation(singlecell)
pluginVariation(singlecell, type="phi")
pluginVariation(singlecell, type="rho")
```

`singlecell` *Single cell sequencing data from mouse embryonic stem cells in G1 phase*

Description

A subset of single cell data from Buettner et al. 2015. Contains single cell measurements from 96 mouse embryonic stem cells all in G1 phase.

Usage

```
data(singlecell)
```

Format

‘singlecell‘ A matrix with 96 rows and 10 columns.

Source

<https://www.ebi.ac.uk/biostudies/arrayexpress/studies/E-MTAB-2805>

Examples

```
data(singlecell)
```

Index