Package ‘disclapmix’

October 13, 2022

Type Package

Title Discrete Laplace Mixture Inference using the EM Algorithm

Version 1.7.4

Author Mikkel Meyer Andersen [aut, cre],
Poul Svante Eriksen [aut]

Maintainer Mikkel Meyer Andersen <mikl@math.aau.dk>

Description Make inference in a mixture of discrete Laplace distributions using the EM algorithm. This can e.g. be used for modelling the distribution of Y chromosomal haplotypes as described in [1, 2] (refer to the URL section).

License GPL-2 | file LICENSE

LinkingTo Rcpp, RcppProgress

Imports Rcpp (>= 0.11), disclap (>= 1.4), cluster (>= 1.14.4), MASS, stats, graphics, methods, utils

Suggests knitr, ggplot2, gridExtra, ggdendro, scales, seriation, fwsim, testthat, rmarkdown

LazyLoad yes

BugReports https://github.com/mikldk/disclapmix/issues

VignetteBuilder knitr

SystemRequirements C++11

Encoding UTF-8

URL http://dx.doi.org/10.1016/j.jtbi.2013.03.009

https://arxiv.org/abs/1304.2129

RoxygenNote 7.2.0

NeedsCompilation yes

Repository CRAN

Date/Publication 2022-06-29 14:10:02 UTC
Description

Discrete Laplace Mixture Inference using the EM Algorithm. A central function is `disclapmix_adaptive` (and the underlying `disclapmixfit`).

Author(s)

Mikkel Meyer Andersen <mikl@math.aau.dk> and Poul Svante Eriksen <svante@math.aau.dk>
clusterdist  

*Calculate distance between clusters*

**Description**

clusterdist calculates the distance between each pair of clusters. The distance measure is based on a symmetric Kullback-Leibler divergence.

**Usage**

```
clusterdist(fit, ...)
```

**Arguments**

- **fit**  
  A `disclapmixfit` object.
- **...**  
  Not used

**Value**

A distance matrix

**See Also**

disclapmix-package disclapmix disclapmixfit clusterprob predict.disclapmixfit print.disclapmixfit summary.disclapmixfit simulate.disclapmixfit
disclap

---

clusterprob  

*Cluster origin probabilities for haplotypes*

**Description**

clusterprob calculates the cluster origin probabilities for haplotypes.

**Usage**

```
clusterprob(fit, newdata, ...)
```

**Arguments**

- **fit**  
  A `disclapmixfit` object.
- **newdata**  
  The haplotypes to predict the cluster origin probabilities for.
- **...**  
  Not used
contributor_pairs

Value

A matrix where the rows correspond to the rows in newdata and the sum of each row is 1.

See Also

disclapmix-package disclapmix disclapmixfit clusterdist predict.disclapmixfit print.disclapmixfit summary.disclapmixfit simulate.disclapmixfit
disclap

ccontributor_pairs Contributor pairs from a 2 person mixture

Description

Get all possible contributor pairs from a 2 person mixture

Usage

contributor_pairs(mixture)

Arguments

mixture A list of integer vectors. The k’th element in the list is an integer vector with the alleles in the mixture at locus k.

Value

A contrib_pairs object that is an unordered list of pairs. Note, that contributor order is disregarded so that each contributor pair is only present once (and not twice as would be the case if taking order into consideration). See example usage at rank_contributor_pairs.

See Also

rank_contributor_pairs generate_mixture disclapmix-package disclapmix disclapmixfit clusterprob predict.disclapmixfit print.disclapmixfit summary.disclapmixfit simulate.disclapmixfit disclap
**danes**

### Description

185 Y-STR 10 loci haplotypes

### Format

A data frame with 185 observations on the following 10 loci (n is the number of times each haplotype has been observed)

- **DYS19**
- **DYS389I**
- **DYS389II**
- **DYS390**
- **DYS391**
- **DYS392**
- **DYS393**
- **DYS437**
- **DYS438**
- **DYS439**

n

### Source


---

**disclapmix**

### Discrete Laplace mixture inference using the EM algorithm

### Description

disclapmix makes inference in a mixture of Discrete Laplace distributions using the EM algorithm. After the EM algorithm has converged, the centers are moved if the marginal likelihood increases by doing so. And then the EM algorithm is run again. This continues until the centers are not moved.
Usage

disclapmix(
  x,
  clusters,
  init_y = NULL,
  iterations = 100L,
  eps = 0.001,
  verbose = 0L,
  glm_method = "internal_coef",
  glm_control_maxit = 50L,
  glm_control_eps = 1e-06,
  init_y_method = "pam",
  init_v = NULL,
  ret_x = FALSE,
  ...
)

Arguments

x               Dataset.
clusters        The number of clusters/components to fit the model for.
init_y          Initial central haplotypes, if NULL, these will be estimated as described under the init_y_method argument.
iterations      Maximum number of iterations in the EM-algorithm.
eps             Convergence stop criteria in the EM algorithm which is compared to \( \max\{v_{new} - v_{old}\} / \max\{v_{old}\} \), where \( v \) is a matrix of each observation’s probability of belonging to a certain center.
verbose         from 0 to 2 (both including): 0 for silent, 2 for extra verbose.
glm_method      internal_coef, internal_dev or glm.fit. Please see details.
glm_control_maxit Integer giving the maximal number of IWLS iterations.
glm_control_eps Positive convergence tolerance epsilon; the iterations converge when \( |x - x_{old}|/(|x| + 0.1) < \text{epsilon} \), where \( x = \text{beta_correction} \) for internal_coef and \( x = \text{deviance} \) otherwise.
init_y_method   Which cluster method to use for finding initial central haplotypes, y: pam (recommended) or clara. Ignored if init_y is supplied.
init_v          Matrix with ‘nrow(x)’ rows and ‘clusters’ columns specifying initial posterior probabilities to get EM started, if none specified, then ‘matrix(1/clusters, nrow = nrow(x), ncol = clusters)’ is used.
ret_x           Return data ‘x’
...             Used to detect obsolete usage (when using parameters centers, use.parallel, calculate.logLs or plots.prefix).
Details

`glm_method`: `internal_coef` is the fastest as it uses the relative changes in the coefficients as a stopping criterium, hence it does not need to compute the deviance until the very end. In normal situations, it would not be a problem to use this method. `internal_dev` is the reasonably fast method that uses the deviance as a stopping criterium (like `glm.fit`). `glm.fit` to use the traditional `glm.fit` IWLS implementation and is slow compared to the other two methods.

`init_y_method`: For `init_y_method = 'clara'`, the sampling parameters are: samples = 100, sampsize = min(ceiling(nrow(x)/2), 100 + 2*clusters) and the random number generator in R is used.

Value

A `disclapmixfit` object:

- list("glm_method") The supplied GLM method.
- list("init_y") The supplied initial central haplotypes, `init_y`.
- list("init_y_method") The supplied method for choosing initial central haplotypes (only used if `init_y` is NULL).
- list("converged") Whether the estimation converged or not.
- list("x") Dataset used to fit the model if 'ret_x' is 'TRUE', else 'NULL'.
- list("y") The central haplotypes, y.
- list("tau") The prior probabilities of belonging to a cluster, tau.
- list("v_matrix") The matrix v of each observation's probability of belonging to a certain cluster. The rows are in the same order as the observations in x used to generate this fit.
- list("disclap_parameters") A matrix with the estimated dicrete Laplace parameters.
- list("glm_coef") The coefficients from the last GLM fit (used to calculate `disclap_parameters`).
- list("model_observations") Number of observations.
- list("model_parameters") Number of parameters in the model.
- list("iterations") Number of iterations performed in total (including moving centers and re-estimating using the EM algorithm).
- list("logL_full") Full log likelihood of the final model.
- list("logL_marginal") Marginal log likelihood of the final model.
- list("BIC_full") BIC based on the full log likelihood of the final model.
- list("BIC_marginal") BIC based on the marginal log likelihood of the final model.
- list("v_gain_iterations") The gain \(\frac{\max\{v_{new} - v_{old}\}}{\max\{v_{old}\}}\) during the iterations.
- list("tau_iterations") The prior probability of belonging to the centers during the iterations.
- list("logL_full_iterations") Full log likelihood of the models during the iterations (only calculated when `verbose = 2L`).
- list("logL_marginal_iterations") Marginal log likelihood of the models during the iterations (only calculated when `verbose = 2L`).
- list("BIC_full_iterations") BIC based on full log likelihood of the models during the iterations (only calculated when `verbose = 2L`).
- list("BIC_marginal_iterations") BIC based on marginal log likelihood of the models during the iterations (only calculated when `verbose = 2L`).
See Also

disclapmix-package disclapmix disclapmixfit predict.disclapmixfit print.disclapmixfit summary.disclapmixfit simulate.disclapmixfit clusterdist clusterprob glm.fit disclap pam clara

Examples

# Generate sample database
db <- matrix(disclap::rdisclap(1000, 0.3), nrow = 250, ncol = 4)

# Add location parameters
db <- sapply(1:ncol(db), function(i) as.integer(db[, i]+13+i))

head(db)

fit1 <- disclapmix(db, clusters = 1L, verbose = 1L, glm_method = "glm.fit")
fit1$disclap_parameters
fit1$y

fit1b <- disclapmix(db, clusters = 1L, verbose = 1L, glm_method = "internal_coef")
fit1b$disclap_parameters
fit1b$y

max(abs(fit1$disclap_parameters - fit1b$disclap_parameters))

# Generate another type of database

db2 <- matrix(disclap::rdisclap(2000, 0.1), nrow = 500, ncol = 4)
db2 <- sapply(1:ncol(db2), function(i) as.integer(db2[, i]+14+i))

fit2 <- disclapmix(rbind(db, db2), clusters = 2L, verbose = 1L)
fit2$disclap_parameters
fit2$y
fit2$tau

---

Disclapmix_adaptive  Adaptive fitting

Description

A wrapper around `disclapmix_robust()` that instead of fitting one model for a given number of clusters, fits models until the best model (lowest marginal BIC) is in the interior (with margin ‘M’) of all number of clusters tried.

Usage

disclapmix_adaptive(x, margin = 5L, criteria = "BIC_marginal", ...)

**Arguments**

- **x**: Dataset.
- **margin**: Fit models until there is at least this margin
- **criteria**: The slot to chose the best model from (small values indicate better model)
- **...**: Passed on to `disclapmix_robust()` (and further to `disclapmix()`)

**Details**

E.g., the best model has 3 clusters and the margin `M = 5`, then this function ensures that models with 1, 2, ..., 3+5 = 8 clusters are fitted. If e.g. 7 is better than 3, then it continues such that also models with up to 7+5 = 12 clusters are fitted.

Note that models with 1-5 clusters are always fitted.

**Value**

A list of all `disclapmix` fits

**Examples**

```r
data(danes)
db <- as.matrix(danes[rep(1:nrow(danes), danes$n), 1:(ncol(danes)-1)])
fits <- disclapmix_adaptive(db, margin = 5L)
fits
BICs <- sapply(fits, function(x) x$BIC_marginal)
BICs
ks <- sapply(fits, function(x) nrow(x$y)) # Always same as seq_along(fits)
ks
max_k <- max(ks)
best_k <- which.min(BICs)
max_k
best_k
max_k - best_k # = margin = 5
```

---

**disclapmix_robust**

**Robust fitting**

**Description**

A wrapper around `disclapmix()` that tries to avoid errors. Can sometimes avoid errors with SVD problems happening with `glm_method = 'internal_coef'` and `glm_method = 'internal_dev'`.

**Usage**

`disclapmix_robust(x, clusters, ...)`
**generate_mixture**

**Arguments**

- **x**  
  Dataset.
- **clusters**  
  The number of clusters/components to fit the model for.
- **...**  
  Passed on to ‘disclapmix()’

**Examples**

```r
data(danes)
db <- as.matrix(danes[rep(1:nrow(danes), danes$n), 1:(ncol(danes)-1)])
fit <- disclapmix_robust(db, 3L)
fit
```

---

**generate_mixture**

*Generate a mixture*

**Description**

This function can generate a mixture given a list of contributors.

**Usage**

`generate_mixture(profiles)`

**Arguments**

- **profiles**  
  A list with profiles to mix.

**Value**

A list, e.g. for use with `contributor_pairs`. See example usage at `rank_contributor_pairs`.

**See Also**

`contributor_pairs` `rank_contributor_pairs` `disclapmix-package` `disclapmix` `disclapmixfit` `clusterprob` `predict.disclapmixfit` `print.disclapmixfit` `summary.disclapmixfit` `simulate.disclapmixfit` `disclap`
get_rank

Description
Get rank of pair

Usage
get_rank(x, haplotype)

Arguments
x A ranked_contrib_pairs object.
haplotype A haplotype.

haplotype_diversity

Calculate haplotype diversity from a disclapmixfit

Description
Calculate haplotype diversity from a disclapmixfit object. The method is based on simulating a huge database that approximates the population.

Usage
haplotype_diversity(object, nsim = 10000L)

Arguments
object a disclapmixfit object, usually from a result of a call to disclapmix.
nsim number of haplotypes to generate for calculating the haplotype diversity.

Value
The calculated haplotype diversity.

See Also
disclapmix disclapmixfit predict.disclapmixfit print.disclapmixfit summary.disclapmixfit simulate.disclapmixfit
plot.disclapmixfit  

Plot a disclapmixfit

Description

Plot a disclapmixfit object.

Usage

```r
## S3 method for class 'disclapmixfit'
plot(x, which = 1L, clusdist = clusterdist(x), ...)
```

Arguments

- `x` : a disclapmixfit object, usually from a result of a call to disclapmix.
- `which` : What plot to make. 1L = clusters and their distances.
- `clusdist` : To use previously computed cluster distances to avoid doing the same computations twice.
- `...` : not used

Value

A data frame with discrete Laplace distributions for each cluster and locus. Side effect: A plot.

See Also

disclapmix disclapmixfit predict.disclapmixfit print.disclapmixfit simulate.disclapmixfit summary.disclapmixfit

Examples

```r
data(danes)
db <- as.matrix(danes[rep(1:nrow(danes), danes$n), 1:(ncol(danes)-1)])
fit <- disclapmix(db, clusters = 4L)
plot(fit)
```
### plot.ranked_contrib_pairs

**Description**

Plot ranked contributor pairs

**Usage**

```r
## S3 method for class 'ranked_contrib_pairs'
plot(x, top = NULL, ..., xlab = "Rank", ylab = "P(H1)P(H2)")
```

**Arguments**

- **x**: A `ranked_contrib_pairs` object.
- **top**: The top ranked number of pairs to print. `NULL` for all.
- **...**: Delegated to the generic `plot` function.
- **xlab**: Graphical parameter.
- **ylab**: Graphical parameter.

---

### predict.disclapmixfit

**Description**

Is able to predict haplotype frequencies using a `disclapmixfit` object.

**Usage**

```r
## S3 method for class 'disclapmixfit'
predict(object, newdata, ...)
```

**Arguments**

- **object**: a `disclapmixfit` object
- **newdata**: the haplotypes in matrix format to estimate haplotype probabilities for
- **...**: not used

**See Also**

- `disclapmix`
- `disclapmixfit`
- `print.disclapmixfit`
- `summary.disclapmixfit`
- `simulate.disclapmixfit`
- `plot.disclapmixfit`
- `clusterprob`
print.contrib_pairs  Print contributor pairs

Description
Print contributor pairs

Usage
## S3 method for class 'contrib_pairs'
print(x, ...)

Arguments
x  A contrib_pairs object.
...  Ignored

print.disclapmixfit  Print a disclapmixfit

Description
Prints a disclapmixfit object.

Usage
## S3 method for class 'disclapmixfit'
print(x, ...)

Arguments
x  a disclapmixfit object, usually from a result of a call to disclapmix.
...  not used

See Also
disclapmix disclapmixfit predict.disclapmixfit summary.disclapmixfit simulate.disclapmixfit plot.disclapmixfit
print.ranked_contrib_pairs

Print ranked contributor pairs

Description
Print ranked contributor pairs

Usage
## S3 method for class 'ranked_contrib_pairs'
print(x, top = 5L, hide_non_varying_loci = TRUE, ...)

Arguments
x A ranked_contrib_pairs object.
top The top ranked number of pairs to print/plot. NULL for all.
hide_non_varying_loci Whether to hide alleles on loci that do not vary.
... Ignored

rank_contributor_pairs
Separate a 2 person mixture

Description
Separate a 2 person mixture by ranking the possible contributor pairs.

Usage
rank_contributor_pairs(contri_pairs, fit, max_rank = NULL)

Arguments
contri_pairs A contrib_pairs object obtained from contributor_pairs.
fit A disclapmixfit object.
max_rank Not used. Reserved for future use.

Value
A ranked_contrib_pairs object that is basically an order vector and the probabilities for each pair (in the same order as given in contrib_pairs), found by using fit. Note, that contributor order is disregarded so that each contributor pair is only present once (and not twice as would be the case if taking order into consideration).
simulate.disclapmixfit

Simulate from a disclapmixfit

Description

Simulate from a disclapmixfit object.
**Usage**

```r
## S3 method for class 'disclapmixfit'
simulate(object, nsim = 1L, seed = NULL, ...)
```

**Arguments**

- **object**: A `disclapmixfit` object, usually from a result of a call to `disclapmix`.
- **nsim**: Number of haplotypes to generate.
- **seed**: Not used
- **...**: Not used

**Value**

A matrix where the rows correspond to the simulated haplotypes.

**See Also**

- `disclapmix`
- `disclapmixfit`
- `predict.disclapmixfit`
- `print.disclapmixfit`
- `plot.disclapmixfit`
- `summary.disclapmixfit`
- `clusterdist`

---

**Summary of a disclapmixfit**

**Description**

Summary of a `disclapmixfit` object.

**Usage**

```r
## S3 method for class 'disclapmixfit'
summary(object, ...)```

**Arguments**

- **object**: A `disclapmixfit` object, usually from a result of a call to `disclapmix`.
- **...**: Not used

**See Also**

- `disclapmix`
- `disclapmixfit`
- `predict.disclapmixfit`
- `print.disclapmixfit`
- `plot.disclapmixfit`
- `summary.disclapmixfit`
- `clusterdist`
Index

* clusters
  clusterdist, 3
  clusterprob, 3
  disclapmix, 5
* datasets
  danes, 5
* deconvolution
  contributor_pairs, 4
  generate_mixture, 10
  rank_contributor_pairs, 15
* disclapmix
  disclapmix, 5
* distance
  clusterdist, 3
  clusterprob, 3
* eps
  disclapmix, 5
* mixture
  contributor_pairs, 4
  generate_mixture, 10
  rank_contributor_pairs, 15
* plot
  plot.disclapmixfit, 12
* predict
  predict.disclapmixfit, 13
* print
  haplotype_diversity, 11
  print.disclapmixfit, 14
  simulate.disclapmixfit, 16
  summary.disclapmixfit, 17
* separation
  contributor_pairs, 4
  generate_mixture, 10
  rank_contributor_pairs, 15

calla, 8
clusterdist, 3, 4, 8, 17
clusterprob, 3, 4, 8, 10, 13, 16
contributor_pairs, 4, 10, 15, 16
danes, 5
disclap, 3, 4, 8, 10, 16
disclapmix, 3, 4, 5, 8, 10–14, 16, 17
disclapmix-package, 2
disclapmix_adaptive, 2, 8
disclapmix_robust, 9
disclapmixfit, 2–4, 7, 8, 10–17
disclapmixfit(disclapmix), 5
generate_mixture, 4, 10, 16
get_rank, 11
glm.fit, 8
haplotype_diversity, 11
pam, 8
plot, 13
plot.disclapmixfit, 12, 13, 14, 17
plot.ranked_contrib_pairs, 15
predict.disclapmixfit, 3, 4, 8, 10–12, 13,
  14, 16, 17
print.contrib_pairs, 14
print.disclapmixfit, 3, 4, 8, 10–13, 14, 16,
  17
print.ranked_contrib_pairs, 15
rank_contributor_pairs, 4, 10, 15
simulate.disclapmixfit, 3, 4, 8, 10–14, 16,
  16, 17
summary.disclapmixfit, 3, 4, 8, 10–14, 16,
  17, 17