Package 'PLMIX'

September 4, 2019

Type Package

Title Bayesian Analysis of Finite Mixtures of Plackett-Luce Models for Partial Rankings/Orderings

Version 2.1.1

Date 2019-09-04

Description Fit finite mixtures of Plackett-Luce models for partial top rankings/orderings within the Bayesian framework. It provides MAP point estimates via EM algorithm and posterior MCMC simulations via Gibbs Sampling. It also fits MLE as a special case of the noninformative Bayesian analysis with vague priors. In addition to inferential techniques, the package assists other fundamental phases of a model-based analysis for partial rankings/orderings, by including functions for data manipulation, simulation, descriptive summary, model selection and goodness-of-fit evaluation. Main references on the methods are Mollica and Tardella (2017) <doi.org/10.1007/s11336-016-9530-0> and Mollica and Tardella (2014) <doi/10.1002/sim.6224>.

License GPL (>= 2)

Imports Rcpp (>= 1.0.0), abind (>= 1.4-5), foreach (>= 1.4.4), ggplot2 (>= 2.2.1), ggmcmc (>= 1.2), coda (>= 0.19-1), reshape2 (>= 1.4.3), rcdd (>= 1.2), gridExtra (>= 2.3), MCMCpack (>= 1.4-2), gtools (>= 3.8.1), label.switching (>= 1.6), radarchart (>= 0.3.1), prefmod (>= 0.8-34), rankdist (>= 1.1.3), StatRank (>= 0.0.6), pmr (>= 1.2.5), PlackettLuce (>= 0.2-3), stats, utils

LinkingTo Rcpp

RoxygenNote 6.1.0

Suggests doParallel

LazyData true

NeedsCompilation yes

Author Cristina Mollica [aut, cre],
Luca Tardella [aut]

Maintainer Cristina Mollica <cristina.mollica@uniroma1.it>

Repository CRAN

Date/Publication 2019-09-04 11:50:02 UTC

R topics documented:

PLMIX-package 3
as.top_ordering 5
bicPLMIX 7
binary_group_ind 9
d_apa 9
d_carconf 10
d_dublinwest 11
d_gaming 12
d_german 13
d_nascar 14
d_occup 15
d_rice 16
freq_to_unit 16
gibbsPLMIX 17
gsPLMIX_to_mcmc 19
is.top_ordering 20
label_switchPLMIX 21
Loglikelihood 23
make_complete 25
make_partial 26
mapPLMIX 27
mapPLMIX_multistart 29
paired_comparisons 31
plot.gsPLMIX 32
plot.mpPLMIX 34
ppcheckPLMIX 36
ppcheckPLMIX_cond 38
print.gsPLMIX 40
print.mpPLMIX 40
rank_ord_switch 41
rank_summaries 42
rPLMIX 43
selectPLMIX 45
summary.gsPLMIX 47
summary.mpPLMIX 49
unit_to_freq 50

Index **51**

PLMIX-package — *Bayesian Analysis of Finite Mixtures of Plackett-Luce Models for Partial Rankings/Orderings*

Description

The **PLMIX** package for R provides functions to fit and analyze finite mixtures of Plackett-Luce models for partial top rankings/orderings within the Bayesian framework. It provides MAP point estimates via EM algorithm and posterior MCMC simulations via Gibbs Sampling. It also fits MLE as a special case of the noninformative Bayesian analysis with vague priors.

In addition to inferential techniques, the package assists other fundamental phases of a model-based analysis for partial rankings/orderings, by including functions for data manipulation, simulation, descriptive summary, model selection and goodness-of-fit evaluation.

Specific S3 classes and methods are also supplied to enhance the usability and foster exchange with other packages. Finally, to address the issue of computationally demanding procedures typical in ranking data analysis, **PLMIX** takes advantage of a hybrid code linking the R environment with the C++ programming language.

Details

The Plackett-Luce model is one of the most popular and frequently applied parametric distributions to analyze partial top rankings/orderings of a finite set of items. The present package allows to account for unobserved sample heterogeneity of partially ranked data with a model-based analysis relying on Bayesian finite mixtures of Plackett-Luce models. The package provides a suite of functions that covers the fundamental phases of a model-based analysis:

Ranking data manipulation

`binary_group_ind` Binary group membership matrix from the mixture component labels.

`freq_to_unit` From the frequency distribution to the dataset of individual orderings/rankings.

`make_complete` Random completion of partial orderings/rankings data.

`make_partial` Censoring of complete orderings/rankings data.

`rank_ord_switch` From rankings to orderings and vice-versa.

`unit_to_freq` From the dataset of individual orderings/rankings to the frequency distribution.

Ranking data simulation

`rPLMIX` Random sample from a finite mixture of Plackett-Luce models.

Ranking data description

`paired_comparisons` Paired comparison frequencies.

`rank_summaries` Summary statistics of partial ranking/ordering data.

Model estimation

`gibbsPLMIX` Bayesian analysis with MCMC posterior simulation via Gibbs sampling.

`label_switchPLMIX` Label switching adjustment of the Gibbs sampling simulations.

`likPLMIX` Likelihood evaluation for a mixture of Plackett-Luce models.

`loglikPLMIX` Log-likelihood evaluation for a mixture of Plackett-Luce models.

`mapPLMIX` MAP estimation via EM algorithm.

`mapPLMIX_multistart` MAP estimation via EM algorithm with multiple starting values.

Class coercion and membership

`as.top_ordering` Coercion into top-ordering datasets.

`gsPLMIX_to_mcmc` From the Gibbs sampling simulation to an MCMC class object.

`is.top_ordering` Test for the consistency of input data with a top-ordering dataset.

S3 class methods

`plot.gsPLMIX` Plot of the Gibbs sampling simulations.

`plot.mpPLMIX` Plot of the MAP estimates.

`print.gsPLMIX` Print of the Gibbs sampling simulations.

`print.mpPLMIX` Print of the MAP estimation algorithm.

`summary.gsPLMIX` Summary of the Gibbs sampling procedure.

`summary.mpPLMIX` Summary of the MAP estimation.

Model selection

`bicPLMIX` BIC value for the MLE of a mixture of Plackett-Luce models.

`selectPLMIX` Bayesian model selection criteria.

Model assessment

`ppcheckPLMIX` Posterior predictive diagnostics.

`ppcheckPLMIX_cond` Posterior predictive diagnostics conditionally on the number of ranked items.

Datasets

`d_apa` American Psychological Association Data (partial orderings).

`d_carconf` Car Configurator Data (partial orderings).

`d_dublinwest` Dublin West Data (partial orderings).

`d_gaming` Gaming Platforms Data (complete orderings).

`d_german` German Sample Data (complete orderings).

`d_nascar` NASCAR Data (partial orderings).

`d_occup` Occupation Data (complete orderings).

`d_rice` Rice Voting Data (partial orderings).

Data have to be supplied as an object of class `matrix`, where missing positions/items are denoted with zero entries and Rank = 1 indicates the most-liked alternative. For a more efficient implementation of the methods, partial sequences with a single missing entry should be preliminarily filled in, as they correspond to complete rankings/orderings. In the present setting, ties are not allowed. Some quantities frequently recalled in the manual are the following:

N	Sample size.
K	Number of possible items.
G	Number of mixture components.
L	Size of the final posterior MCMC sample (after burn-in phase).

Author(s)

Cristina Mollica and Luca Tardella

Maintainer: Cristina Mollica <cristina.mollica@uniroma1.it>

References

Mollica, C. and Tardella, L. (2017). Bayesian Plackett-Luce mixture models for partially ranked data. *Psychometrika*, **82**(2), pages 442–458, ISSN: 0033-3123, `http://dx.doi.org/10.1007/s11336-016-9530-0`.

Mollica, C. and Tardella, L. (2014). Epitope profiling via mixture modeling for ranked data. *Statistics in Medicine*, **33**(21), pages 3738–3758, ISSN: 0277-6715, `http://onlinelibrary.wiley.com/doi/10.1002/sim.6224/full`.

`as.top_ordering` *Coercion into top-ordering datasets*

Description

Attempt to coerce the input data into a top-ordering dataset.

Usage

```
as.top_ordering(data, format_input = NULL, aggr = NULL,
  freq_col = NULL, ties_method = "random", ...)
```

Arguments

`data`	An object containing the partial sequences to be coerced into an object of class `top_ordering`. The following classes are admissible for `data`: numeric `matrix`, `data.frame`, `RandData` from the `rankdist` package and `rankings` from the `PlackettLuce` package.
`format_input`	Character string indicating the format of the `data` input, namely `"ordering"` or `"ranking"`. Used only when the class of the `data` argument is matrix or data frame. Default is `NULL`.
`aggr`	Logical: whether the `data` argument collects the distinct observed sequences with the corresponding frequencies (aggregated format). Used only when the class of the `data` aargument is matrix or data frame. Default is `NULL`.
`freq_col`	Integer indicating the column of the `data` argument containing the frequencies of the distinct observed sequences. Used only when the class of the `data` argument is matrix or data frame and `aggr` argument is `TRUE`. Default is `NULL`.

`ties_method`	Character string indicating the treatment of sequences with ties (not used for data of class `RankData`). If `"remove"`, the sequences with ties are removed before acting the coercion; if `"random"` (default), tied positions are re-assigned at random before acting the coercion.
`...`	Further arguments passed to or from other methods (not used).

Details

The coercion function `as.top_ordering` tries to coerce the input data into an object of class `top_ordering` after checking for possible partial sequences that do not satisfy the top-ordering requirements. If none of the supplied sequences satisfies the top-ordering conditions, an error message is returned. `NA`'s in the input `data` are tacitly converted into zero entries.

Value

An object of S3 class `c("top_ordering","matrix")`.

Author(s)

Cristina Mollica and Luca Tardella

References

Turner, H., Kormidis, I. and Firth, D. (2018). PlackettLuce: Plackett-Luce Models for Rankings. R package version 0.2-3. https://CRAN.R-project.org/package=PlackettLuce

Qian, Z. (2018). rankdist: Distance Based Ranking Models. R package version 1.1.3. https://CRAN.R-project.org/package=rankdist

See Also

`is.top_ordering`, `as.rankings` and `rankings`

Examples

```
## Coerce an object of class 'rankings' into an object of class 'top_ordering'
library(PlackettLuce)
RR <- matrix(c(1, 2, 0, 0,
4, 1, 2, 3,
2, 1, 1, 1,
1, 2, 3, 0,
2, 1, 1, 0,
1, 0, 3, 2), nrow = 6, byrow = TRUE)
RR_rank=as.rankings(RR)
RR_rank
as.top_ordering(RR_rank, ties_method="random")

## Coerce an object of class 'RankData' into an object of class 'top_ordering'
library(rankdist)
data(apa_partial_obj)
d_apa_top_ord=as.top_ordering(data=apa_partial_obj)
```

```
identical(d_apa,d_apa_top_ord)

## Coerce a data frame from the package prefmod into an object of class 'top_ordering'
library(prefmod)
data(carconf)
carconf_rank=carconf[,1:6]
carconf_top_ord=as.top_ordering(data=carconf_rank,format_input="ranking",aggr=FALSE)
identical(d_carconf,carconf_top_ord)

## Coerce a data frame from the package pmr into an object of class 'top_ordering'
library(pmr)
data(big4)
head(big4)
big4_top_ord=as.top_ordering(data=big4,format_input="ranking",aggr=TRUE,freq_col=5)
head(big4_top_ord)
```

---

bicPLMIX *BIC for the MLE of a mixture of Plackett-Luce models*

Description

Compute BIC value for the MLE of a mixture of Plackett-Luce models fitted to partial orderings.

Usage

```
bicPLMIX(max_log_lik, pi_inv, G, ref_known = TRUE, ref_vary = FALSE)
```

Arguments

`max_log_lik`	Maximized log-likelihood value.
`pi_inv`	An object of class `top_ordering`, collecting the numeric $N \times K$ data matrix of partial orderings, or an object that can be coerced with `as.top_ordering`.
`G`	Number of mixture components.
`ref_known`	Logical: whether the component-specific reference orders are known (not to be estimated). Default is `TRUE`.
`ref_vary`	Logical: whether the reference orders vary across mixture components. Default is `FALSE`.

Details

The `max_log_lik` and the BIC values can be straightforwardly obtained from the output of the `mapPLMIX` and `mapPLMIX_multistart` functions when the default noninformative priors are adopted in the MAP procedure. So, the `bicPLMIX` function is especially useful to compute the BIC value from the output of alternative MLE methods for mixtures of Plackett-Luce models implemented, for example, with other softwares.

The `ref_known` and `ref_vary` arguments accommodate for the more general mixture of Extended Plackett-Luce models (EPL), involving the additional reference order parameters (Mollica and Tardella 2014). Since the Plackett-Luce model is a special instance of the EPL with the reference order equal to the identity permutation $(1, \ldots, K)$, the default values of `ref_known` and `ref_vary` are set equal, respectively, to `TRUE` and `FALSE`.

Value

A list of two named objects:

`max_log_lik`	The `max_log_lik` argument.
`bic`	BIC value.

Author(s)

Cristina Mollica and Luca Tardella

References

Mollica, C. and Tardella, L. (2017). Bayesian Plackett-Luce mixture models for partially ranked data. *Psychometrika*, **82**(2), pages 442–458, ISSN: 0033-3123, DOI: 10.1007/s11336-016-9530-0.

Mollica, C. and Tardella, L. (2014). Epitope profiling via mixture modeling for ranked data. *Statistics in Medicine*, **33**(21), pages 3738–3758, ISSN: 0277-6715, DOI: 10.1002/sim.6224.

Schwarz, G. (1978). Estimating the dimension of a model. *Ann. Statist.*, **6**(2), pages 461–464, ISSN: 0090-5364, DOI: 10.1002/sim.6224.

See Also

`mapPLMIX` and `mapPLMIX_multistart`

Examples

```
data(d_carconf)
K <- ncol(d_carconf)
MAP_mult <- mapPLMIX_multistart(pi_inv=d_carconf, K=K, G=3, n_start=2, n_iter=400*3)
bicPLMIX(max_log_lik=MAP_mult$mod$max_objective, pi_inv=d_carconf, G=3)$bic

## Equivalently
MAP_mult$mod$bic
```

`binary_group_ind` *Binary group membership matrix*

Description

Construct the binary group membership matrix from the multinomial classification vector.

Usage

```
binary_group_ind(class, G)
```

Arguments

`class`	Numeric vector of class memberships.
`G`	Number of possible different classes.

Value

Numeric `length(class)`$\times G$ matrix of binary group memberships.

Author(s)

Cristina Mollica and Luca Tardella

Examples

```
binary_group_ind(class=c(3,1,5), G=6)
```

`d_apa` *American Psychological Association Data (partial orderings)*

Description

The popular American Psychological Association dataset (`d_apa`) contains the results of the voting ballots of the 1980 presidential election. A total of $N = 15449$ voters ranked a maximum of $K = 5$ candidates, conventionally classified as research psychologists (candidate 1 and 3), clinical psychologists (candidate 4 and 5) and community psychologists (candidate 2). The winner of the election was candidate 3. The dataset is composed of partial top orderings of varying lengths. Missing positions are denoted with zero entries.

Usage

```
data(d_apa)
```

Format

Object of S3 class c("top_ordering","matrix") gathering a matrix of partial orderings with $N = 15449$ rows and $K = 5$ columns Each row lists the candidates from the most-liked (Rank_1) to the least-liked (Rank_5) in a given voting ballot.

References

Mollica, C. and Tardella, L. (2017). Bayesian Plackett-Luce mixture models for partially ranked data. *Psychometrika*, **82**(2), pages 442–258, ISSN: 0033-3123, DOI: 10.1007/s11336-016-9530-0.

Diaconis, P. W. (1988). Group representations in probability and statistics. *Lecture Notes-Monograph Series*, pages 94–96.

Diaconis, P. W. (1987). Spectral analysis for ranked data. Technical Report 282, Dept of Statistics, Stanford University.

Examples

```
data(d_apa)
head(d_apa)

## Subset of complete sequences
d_apa_compl=d_apa[rowSums(d_apa!=0)>=(ncol(d_apa)-1),]
head(d_apa_compl)
```

---

d_carconf *Car Configurator Data (partial orderings)*

Description

The Car Configurator dataset (d_carconf) came up from a marketing study aimed at investigating customer preferences toward different car features. A sample of $N = 435$ customers were asked to construct their car by using an online configurator system and choose among $K = 6$ car modules in order of preference. The car features are labeled as: 1 = price, 2 = exterior design, 3 = brand, 4 = technical equipment, 5 = producing country and 6 = interior design. The survey did not require a complete ranking elicitation, therefore the dataset is composed of partial top orderings of varying lengths. Missing positions are denoted with zero entries.

Usage

```
data(d_carconf)
```

Format

Object of S3 class c("top_ordering","matrix") gathering a matrix of partial orderings with $N = 435$ rows and $K = 6$ columns. Each row lists the car features from the most important (Rank_1) to the least important (Rank_6) for a given customer.

References

Mollica, C. and Tardella, L. (2017). Bayesian Plackett-Luce mixture models for partially ranked data. *Psychometrika*, **82**(2), pages 442–458, ISSN: 0033-3123, DOI: 10.1007/s11336-016-9530-0.

Hatzinger, R. and Dittrich, R. (2012). Prefmod: An R package for modeling preferences based on paired comparisons, rankings, or ratings. *Journal of Statistical Software*, **48**(10), pages 1–31.

Dabic, M. and Hatzinger, R. (2009). Zielgruppenadaequate Ablaeufe in Konfigurationssystemen - eine empirische Studie im Automobilmarkt - Partial Rankings. In Hatzinger, R., Dittrich, R. and Salzberger, T. (eds), *Praeferenzanalyse mit R: Anwendungen aus Marketing, Behavioural Finance und Human Resource Management*. Wien: Facultas.

Examples

```
data(d_carconf)
head(d_carconf)

## Subset of complete sequences
d_carconf_compl=d_carconf[rowSums(d_carconf!=0)>=(ncol(d_carconf)-1),]
head(d_carconf_compl)
```

`d_dublinwest` *Dublin West Data (partial orderings)*

Description

The Dublin West dataset (`d_dublinwest`) contains the results of the voting ballots of the 2002 Irish general election from the Dublin West constituency. The Irish voting system allows voters to rank the candidates in order of preferences, rather than only specify the favorite one. In the Dublin West constituency, $N = 29988$ voters ranked a maximum of $K = 9$ candidates, labeled as: 1 = Bonnie R., 2 = Burton J., 3 = Doherty-Ryan D., 4 = Higgins J., 5 = Lenihan B., 6 = McDonald M., 7 = Morrissey T., 8 = Smyth J. and 9 = Terry S.. The dataset is composed of partial top orderings of varying lengths. Missing positions are denoted with zero entries.

Usage

```
data(d_dublinwest)
```

Format

Object of S3 class `c("top_ordering","matrix")` gathering a partial orderings with $N = 29988$ rows and $K = 9$ columns. Each row lists the candidates from the most-liked (`Rank_1`) to the least-liked (`Rank_9`) in a given voting ballot.

Source

The 2002 Dublin West data have been downloaded from `http://www.preflib.org/` PrefLib: A Library for Preferences. In that repository, preferences with ties are also included. The original source was publicly available from the Dublin County Returning Officer at the following URL: `https://dublincountyreturningofficer.com/`.

References

Mattei, N. and Walsh, T. (2013) PrefLib: A Library of Preference Data. *Proceedings of Third International Conference on Algorithmic Decision Theory* (ADT 2013). Springer, Lecture Notes in Artificial Intelligence, November 13-15, 2013.

Gormley, I. C. and Murphy, T. B. (2009). A grade of membership model for rank data. *Bayesian Analysis*, **4**(2), pages 65–295.

Gormley, I. C. and Murphy, T. B. (2008). Exploring Voting Blocs Within the Irish Electorate: A Mixture Modeling Approach. *Journal of the America Statistical Association*, **103**(483), pages 1014–1027.

Examples

```
data(d_dublinwest)
head(d_dublinwest)

## Subset of complete sequences
d_dublinwest_compl=d_dublinwest[rowSums(d_dublinwest!=0)>=(ncol(d_dublinwest)-1),]
head(d_dublinwest_compl)
```

---

d_gaming *Gaming Platforms Data (complete orderings)*

Description

The Gaming Platforms dataset (d_gaming) collects the results of a survey conducted on a sample of $N = 91$ Dutch students, who were asked to rank $K = 6$ gaming platforms in order of preference, namely: 1 = X-Box, 2 = PlayStation, 3 = PSPortable, 4 = GameCube, 5 = GameBoy and 6 = Personal Computer. The dataset is composed of complete orderings.

Usage

```
data(d_gaming)
```

Format

Object of S3 class c("top_ordering","matrix") gathering a matrix of complete orderings with $N = 91$ rows and $K = 6$ columns. Each row lists the gaming platforms from the most-liked (Rank_1) to the least-liked (Rank_6) for a given student.

Source

The Gaming Platforms dataset in .csv format can be downloaded from http://qed.econ.queensu.ca/jae/2012-v27.5/fok-paap-van_dijk/. The .csv files contains the preference data in ranking format and some covariates collected for each student.

References

Fok, D., Paap, R. and Van Dijk, B. (2012). A Rank-Ordered Logit Model With Unobserved Heterogeneity In Ranking Capatibilities. *Journal of Applied Econometrics*, **27**(5), pages 831–846.

Examples

```
data(d_gaming)
head(d_gaming)
```

`d_german` *German Sample Data (complete orderings)*

Description

The German Sample dataset (`d_german`) is part of a comparative cross-sectional study on political actions and mass participation involving five Western countries. The dataset regards a sample of $N = 2262$ German respondents who were asked to rank $K = 4$ political goals in order of desirability, namely: 1 = maintaining order in the nation, 2 = giving people more say in the decisions of government, 3 = fighting rising prices and 4 = protecting freedom of speech. The dataset is composed of complete orderings.

Usage

```
data(d_german)
```

Format

Object of S3 class `c("top_ordering","matrix")` gathering a matrix of complete orderings with $N = 2262$ rows and $K = 4$ columns. Each row lists the political goals from the most desiderable (`Rank_1`) to the least desiderable (`Rank_4`) for a given respondent.

References

Croon, M. A. (1989). Latent class models for the analysis of rankings. In De Soete, G., Feger, H. and Klauer, K. C. (eds), *New Developments in Psychological Choice Modeling*, pages 99–121. North-Holland: Amsterdam.

Barnes, S. H. et al. (1979). Political action. Mass participation in five Western democracies. London: Sage.

Examples

```
data(d_german)
head(d_german)
```

d_nascar *NASCAR Data (partial orderings)*

Description

The NASCAR dataset (d_nascar) collects the results of the 2002 season of stock car racing held in the United States. The 2002 championship consisted of $N = 36$ races, with 43 car drivers competing in each race. A total of $K = 87$ drivers participated in the 2002 season, taking part to a variable number of races: some of them competed in all the races, some others in only one. The results of the entire 2002 season were collected in the form of top-43 orderings, where the position of the not-competing drivers in each race is assumed lower than the 43th, but undetermined. Missing positions are denoted with zero entries.

Usage

```
data(d_nascar)
```

Format

Object of S3 class c("top_ordering","matrix") gathering a matrix of partial orderings with $N = 36$ rows and $K = 87$ columns. Each row lists the car drivers from the top position (Rank_1) to the bottom one (Rank_87) in a given race. Columns from the 44th to the 87th are filled with zeros, because only 43 drivers competed in each race.

Source

The NASCAR dataset in the MATLAB format used by Hunter, D. R. (2004) can be downloaded from http://sites.stat.psu.edu/~dhunter/code/btmatlab/. At the same link, a .xls file with drivers' names is also available.

References

Caron, F. and Doucet, A. (2012). Efficient Bayesian inference for Generalized Bradley-Terry models. *J. Comput. Graph. Statist.*, **21**(1), pages 174–196.

Guiver, J. and Snelson, E. (2009). Bayesian inference for Plackett-Luce ranking models. In Bottou, L. and Littman, M., editors, *Proceedings of the 26th International Conference on Machine Learning - ICML 2009*, pages 377–384. Omnipress.

Hunter, D. R. (2004). MM algorithms for Generalized Bradley-Terry models. *Ann. Statist.*, **32**(1), pages 384–406.

Examples

```
data(d_nascar)
head(d_nascar)

## Compute the number of races for each of the 87 drivers
table(c(d_nascar[,1:43]))
```

```
## Identify drivers arrived last (43th position) in all the races
which(colSums(rank_summaries(d_nascar, format="ordering")$marginals[1:42,])==0)

## Obscure drivers 84, 85, 86 and 87 to get the reduced dataset
## with 83 racers employed by Hunter, D. R. (2004)
d_nascar_hunter=d_nascar[,1:83]
d_nascar_hunter[is.element(d_nascar_hunter,84:87)]=0
```

d_occup — *Occupation Data (complete orderings)*

Description

The Occupation dataset (`d_occup`) came up from a survey conducted on graduates from the Technion-Insrael Institute of Tecnology. A sample of $N = 143$ graduates were asked to rank $K = 10$ professions according to the perceived prestige. The occupations are labeled as: 1 = faculty member, 2 = owner of a business, 3 = applied scientist, 4 = operations researcher, 5 = industrial engineer, 6 = manager, 7 = mechanical engineer, 8 = supervisor, 9 = technician and 10 = foreman. The dataset is composed of complete orderings.

Usage

```
data(d_occup)
```

Format

Object of S3 class `c("top_ordering","matrix")` gathering a matrix of complete orderings with $N = 143$ rows and $K = 10$ columns. Each row lists the professions from the most-liked (`Rank_1`) to the least-liked (`Rank_10`) for a given graduate.

References

Cohen, A. and Mallows, C. L. (1983). Assessing goodness of fit of ranking models to data. *Journal of the Royal Statistical Society: Series D (The Statistician)*, **32**(4), pages 361–374, ISSN: 0039-0526.

Cohen, A. (1982). Analysis of large sets of ranking data. *Communications in Statistics – Theory and Methods*, **11**(3), pages 235–256.

Goldberg, A. I. (1976). The relevance of cosmopolitan/local orientations to professional values and behavior. *Sociology of Work and Occupations*, **3**(3), pages 331–356.

Examples

```
data(d_occup)
head(d_occup)
```

d_rice *Rice Voting Data (partial orderings)*

Description

The Rice Voting dataset (d_rice) collects the results of the 1992 election of a faculty member to serve on the Presidential Search Committee in the Rice University. A total of $N = 300$ people casted their vote in the ballots by ranking the $K = 5$ candidates in the short list in a preferential manner. The dataset is composed of partial top orderings of varying lengths. Missing positions are denoted with zero entries.

Usage

```
data(d_rice)
```

Format

Object of S3 class c("top_ordering","matrix") gathering a matrix of partial orderings with $N = 300$ rows and $K = 5$ columns. Each row lists the faculty members from the most-liked (Rank_1) to the least-liked (Rank_5) in a given voting ballot.

References

Marcus, P., Heiser, W. J. and D'Ambrosio, A. (2013). Comparison of heterogeneous probability models for ranking data, Master Thesis, Leiden University.

Baggerly, K. A. (1995). Visual estimation of structure in ranked data, PhD thesis, Rice University.

Examples

```
data(d_rice)
head(d_rice)

## Subset of complete sequences
d_rice_compl=d_rice[rowSums(d_rice!=0)>=(ncol(d_rice)-1),]
head(d_rice_compl)
```

freq_to_unit *Individual rankings/orderings from the frequency distribution*

Description

Construct the dataset of individual rankings/orderings from the frequency distribution of the distinct observed sequences.

Usage

```
freq_to_unit(freq_distr)
```

Arguments

`freq_distr`	Numeric matrix of the distinct observed sequences with the corresponding frequencies indicated in the last $(K+1)$-th column.

Value

Numeric $N \times K$ data matrix of observed individual sequences.

Author(s)

Cristina Mollica and Luca Tardella

Examples

```
library(gtools)
K <- 4
perm_matrix <- permutations(n=K, r=K)
freq_data <- cbind(perm_matrix, sample(1:factorial(K)))
freq_data
freq_to_unit(freq_distr=freq_data)
```

`gibbsPLMIX` *Gibbs sampling for a Bayesian mixture of Plackett-Luce models*

Description

Perform Gibbs sampling simulation for a Bayesian mixture of Plackett-Luce models fitted to partial orderings.

Usage

```
gibbsPLMIX(pi_inv, K, G, init = list(z = NULL, p = NULL),
  n_iter = 1000, n_burn = 500, hyper = list(shape0 = matrix(1, nrow =
  G, ncol = K), rate0 = rep(0.001, G), alpha0 = rep(1, G)),
  centered_start = FALSE)
```

Arguments

`pi_inv`	An object of class `top_ordering`, collecting the numeric $N \times K$ data matrix of partial orderings, or an object that can be coerced with `as.top_ordering`.
`K`	Number of possible items.
`G`	Number of mixture components.
`init`	List of named objects with initialization values: `z` is a numeric $N \times G$ matrix of binary mixture component memberships; `p` is a numeric $G \times K$ matrix of component-specific support parameters. If starting values are not supplied (`NULL`), they are randomly generated with a uniform distribution. Default is `NULL`.
`n_iter`	Total number of MCMC iterations.
`n_burn`	Number of initial burn-in drawings removed from the returned MCMC sample.
`hyper`	List of named objects with hyperparameter values for the conjugate prior specification: `shape0` is a numeric $G \times K$ matrix of shape hyperparameters; `rate0` is a numeric vector of G rate hyperparameters; `alpha0` is a numeric vector of G Dirichlet hyperparameters. Default is vague prior setting.
`centered_start`	Logical: whether a random start whose support parameters and weights should be centered around the observed relative frequency that each item has been ranked top. Default is `FALSE`. Ignored when `init` is not `NULL`.

Details

The size L of the final MCMC sample is equal to `n_iter`-`n_burn`.

Value

A list of S3 class `gsPLMIX` with named elements:

`W`	Numeric $L \times G$ matrix with MCMC samples of the mixture weights.
`P`	Numeric $L \times (G * K)$ matrix with MCMC samples of the component-specific support parameters.
`log_lik`	Numeric vector of L posterior log-likelihood values.
`deviance`	Numeric vector of L posterior deviance values ($-2*$`log_lik`).
`objective`	Numeric vector of L objective function values (that is the kernel of the log-posterior distribution).
`call`	The matched call.

Author(s)

Cristina Mollica and Luca Tardella

References

Mollica, C. and Tardella, L. (2017). Bayesian Plackett-Luce mixture models for partially ranked data. *Psychometrika*, **82**(2), pages 442–458, ISSN: 0033-3123, DOI: 10.1007/s11336-016-9530-0.

Examples

```
data(d_carconf)
GIBBS <- gibbsPLMIX(pi_inv=d_carconf, K=ncol(d_carconf), G=3, n_iter=30, n_burn=10)
str(GIBBS)
GIBBS$P
GIBBS$W
```

`gsPLMIX_to_mcmc` *MCMC class objects from the Gibbs sampling simulations of a Bayesian mixture of Plackett-Luce models*

Description

Coerce the Gibbs sampling simulations for a Bayesian mixture of Plackett-Luce models into an `mcmc` class object.

Usage

```
gsPLMIX_to_mcmc(gsPLMIX_out)
```

Arguments

`gsPLMIX_out`	Object of class `gsPLMIX` returned by the `gibbsPLMIX` function.

Details

`gsPLMIX_to_mcmc` attemps to coerce its argument by recalling the `as.mcmc` function of the `coda` package.

Value

An `mcmc` class object.

Author(s)

Cristina Mollica and Luca Tardella

References

Plummer, M., Best, N., Cowles, K. and Vines, K. (2006). CODA: Convergence Diagnosis and Output Analysis for MCMC, *R News*, **6**, pages 7–11, ISSN: 1609-3631.

See Also

`as.mcmc`

Examples

```
data(d_carconf)
GIBBS <- gibbsPLMIX(pi_inv=d_carconf, K=ncol(d_carconf), G=3, n_iter=30, n_burn=10)

## Coerce the posterior samples into an mcmc class object
gsPLMIX_to_mcmc(GIBBS)
```

`is.top_ordering` *Top-ordering datasets*

Description

Check the consistency of partial ordering data with a top-ordering dataset.

Usage

```
is.top_ordering(data, ...)
```

Arguments

`data`	An object containing the partial orderings whose consistency with a top-ordering dataset has to be tested. The following classes are admissible for `data`: numeric `matrix`, `data.frame`, `RandData` from the `rankdist` package and `rankings` from the `PlackettLuce` package.
`...`	Further arguments passed to or from other methods (not used).

Details

The argument `data` requires the partial sequences expressed in ordering format. When the value of `is.top-ordering` is `FALSE`, the membership function returns also a message with the conditions that are not met for the `data` to be a top-ordering dataset. `NA`'s in the input `data` are tacitly converted into zero entries.

Value

Logical: `TRUE` if the `data` argument is consistent with a top-ordering dataset (with a possible warning message if the supplied data need a further treatment with the coercion function `as.top_ordering` before being processed with the core functions of **PLMIX**) and `FALSE` otherwise.

Author(s)

Cristina Mollica and Luca Tardella

References

Turner, H., Kormidis, I. and Firth, D. (2018). PlackettLuce: Plackett-Luce Models for Rankings. R package version 0.2-3. `https://CRAN.R-project.org/package=PlackettLuce`

Qian, Z. (2018). rankdist: Distance Based Ranking Models. R package version 1.1.3. `https://CRAN.R-project.org/package=rankdist`

See Also

`rankings` and `rankings`

Examples

```
## A toy example of data matrix not satisfying the conditions to be a top-ordering dataset
toy_data=rbind(1:5,
c(0,4,3,2,1),
c(4,3.4,2,1,5),
c(2,3,0,0,NA),
c(4,4,3,2,5),
c(3,5,4,2,6),
c(2,-3,1,4,5),
c(2,0,1,4,5),
c(2,3,1,1,1),
c(2,3,0,4,0))

is.top_ordering(data=toy_data)

## A dataset from the StatRank package satisfying the conditions to be a top-ordering dataset
library(StatRank)
data(Data.Election9)
is.top_ordering(data=Data.Election9)
```

`label_switchPLMIX` *Label switching adjustment of the Gibbs sampling simulations for Bayesian mixtures of Plackett-Luce models*

Description

Remove the label switching phenomenon from the MCMC samples of Bayesian mixtures of Plackett-Luce models with $G > 1$ components.

Usage

```
label_switchPLMIX(pi_inv, seq_G, MCMCsampleP, MCMCsampleW, MAPestP,
  MAPestW, parallel = FALSE)
```

Arguments

`pi_inv`	An object of class `top_ordering`, collecting the numeric $N \times K$ data matrix of partial orderings, or an object that can be coerced with `as.top_ordering`.
`seq_G`	Numeric vector with the number of components of the Plackett-Luce mixtures to be assessed.
`MCMCsampleP`	List of size `length(seq_G)`, whose generic element is a numeric $L \times (G * K)$ matrix with the MCMC samples of the component-specific support parameters to be processed.
`MCMCsampleW`	List of size `length(seq_G)`, whose generic element is a numeric $L \times G$ matrix with the MCMC samples of the mixture weights to be processed.
`MAPestP`	List of size `length(seq_G)`, whose generic element is a numeric $G \times K$ matrix with the MAP estimates of the component-specific support parameters to be used as a pivot in the PRA method (see 'Details').
`MAPestW`	List of size `length(seq_G)`, whose generic element is a numeric vector with the MAP estimates of the G mixture weights to be used as a pivot in the PRA method (see 'Details').
`parallel`	Logical: whether parallelization should be used. Default is `FALSE`.

Details

The `label_switchPLMIX` function performs the label switching adjustment of the MCMC samples via the Pivotal Reordering Algorithm (PRA) described in Marin et al (2005), by recalling the `pra` function from the `label.switching` package.

Value

A list of named objects:

`final_sampleP`	List of size `length(seq_G)`, whose generic element is a numeric $G \times K \times L$ array with the MCMC samples of the component-specific support parameters adjusted for label switching.
`final_sampleW`	List of size `length(seq_G)`, whose generic element is a numeric $L \times G$ matrix with the MCMC samples of the mixture weights adjusted for label switching.

Author(s)

Cristina Mollica and Luca Tardella

References

Mollica, C. and Tardella, L. (2017). Bayesian Plackett-Luce mixture models for partially ranked data. *Psychometrika*, **82**(2), pages 442–458, ISSN: 0033-3123, DOI: 10.1007/s11336-016-9530-0.

Papastamoulis, P. (2016). label.switching: An R Package for Dealing with the Label Switching Problem in MCMC Outputs. *Journal of Statistical Software*, **69**(1), pages 1–24, DOI: 10.18637/jss.v069.c01.

Marin, J. M., Mengersen, K. and Robert, C.P. (2005). Bayesian modelling and inference on mixtures of distributions. *Handbook of Statistics* (25), D. Dey and C.R. Rao (eds). Elsevier-Sciences.

See Also

pra

Examples

```
data(d_carconf)
K <- ncol(d_carconf)

## Fit 1- and 2-component PL mixtures via MAP estimation
MAP_1 <- mapPLMIX_multistart(pi_inv=d_carconf, K=K, G=1,
                             n_start=2, n_iter=400*1)

MAP_2 <- mapPLMIX_multistart(pi_inv=d_carconf, K=K, G=2,
                             n_start=2, n_iter=400*2)

MAP_3 <- mapPLMIX_multistart(pi_inv=d_carconf, K=K, G=3,
                             n_start=2, n_iter=400*3)

mcmc_iter <- 30
burnin <- 10

## Fit 1- and 2-component PL mixtures via Gibbs sampling procedure
GIBBS_1 <- gibbsPLMIX(pi_inv=d_carconf, K=K, G=1, n_iter=mcmc_iter,
                      n_burn=burnin, init=list(p=MAP_1$mod$P_map,
                      z=binary_group_ind(MAP_1$mod$class_map,G=1)))
GIBBS_2 <- gibbsPLMIX(pi_inv=d_carconf, K=K, G=2, n_iter=mcmc_iter,
                      n_burn=burnin, init=list(p=MAP_2$mod$P_map,
                      z=binary_group_ind(MAP_2$mod$class_map,G=2)))
GIBBS_3 <- gibbsPLMIX(pi_inv=d_carconf, K=K, G=3, n_iter=mcmc_iter,
                      n_burn=burnin, init=list(p=MAP_3$mod$P_map,
                      z=binary_group_ind(MAP_3$mod$class_map,G=3)))

## Adjusting the MCMC samples for label switching
LS <- label_switchPLMIX(pi_inv=d_carconf, seq_G=1:3,
                   MCMCsampleP=list(GIBBS_1$P, GIBBS_2$P, GIBBS_3$P),
                   MCMCsampleW=list(GIBBS_1$W, GIBBS_2$W, GIBBS_3$W),
                   MAPestP=list(MAP_1$mod$P_map, MAP_2$mod$P_map, MAP_3$mod$P_map),
                   MAPestW=list(MAP_1$mod$W_map, MAP_2$mod$W_map, MAP_3$mod$W_map))
str(LS)
```

Loglikelihood — *Likelihood and log-likelihood evaluation for a mixture of Plackett-Luce models*

Description

Compute either the likelihood or the log-likelihood of the Plackett-Luce mixture model parameters for a partial ordering dataset.

Usage

```
likPLMIX(p, ref_order, weights, pi_inv)

loglikPLMIX(p, ref_order, weights, pi_inv)
```

Arguments

`p`	Numeric $G \times K$ matrix of component-specific support parameters.
`ref_order`	Numeric $G \times K$ matrix of component-specific reference orders.
`weights`	Numeric vector of G mixture weights.
`pi_inv`	An object of class `top_ordering`, collecting the numeric $N \times K$ data matrix of partial orderings, or an object that can be coerced with `as.top_ordering`.

Details

The `ref_order` argument accommodates for the more general mixture of Extended Plackett-Luce models (EPL), involving the additional reference order parameters (Mollica and Tardella 2014). A permutation of the first K integers can be specified in each row of the `ref_order` argument. Since the Plackett-Luce model is a special instance of the EPL with the reference order equal to the identity permutation, the `ref_order` argument must be a matrix with G rows equal to $(1, \dots, K)$ when dealing with Plackett-Luce mixtures.

Value

Either the likelihood or the log-likelihood value of the Plackett-Luce mixture model parameters for a partial ordering dataset.

Author(s)

Cristina Mollica and Luca Tardella

References

Mollica, C. and Tardella, L. (2017). Bayesian Plackett-Luce mixture models for partially ranked data. *Psychometrika*, **82**(2), pages 442–458, ISSN: 0033-3123, DOI: 10.1007/s11336-016-9530-0.

Mollica, C. and Tardella, L. (2014). Epitope profiling via mixture modeling for ranked data. *Statistics in Medicine*, **33**(21), pages 3738–3758, ISSN: 0277-6715, DOI: 10.1002/sim.6224.

Examples

```
data(d_apa)
K <- ncol(d_apa)
G <- 3
support_par <- matrix(1:(G*K), nrow=G, ncol=K)
weights_par <- c(0.50, 0.25, 0.25)
loglikPLMIX(p=support_par, ref_order=matrix(1:K, nrow=G, ncol=K, byrow=TRUE),
            weights=weights_par, pi_inv=d_apa)
```

make_complete — *Completion of partial rankings/orderings*

Description

Return complete rankings/orderings from partial sequences relying on a random generation of the missing positions/items.

Usage

```
make_complete(data, format_input, nranked = NULL, probitems = rep(1,
  ncol(data)))
```

Arguments

`data`	Numeric $N \times K$ data matrix of partial sequences to be completed.
`format_input`	Character string indicating the format of the `data` input, namely `"ordering"` or `"ranking"`.
`nranked`	Optional numeric vector of length N with the number of items ranked by each sample unit.
`probitems`	Numeric vector with the K item-specific probabilities to be employed for the random generation of the missing positions/items (see 'Details'). Default is equal probabilities.

Details

The completion of the partial top rankings/orderings is performed according to the Plackett-Luce scheme, that is, with a sampling without replacement of the not-ranked items by using the positive values in the `probitems` argument as support parameters (normalization is not necessary).

Value

A list of two named objects:

`completedata`	Numeric $N \times K$ data matrix of complete sequences with the same format of the input `data`.
`nranked`	Numeric vector of length N with the number of items ranked by each sample unit of the input `data`.

Author(s)

Cristina Mollica and Luca Tardella

Examples

```
## Completion based on the top item frequencies
data(d_dublinwest)
head(d_dublinwest)
top_item_freq <- rank_summaries(data=d_dublinwest, format_input="ordering", mean_rank=FALSE,
                                pc=FALSE)$marginals["Rank_1",]

d_dublinwest_compl <- make_complete(data=d_dublinwest, format_input="ordering",
                                    probitems=top_item_freq)
head(d_dublinwest_compl$completedata)
```

`make_partial` *Censoring of complete rankings/orderings*

Description

Return partial top rankings/orderings from complete sequences obtained either with user-specified censoring patterns or with a random truncation.

Usage

```
make_partial(data, format_input, nranked = NULL, probcens = rep(1,
  ncol(data) - 1))
```

Arguments

`data`	Numeric $N \times K$ data matrix of complete sequences to be censored.
`format_input`	Character string indicating the format of the `data` input, namely `"ordering"` or `"ranking"`.
`nranked`	Numeric vector of length N with the desired number of items ranked by each sample unit after censoring. If not supplied (`NULL`), the censoring patterns are randomly generated according to the probabilities in the `probcens` argument.
`probcens`	Numeric vector of length $(K-1)$ with the probability of each censoring pattern to be employed for the random truncation of the complete sequences (normalization is not necessary). It works only if `nranked` argument is `NULL` (see 'Details'). Default is equal probabilities.

Details

The censoring of the complete sequences can be performed in: (i) a deterministic way, by specifying the number of top positions to be retained for each sample unit in the `nranked` argument; (ii) a random way, by sequentially specifying the probabilities of the top-1, top-2, ..., top-$(K-1)$ censoring patterns in the `probcens` argument. Recall that a top-$(K-1)$ sequence corresponds to a complete ordering/ranking.

Value

A list of two named objects:

`partialdata`	Numeric $N \times K$ data matrix of partial (censored) sequences with the same format of the input `data` and missing positions/items denoted with zero entries.
`nranked`	Numeric vector of length N with the number of items ranked by each sample unit after censoring.

Author(s)

Cristina Mollica and Luca Tardella

Examples

```
data(d_german)
head(d_german)
d_german_cens <- make_partial(data=d_german, format_input="ordering",
                              probcens=c(0.3, 0.3, 0.4))
head(d_german_cens$partialdata)

## Check consistency with the nominal censoring probabilities
round(prop.table(table(d_german_cens$nranked)), 2)
```

`mapPLMIX` *MAP estimation for a Bayesian mixture of Plackett-Luce models*

Description

Perform MAP estimation via EM algorithm for a Bayesian mixture of Plackett-Luce models fitted to partial orderings.

Usage

```
mapPLMIX(pi_inv, K, G, init = list(p = NULL, omega = NULL),
  n_iter = 1000, hyper = list(shape0 = matrix(1, nrow = G, ncol = K),
  rate0 = rep(0, G), alpha0 = rep(1, G)), eps = 10^(-6),
  centered_start = FALSE, plot_objective = FALSE)
```

Arguments

`pi_inv`	An object of class `top_ordering`, collecting the numeric $N \times K$ data matrix of partial orderings, or an object that can be coerced with `as.top_ordering`.
`K`	Number of possible items.
`G`	Number of mixture components.

`init`	List of named objects with initialization values: `p` is a numeric $G \times K$ matrix of component-specific support parameters; `omega` is a numeric vector of G mixture weights. If starting values are not supplied (`NULL`), they are randomly generated with a uniform distribution. Default is `NULL`.
`n_iter`	Maximum number of EM iterations.
`hyper`	List of named objects with hyperparameter values for the conjugate prior specification: `shape0` is a numeric $G \times K$ matrix of shape hyperparameters; `rate0` is a numeric vector of G rate hyperparameters; `alpha0` is a numeric vector of G Dirichlet hyperparameters. Default is noninformative (flat) prior setting.
`eps`	Tolerance value for the convergence criterion.
`centered_start`	Logical: whether a random start whose support parameters and weights should be centered around the observed relative frequency that each item has been ranked top. Default is `FALSE`. Ignored when `init` is not `NULL`.
`plot_objective`	Logical: whether the objective function (that is the kernel of the log-posterior distribution) should be plotted. Default is `FALSE`.

Details

Under noninformative (flat) prior setting, the EM algorithm for MAP estimation corresponds to the EMM algorithm described by Gormley and Murphy (2006) to perform frequentist inference. In this case, the MAP solution coincides with the MLE and the output vectors `log_lik` and `objective` coincide as well.

The `mapPLMIX` function performs the MAP procedure with a single starting value. To address the issue of local maxima in the posterior distribution, see the `mapPLMIX_multistart` function.

Value

A list of S3 class `mpPLMIX` with named elements:

`W_map`	Numeric vector with the MAP estimates of the G mixture weights.
`P_map`	Numeric $G \times K$ matrix with the MAP estimates of the component-specific support parameters.
`z_hat`	Numeric $N \times G$ matrix of estimated posterior component membership probabilities.
`class_map`	Numeric vector of N mixture component memberships based on MAP allocation from the `z_hat` matrix.
`log_lik`	Numeric vector of the log-likelihood values at each iteration.
`objective`	Numeric vector of the objective function values (that is the kernel of the log-posterior distribution) at each iteration.
`max_objective`	Maximized objective function value.
`bic`	BIC value (only for the default flat priors, otherwise `NULL`).
`conv`	Binary convergence indicator: 1 = convergence has been achieved, 0 = otherwise.
`call`	The matched call.

Author(s)

Cristina Mollica and Luca Tardella

References

Mollica, C. and Tardella, L. (2017). Bayesian Plackett-Luce mixture models for partially ranked data. *Psychometrika*, **82**(2), pages 442–458, ISSN: 0033-3123, DOI: 10.1007/s11336-016-9530-0.

Gormley, I. C. and Murphy, T. B. (2006). Analysis of Irish third-level college applications data. *Journal of the Royal Statistical Society: Series A*, **169**(2), pages 361–379, ISSN: 0964-1998, DOI: 10.1111/j.1467-985X.2006.00412.x.

See Also

mapPLMIX_multistart

Examples

```
data(d_carconf)
MAP <- mapPLMIX(pi_inv=d_carconf, K=ncol(d_carconf), G=3, n_iter=400*3)
str(MAP)
MAP$P_map
MAP$W_map
```

---

mapPLMIX_multistart *MAP estimation for a Bayesian mixture of Plackett-Luce models with multiple starting values*

Description

Perform MAP estimation via EM algorithm with multiple starting values for a Bayesian mixture of Plackett-Luce models fitted to partial orderings.

Usage

```
mapPLMIX_multistart(pi_inv, K, G, n_start = 1, init = rep(list(list(p =
  NULL, omega = NULL)), times = n_start), n_iter = 200,
  hyper = list(shape0 = matrix(1, nrow = G, ncol = K), rate0 = rep(0, G),
  alpha0 = rep(1, G)), eps = 10^(-6), plot_objective = FALSE,
  init_index = 1:n_start, parallel = FALSE, centered_start = FALSE)
```

Arguments

`pi_inv`	An object of class `top_ordering`, collecting the numeric $N \times K$ data matrix of partial orderings, or an object that can be coerced with `as.top_ordering`.
`K`	Number of possible items.
`G`	Number of mixture components.
`n_start`	Number of starting values.
`init`	List of `n_start` lists of named objects with initialization values: `p` is a numeric $G \times K$ matrix of component-specific support parameters; `omega` is a numeric vector of G mixture weights. If starting values are not supplied (`NULL`), they are randomly generated with a uniform distribution. Default is `NULL`.
`n_iter`	Maximum number of EM iterations.
`hyper`	List of named objects with hyperparameter values for the conjugate prior specification: `shape0` is a numeric $G \times K$ matrix of shape hyperparameters; `rate0` is a numeric vector of G rate hyperparameters; `alpha0` is a numeric vector of G Dirichlet hyperparameters. Default is noninformative (flat) prior setting.
`eps`	Tolerance value for the convergence criterion.
`plot_objective`	Logical: whether the objective function (that is the kernel of the log-posterior distribution) should be plotted. Default is `FALSE`.
`init_index`	Numeric vector indicating the positions of the starting values in the `init` list to be actually launched. Useful to launch the most promising starting values identified after a preliminary run. Default is run all the starting points in the `init` list.
`parallel`	Logical: whether parallelization should be used. Default is `FALSE`.
`centered_start`	Logical: whether a random start whose support parameters and weights should be centered around the observed relative frequency that each item has been ranked top. Default is `FALSE`. Ignored when `init` is not `NULL`.

Details

Under noninformative (flat) prior setting, the EM algorithm for MAP estimation corresponds to the EMM algorithm described by Gormley and Murphy (2006) to perform frequentist inference. In this case the MAP solution coincides with the MLE. The best model in terms of maximized posterior distribution is returned.

Value

A list of S3 class `mpPLMIX` with named elements:

`mod`	List of named objects describing the best model in terms of maximized posterior distribution. See output values of the single-run `mapPLMIX` function for a detailed explanation of the list elements.
`max_objective`	Numeric vector of the maximized objective function values for each initialization.
`convergence`	Binary vector with `length(init_index)` convergence indicators for each initialization: 1 = convergence has been achieved, 0 = otherwise.
`call`	The matched call.

Author(s)

Cristina Mollica and Luca Tardella

References

Mollica, C. and Tardella, L. (2017). Bayesian Plackett-Luce mixture models for partially ranked data. *Psychometrika*, **82**(2), pages 442–458, ISSN: 0033-3123, DOI: 10.1007/s11336-016-9530-0.

Gormley, I. C. and Murphy, T. B. (2006). Analysis of Irish third-level college applications data. *Journal of the Royal Statistical Society: Series A*, **169**(2), pages 361–379, ISSN: 0964-1998, DOI: 10.1111/j.1467-985X.2006.00412.x.

See Also

`mapPLMIX`

Examples

```
data(d_carconf)
MAP_mult <- mapPLMIX_multistart(pi_inv=d_carconf, K=ncol(d_carconf), G=3,
                                               n_start=2, n_iter=400*3)
str(MAP_mult)
MAP_mult$mod$P_map
MAP_mult$mod$W_map
```

`paired_comparisons` *Paired comparison matrix for a partial ordering/ranking dataset*

Description

Construct the paired comparison matrix for a partial ordering/ranking dataset.

Usage

```
paired_comparisons(data, format_input, nranked = NULL)
```

Arguments

`data`	Numeric $N \times K$ data matrix of partial sequences.
`format_input`	Character string indicating the format of the `data` input, namely `"ordering"` or `"ranking"`.
`nranked`	Optional numeric vector of length N with the number of items ranked by each sample unit.

Value

Numeric $K \times K$ paired comparison matrix: the (i, i')-th entry indicates the number of sample units that preferred item i to item i'.

Author(s)

Cristina Mollica and Luca Tardella

References

Mollica, C. and Tardella, L. (2017). Bayesian Plackett-Luce mixture models for partially ranked data. *Psychometrika*, **82**(2), pages 442–458, ISSN: 0033-3123, DOI: 10.1007/s11336-016-9530-0.

See Also

rank_summaries

Examples

```
data(d_dublinwest)
paired_comparisons(data=d_dublinwest, format_input="ordering")
```

`plot.gsPLMIX` *Plot the Gibbs sampling simulations for a Bayesian mixture of Plackett-Luce models*

Description

`plot` method for class `gsPLMIX`. It builds a suite of plots, visual convergence diagnostics and credible intervals for the MCMC samples of a Bayesian mixture of Plackett-Luce models. Graphics can be plotted directly into the current working device or stored into an external file placed into the current working directory.

Usage

```
## S3 method for class 'gsPLMIX'
plot(x, file = "ggmcmc-output.pdf", family = NA,
  plot = NULL, param_page = 5, width = 7, height = 10,
  dev_type_html = "png", post_est = "mean", max_scale_radar = NULL,
  ...)
```

Arguments

`x`	Object of class `gsPLMIX` returned by the `gibbsPLMIX` function.
`file`	Character vector with the name of the file to be created in the current working directory. Defaults is "ggmcmc-output.pdf". When NULL, plots are directly returned into the current working device (not recommended). This option allows also the user to work with an opened pdf (or other) device. When the file has an html file extension, the output is an Rmarkdown report with the figures embedded in the html file.
`family`	Character string indicating the name of the family of parameters to be plotted. A family of parameters is considered to be any group of parameters with the same name but different numerical values (for example `w[1]`, `w[2]`, etc). Default is `NA` meaning that all the parameters in the chain are plotted. Alternatively, one can choose `"w"`, `"p"`, `"log_lik"`, `"deviance"` or `"objective"`.
`plot`	Character vector containing the names of the desired plots. Default is `NULL` meaning that all the plots and convergence diagnostics are built (see 'Details').
`param_page`	Number of parameters to be plotted in each page. Defaults is 5.
`width`	Numeric scalar indicating the width of the pdf display in inches. Defaults is 7.
`height`	Numeric scalar indicating the height of the pdf display in inches. Defaults is 10.
`dev_type_html`	Character vector indicating the type of graphical device for the html output. Default is `"png"`. Alternatively, one can choose `"svg"`.
`post_est`	Character string indicating the point estimates of the Plackett-Luce mixture parameters to be computed from the `gsPLMIX` class object and then plotted in the current working device. Default is `"mean"`. Alternatively, one can choose `"median"`.
`max_scale_radar`	Numeric scalar indicating the maximum value on each axis of the radar plot for the support parameter point estimates. Default is `NULL` meaning that the maximum of the estimated support parameters is used.
`...`	Further arguments passed to or from other methods (not used).

Details

Plots of the MCMC samples include histograms, densities, traceplots, running means plots, overlapped densities comparing the complete and partial samples, autocorrelation functions, crosscorrelation plots and caterpillar plots of the 90 and 95% equal-tails credible intervals. Note that the latter are created for the support parameters (when either `family=NA` or `family="p"`), for the mixture weights in the case $G > 1$ (when either `family=NA` or `family="w"`), for the log-likelihood values (when `family="log_lik"`), for the deviance values (when `family="deviance"`). Convergence tools include the potential scale reduction factor and the Geweke z-score. These functionalities are implemented with a call to the `ggs` and `ggmcmc` functions of the `ggmcmc` package (see 'Examples' for the specification of the `plot` argument) and for the objective function values (when `family="objective"`).

By recalling the `chartJSRadar` function from the `radarchart` package and the routines of the `ggplot2` package, `plot.gsPLMIX` additionally produces a radar plot of the support parameters and, when $G > 1$, a donut plot of the mixture weights based on the posterior point estimates. The radar chart is returned in the Viewer Pane.

Author(s)

Cristina Mollica and Luca Tardella

References

Ashton, D. and Porter, S. (2016). radarchart: Radar Chart from 'Chart.js'. R package version 0.3.1. https://CRAN.R-project.org/package=radarchart

Wickham, H. (2009). ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York.

Fernandez-i-Marin, X. (2006). ggmcmc: Analysis of MCMC Samples and Bayesian Inference, *Journal of Statistical Software*, **70**(9), pages 1–20, DOI: 10.18637/jss.v070.i09.

See Also

ggs, ggmcmc, chartJSRadar and ggplot

Examples

```
# Not run:
data(d_carconf)
GIBBS <- gibbsPLMIX(pi_inv=d_carconf, K=ncol(d_carconf), G=5, n_iter=30, n_burn=10)

# Not run:
# Plot posterior samples supplied as an gsPLMIX class object
# plot(GIBBS)

# Selected plots of the posterior samples of the support parameters
# plot(GIBBS, family="p", plot=c("compare_partial","Rhat","caterpillar"), param_page=6)

# Selected plots of the posterior samples of the mixture weights
# plot(GIBBS, family="w", plot=c("histogram","running","crosscorrelation","caterpillar"))

# Selected plots of the posterior log-likelihood values
# plot(GIBBS, family="log_lik", plot=c("autocorrelation","geweke"), param_page=1)

# Selected plots of the posterior deviance values
# plot(GIBBS, family="deviance", plot=c("traceplot","density"), param_page=1)
```

---

plot.mpPLMIX *Plot the MAP estimates for a Bayesian mixture of Plackett-Luce models*

Description

plot method for class mpPLMIX.

Usage

```
## S3 method for class 'mpPLMIX'
plot(x, max_scale_radar = NULL, ...)
```

Arguments

`x`	Object of class `mpPLMIX` returned by the `mpPLMIX` function.
`max_scale_radar`	Numeric scalar indicating the maximum value on each axis of the radar plot for the support parameter point estimates. Default is `NULL` meaning that the maximum of the estimated support parameters is used.
`...`	Further arguments passed to or from other methods (not used).

Details

By recalling the `chartJSRadar` function from the `radarchart` package and the routines of the `ggplot2` package, `plot.mpPLMIX` produces a radar plot of the support parameters and, when $G > 1$, a donut plot of the mixture weights and a heatmap of the component membership probabilities based on the MAP estimates. The radar chart is returned in the Viewer Pane.

Author(s)

Cristina Mollica and Luca Tardella

References

Ashton, D. and Porter, S. (2016). radarchart: Radar Chart from 'Chart.js'. R package version 0.3.1. https://CRAN.R-project.org/package=radarchart

Wickham, H. (2009). ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York.

See Also

`chartJSRadar` and `ggplot`

Examples

```
# Not run:
data(d_carconf)
MAP <- mapPLMIX(pi_inv=d_carconf, K=ncol(d_carconf), G=3)
plot(MAP)

# Not run:
MAP_multi <- mapPLMIX_multistart(pi_inv=d_carconf, K=ncol(d_carconf), G=3, n_start=5)
plot(MAP_multi)
```

ppcheckPLMIX *Posterior predictive check for Bayesian mixtures of Plackett-Luce models*

Description

Perform posterior predictive check to assess the goodness-of-fit of Bayesian mixtures of Plackett-Luce models with a different number of components.

Usage

```
ppcheckPLMIX(pi_inv, seq_G, MCMCsampleP, MCMCsampleW, top1 = TRUE,
  paired = TRUE, parallel = FALSE)
```

Arguments

pi_inv	An object of class top_ordering, collecting the numeric $N \times K$ data matrix of partial orderings, or an object that can be coerced with as.top_ordering.
seq_G	Numeric vector with the number of components of the Plackett-Luce mixtures to be assessed.
MCMCsampleP	List of size length(seq_G), whose generic element is a numeric $L \times (G * K)$ matrix with the MCMC samples of the component-specific support parameters.
MCMCsampleW	List of size length(seq_G), whose generic element is a numeric $L \times G$ matrix with the MCMC samples of the mixture weights.
top1	Logical: whether the posterior predictive p-value based on the top item frequencies has to be computed. Default is TRUE.
paired	Logical: whether the posterior predictive p-value based on the paired comparison frequencies has to be computed. Default is TRUE.
parallel	Logical: whether parallelization should be used. Default is FALSE.

Details

The ppcheckPLMIX function returns two posterior predictive p-values based on two chi squared discrepancy variables involving: (i) the top item frequencies and (ii) the paired comparison frequencies. In the presence of partial sequences in the pi_inv matrix, the same missingness patterns observed in the dataset (i.e., the number of items ranked by each sample unit) are reproduced on the replicated datasets from the posterior predictive distribution.

Value

A list with a named element:

post_pred_pvalue	Numeric length(seq_G)$\times 2$ matrix of posterior predictive p-values based on the top item and paired comparison frequencies. If either top1 or paired argument is FALSE, the corresponding matrix entries are NA.

Author(s)

Cristina Mollica and Luca Tardella

References

Mollica, C. and Tardella, L. (2017). Bayesian Plackett-Luce mixture models for partially ranked data. *Psychometrika*, **82**(2), pages 442–458, ISSN: 0033-3123, DOI: 10.1007/s11336-016-9530-0.

See Also

ppcheckPLMIX_cond

Examples

```
data(d_carconf)
K <- ncol(d_carconf)

## Fit 1- and 2-component PL mixtures via MAP estimation
MAP_1 <- mapPLMIX_multistart(pi_inv=d_carconf, K=K, G=1,
                             n_start=2, n_iter=400*1)

MAP_2 <- mapPLMIX_multistart(pi_inv=d_carconf, K=K, G=2,
                             n_start=2, n_iter=400*2)

MAP_3 <- mapPLMIX_multistart(pi_inv=d_carconf, K=K, G=3,
                             n_start=2, n_iter=400*3)

mcmc_iter <- 30
burnin <- 10

## Fit 1- and 2-component PL mixtures via Gibbs sampling procedure
GIBBS_1 <- gibbsPLMIX(pi_inv=d_carconf, K=K, G=1, n_iter=mcmc_iter,
                      n_burn=burnin, init=list(p=MAP_1$mod$P_map,
                      z=binary_group_ind(MAP_1$mod$class_map,G=1)))
GIBBS_2 <- gibbsPLMIX(pi_inv=d_carconf, K=K, G=2, n_iter=mcmc_iter,
                      n_burn=burnin, init=list(p=MAP_2$mod$P_map,
                      z=binary_group_ind(MAP_2$mod$class_map,G=2)))
GIBBS_3 <- gibbsPLMIX(pi_inv=d_carconf, K=K, G=3, n_iter=mcmc_iter,
                      n_burn=burnin, init=list(p=MAP_3$mod$P_map,
                      z=binary_group_ind(MAP_3$mod$class_map,G=3)))

## Checking goodness-of-fit of the estimated mixtures
CHECK <- ppcheckPLMIX(pi_inv=d_carconf, seq_G=1:3,
                      MCMCsampleP=list(GIBBS_1$P, GIBBS_2$P, GIBBS_3$P),
                      MCMCsampleW=list(GIBBS_1$W, GIBBS_2$W, GIBBS_3$W))
CHECK$post_pred_pvalue
```

ppcheckPLMIX_cond — *Conditional posterior predictive check for Bayesian mixtures of Plackett-Luce models*

Description

Perform conditional posterior predictive check to assess the goodness-of-fit of Bayesian mixtures of Plackett-Luce models with a different number of components.

Usage

```
ppcheckPLMIX_cond(pi_inv, seq_G, MCMCsampleP, MCMCsampleW, top1 = TRUE,
  paired = TRUE, parallel = FALSE)
```

Arguments

`pi_inv`	An object of class `top_ordering`, collecting the numeric $N \times K$ data matrix of partial orderings, or an object that can be coerced with `as.top_ordering`.
`seq_G`	Numeric vector with the number of components of the Plackett-Luce mixtures to be assessed.
`MCMCsampleP`	List of size `length(seq_G)`, whose generic element is a numeric $L \times (G * K)$ matrix with the MCMC samples of the component-specific support parameters.
`MCMCsampleW`	List of size `length(seq_G)`, whose generic element is a numeric $L \times G$ matrix with the MCMC samples of the mixture weights.
`top1`	Logical: whether the posterior predictive p-value based on the top item frequencies has to be computed. Default is `TRUE`.
`paired`	Logical: whether the posterior predictive p-value based on the paired comparison frequencies has to be computed. Default is `TRUE`.
`parallel`	Logical: whether parallelization should be used. Default is `FALSE`.

Details

The `ppcheckPLMIX_cond` function returns two posterior predictive p-values based on two chi squared discrepancy variables involving: (i) the top item frequencies and (ii) the paired comparison frequencies. In the presence of partial sequences in the `pi_inv` matrix, the same missingness patterns observed in the dataset (i.e., the number of items ranked by each sample unit) are reproduced on the replicated datasets from the posterior predictive distribution. Differently from the `ppcheckPLMIX` function, the condional discrepancy measures are obtained by summing up the chi squared discrepancies computed on subsamples of observations with the same number of ranked items.

Value

A list with a named element:

`post_pred_pvalue_cond`
: Numeric `length(seq_G)`$\times 2$ matrix of posterior predictive p-values based on the top item and paired comparison frequencies. If either `top1` or `paired` argument is `FALSE`, the corresponding matrix entries are `NA`.

Author(s)

Cristina Mollica and Luca Tardella

References

Mollica, C. and Tardella, L. (2017). Bayesian Plackett-Luce mixture models for partially ranked data. *Psychometrika*, **82**(2), pages 442–458, ISSN: 0033-3123, DOI: 10.1007/s11336-016-9530-0.

See Also

ppcheckPLMIX

Examples

```

data(d_carconf)
K <- ncol(d_carconf)

## Fit 1- and 2-component PL mixtures via MAP estimation
MAP_1 <- mapPLMIX_multistart(pi_inv=d_carconf, K=K, G=1,
                             n_start=2, n_iter=400*1)

MAP_2 <- mapPLMIX_multistart(pi_inv=d_carconf, K=K, G=2,
                             n_start=2, n_iter=400*2)

MAP_3 <- mapPLMIX_multistart(pi_inv=d_carconf, K=K, G=3,
                             n_start=2, n_iter=400*3)

mcmc_iter <- 30
burnin <- 10

## Fit 1- and 2-component PL mixtures via Gibbs sampling procedure
GIBBS_1 <- gibbsPLMIX(pi_inv=d_carconf, K=K, G=1, n_iter=mcmc_iter,
                      n_burn=burnin, init=list(p=MAP_1$mod$P_map,
                      z=binary_group_ind(MAP_1$mod$class_map,G=1)))
GIBBS_2 <- gibbsPLMIX(pi_inv=d_carconf, K=K, G=2, n_iter=mcmc_iter,
                      n_burn=burnin, init=list(p=MAP_2$mod$P_map,
                      z=binary_group_ind(MAP_2$mod$class_map,G=2)))
GIBBS_3 <- gibbsPLMIX(pi_inv=d_carconf, K=K, G=3, n_iter=mcmc_iter,
                      n_burn=burnin, init=list(p=MAP_3$mod$P_map,
                      z=binary_group_ind(MAP_3$mod$class_map,G=3)))

## Checking goodness-of-fit of the estimated mixtures
CHECKCOND <- ppcheckPLMIX_cond(pi_inv=d_carconf, seq_G=1:3,
                               MCMCsampleP=list(GIBBS_1$P, GIBBS_2$P, GIBBS_3$P),
                               MCMCsampleW=list(GIBBS_1$W, GIBBS_2$W, GIBBS_3$W))
CHECKCOND$post_pred_pvalue
```

print.gsPLMIX — *Print of the Gibbs sampling simulation of a Bayesian mixture of Plackett-Luce models*

Description

`print` method for class `gsPLMIX`. It shows some general information on the Gibbs sampling simulation for a Bayesian mixture of Plackett-Luce models.

Usage

```
## S3 method for class 'gsPLMIX'
print(x, ...)
```

Arguments

`x`	Object of class `gsPLMIX` returned by the `gibbsPLMIX` function.
`...`	Further arguments passed to or from other methods (not used).

Author(s)

Cristina Mollica and Luca Tardella

See Also

`gibbsPLMIX`

Examples

```
## Print of the Gibbs sampling procedure
data(d_carconf)
GIBBS <- gibbsPLMIX(pi_inv=d_carconf, K=ncol(d_carconf), G=3, n_iter=30, n_burn=10)
print(GIBBS)
```

print.mpPLMIX — *Print of the MAP estimation algorithm for a Bayesian mixture of Plackett-Luce models*

Description

`print` method for class `mpPLMIX`. It shows some general information on the MAP estimation procedure for a Bayesian mixture of Plackett-Luce models.

Usage

```
## S3 method for class 'mpPLMIX'
print(x, ...)
```

Arguments

`x`	Object of class `mpPLMIX` returned by the `mapPLMIX` or `mapPLMIX_multistart` function.
`...`	Further arguments passed to or from other methods (not used).

Author(s)

Cristina Mollica and Luca Tardella

See Also

`mapPLMIX` and `mapPLMIX_multistart`

Examples

```
## Print of the MAP procedure with a single starting point
data(d_carconf)
MAP <- mapPLMIX(pi_inv=d_carconf, K=ncol(d_carconf), G=3)
print(MAP)

## Print of the MAP procedure with 5 starting points
MAP_multi <- mapPLMIX_multistart(pi_inv=d_carconf, K=ncol(d_carconf), G=3, n_start=5)
print(MAP_multi)
```

`rank_ord_switch` *Switch from orderings to rankings and vice versa*

Description

Convert the format of the input dataset from orderings to rankings and vice versa.

Usage

```
rank_ord_switch(data, format_input, nranked = NULL)
```

Arguments

`data`	Numeric $N \times K$ data matrix of partial sequences whose format has to be converted.
`format_input`	Character string indicating the format of the `data` input, namely `"ordering"` or `"ranking"`.
`nranked`	Optional numeric vector of length N with the number of items ranked by each sample unit.

Value

Numeric $N \times K$ data matrix of partial sequences with inverse format.

Author(s)

Cristina Mollica and Luca Tardella

Examples

```
## From orderings to rankings for the Dublin West dataset
data(d_dublinwest)
head(d_dublinwest)
rank_ord_switch(data=head(d_dublinwest), format_input="ordering")
```

---

rank_summaries *Descriptive summaries for a partial ordering/ranking dataset*

Description

Compute rank summaries and censoring patterns for a partial ordering/ranking dataset.

Usage

```
rank_summaries(data, format_input, mean_rank = TRUE, marginals = TRUE,
  pc = TRUE)
```

Arguments

`data`	Numeric $N \times K$ data matrix of partial sequences.
`format_input`	Character string indicating the format of the `data` input, namely `"ordering"` or `"ranking"`.
`mean_rank`	Logical: whether the mean rank vector has to be computed. Default is `TRUE`.
`marginals`	Logical: whether the marginal rank distributions have to be computed. Default is `TRUE`.
`pc`	Logical: whether the paired comparison matrix has to be computed. Default is `TRUE`.

Value

A list of named objects:

`nranked`	Numeric vector of length N with the number of items ranked by each sample unit.
`nranked_distr`	Frequency distribution of the `nranked` vector.

`na_or_not`	Numeric $3{\times}K$ matrix with the counts of sample units that ranked or not each item. The last row contains the total by column, corresponding to the sample size N.
`mean_rank`	Numeric vector of length K with the mean rank of each item.
`marginals`	Numeric $K{\times}K$ matrix of the marginal rank distributions: the (i,j)-th entry indicates the number of units that ranked item i in the j-th position.
`pc`	Numeric $K{\times}K$ paired comparison matrix: the (i,i')-th entry indicates the number of sample units that preferred item i to item i'.

Author(s)

Cristina Mollica and Luca Tardella

References

Marden, J. I. (1995). Analyzing and modeling rank data. *Monographs on Statistics and Applied Probability* (64). Chapman & Hall, ISSN: 0-412-99521-2. London.

Examples

```
data(d_carconf)
rank_summaries(data=d_carconf, format_input="ordering")
```

`rPLMIX` *Random sample from a mixture of Plackett-Luce models*

Description

Draw a random sample of complete orderings/rankings from a G-component mixture of Plackett-Luce models.

Usage

```
rPLMIX(n = 1, K, G, p = t(matrix(1/K, nrow = K, ncol = G)),
  ref_order = t(matrix(1:K, nrow = K, ncol = G)), weights = rep(1/G,
  G), format_output = "ordering")
```

Arguments

`n`	Number of observations to be sampled. Default is 1.
`K`	Number of possible items.
`G`	Number of mixture components.
`p`	Numeric $G{\times}K$ matrix of component-specific support parameters. Default is equal support parameters (uniform mixture components).

`ref_order`	Numeric $G \times K$ matrix of component-specific reference orders. Default is forward orders (identity permutations) in each row, corresponding to Plackett-Luce mixture components (see 'Details').
`weights`	Numeric vector of G mixture weights. Default is equal weights.
`format_output`	Character string indicating the format of the returned simulated dataset (`"ordering"` or `"ranking"`). Default is `"ordering"`.

Details

Positive values are required for `p` and `weights` arguments (normalization is not necessary).

The `ref_order` argument accommodates for the more general mixture of Extended Plackett-Luce models (EPL), involving the additional reference order parameters (Mollica and Tardella 2014). A permutation of the first K integers can be specified in each row of the `ref_order` argument to generate a sample from a G-component mixture of EPL. Since the Plackett-Luce model is a special instance of the EPL with the reference order equal to the identity permutation $(1, \dots, K)$, the default value of the `ref_order` argument is forward orders.

Value

If $G = 1$, a numeric $N \times K$ matrix of simulated complete sequences. If $G > 1$, a list of two named objects:

`comp`	Numeric vector of N mixture component memberships.
`sim_data`	Numeric $N \times K$ matrix of simulated complete sequences.

Author(s)

Cristina Mollica and Luca Tardella

Examples

```
K <- 6
G <- 3
support_par <- matrix(1:(G*K), nrow=G, ncol=K)
weights_par <- c(0.50, 0.25, 0.25)

set.seed(47201)
simulated_data <- rPLMIX(n=5, K=K, G=G, p=support_par, weights=weights_par)
simulated_data$comp
simulated_data$sim_data
```

`selectPLMIX`	*Bayesian selection criteria for mixtures of Plackett-Luce models*

Description

Compute Bayesian comparison criteria for mixtures of Plackett-Luce models with a different number of components.

Usage

```
selectPLMIX(pi_inv, seq_G, MCMCsampleP = vector(mode = "list", length =
  length(seq_G)), MCMCsampleW = vector(mode = "list", length =
  length(seq_G)), MAPestP, MAPestW, deviance, post_est = "mean",
  parallel = FALSE)
```

Arguments

`pi_inv`	An object of class `top_ordering`, collecting the numeric $N \times K$ data matrix of partial orderings, or an object that can be coerced with `as.top_ordering`.
`seq_G`	Numeric vector with the number of components of the Plackett-Luce mixtures to be compared.
`MCMCsampleP`	List of size `length(seq_G)`, whose generic element is a numeric $L \times (G * K)$ matrix with the MCMC samples of the component-specific support parameters. Default is list of `NULL` elements.
`MCMCsampleW`	List of size `length(seq_G)`, whose generic element is a numeric $L \times G$ matrix with the MCMC samples of the mixture weights. Default is list of `NULL` elements.
`MAPestP`	List of size `length(seq_G)`, whose generic element is a numeric $G \times K$ matrix with the MAP estimates of the component-specific support parameters.
`MAPestW`	List of size `length(seq_G)`, whose generic element is a numeric vector with the MAP estimates of the G mixture weights.
`deviance`	List of size `length(seq_G)`, whose generic element is a numeric vector of posterior deviance values.
`post_est`	Character string indicating the point estimates of the Plackett-Luce mixture parameters to be computed from the MCMC sample. This argument is ignored when MAP estimates are supplied in the `MAPestP` and `MAPestW` arguments. Default is `"mean"`. Alternatively, one can choose `"median"` (see 'Details').
`parallel`	Logical: whether parallelization should be used. Default is `FALSE`.

Details

The `selectPLMIX` function privileges the use of the MAP point estimates to compute the Bayesian model comparison criteria, since they are not affected by the label switching issue. By setting both the `MAPestP` and `MAPestW` arguments equal to NULL, the user can alternatively compute the selection measures by relying on a different posterior summary (`"mean"` or `"median"`) specified in the `post_est` argument. In the latter case, the MCMC samples for each Plackett-Luce mixture must be supplied in the lists `MCMCsampleP` and `MCMCsampleW`. The drawback when working with point estimates other than the MAP is that the possible presence of label switching has to be previously removed from the traces to obtain meaningful results. See the `label_switchPLMIX` function to perfom label switching adjustment of the MCMC samples.

Several model selection criteria are returned. The two versions of DIC correspond to alternative ways of computing the effective number of parameters: DIC1 was proposed by Spiegelhalter et al. (2002) with penalty named `pD`, whereas DIC2 was proposed by Gelman et al. (2004) with penalty named `pV`. The latter coincides with the AICM introduced by Raftery et al. (2007), that is, the Bayesian counterpart of AIC. BPIC1 and BPIC2 are obtained from the two DIC by simply doubling the penalty term, as suggested by Ando (2007) to contrast DIC's tendency to overfitting. BICM1 is the Bayesian variant of the BIC, originally presented by Raftery et al. (2007) and entirely based on the MCMC sample. The BICM2, instead, involved the MAP estimate without the need of its approximation from the MCMC sample as for the BICM1.

Value

A list of named objects:

`point_estP`	List of size `length(seq_G)`, whose generic element is a numeric $G \times K$ matrix with the point estimates of the component-specific support parameters employed for the computation of the criteria.
`point_estW`	List of size `length(seq_G)`, whose generic element is a numeric vector with the G point estimates of the mixture weights employed for the computation of the criteria.
`fitting`	Numeric `length(seq_G)`$\times 2$ matrix with the fitting terms of the comparison measures, given by the posterior expected deviance `D_bar` and the deviance `D_hat` evaluated at the point estimate.
`penalties`	Numeric `length(seq_G)`$\times 2$ matrix with the penalty terms `pD` and `pV` (effective number of parameters).
`criteria`	Numeric `length(seq_G)`$\times 6$ matrix of Bayesian model selection criteria: `DIC1`, `DIC2`, `BPIC1`, `BPIC2`, `BICM1` and `BICM2` (see 'Details').

Author(s)

Cristina Mollica and Luca Tardella

References

Mollica, C. and Tardella, L. (2017). Bayesian Plackett-Luce mixture models for partially ranked data. *Psychometrika*, **82**(2), pages 442–458, ISSN: 0033-3123, DOI: 10.1007/s11336-016-9530-0.

Ando, T. (2007). Bayesian predictive information criterion for the evaluation of hierarchical Bayesian and empirical Bayes models. *Biometrika*, **94**(2), pages 443–458.

Raftery, A. E, Satagopan, J. M., Newton M. A. and Krivitsky, P. N. (2007). BAYESIAN STATISTICS 8. *Proceedings of the eighth Valencia International Meeting 2006*, pages 371–416. Oxford University Press.

Gelman, A., Carlin, J. B., Stern, H. S. and Rubin, D. B. (2004). Bayesian data analysis. Chapman & Hall/CRC, Second Edition, ISBN: 1-58488-388-X. New York.

Spiegelhalter, D. J., Best, N. G., Carlin, B. P. and Van Der Linde, A. (2002). Bayesian measures of model complexity and fit. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **64**(4), pages 583–639.

Examples

```
data(d_carconf)
K <- ncol(d_carconf)

## Fit 1- and 2-component PL mixtures via MAP estimation
MAP_1 <- mapPLMIX_multistart(pi_inv=d_carconf, K=K, G=1,
                                   n_start=2, n_iter=400*1)

MAP_2 <- mapPLMIX_multistart(pi_inv=d_carconf, K=K, G=2,
                                   n_start=2, n_iter=400*2)

mcmc_iter <- 30
burnin <- 10

## Fit 1- and 2-component PL mixtures via Gibbs sampling procedure
GIBBS_1 <- gibbsPLMIX(pi_inv=d_carconf, K=K, G=1, n_iter=mcmc_iter,
                      n_burn=burnin, init=list(p=MAP_1$mod$P_map,
                      z=binary_group_ind(MAP_1$mod$class_map,G=1)))
GIBBS_2 <- gibbsPLMIX(pi_inv=d_carconf, K=K, G=2, n_iter=mcmc_iter,
                      n_burn=burnin, init=list(p=MAP_2$mod$P_map,
                      z=binary_group_ind(MAP_2$mod$class_map,G=2)))
## Select the optimal number of components
SELECT <- selectPLMIX(pi_inv=d_carconf, seq_G=1:2,
                      MAPestP=list(MAP_1$mod$P_map, MAP_2$mod$P_map),
                      MAPestW=list(MAP_1$mod$W_map, MAP_2$mod$W_map),
                      deviance=list(GIBBS_1$deviance, GIBBS_2$deviance))
SELECT$criteria
```

`summary.gsPLMIX` *Summary of the Gibbs sampling procedure for a Bayesian mixture of Plackett-Luce models*

Description

`summary` method for class gsPLMIX. It provides summary statistics and credible intervals for the Gibbs sampling simulation of a Bayesian mixture of Plackett-Luce models.

Usage

```
## S3 method for class 'gsPLMIX'
summary(object, quantiles = c(0.025, 0.25, 0.5, 0.75,
  0.975), hpd_prob = 0.95, digits = 2, ...)
```

Arguments

`object`	Object of class `gsPLMIX` returned by the `gibbsPLMIX` function.
`quantiles`	Numeric vector of quantile probabilities.
`hpd_prob`	Numeric scalar in the grid of values spanning the interval (0,1) by 0.05, giving the posterior probability content of the HPD intervals. Supplied values outside the grid are rounded.
`digits`	Number of decimal places for rounding the posterior summaries.
`...`	Further arguments passed to or from other methods (not used).

Details

Posterior summaries include means, standard deviations, naive standard errors of the means (ignoring autocorrelation of the chain) and time-series standard errors based on an estimate of the spectral density at 0. They correspond to the `statistics` element of the output returned by the `summary.mcmc` function of the `coda` package. Highest posterior density (HPD) intervals are obtained by recalling the `HPDinterval` function of the `coda` package.

Value

A list of summary statistics for the `gsPLMIX` class object:

`statistics`	Numeric matrix with posterior summaries in each row (see 'Details').
`quantiles`	Numeric matrix with posterior quantiles at the given `quantiles` probabilities in each row.
`HPDintervals`	Numeric matrix with 100$*$`hpd_prob`% HPD intervals in each row.
`Modal_orderings`	Numeric $G \times K$ matrix with the estimated posterior modal orderings of each mixture component.
`call`	The matched call.

Author(s)

Cristina Mollica and Luca Tardella

References

Plummer, M., Best, N., Cowles, K. and Vines, K. (2006). CODA: Convergence Diagnosis and Output Analysis for MCMC, *R News*, **6**, pages 7–11, ISSN: 1609-3631.

See Also

`summary.mcmc` and `HPDinterval`

Examples

```
data(d_carconf)
GIBBS <- gibbsPLMIX(pi_inv=d_carconf, K=ncol(d_carconf), G=3, n_iter=30, n_burn=10)

## Summary of the Gibbs sampling procedure
summary(GIBBS)
```

---

summary.mpPLMIX   *Summary of the MAP estimation for a Bayesian mixture of Plackett-Luce models*

Description

`summary` method for class `mpPLMIX`. It provides summaries for the MAP estimation of a Bayesian mixture of Plackett-Luce models.

Usage

```
## S3 method for class 'mpPLMIX'
summary(object, digits = 2, ...)
```

Arguments

`object`	Object of class `mpPLMIX` returned by the `mapPLMIX` or `mapPLMIX_multistart` function.
`digits`	Number of decimal places for rounding the summaries.
`...`	Further arguments passed to or from other methods (not used).

Value

A list of summaries for the `mpPLMIX` class object:

`MAP_w`	Numeric vector with the MAP estimates of the G mixture weights. Returned only when when $G > 1$.
`MAP_p`	Numeric $G \times K$ matrix with the MAP estimates of the component-specific support parameters.
`MAP_modal_orderings`	Numeric $G \times K$ matrix with the estimated modal orderings of each mixture component.
`group_distr`	Numeric vector with the relative frequency distribution of the mixture component memberships based on MAP allocation. Returned only when when $G > 1$.
`perc_conv_rate`	Numeric scalar with the percentage of MAP algorithm convergence over the multiple starting points. Returned only when `summary.mpPLMIX` is applied to the output of the `mapPLMIX_multistart` function.

Author(s)

Cristina Mollica and Luca Tardella

Examples

```
## Summary of the MAP procedure with a single starting point
data(d_carconf)
MAP <- mapPLMIX(pi_inv=d_carconf, K=ncol(d_carconf), G=3)
summary(MAP)

## Summary of the MAP procedure with 5 starting points
MAP_multi <- mapPLMIX_multistart(pi_inv=d_carconf, K=ncol(d_carconf), G=3, n_start=5)
summary(MAP_multi)
```

---

unit_to_freq *Frequency distribution from the individual rankings/orderings*

Description

Construct the frequency distribution of the distinct observed sequences from the dataset of individual rankings/orderings.

Usage

```
unit_to_freq(data)
```

Arguments

data	Numeric $N \times K$ data matrix of observed individual sequences.

Value

Numeric matrix of the distinct observed sequences with the corresponding frequencies indicated in the last $(K+1)$-th column.

Author(s)

Cristina Mollica and Luca Tardella

Examples

```
## Frequency distribution for the APA top-ordering dataset
data(d_apa)
unit_to_freq(data=d_apa)
```

Index

∗Topic **datasets**
  d_apa, 9
  d_carconf, 10
  d_dublinwest, 11
  d_gaming, 12
  d_german, 13
  d_nascar, 14
  d_occup, 15
  d_rice, 16

as.mcmc, *19*
as.rankings, *6*
as.top_ordering, *4*, 5, *7*, *18*, *20*, *22*, *24*, *27*, *30*, *36*, *38*, *45*

bicPLMIX, *4*, 7
binary_group_ind, *3*, 9

chartJSRadar, *34, 35*

d_apa, *4*, 9
d_carconf, *4*, 10
d_dublinwest, *4*, 11
d_gaming, *4*, 12
d_german, *4*, 13
d_nascar, *4*, 14
d_occup, *4*, 15
d_rice, *4*, 16

freq_to_unit, *3*, 16

ggmcmc, *34*
ggplot, *34, 35*
ggs, *34*
gibbsPLMIX, *3*, 17, *40*
gsPLMIX_to_mcmc, *4*, 19

HPDinterval, *48*

is.top_ordering, *4*, *6*, 20

label.switching, *22*
label_switchPLMIX, *4*, 21, *46*
Likelihood *(*Loglikelihood*)*, 23
likelihood *(*Loglikelihood*)*, 23
likPLMIX, *4*
likPLMIX *(*Loglikelihood*)*, 23
Loglikelihood, 23
loglikelihood *(*Loglikelihood*)*, 23
loglikPLMIX, *4*
loglikPLMIX *(*Loglikelihood*)*, 23

make_complete, *3*, 25
make_partial, *3*, 26
mapPLMIX, *4*, *7, 8*, 27, *28*, *30, 31*, *41*
mapPLMIX_multistart, *4*, *7, 8*, *28*, *29*, 29, *41*

paired_comparisons, *3*, 31
PLMIX *(*PLMIX-package*)*, 3
PLMIX-package, 3
plot.gsPLMIX, *4*, 32
plot.mpPLMIX, *4*, 34
ppcheckPLMIX, *4*, 36, *39*
ppcheckPLMIX_cond, *4*, *37*, 38
pra, *22, 23*
print.gsPLMIX, *4*, 40
print.mpPLMIX, *4*, 40

rank_ord_switch, *3*, 41
rank_summaries, *3*, *32*, 42
rankings, *6*, *21*
rPLMIX, *3*, 43

selectPLMIX, *4*, 45
summary.gsPLMIX, *4*, 47
summary.mcmc, *48*
summary.mpPLMIX, *4*, 49

unit_to_freq, *3*, 50