Package: MSmix (via r-universe)

September 14, 2024

Title Finite Mixtures of Mallows Models with Spearman Distance for Full and Partial Rankings

Version 1.0.2

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

Description Fit and analysis of finite Mixtures of Mallows models with Spearman Distance for full and partial rankings with arbitrary missing positions. Inference is conducted within the maximum likelihood framework via Expectation-Maximization algorithms. Estimation uncertainty is tackled via diverse versions of bootstrapping as well as via Hessian-based standard errors calculations. The most relevant reference of the methods is Crispino, Mollica, Astuti and Tardella (2023) <doi:10.1007/s11222-023-10266-8>.

License GPL (>= 3)

LinkingTo Rcpp

Imports BayesMallows (>= 2.0.1), bmixture (>= 1.7.0), data.table (>= 1.15.0), factoextra (>= 1.0.7), fields (>= 15.2.0), foreach (>= 1.5.2), ggbump (>= 0.1.0), ggplot2 (>= 3.4.4), gmp (>= 0.7.4), gridExtra (>= 2.3.0), methods (>= 4.3.1), magrittr (>= 2.0.3), Rankcluster (>= 0.98.0), RColorBrewer (>= 1.1.3), Rcpp (>= 1.0.12), reshape (>= 0.8.9), rlang (>= 1.1.3), scales (>= 1.3.0), spsUtil (>= 0.2.2), stats (>= 4.3.1)

Depends R (>= 4.3.0), dplyr (>= 1.1.4)

Suggests doParallel

NeedsCompilation yes

Author Cristina Mollica [aut, cre, cph] (<`https://orcid.org/0000-0001-9152-6021`>), Marta Crispino [aut, cph] (<`https://orcid.org/0000-0002-1818-1298`>), Lucia Modugno [ctb] (<`https://orcid.org/0009-0003-7212-3237`>), Luca Tardella [ctb] (<`https://orcid.org/0000-0002-5703-3909`>)

LazyData true

Encoding UTF-8

RoxygenNote 7.3.1

Date/Publication 2024-06-15 14:00:01 UTC

Repository https://cmollica.r-universe.dev

RemoteUrl https://github.com/cran/MSmix

RemoteRef HEAD

RemoteSha ca2d579e4c68a6d3b21de259ca0c25f9b74062c5

Contents

`MSmix-package` *Finite Mixtures of Mallows Models with Spearman Distance for Full and Partial Rankings*

Description

The **MSmix** package provides functions to fit and analyze finite Mixtures of Mallows models with Spearman distance (a.k.a. θ-model) for full and partial rankings with arbitrary missing positions. Inference is conducted within the maximum likelihood (ML) framework via EM algorithms. Estimation uncertainty is tackled via diverse versions of bootstrapping as well as via Hessian-based standard errors calculations.

Details

The *Mallows model* is one of the most popular and frequently applied parametric distributions to analyze rankings of a finite set of items. However, inference for this model is challenging due to the intractability of the normalizing constant, also referred to as *partition function*. The present package performs ML estimation (MLE) of the Mallows model with Spearman distance from full and partial rankings with arbitrary censoring patterns. Thanks to the novel approximation of the model normalizing constant introduced by Crispino, Mollica, Astuti and Tardella (2023), as well as the existence of a closed-form expression of the MLE of the consensus ranking, **MSmix** can address inference even for a large number of items. The package also allows to account for unobserved sample heterogeneity through MLE of *finite mixtures of Mallows models with Spearman distance* via EM algorithms, in order to perform a model-based clustering of partial rankings into groups with similar preferences.

Computational efficiency is achieved with the use of a hybrid language, combining `R` and `C++` code, and the possibility of parallel computation.

In addition to inferential techniques, the package provides various functions for data manipulation, simulation, descriptive summary and model selection.

Specific S3 classes and methods are also supplied to enhance the usability and foster exchange with other packages.

The suite of functions available in the **MSmix** package is composed of:

Ranking data manipulation

`data_conversion` From rankings to orderings and vice versa.

`data_censoring` Censoring of full rankings.

`data_completion` Deterministic completion of partial rankings with full reference rankings.

`data_augmentation` Generate all full rankings compatible with partial rankings.

Ranking data simulation

`rMSmix` Random samples from finite mixtures of Mallows models with Spearman distance.

Ranking data description

`data_description` Descriptive summaries for partial rankings.

Model estimation

`fitMSmix` MLE of mixtures of Mallows models with Spearman distance via EM algorithms.

`likMSmix` Likelihood evaluation for mixtures of Mallows models with Spearman distance.

Model selection

`bicMSmix` BIC value for the fitted mixture of Mallows models with Spearman distance.

`aicMSmix` AIC value for the fitted mixture of Mallows models with Spearman distance.

Estimation uncertainty

`bootstrapMSmix` Bootstrap confidence intervals for mixtures of Mallows models with Spearman distance.

`confintMSmix` Hessian-based confidence intervals for mixtures of Mallows models with Spearman distance.

Spearman distance utilities

`spear_dist` Spearman distance computation for full rankings.

`spear_dist_distr` Spearman distance distribution under the uniform (null) model.

`partition_fun_spear` Partition function of the Mallows model with Spearman distance.

`expected_spear_dist` Expected Spearman distance under the Mallows model with Spearman distance.

`var_spear_dist` Variance of the Spearman distance under the Mallows model with Spearman distance.

S3 class methods

`print.bootMSmix` Print the bootstrap confidence intervals of mixtures of Mallows models with Spearman distance.

`print.data_descr` Print the descriptive statistics for partial rankings.

`print.emMSmix` Print the MLEs of mixtures of Mallows models with Spearman distance.

`print.summary.emMSmix` Print the summary of the MLEs of mixtures of Mallows models with Spearman distance.

`plot.bootMSmix` Plot the bootstrap confidence intervals of mixtures of Mallows models with Spearman distance.

`plot.data_descr` Plot the descriptive statistics for partial rankings.

`plot.dist` Plot the Spearman distance matrix for full rankings.

`plot.emMSmix` Plot the MLEs of mixtures of Mallows models with Spearman distance.

`summary.emMSmix` Summary of the MLEs of mixtures of Mallows models with Spearman distance.

Datasets

`ranks_antifragility` Antifragility features of innovative startups (full rankings with covariates).

`ranks_horror` Arkham Horror data (full rankings).

`ranks_beers` Beer preference data (partial missing at random rankings with covariate).

`ranks_read_genres` Reading preference data (partial top-5 rankings with covariates).

`ranks_sports` Sport preferences and habits (full rankings with covariates).

Some quantities frequently recalled in the manual are the following:

N Sample size.

n Number of possible items.

G Number of mixture components.

Data must be supplied as an integer $N \times n$ matrix with partial rankings in each row and missing positions denoted as NA (rank = 1 indicates the most-liked item). Partial sequences with a single missing entry are automatically filled in, as they correspond to full rankings. In the present setting, ties are not allowed.

Author(s)

Cristina Mollica, Marta Crispino, Lucia Modugno and Luca Tardella

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

References

Crispino M, Mollica C, Astuti V and Tardella L (2023). Efficient and accurate inference for mixtures of Mallows models with Spearman distance. *Statistics and Computing*, **33**(98), DOI: 10.1007/s11222-023-10266-8.

Crispino M, Mollica C, Modugno L, Casadio Tarabusi E, and Tardella L (2024+). MSmix: An R Package for clustering partial rankings via mixtures of Mallows models with Spearman distance. *(submitted)*.

`bicMSmix` *BIC and AIC for mixtures of Mallows models with Spearman distance*

Description

`bicMSmix` and `aicMSmix` compute, respectively, the Bayesian Information Criterion (BIC) and the Akaike Information Criterion (AIC) for a mixture of Mallow models with Spearman distance fitted on partial rankings.

Usage

```
bicMSmix(rho, theta, weights, rankings)

aicMSmix(rho, theta, weights, rankings)
```

Arguments

`rho`	Integer $G \times n$ matrix with the component-specific consensus rankings in each row.
`theta`	Numeric vector of G non-negative component-specific precision parameters.
`weights`	Numeric vector of G positive mixture weights (normalization is not necessary).
`rankings`	Integer $N \times n$ matrix or data frame with partial rankings in each row. Missing positions must be coded as `NA`.

Details

The (log-)likelihood evaluation is performed by augmenting the partial rankings with the set of all compatible full rankings (see `data_augmentation`), and then the marginal likelihood is computed.

When $n \leq 20$, the (log-)likelihood is exactly computed, otherwise it is approximated with the method introduced by Crispino et al. (2023). If $n > 170$, the approximation is also restricted over a fixed grid of values for the Spearman distance to limit computational burden.

Value

The BIC or AIC value.

References

Crispino M, Mollica C and Modugno L (2024+). MSmix: An R Package for clustering partial rankings via mixtures of Mallows Models with Spearman distance. *(submitted)*

Crispino M, Mollica C, Astuti V and Tardella L (2023). Efficient and accurate inference for mixtures of Mallows models with Spearman distance. *Statistics and Computing*, **33**(98), DOI: 10.1007/s11222-023-10266-8.

Schwarz G (1978). Estimating the dimension of a model. *The Annals of Statistics*, **6**(2), pages 461–464, DOI: 10.1002/sim.6224.

Sakamoto Y, Ishiguro M, and Kitagawa G (1986). *Akaike Information Criterion Statistics*. Dordrecht, The Netherlands: D. Reidel Publishing Company.

See Also

`likMSmix`, `data_augmentation`

Examples

```
## Example 1. Simulate rankings from a 2-component mixture of Mallow models
## with Spearman distance.
set.seed(12345)
rank_sim <- rMSmix(sample_size = 50, n_items = 12, n_clust = 2)
str(rank_sim)
rankings <- rank_sim$samples
# Fit the true model.
set.seed(12345)
fit <- fitMSmix(rankings = rankings, n_clust = 2, n_start = 10)
# Comparing the BIC at the true parameter values and at the MLE.
```

```
bicMSmix(rho = rank_sim$rho, theta = rank_sim$theta, weights = rank_sim$weights,
       rankings = rank_sim$samples)
bicMSmix(rho = fit$mod$rho, theta = fit$mod$theta, weights = fit$mod$weights,
       rankings = rank_sim$samples)
aicMSmix(rho = rank_sim$rho, theta = rank_sim$theta, weights = rank_sim$weights,
       rankings = rank_sim$samples)
aicMSmix(rho = fit$mod$rho, theta = fit$mod$theta, weights = fit$mod$weights,
       rankings = rank_sim$samples)


## Example 2. Simulate rankings from a basic Mallow model with Spearman distance.
set.seed(54321)
rank_sim <- rMSmix(sample_size = 50, n_items = 8, n_clust = 1)
str(rank_sim)
# Let us censor the observations to be top-5 rankings.
rank_sim$samples[rank_sim$samples > 5] <- NA
rankings <- rank_sim$samples
# Fit the true model with the two EM algorithms.
set.seed(54321)
fit_em <- fitMSmix(rankings = rankings, n_clust = 1, n_start = 10)
set.seed(54321)
fit_mcem <- fitMSmix(rankings = rankings, n_clust = 1, n_start = 10, mc_em = TRUE)
# Compare the BIC at the true parameter values and at the MLEs.
bicMSmix(rho = rank_sim$rho, theta = rank_sim$theta, weights = rank_sim$weights,
       rankings = rank_sim$samples)
bicMSmix(rho = fit_em$mod$rho, theta = fit_em$mod$theta, weights = fit_em$mod$weights,
       rankings = rank_sim$samples)
bicMSmix(rho = fit_mcem$mod$rho, theta = fit_mcem$mod$theta, weights = fit_mcem$mod$weights,
       rankings = rank_sim$samples)
aicMSmix(rho = rank_sim$rho, theta = rank_sim$theta, weights = rank_sim$weights,
       rankings = rank_sim$samples)
aicMSmix(rho = fit_em$mod$rho, theta = fit_em$mod$theta, weights = fit_em$mod$weights,
       rankings = rank_sim$samples)
aicMSmix(rho = fit_mcem$mod$rho, theta = fit_mcem$mod$theta, weights = fit_mcem$mod$weights,
       rankings = rank_sim$samples)
```

`bootstrapMSmix` *Bootstrap confidence intervals for mixtures of Mallows models with Spearman distance*

Description

Return the bootstrap confidence intervals for the parameters of a mixture of Mallow models with Spearman distance fitted on partial rankings.

`plot` method for class `"bootMSmix"`.

Usage

```
bootstrapMSmix(
  object,
  n_boot = 50,
  type = (if (object$em_settings$n_clust == 1) "non-parametric" else "soft"),
  conf_level = 0.95,
  all = FALSE,
  n_start = 10,
  parallel = FALSE
)

## S3 method for class 'bootMSmix'
plot(x, ...)
```

Arguments

`object`	An object of class `"emMSmix"` returned by `fitMSmix`.
`n_boot`	Number of desired bootstrap samples. Defaults to 50.
`type`	Character indicating which bootstrap method must be used. Available options are: `"non-parametric"` or `"parametric"` for the $G = 1$ case, and `"soft"` or `"separated"` for the $G > 1$ case. Defaults to `"non-parametric"` when `n_clust` = 1 and to `"soft"` when `n_clust` > 1. See Details.
`conf_level`	Value in the interval (0,1] indicating the desired confidence level of the interval estimates. Defaults to 0.95.
`all`	Logical: whether the bootstrap samples of the MLEs for all the parameters must be returned. Defaults to `FALSE`.
`n_start`	Number of starting points for the MLE on each bootstrap sample. Defaults to 10.
`parallel`	Logical: whether parallelization over multiple initializations of the EM algorithm must be used. Used when `rankings` contains some partial rankings. Defaults to `FALSE`.
`x`	An object of class `"bootMSmix"` returned by `bootstrapMSmix`.
`...`	Further arguments passed to or from other methods (not used).

Details

When `n_clust` = 1, two types of bootstrap are available: 1) `type` = `"non-parametric"` (default); `type` = `"parametric"`, where the latter supports full rankings only.

When `n_clust` > 1, two types of bootstrap are available: 1) `type` = `"soft"` (default), which is the soft-separated bootstrap (Crispino et al., 2024+) and returns confidence intervals for all the parameters of the mixture of Mallow models with Spearman distance; 2) `type` = `"separated"`, which is the separated bootstrap (Taushanov and Berchtold, 2019) and returns bootstrap samples for the component-specific consensus rankings and precisions.

Value

An object of class `"bootMSmix"`, namely a list with the following named components:

`itemwise_ci_rho` The bootstrap itemwise confidence intervals for the component-specific consensus rankings.

`ci_boot_theta` The bootstrap confidence intervals for the component-specific precisions.

`ci_boot_weights` The bootstrap confidence intervals for the mixture weights. Returned when `n_clust` > 1 and `type = "soft"`, otherwise `NULL`.

`boot` List containing all the `n_boot` bootstrap parameters. Returned when `all = TRUE`, otherwise `NULL`.

The `boot` sublist contains the following named components:

`rho_boot` List of length `n_clust` with the bootstrap MLEs of the consensus rankings. Each element of the list is an integer `n_boot` × `n_items` matrix containing, in each row, the bootstrap MLEs of the consensus ranking for a specific component.

`theta_boot` Numeric `n_boot`× `n_clust` matrix with the bootstrap MLEs of the component-specific precision parameters in each row.

`weights_boot` Numeric `n_boot`× `n_clust` matrix with the bootstrap MLEs of the mixture weights in each row. Returned when `n_clust` > 1 and `type = "soft"`, otherwise `NULL`.

For the component-specific bootstrap consensus ranking estimates, a heatmap is returned.

For the component-specific precisions and weights (for the latter when $G > 1$), a kernel density plot is returned.

References

Crispino M, Mollica C and Modugno L (2024+). MSmix: An R Package for clustering partial rankings via mixtures of Mallows Models with Spearman distance. *(submitted)*

Taushanov Z and Berchtold A (2019). Bootstrap validation of the estimated parameters in mixture models used for clustering. *Journal de la société française de statistique*, **160**(1).

Efron B (1982). The Jackknife, the Bootstrap, and Other Resampling Plans. Philadelphia, *Pa. :Society for Industrial and Applied Mathematics*.

Examples

```
## Example 1. Compute the bootstrap 95% confidence intervals for the Antifragility dataset.
# Let us assume no clusters.
r_antifrag <- ranks_antifragility[, 1:7]
set.seed(12345)
fit <- fitMSmix(rankings = r_antifrag, n_clust = 1, n_start = 1)
# Apply non-parametric bootstrap procedure.
set.seed(12345)
boot_np <- bootstrapMSmix(object = fit, n_boot = 200)
print(boot_np)
# Apply parametric bootstrap procedure and set all = TRUE
# to return the bootstrap MLEs of the consensus ranking.
set.seed(12345)
```

```
boot_p <- bootstrapMSmix(object = fit, n_boot = 200,
                         type = "parametric", all = TRUE)
print(boot_p)
plot(boot_p)

## Example 2. Compute the bootstrap 95% confidence intervals for the Antifragility dataset.
# Let us assume two clusters and apply soft bootstrap.
r_antifrag <- ranks_antifragility[, 1:7]
set.seed(12345)
fit <- fitMSmix(rankings = r_antifrag, n_clust = 2, n_start = 20)
set.seed(12345)
boot_soft <- bootstrapMSmix(object = fit, n_boot = 500,
                          n_start = 20, all = TRUE)
plot(boot_soft)
# Apply separated bootstrap and compare results.
set.seed(12345)
boot_sep <- bootstrapMSmix(object = fit, n_boot = 500,
                         n_start = 20, type = "separated", all = TRUE)
plot(boot_sep)
print(boot_soft)
print(boot_sep)
```

`confintMSmix` *Hessian-based confidence intervals for mixtures of Mallows models with Spearman distance*

Description

Return the Hessian-based confidence intervals of the continuous parameters of a mixture of Mallow models with Spearman distance fitted to full rankings, namely the component-specific precisions and weights.

Usage

```
confintMSmix(object, conf_level = 0.95)
```

Arguments

`object`	An object of class "`emMSmix`" returned by `fitMSmix`.
`conf_level`	Value in the interval (0,1] indicating the desired confidence level of the interval estimates. Defaults to 0.95.

Details

The current implementation of the hessian-based confidence intervals assumes that the observed rankings are complete.

Value

A list with the following named components:

`ci_theta`	The confidence intervals for the precision parameters.
`ci_weights`	The confidence intervals for the mixture weights. Returned when $G > 1$.

References

Crispino M, Mollica C and Modugno L (2024+). MSmix: An R Package for clustering partial rankings via mixtures of Mallows Models with Spearman distance. *(submitted)*

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

Mclachlan G and Peel D (2000). Finite Mixture Models. *Vol. 299. New York: Wiley*.

Examples

```
## Example 1. Simulate rankings from a 2-component mixture of Mallow models
## with Spearman distance.
set.seed(123)
d_sim <- rMSmix(sample_size = 75, n_items = 8, n_clust = 2)
rankings <- d_sim$samples
# Fit the basic Mallows model with Spearman distance.
set.seed(123)
fit1 <- fitMSmix(rankings = rankings, n_clust = 1, n_start = 10)
# Compute the hessian-based confidence intervals for the MLEs of the precision.
confintMSmix(object = fit1)
# Fit the true model.
set.seed(123)
fit2 <- fitMSmix(rankings = rankings, n_clust = 2, n_start = 10)
# Compute the hessian-based confidence intervals for the MLEs of the weights and precisions.
confintMSmix(object = fit2)
```

`data_augmentation` *Data augmentation of partial rankings*

Description

For a given partial ranking matrix, generate all possible full rankings which are compatible with each partially ranked sequence. Partial rankings with at most 10 missing positions and arbitrary patterns of censoring are supported.

Usage

```
data_augmentation(rankings, fill_single_na = TRUE)
```

Arguments

`rankings`	Integer $N \times n$ matrix or data frame with partial rankings in each row. Missing positions must be coded as `NA`.
`fill_single_na`	Logical: whether single missing positions in the row of `rankings` must be filled in prior to data augmentation. Defaults to `TRUE`.

Details

The data augmentation of a full ranking returns the complete ranking itself arranged in a row vector. The function can be applied on partial observations expressed in ordering format as well. A message informs the user when data augmentation may be heavy.

Value

A list of N elements corresponding to the matrices of full rankings compatible with each partial sequence.

References

Crispino M, Mollica C, Astuti V and Tardella L (2023). Efficient and accurate inference for mixtures of Mallows models with Spearman distance. *Statistics and Computing*, **33**(98), DOI: 10.1007/s11222-023-10266-8.

Examples

```
## Example 1. Data augmentation of a single partial top-9 ranking.
data_augmentation(c(3, 7, 5, 1, NA, 4, NA, 8, 2, 6, NA, 9))

## Example 2. Data augmentation of partial rankings with different censoring patterns.
rank_data <- rbind(c(NA, 4, NA, 1, NA),
                   c(NA, NA, NA, NA, 1),
                   c(2, NA, 1, NA, 3),
                   c(4, 2, 3, 5, 1),
                   c(NA, 4, 1, 3, 2))
data_augmentation(rank_data)
```

`data_censoring` *Censoring of full rankings*

Description

Convert full rankings into either top-k or MAR (missing at random) partial rankings.

Usage

```
data_censoring(
  rankings,
  type = "topk",
  nranked = NULL,
  probs = rep(1, ncol(rankings) - 1)
)
```

Arguments

`rankings`	Integer $N \times n$ matrix or data frame with full rankings in each row.
`type`	Character indicating which censoring process must be used. Options are: `"topk"` and `"mar"`. Defaults to `"topk"`.
`nranked`	Integer vector of length N with the desired number of positions to be retained in each partial sequence after censoring. If not supplied (`NULL`), the number of positions are randomly generated according to the probabilities in the `probs` argument. Defaults to `NULL`.
`probs`	Numeric vector of the $(n - 1)$ probabilities for the random generation of the number of positions to be retained in each partial sequence after censoring (normalization is not necessary). Used only if `nranked` argument is `NULL` (see Details). Default is equal probabilities.

Details

Both forms of partial rankings can be obtained into two ways: (i) by specifying, in the `nranked` argument, the number of positions to be retained in each partial ranking; (ii) by setting `nranked` = `NULL` (default) and specifying, in the `probs` argument, the probabilities of retaining respectively $1, 2, ..., (n-1)$ positions in the partial rankings (recall that a partial sequence with $(n-1)$ observed entries corresponds to a full ranking).

In the censoring process of full rankings into MAR partial sequences, the positions to be retained are uniformly generated.

Value

A list of two named objects:

`part_rankings` Integer $N \times n$ matrix with partial (censored) rankings in each row. Missing positions must be coded as `NA`.

`nranked` Integer vector of length N with the actual number of items ranked in each partial sequence after censoring.

Examples

```
## Example 1. Censoring the Antifragility dataset into partial top rankings
# Top-3 censoring (assigned number of top positions to be retained)
n <- 7
r_antifrag <- ranks_antifragility[, 1:n]
data_censoring(r_antifrag, type = "topk", nranked = rep(3,nrow(r_antifrag)))
```

```
# Random top-k censoring with assigned probabilities
set.seed(12345)
data_censoring(r_antifrag, type = "topk", probs = 1:(n-1))

## Example 2. Simulate full rankings from a basic Mallows model with Spearman distance
n <- 10
N <- 100
set.seed(12345)
rankings <- rMSmix(sample_size = N, n_items = n)$samples
# MAR censoring with assigned number of positions to be retained
set.seed(12345)
nranked <- round(runif(N,0.5,1)*n)
set.seed(12345)
mar_ranks1 <- data_censoring(rankings, type = "mar", nranked = nranked)
mar_ranks1
identical(mar_ranks1$nranked, nranked)
# MAR censoring with assigned probabilities
set.seed(12345)
probs <- runif(n-1, 0, 0.5)
set.seed(12345)
mar_ranks2 <- data_censoring(rankings, type = "mar", probs = probs)
mar_ranks2
prop.table(table(mar_ranks2$nranked))
round(prop.table(probs), 2)
```

`data_completion` *Completion of partial rankings with reference full rankings*

Description

Deterministic completion of partial rankings with the relative positions of the unranked items in the reference full rankings. Partial rankings with arbitrary patterns of censoring are supported.

Usage

```
data_completion(rankings, ref_rho)
```

Arguments

`rankings`	Integer $N \times n$ matrix or data frame with the partial rankings to be completed in each row. Missing positions must be coded as `NA`.
`ref_rho`	Integer $N \times n$ matrix or data frame whose rows represent the reference full rankings to be used to complete the partial rankings.

Details

The arguments `rankings` and `ref_rho` must be objects with the same class (matrix or data frame) and same dimensions.

The completion of a full ranking returns the complete ranking itself.

Value

Integer $N \times n$ matrix with the completed rankings in each row.

References

Crispino M, Mollica C and Modugno L (2024+). MSmix: An R Package for clustering partial rankings via mixtures of Mallows Models with Spearman distance. *(submitted)*

Examples

```
## Example 1. Completion of a single partial ranking.
data_completion(rankings = c(3, NA, NA, 1, NA), ref_rho = c(4, 5, 1, 3, 2))

## Example 2. Completion of partial rankings with arbitrary censoring patterns.
rankings <- rbind(c(3, NA, NA, 7, 2, NA, NA), c(NA, 6, NA, 5, NA, NA, 1),7:1)
data_completion(rankings = rankings, ref_rho = rbind(c(4, 5, 6, 1, 3, 7, 2),
                7:1, 1:7))
```

`data_conversion` *Switch data format from rankings to orderings and vice versa*

Description

Convert the format of the input dataset from rankings to orderings and vice versa. Differently from existing analogous functions supplied by other `R` packages, `data_conversion` supports also partial rankings/orderings with arbitrary patterns of censoring.

Usage

```
data_conversion(data, subset = NULL)
```

Arguments

`data`	Integer $N \times n$ matrix with partial sequences whose format has to be converted in each row. Missing entries must be coded as `NA`.
`subset`	Optional logical or integer vector specifying the subset of observations, i.e. rows of `rankings`, to be considered. Missing values are taken as `FALSE`.

Value

Integer $N \times n$ matrix of partial sequences with the inverse format.

Examples

```
## Example 1. Switch the data format for a single complete observation.
data_conversion(c(4, 5, 1, 3, 2))

## Example 2. Switch the data format for partial sequences with arbitrary censoring patterns.
data_conversion(rbind(c(NA, 2, 5, NA, NA), c(4, NA, 2, NA, 3), c(4, 5, 1, NA, NA),
                      c(NA, NA, NA, NA, 2), c(NA, 5, 2, 1, 3), c(3, 5, 1, 2, 4)))
```

`data_description` *Descriptive summaries for partial rankings*

Description

Compute various data summaries for a partial ranking dataset. Differently from existing analogous functions supplied by other `R` packages, `data_description` supports partial observations with arbitrary patterns of censoring.

`print` method for class `"data_descr"`.

Usage

```
data_description(
  rankings,
  marg = TRUE,
  borda_ord = FALSE,
  paired_comp = TRUE,
  subset = NULL,
  item_names = NULL
)

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

Arguments

`rankings`	Integer $N \times n$ matrix or data frame with partial rankings in each row. Missing positions must be coded as `NA`.
`marg`	Logical: whether the first-order marginals have to be computed. Defaults to `TRUE`.
`borda_ord`	Logical: whether, in the summary statistics, the items must be ordered according to the Borda ranking (i.e., mean rank vector). Defaults to `FALSE`.
`paired_comp`	Logical: whether the pairwise comparison matrix has to be computed. Defaults to `TRUE`.
`subset`	Optional logical or integer vector specifying the subset of observations, i.e. rows of `rankings`, to be considered. Missing values are taken as `FALSE`.

`item_names`	Character vector for the names of the items. Defaults to `NULL`, meaning that `colnames(rankings)` is used and, if not available, `item_names` is set equal to `"Item1","Item2",...`.
`x`	An object of class `"data_descr"` returned by `data_description`.
`...`	Further arguments passed to or from other methods (not used).

Details

The implementation of `data_description` is similar to that of `rank_summaries` from the `PLMIX` package. Differently from the latter, `data_description` works with any kind of partial rankings (not only top rankings) and allows to summarize subsamples thanks to the additional `subset` argument.

The Borda ranking, obtained from the ordering of the mean rank vector, corresponds to the MLE of the consensus ranking of the Mallow model with Spearman distance. If `mean_rank` contains some `NA`s, the corresponding items occupy the bottom positions in the `borda_ordering` according to the order they appear in `item_names`.

Value

An object of class `"data_descr"`, which is a list with the following named components:

`n_ranked`	Integer vector of length N with the number of items ranked in each partial sequence.
`n_ranked_distr`	Frequency distribution of the `n_ranked` vector.
`n_ranks_by_item`	Integer $3{\times}n$ matrix with the number of times that each item has been ranked or not. The last row contains the total by column, i.e. the sample size N.
`mean_rank`	Mean rank vector.
`borda_ordering`	Character vector corresponding to the Borda ordering. This is obtained from the ranking of the mean rank vector.
`marginals`	Integer $n{\times}n$ matrix of the first-order marginals in each column: the (j,i)-th entry indicates the number of times that item i is ranked in position j.
`pc`	Integer $n{\times}n$ pairwise comparison matrix: the (i,i')-th entry indicates the number of times that item i is preferred to item i'.
`rankings`	When `borda_ord = TRUE`, an integer $N{\times}n$ matrix corresponding to `rankings` with columns rearranged according to the Borda ordering, otherwise the input `rankings`.

References

Mollica C and Tardella L (2020). PLMIX: An R package for modelling and clustering partially ranked data. *Journal of Statistical Computation and Simulation*, **90**(5), pages 925–959, ISSN: 0094-9655, DOI: 10.1080/00949655.2020.1711909.

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

See Also

plot.data_descr

Examples

```
## Example 1. Sample statistics for the Antifragility dataset.
r_antifrag <- ranks_antifragility[, 1:7]
descr <- data_description(rankings = r_antifrag)
descr

## Example 2. Sample statistics for the Sports dataset.
r_sports <- ranks_sports[, 1:8]
descr <- data_description(rankings = r_sports, borda_ord = TRUE)
descr

## Example 3. Sample statistics for the Sports dataset by gender.
r_sports <- ranks_sports[, 1:8]
desc_f <- data_description(rankings = r_sports, subset = (ranks_sports$Gender == "Female"))
desc_m <- data_description(rankings = r_sports, subset = (ranks_sports$Gender == "Male"))
desc_f
desc_m
```

`expected_spear_dist` *Expectation of the Spearman distance*

Description

Compute (either the exact or the approximate) (log-)expectation of the Spearman distance under the Mallow model with Spearman distance.

Usage

```
expected_spear_dist(theta, n_items, log = TRUE)
```

Arguments

`theta`	Non-negative precision parameter.
`n_items`	Number of items.
`log`	Logical: whether the expected Spearman distance on the log scale must be returned. Defaults to `TRUE`.

Details

When $n \leq 20$, the expectation is exactly computed by relying on the Spearman distance distribution provided by OEIS Foundation Inc. (2023). When $n > 20$, it is approximated with the method introduced by Crispino et al. (2023) and, if $n > 170$, the approximation is also restricted over a fixed grid of values for the Spearman distance to limit computational burden.

The expected Spearman distance is independent of the consensus ranking of the Mallow model with Spearman distance due to the right-invariance of the metric. When $\theta = 0$, this is equal to $\frac{n^3-n}{6}$, which is the expectation of the Spearman distance under the uniform (null) model.

Value

Either the exact or the approximate (log-)expected value of the Spearman distance under the Mallow model with Spearman distance.

References

Crispino M, Mollica C, Astuti V and Tardella L (2023). Efficient and accurate inference for mixtures of Mallows models with Spearman distance. *Statistics and Computing*, **33**(98), DOI: 10.1007/s11222-023-10266-8.

OEIS Foundation Inc. (2023). The On-Line Encyclopedia of Integer Sequences, Published electronically at `https://oeis.org`.

Kendall MG (1970). Rank correlation methods. 4th ed. Griffin London.

See Also

`spear_dist_distr`, `partition_fun_spear`

Examples

```
## Example 1. Expected Spearman distance under the uniform (null) model,
## coinciding with (n^3-n)/6.
n_items <- 10
expected_spear_dist(theta = 0, n_items = n_items, log = FALSE)
(n_items^3-n_items)/6

## Example 2. Expected Spearman distance.
expected_spear_dist(theta = 0.5, n_items = 10, log = FALSE)

## Example 3. Log-expected Spearman distance as a function of theta.
expected_spear_dist_vec <- Vectorize(expected_spear_dist, vectorize.args = "theta")
curve(expected_spear_dist_vec(x, n_items = 10),
  from = 0, to = 0.1, lwd = 2, col = 2, ylim = c(3, 5.5),
  xlab = expression(theta), ylab = expression(log(E[theta](D))),
  main = "Log-expected Spearman distance")

## Example 4. Log-expected Spearman distance for varying number of items
# and values of the concentration parameter.
expected_spear_dist_vec <- Vectorize(expected_spear_dist, vectorize.args = "theta")
curve(expected_spear_dist_vec(x, n_items = 10),
  from = 0, to = 0.1, lwd = 2, col = 2, ylim = c(3, 9),
  xlab = expression(theta), ylab = expression(log(E[theta](D))),
  main = "Log-expected Spearman distance")
curve(expected_spear_dist_vec(x, n_items = 20), add = TRUE, col = 3, lwd = 2)
curve(expected_spear_dist_vec(x, n_items = 30), add = TRUE, col = 4, lwd = 2)
legend("topright", legend = c(expression(n == 10), expression(n == 20), expression(n == 30)),
  col = 2:4, lwd = 2, bty = "n")
```

`fitMSmix` *MLE of mixtures of Mallows models with Spearman distance via EM algorithms*

Description

Perform the MLE of mixtures of Mallows model with Spearman distance on full and partial rankings via EM algorithms. Partial rankings with arbitrary missing positions are supported.

`print` method for class `"emMSmix"`.

Usage

```
fitMSmix(
  rankings,
  n_clust = 1,
  n_start = 1,
  n_iter = 200,
  mc_em = FALSE,
  eps = 10^(-6),
  init = list(list(rho = NULL, theta = NULL, weights = NULL))[rep(1, n_start)],
  plot_log_lik = FALSE,
  comp_log_lik_part = FALSE,
  plot_log_lik_part = FALSE,
  parallel = FALSE,
  theta_max = 3,
  theta_tol = 1e-05,
  theta_tune = 1,
  subset = NULL,
  item_names = NULL
)

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

Arguments

`rankings`	Integer $N \times n$ matrix or data frame with partial rankings in each row. Missing positions must be coded as `NA`.
`n_clust`	Number of mixture components. Defaults to 1.
`n_start`	Number of starting points. Defaults to 1.
`n_iter`	Maximum number of EM iterations. Defaults to 200.
`mc_em`	Logical: whether the Monte Carlo EM algorithm must be used for MLE on partial rankings completion, see Details. Ignored when `rankings` does not contain any partial sequence. Defaults to `FALSE`.

`eps`	Positive tolerance value for the convergence of the EM algorithm. Defaults to 10^{-6}.
`init`	List of `n_start` lists with the starting values of the parameters to initialize the EM algorithm. Each list must contain three named objects, namely: 1) `rho`: integer $G \times n$ matrix with the component-specific consensus rankings in each row; 2) `theta`: numeric vector of G non-negative component-specific precision parameters; 3) `weights`: numeric vector of G positive mixture weights. Defaults to `NULL`, meaning that the starting points are automatically generated from the uniform distribution.
`plot_log_lik`	Logical: whether the iterative log-likelihood values (based on full or augmented rankings) must be plotted. Defaults to `FALSE`.
`comp_log_lik_part`	Logical: whether the maximized observed-data log-likelihood value (based on partial rankings) must be returned. Ignored when `rankings` does not contain any partial sequence or `data_augmentation` cannot be applied. See Details. Defaults to `FALSE`.
`plot_log_lik_part`	Logical: whether the iterative observed-data log-likelihood values (based on partial rankings) must be plotted. Ignored when `rankings` does not contain any partial sequence. In the presence of partial rankings, this argument is ignored when `comp_log_lik_part = FALSE` or `data_augmentation` cannot be applied. Defaults to `FALSE`.
`parallel`	Logical: whether parallelization over multiple initializations must be used. Defaults to `FALSE`.
`theta_max`	Positive upper bound for the precision parameters. Defaults to 3.
`theta_tol`	Positive convergence tolerance for the Mstep on theta. Defaults to 10^{-5}.
`theta_tune`	Positive tuning constant affecting the precision parameters in the Monte Carlo step. Ignored when `rankings` does not contain any partial sequence or `mc_em = FALSE`. Defaults to 1.
`subset`	Optional logical or integer vector specifying the subset of observations, i.e. rows of the `rankings`, to be kept. Missing values are taken as `FALSE`.
`item_names`	Character vector for the names of the items. Defaults to `NULL`, meaning that `colnames(rankings)` is used and, if not available, `item_names` is set equal to `"Item1","Item2",...`.
`x`	An object of class `"emMSmix"` returned by `fitMSmix`.
`...`	Further arguments passed to or from other methods (not used).

Details

The EM algorithms are launched from `n_start` initializations and the best solution in terms of maximized log-likelihood value (based on full or augmented rankings) is returned.

When `mc_em = FALSE`, the scheme introduced by Crispino et al. (2023) is performed, where partial rankings are augmented with all compatible full rankings. This type of data augmentation is supported up to 10 missing positions in the partial rankings.

When `mc_em = TRUE`, the - computationally more efficient - Monte Carlo EM algorithm introduced by Crispino et al. (2024+) is implemented. In the case of a large number of censored positions and sample sizes, the `mc_em = TRUE` must be preferred.

Regardless of the fitting method adopted for inference on partial rankings, note that setting the argument `comp_log_lik_part = TRUE` for the computation of the observed-data log-likelihood values (based on partial rankings) can slow down the procedure in the case of a large number of censored positions and sample sizes.

Value

An object of class `"emMSmix"`, namely a list with the following named components:

`mod` List of named objects describing the best fitted model in terms of maximized log-likelihood over the `n_start` initializations. See Details.

`max_log_lik` Maximized log-likelihood values for each initialization.

`partial_data` Logical: whether the dataset includes some partially-ranked sequences.

`convergence` Binary convergence indicators of the EM algorithm for each initialization: 1 = convergence has been achieved, 0 = otherwise.

`record` Best log-likelihood values sequentially achieved over the `n_start` initializations.

`em_settings` List of settings used to fit the model.

`call` The matched call.

The `mod` sublist contains the following named objects:

`rho` Integer $G{\times}n$ matrix with the MLEs of the component-specific consensus rankings in each row.

`theta` Numeric vector with the MLEs of the G component-specific precision parameters.

`weights` Numeric vector with the MLEs of the G mixture weights.

`z_hat` Numeric $N{\times}G$ matrix of the estimated posterior component membership probabilities. Returned when `n_clust > 1`, otherwise `NULL`.

`map_classification` Integer vector of N mixture component memberships based on the MAP allocation from the `z_hat` matrix. Returned when `n_clust > 1`, otherwise `NULL`.

`log_lik` Numeric vector of the log-likelihood values (based on full or augmented rankings) at each iteration.

`best_log_lik` Maximized log-likelihood value (based on full or augmented rankings) of the fitted model.

`bic` BIC value of the fitted model based on `best_log_lik`.

`log_lik_part` Numeric vector of the observed-data log-likelihood values (based on partial rankings) at each iteration. Returned when `rankings` contains some partial sequences that can be completed with `data_augmentation` and `plot_log_lik_part = TRUE`, otherwise `NULL`. See Details.

`best_log_lik_part` Maximized observed-data log-likelihood value (based on partial rankings) of the fitted model. Returned when `rankings` contains some partial sequences that can be completed with `data_augmentation`, otherwise `NULL`. See Details.

`bic_part` BIC value of the fitted model based on `best_log_lik_part`. Returned when `rankings` contains some partial sequences that can be completed with `data_augmentation`, otherwise `NULL`. See Details.

`conv` Binary convergence indicator of the best fitted model: 1 = convergence has been achieved, 0 = otherwise.

`augmented_rankings` Integer $N \times n$ matrix with rankings completed through the Monte Carlo step in each row. Returned when `rankings` contains some partial sequences and `mc_em = TRUE`, otherwise `NULL`.

References

Crispino M, Mollica C and Modugno L (2024+). MSmix: An R Package for clustering partial rankings via mixtures of Mallows Models with Spearman distance. *(submitted)*

Crispino M, Mollica C, Astuti V and Tardella L (2023). Efficient and accurate inference for mixtures of Mallows models with Spearman distance. *Statistics and Computing*, **33**(98), DOI: 10.1007/s11222-023-10266-8.

Sørensen Ø, Crispino M, Liu Q and Vitelli V (2020). BayesMallows: An R Package for the Bayesian Mallows Model. *The R Journal*, **12**(1), pages 324–342, DOI: 10.32614/RJ-2020-026.

Beckett LA (1993). Maximum likelihood estimation in Mallows's model using partially ranked data. In *Probability models and statistical analyses for ranking data*, pages 92–107. Springer New York.

See Also

`summary.emMSmix`, `plot.emMSmix`

Examples

```
## Example 1. Fit the 3-component mixture of Mallow models with Spearman distance
## to the Antifragility dataset.
r_antifrag <- ranks_antifragility[, 1:7]
set.seed(123)
mms_fit <- fitMSmix(rankings = r_antifrag, n_clust = 3, n_start = 10)
mms_fit$mod$rho; mms_fit$mod$theta; mms_fit$mod$weights

## Example 2. Fit the Mallow model with Spearman distance
## to simulated partial rankings through data augmentation.
rank_data <- rbind(c(NA, 4, NA, 1, NA), c(NA, NA, NA, NA, 1), c(2, NA, 1, NA, 3),
                   c(4, 2, 3, 5, 1), c(NA, 4, 1, 3, 2))
mms_fit <- fitMSmix(rankings = rank_data, n_start = 10)
mms_fit$mod$rho; mms_fit$mod$theta

## Example 3. Fit the Mallow model with Spearman distance
## to the Reading genres dataset through Monte Carlo EM.
top5_read <- ranks_read_genres[, 1:11]
mms_fit <- fitMSmix(rankings = top5_read, n_start = 10, mc_em = TRUE)
mms_fit$mod$rho; mms_fit$mod$theta
```

`likMSmix` *(Log-)likelihood for mixtures of Mallows models with Spearman distance*

Description

Compute the (log-)likelihood for the parameters of a mixture of Mallow models with Spearman distance on partial rankings. Partial rankings with arbitrary missing positions are supported.

Usage

```
likMSmix(
  rho,
  theta,
  weights = (if (length(theta) == 1) NULL),
  rankings,
  log = TRUE
)
```

Arguments

`rho`	Integer $G \times n$ matrix with the component-specific consensus rankings in each row.
`theta`	Numeric vector of G non-negative component-specific precision parameters.
`weights`	Numeric vector of G positive mixture weights (normalization is not necessary).
`rankings`	Integer $N \times n$ matrix or data frame with partial rankings in each row. Missing positions must be coded as `NA`.
`log`	Logical: whether the log-likelihood must be returned. Defaults to `TRUE`.

Details

The (log-)likelihood evaluation is performed by augmenting the partial rankings with the set of all compatible full rankings (see `data_augmentation`), and then the marginal likelihood is computed.

When $n \leq 20$, the (log-)likelihood is exactly computed. When $n > 20$, the model normalizing constant is not available and is approximated with the method introduced by Crispino et al. (2023). If $n > 170$, the approximation is also restricted over a fixed grid of values for the Spearman distance to limit computational burden.

Value

The (log)-likelihood value.

References

Crispino M, Mollica C and Modugno L (2024+). MSmix: An R Package for clustering partial rankings via mixtures of Mallows Models with Spearman distance. *(submitted)*

Crispino M, Mollica C, Astuti V and Tardella L (2023). Efficient and accurate inference for mixtures of Mallows models with Spearman distance. *Statistics and Computing*, **33**(98), DOI: 10.1007/s11222-023-10266-8.

See Also

`bicMSmix`, `aicMSmix`, `data_augmentation`

Examples

```
## Example 1. Likelihood of a full ranking of n=5 items under the uniform (null) model.
likMSmix(rho = 1:5, theta = 0, weights = 1, rankings = c(3,5,2,1,4), log = FALSE)
# corresponds to...
1/factorial(5)

## Example 2. Simulate rankings from a 2-component mixture of Mallow models
## with Spearman distance.
set.seed(12345)
d_sim <- rMSmix(sample_size = 75, n_items = 8, n_clust = 2)
str(d_sim)
# Fit the true model.
rankings <- d_sim$samples
fit <- fitMSmix(rankings = rankings, n_clust = 2, n_start = 10)
# Compare log-likelihood values of the true parameter values and the MLE.
likMSmix(rho = d_sim$rho, theta = d_sim$theta, weights = d_sim$weights,
       rankings = d_sim$samples)
likMSmix(rho = fit$mod$rho, theta = fit$mod$theta, weights = fit$mod$weights,
       rankings = d_sim$samples)

## Example 3. Simulate rankings from a basic Mallow model with Spearman distance.
set.seed(12345)
d_sim <- rMSmix(sample_size = 25, n_items = 6)
str(d_sim)
# Censor data to be partial top-3 rankings.
rankings <- d_sim$samples
rankings[rankings>3] <- NA
# Fit the true model with data augmentation.
set.seed(12345)
fit <- fitMSmix(rankings = rankings, n_clust = 1, n_start = 10)
# Compare log-likelihood values of the true parameter values and the MLEs.
likMSmix(rho = d_sim$rho, theta = d_sim$theta, weights = d_sim$weights,
       rankings = d_sim$samples)
likMSmix(rho = fit$mod$rho, theta = fit$mod$theta, weights = fit$mod$weights,
       rankings = d_sim$samples)
```

`partition_fun_spear` *Partition function of the Mallows model with Spearman distance*

Description

Compute (either the exact or the approximate) (log-)partition function of the Mallow model with Spearman distance.

Usage

```
partition_fun_spear(theta, n_items, log = TRUE)
```

Arguments

`theta`	Non-negative precision parameter.
`n_items`	Number of items.
`log`	Logical: whether the partition function on the log scale must be returned. Defaults to `TRUE`.

Details

When $n \leq 20$, the partition is exactly computed by relying on the Spearman distance distribution provided by OEIS Foundation Inc. (2023). When $n > 20$, it is approximated with the method introduced by Crispino et al. (2023) and, if $n > 170$, the approximation is also restricted over a fixed grid of values for the Spearman distance to limit computational burden.

The partition function is independent of the consensus ranking of the Mallow model with Spearman distance due to the right-invariance of the metric. When $\theta = 0$, the partition function is equivalent to $n!$, which is the normalizing constant of the uniform (null) model.

Value

Either the exact or the approximate (log-)partition function of the Mallow model with Spearman distance.

References

Crispino M, Mollica C, Astuti V and Tardella L (2023). Efficient and accurate inference for mixtures of Mallows models with Spearman distance. *Statistics and Computing*, **33**(98), DOI: 10.1007/s11222-023-10266-8.

OEIS Foundation Inc. (2023). The On-Line Encyclopedia of Integer Sequences, Published electronically at `https://oeis.org`.

See Also

`spear_dist_distr`, `expected_spear_dist`

Examples

```
## Example 1. Partition function of the uniform (null) model, coinciding with n!.
partition_fun_spear(theta = 0, n_items = 10, log = FALSE)
factorial(10)

## Example 2. Partition function of the Mallow model with Spearman distance.
partition_fun_spear(theta = 0.5, n_items = 10, log = FALSE)

## Example 3. Log-partition function of the Mallow model with Spearman distance
## as a function of theta.
partition_fun_spear_vec <- Vectorize(partition_fun_spear, vectorize.args = "theta")
curve(partition_fun_spear_vec(x, n_items = 10), from = 0, to = 0.1, lwd = 2,
  xlab = expression(theta), ylab = expression(log(Z(theta))),
  main = "Log-partition function of the Mallow model with Spearman distance",
  ylim = c(7, log(factorial(10))))

## Example 4. Log-partition function of the Mallow model with Spearman distance
## for varying number of items
# and values of the concentration parameter.
partition_fun_spear_vec <- Vectorize(partition_fun_spear, vectorize.args = "theta")
curve(partition_fun_spear_vec(x, n_items = 10),
  from = 0, to = 0.1, lwd = 2, col = 2,
  xlab = expression(theta), ylab = expression(log(Z(theta))),
  main = "Log-partition function of the Mallow model with Spearman distance",
  ylim = c(0, log(factorial(30))))
curve(partition_fun_spear_vec(x, n_items = 20), add = TRUE, col = 3, lwd = 2)
curve(partition_fun_spear_vec(x, n_items = 30), add = TRUE, col = 4, lwd = 2)
legend("topright", legend = c(expression(n == 10), expression(n == 20), expression(n == 30)),
  col = 2:4, lwd = 2, bty = "n")
```

`plot.data_descr` *Plot descriptive statistics for partial rankings*

Description

`plot` method for class `"data_descr"`.

Usage

```
## S3 method for class 'data_descr'
plot(
  x,
  cex_text_mean = 1,
  cex_symb_mean = 12,
  marg_by = "item",
  cex_text_pc = 3,
  cex_range_pc = c(8, 20),
  ...
)
```

Arguments

`x`	An object of class `"data_descr"` returned by `data_description`.
`cex_text_mean`	Positive scalar: the magnification to be used for all the labels in the plot for the mean rank vector. Defaults to 1.
`cex_symb_mean`	Positive scalar: the magnification to be used for the symbols in the pictogram of the mean rank vector. Defaults to 12.
`marg_by`	Character indicating whether the marginal distributions must be reported by `"item"` or by `"rank"` in the heatmap. Defaults to `"item"`.
`cex_text_pc`	Positive scalar: the magnification to be used for all the labels in the bubble plot of the paired comparison frequencies. Defaults to 3.
`cex_range_pc`	Numeric vector indicating the range of values to be used on each axis in the bubble plot of the paired comparison frequencies. Defaults to `c(8,20)`.
`...`	Further arguments passed to or from other methods (not used).

Value

Produce 5 plots to display descriptive summaries of the partial ranking dataset, namely: i) a barplot of the frequency distribution (%) of the number of items actually ranked in each partial sequence, ii) a basic pictogram of the mean rank vector, iii) a heatmap of the marginal distirbutions (either by item or by rank), iv) the ecdf of the marginal rank distributions and v) a bubble plot of the pairwise comparison matrix.

References

Wickham H (2016). ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York. ISBN 978-3-319-24277-4, https://ggplot2.tidyverse.org.

See Also

`data_description`

Examples

```
## Example 1. Plot sample statistics for the Antifragility dataset.
r_antifrag <- ranks_antifragility[, 1:7]
desc <- data_description(r_antifrag)
plot(desc)

## Example 2. Plot sample statistics for the Sports dataset.
r_sports <- ranks_sports[, 1:8]
desc <- data_description(rankings = r_sports, borda_ord = TRUE)
plot(desc, cex_text_mean = 1.2)

## Example 3. Plot sample statistics for the Sports dataset by gender.
r_sports <- ranks_sports[, 1:8]
desc_f <- data_description(rankings = r_sports, subset = (ranks_sports$Gender == "Female"))
plot(desc_f, cex_text_mean = 1.2)
desc_m <- data_description(rankings = r_sports, subset = (ranks_sports$Gender == "Male"))
```

```
plot(desc_m, cex_text_mean = 1.2)
```

`plot.dist` *Plot the Spearman distance matrix*

Description

`plot` method for class `"dist"`. It is useful to preliminary explore the presence of patterns (groups) of similar preferences in the ranking dataset.

Usage

```
## S3 method for class 'dist'
plot(
  x,
  order = TRUE,
  show_labels = TRUE,
  lab_size = 3,
  gradient = list(low = "#00AFBB", mid = "white", high = "#FC4E07"),
  ...
)
```

Arguments

`x`	An object of class `"dist"`, returned by `spear_dist` when setting the argument `rho = NULL`.
`order`	Logical: whether the rows of the distance matrix must be ordered. Defaults to `TRUE`.
`show_labels`	Logical: whether the labels must be displayed on the axes. Defaults to `TRUE`.
`lab_size`	Positive scalar: the magnification of the labels on the axes. Defaults to 3.
`gradient`	List of three elements with the colors for low, mid and high values of the distances in the heatmap. The element `mid` can take the value of NULL.
`...`	Further arguments passed to or from other methods (not used).

Details

`plot.dist` can visualize a distance matrix of any metric, provided that its class is `"dist"`. It can take a few seconds if the size of the distance matrix is large.

The heatmap can be also obtained by setting the arguments `rho = NULL` and `plot_dist_mat = TRUE` when applying `spear_dist`.

Value

Produce a heatmap of the Spearman distance matrix between all pairs of full rankings.

References

Alboukadel K and Mundt F (2020). factoextra: Extract and Visualize the Results of Multivariate Data Analyses. R package version 1.0.7. https://CRAN.R-project.org/package=factoextra

See Also

`spear_dist`

Examples

```
## Example 1. Plot the Spearman distance matrix of the Antifragility ranking dataset.
r_antifrag <- ranks_antifragility[, 1:7]
dist_mat <- spear_dist(rankings = r_antifrag)
plot(dist_mat, show_labels = FALSE)

## Example 2. Plot the Spearman distance matrix of the Sports ranking dataset.
r_sports <- ranks_sports[, 1:8]
dist_mat <- spear_dist(rankings = r_sports)
plot(dist_mat, show_labels = FALSE)
# Plot the Spearman distance matrix for the subsample of males.
dist_m <- spear_dist(rankings = r_sports, subset = (ranks_sports$Gender == "Male"))
plot(dist_m)
```

`plot.emMSmix` *Plot the MLEs for the fitted mixture of Mallows models with Spearman distance*

Description

`plot` method for class `"emMSmix"`.

Usage

```
## S3 method for class 'emMSmix'
plot(x, max_scale_w = 20, mar_lr = 0.4, mar_tb = 0.2, ...)
```

Arguments

`x`	An object of class `"emMSmix"` returned by `fitMSmix`.
`max_scale_w`	Positive scalar: maximum magnification of the dots in the bump plot, set proportional to the MLEs of the weights. Defaults to 20.
`mar_lr`	Numeric: margin for the left and right side of the plot. Defaults to 0.4.
`mar_tb`	Numeric: margin for the bottom and top side of the plot. Defaults to 0.2.
`...`	Further arguments passed to or from other methods (not used).

Value

Produce a bump plot to compare the component-specific consensus rankings of the fitted mixture of Mallow models with Spearman distance. The size of the dots of each consensus ranking is proportional to the weight of the corresponding component. When `n_clust` > 1, It also returns a heatmap of the estimated coponent membership probabilities.

References

Sjoberg D (2020). ggbump: Bump Chart and Sigmoid Curves. R package version 0.1.10. `https://CRAN.R-project.org/package=ggbump`.

Wickham H et al. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, **4**(43), 1686, DOI: 10.21105/joss.01686.

Wickham H (2016). ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York. ISBN 978-3-319-24277-4, `https://ggplot2.tidyverse.org`.

See Also

`fitMSmix`, `summary.emMSmix`

Examples

```
## Example 1. Fit a 3-component mixture of Mallow models with Spearman distance
## to the Antifragility dataset.
r_antifrag <- ranks_antifragility[, 1:7]
set.seed(123)
mms_fit <- fitMSmix(rankings = r_antifrag, n_clust = 3, n_start = 10)
plot(mms_fit)
```

`print.bootMSmix` *Print of the bootstrap confidence intervals for mixtures of Mallows models with Spearman distance*

Description

`print` method for class `"bootMSmix"`.

Usage

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

Arguments

`x`	An object of class `"bootMSmix"` returned by `bootstrapMSmix`.
`...`	Further arguments passed to or from other methods (not used).

`ranks_antifragility` *Antifragility Data (complete rankings with covariates)*

Description

The Antifragility dataset came up from an on-line survey conducted during spring 2021 by Sapienza University of Rome in collaboration with the Italian incubator Digital Magics, to investigate the construct of antifragility in innovative startups. Antifragility reflects the capacity of a company to adapt and improve its activity in the case of stresses, volatility and disorders triggered by critical and unexpected events, such as the COVID-19 outbreak which motivated the survey. On the basis of their experience and knowledge, a sample of $N = 99$ startups provided their complete rankings of $n = 7$ desirable antifragility properties in order of importance. The antifragility features are: 1 = Absorption, 2 = Redundancy, 3 = Small stressors, 4 = Non-monotonicity, 5 = Requisite variety, 6 = Emergence and 7 = Uncoupling.

Usage

```
data(ranks_antifragility)
```

Format

A data frame gathering $N = 99$ complete rankings of the $n = 7$ antifragility features in each row (rank 1 = most preferred item). The definition of the antifragility aspects is detailed below:

Absorption Ability to absorb stress and shocks while remaining in the planned state.

Redundancy Overcapacity to defend from risks and prevent faults.

Small_stressors Ability to exert low levels of stress on the organization.

Non_monotonicity Capacity to learn from failures and errors.

Requisite_variety Need for regulatory agents (i.e., government agency) to monitor and control organization's outcomes and behaviors.

Emergence Existence of cause-effect relationships between organization's activity at micro level and its outcomes at macro level.

Uncoupling Existence of strong interconnection between agents inside and outside the organization.

Industry_sector Industry sector of the startup.

Market Market type in which the startup operates.

Innovation_type Main innovation type of the startup.

Approach_to_crisis Main approach implemented by the startup during Covid-19 outbreak.

Crisis_impact Impact of Covid-19 outbreak on the startup.

Age Age of the startup (years).

N_employees Number of employees in the startup.

Region Italian region of the startup.

Job_title Job title of the startup participant in the survey.

Experience Years of job experience of the startup participant in the survey.

References

Ghasemi A and Alizadeh M (2017). Evaluating organizational antifragility via fuzzy logic. The case of an Iranian company producing banknotes and security paper. *Operations research and decisions*, **27**(2), pages 21–43, DOI: 10.5277/ord170202.

Examples

```
str(ranks_antifragility)
head(ranks_antifragility)
```

`ranks_beers` *Beers Data (partial rankings with covariate)*

Description

The Beers dataset was collected through an on-line survey on beer preferences administrated to the participants of the 2018 Pint of Science festival held in Grenoble. A sample of $N = 105$ respondents provided their partial rankings of $n = 20$ beers according to their personal preferences. The dataset also includes a covariate concerning respondents' residence.

Usage

```
data(ranks_beers)
```

Format

A data frame gathering $N = 105$ partial rankings of the beers in the first $n = 20$ columns (rank 1 = most preferred item) and an individual covariate in the last column. Partial rankings have missing at random positions coded as `NA`. The variables are detailed below:

Stella Rank assigned to Stella Artois.

Kwak Rank assigned to Kwak Brasserie.

KronKron Rank assigned to Kronenbourg (Kronenbourg).

Faro Rank assigned to Faro Timmermans.

Kron1664 Rank assigned to 1664 (Kronenbourg).

Chimay Rank assigned to Chimay Triple.

Pelforth Rank assigned to Pelforth Brune.

KronCarls Rank assigned to Carlsberg (Kronenbourg).

KronKanter Rank assigned to Kanterbraeu (Kronenbourg).

Hoegaarden Rank assigned to Hoegaarden Blanche.

Grimbergen Rank assigned to Grimbergen Blonde.

Pietra Rank assigned to Pietra Brasserie.

Affligem Rank assigned to Affligem Brasserie.

Goudale Rank assigned to La Goudale.

Leffe Rank assigned to Leffe Blonde.

Heineken Rank assigned to Heineken.

Duvel Rank assigned to Duvel Brasserie.

Choulette Rank assigned to La Choulette.

Orval Rank assigned to Orval.

Karmeliet Rank assigned to Karmeliet Triple.

Residence Residence.

References

Crispino M (2018). On-line questionnaire of the 2018 *Pint of Science festival* in Grenoble available at `https://docs.google.com/forms/d/1TiOIyc-jFXZF2Hb9echxZL0ZOcmr95LIBIieQuI-UJc/viewform?ts=5ac3a382&edit_requested=true`.

Examples

```
str(ranks_beers)
head(ranks_beers)
```

`ranks_horror` *Arkham Horror Data (complete rankings)*

Description

The Arkham Horror dataset came up from an on-line survey conducted by Curtis Miller to investigate popularity of different player roles of *Arkham Horror: The Card Game*. A sample of $N = 241$ respondents provided their complete rankings of $n = 5$ player modes in order of preference. The player roles are: 1 = Survivor, 2 = Rogue, 3 = Guardian, 4 = Seeker, 5 = Mystic. Statistical analyses of these data can be found at the links provided in References.

Usage

```
data(ranks_horror)
```

Format

A data frame gathering $N = 421$ complete rankings of the $n = 5$ player roles in each row (rank 1 = most preferred item). The variables are detailed below:

Survivor Rank assigned to Survivor role.

Rogue Rank assigned to Rogue role.

Guardian Rank assigned to Guardian role.

Seeker Rank assigned to Seeker role.

Mystic Rank assigned to Mystic role.

References

Curtis Miller's personal website: https://ntguardian.wordpress.com/2019/02/18/.

Examples

```
str(ranks_horror)
head(ranks_horror)
```

`ranks_read_genres` *Reading Genres Data (partial rankings with covariates)*

Description

The Reading Genres dataset was collected through an on-line survey conducted in Italy to investigate reading preferences in the context of the 2019 project *Patto per la lettura – Conta chi legge*. The questionnaire was administrated by the municipality of Latina (Latium, Italy), in collaboration with Sapienza University of Rome and the School of Government of the University of Tor Vergata. A sample of $N = 507$ respondents provided their partial top-5 rankings of $n = 11$ reading genres according to their personal preferences. The reading genres are: 1 = Classic, 2 = Novel, 3 = Thriller, 4 = Fantasy, 5 = Biography, 6 = Teenage, 7 = Horror, 8 = Comics, 9 = Poetry, 10 = Essay and 11 = Humor. The dataset also includes several covariates concerning respondents' socio-demographics characteristics and other free time activities.

Usage

```
data(ranks_read_genres)
```

Format

A data frame gathering $N = 507$ partial top-5 rankings of the reading genres in the first $n = 11$ columns (rank 1 = most preferred item) and individual covariates in the remaining columns. Missing positions are coded as `NA`. The variables are detailed below:

Classic Rank assigned to Classic.

Novel Rank assigned to Novel.

Thriller Rank assigned to Thriller.

Fantasy Rank assigned to Fantasy.

Biography Rank assigned to Biography.

Teenage Rank assigned to Teenage.

Horror Rank assigned to Horror.

Comics Rank assigned to Comics.

Poetry Rank assigned to Poetry.

Essay Rank assigned to Essay.

Humor Rank assigned to Humor.

Gender Gender.

Region Italian region of residence.

Age Age (years).

N_children Number of children.

Education Education level.

Final_mark Final grade of the education degree, scaled in the interval [6,10].

N_books Number of books read in the last 12 months.

References

Crispino M, Mollica C, Astuti V and Tardella L (2023). Efficient and accurate inference for mixtures of Mallows models with Spearman distance. *Statistics and Computing*, **33**(98), DOI: 10.1007/s11222-023-10266-8.

Mollica (2019). On-line questionnaire of the Italian 2019 project *Patto per la lettura – Conta chi legge* available at `https://form.jotformeu.com/90275118835359`.

Examples

```
str(ranks_read_genres)
head(ranks_read_genres)
```

`ranks_sports` *Sports Data (complete rankings with covariates)*

Description

The Sports dataset was collected through an on-line questionnaire on sport preferences and habits administered within the 2016 Big Data Analystics in Sports (BDsports) project, developed by the Big and Open Data Innovation Laboratory (BODaI-Lab) of the University of Brescia. A sample of $N = 647$ respondents provided their complete rankings of $n = 8$ popular sports according to their personal preferences. The sports are: 1 = Soccer, 2 = Swimming, 3 = Volleyball, 4 = Cycling, 5 = Basket, 6 = Boxe and martial arts, 7 = Tennis and 8 = Jogging. The dataset also includes several covariates concerning respondents' socio-demographics characteristics and other sport-related information.

Usage

```
data(ranks_sports)
```

Format

A data frame gathering $N = 647$ complete rankings of the sports in the first $n = 8$ columns (rank 1 = most preferred item) and individual covariates in the remaining columns. The variables are detailed below:

Soccer Rank assigned to Soccer.

Swimming Rank assigned to Swimming.

Volleyball Rank assigned to Volleyball.

Cycling Rank assigned to Cycling.

Basket Rank assigned to Basket.

Boxe_and_martial_arts Rank assigned to Boxe and Martial Arts.

Tennis Rank assigned to Tennis.

Jogging Rank assigned to Jogging.

Gender Gender.

Birth_month Month of birth.

Birth_year Year of birth.

Education Education level.

Residence Geographical area of residence.

Work Type of work.

Smoking Smoking status.

Sport_frequency Number of times per week that the respondent plays sports.

Sport_hours Number of hours per week that the respondent watches sports.

Sport_played Sport played by the respondent.

Personality Main aspect of respondent's personality.

Sport_motivation Main reason why the respondent plays sport.

Sport_type Favorite sport type.

Sport_relationships Do you think that sport, especially in team games, allows you to make new friends?

Water Quantity of water consumed per day.

Alcohol Frequency of alcohol consumption.

Fastfood Frequency of fast food consumption.

Food_supplements Opinion about the use of food supplements in sports.

Massmedia Prevalent mass media used to inquire about sport.

TV_space Do you think that sport currently occupies the space it deserves on TV? Rating from 1=definitely not to 7=definitely yes with 4=indifferent.

Magazine_space Do you think that sport currently occupies the space it deserves on the magazines? Rating from 1=definitely not to 7=definitely yes with 4=indifferent.

Radio_space Do you think that sport currently occupies the space it deserves on the radio? Rating from 1=definitely not to 7=definitely yes with 4=indifferent.

Internet_space Do you think that sport currently occupies the space it deserves on internet? Rating from 1=definitely not to 7=definitely yes with 4=indifferent.

Paid_channels Do you think it is right that some sports are only accessible on paid channels? Rating from 1=definitely not to 7=definitely yes with 4=indifferent.

Subscriptions Any past or current subscription to a sport magazine/channel.

Psycol_well_being Do you think that practicing sports affects psychological well-being? Rating from 1=definitely not to 7=definitely yes with 4=indifferent.

Physical_well_being Do you think that practicing sports affects physical well-being? Rating from 1=definitely not to 7=definitely yes with 4=indifferent.

Sport_nutrition Do you think nutrition affects sport? Rating from 1=definitely not to 7=definitely yes with 4=indifferent.

Overall_health Do you think that practicing sports affects health? Rating from 1=definitely not to 7=definitely yes with 4=indifferent.

Stress Self-reported stress level on a scale between 0 and 100.

Economic_status Level of satisfaction for one's own economic status: 0=not at all, 1=a little bit, 2=enough, 3=satisfied, 4=a lot. It is the only covariate with some `NA`'s.

References

Simone, R., Cappelli, C. and Di Iorio, F., (2019). Modelling marginal ranking distributions: the uncertainty tree. *Pattern Recognition Letters*, **125**, pages 278–288, DOI: 10.1016/j.patrec.2019.04.026.

Simone, R. and Iannario, M., (2018). Analysing sport data with clusters of opposite preferences. *Statistical Modelling*, **18**(5-6), pages 505–524, DOI: 10.1177/1471082X18798455.

Examples

```
str(ranks_sports)
head(ranks_sports)
```

`rMSmix` *Random samples from a mixture of Mallows models with Spearman distance*

Description

Draw random samples of full rankings from a mixture of Mallow models with Spearman distance.

Usage

```
rMSmix(
  sample_size = 1,
  n_items,
  n_clust = 1,
  rho = NULL,
  theta = NULL,
  weights = NULL,
  uniform = FALSE,
  mh = TRUE
)
```

Arguments

`sample_size`	Number of full rankings to be sampled. Defaults to 1.
`n_items`	Number of items.
`n_clust`	Number of mixture components. Defaults to 1.
`rho`	Integer $G \times n$ matrix with the component-specific consensus rankings in each row. Defaults to `NULL`, meaning that the consensus rankings are randomly generated according to the sampling scheme indicated by the `uniform` argument. See Details.
`theta`	Numeric vector of G non-negative component-specific precision parameters. Defaults to `NULL`, meaning that the concentrations are uniformly generated from an interval containing typical values for the precisions. See Details.
`weights`	Numeric vector of G positive mixture weights (normalization is not necessary). Defaults to `NULL`, meaning that the mixture weights are randomly generated according to the sampling scheme indicated by the `uniform` argument. See Details.
`uniform`	Logical: whether `rho` or `weights` have to be sampled uniformly on their support. When `uniform = FALSE` they are sampled, respectively, to ensure separation among mixture components and populated weights. Used when $G > 1$ and either `rho` or `weights` are `NULL` (see Details). Defaults to `FALSE`.
`mh`	Logical: whether the samples must be drawn with the Metropolis-Hastings (MH) scheme implemented in the `BayesMallows` package, rather by direct sampling from the Mallows probability distribution. For `n_items > 10`, the MH is always applied to speed up the sampling procedure. Defaults to `TRUE`.

Details

When `n_items > 10` or `mh = TRUE`, the random samples are obtained by using the Metropolis-Hastings algorithm, described in Vitelli et al. (2018) and implemented in the `sample_mallows` function of the package `BayesMallows` package.

When `theta = NULL` is not provided by the user, the concentration parameters are randomly generated from a uniform distribution on the interval $(1/n^2, 3/n^{1.5})$ of some typical values for the precisions.

When `uniform = FALSE`, the mixing weights are sampled from a symmetric Dirichlet distribution with shape parameters all equal to $2G$, to favor populated and balanced clusters; the consensus parameters are sampled to favor well-separated clusters, i. e., at least at Spearman distance $\frac{2}{G}\binom{n+1}{3}$ from each other.

Value

A list of the following named components:

`samples`	Integer $N \times n$ matrix with the `sample_size` simulated full rankings in each row.
`rho`	Integer $G \times n$ matrix with the component-specific consensus rankings used for the simulation in each row.
`theta`	Numeric vector of the G component-specific precision parameters used for the simulation.

`weights`	Numeric vector of the G mixture weights used for the simulation.
`classification`	Integer vector of the `sample_size` component membership labels.

References

Vitelli V, Sørensen Ø, Crispino M, Frigessi A and Arjas E (2018). Probabilistic Preference Learning with the Mallows Rank Model. *Journal of Machine Learning Research*, **18**(158), pages 1–49, ISSN: 1532-4435, https://jmlr.org/papers/v18/15-481.html.

Sørensen Ø, Crispino M, Liu Q and Vitelli V (2020). BayesMallows: An R Package for the Bayesian Mallows Model. *The R Journal*, **12**(1), pages 324–342, DOI: 10.32614/RJ-2020-026.

Chenyang Zhong (2021). Mallows permutation model with L1 and L2 distances I: hit and run algorithms and mixing times. arXiv: 2112.13456.

Examples

```
## Example 1. Drawing from a mixture with randomly generated parameters of separated clusters.
set.seed(12345)
rMSmix(sample_size = 50, n_items = 25, n_clust = 5)


## Example 2. Drawing from a mixture with uniformly generated parameters.
set.seed(12345)
rMSmix(sample_size = 100, n_items = 9, n_clust = 3, uniform = TRUE)


## Example 3.  Drawing from a mixture with customized parameters.
r_par <- rbind(1:5, c(4, 5, 2, 1, 3))
t_par <- c(0.01, 0.02)
w_par <- c(0.4, 0.6)
set.seed(12345)
rMSmix(sample_size = 50, n_items = 5, n_clust = 2, theta = t_par, rho = r_par, weights = w_par)
```

`spear_dist` *Spearman distance*

Description

Compute either the Spearman distance between each row of a full ranking matrix and a reference complete ranking, or the Spearman distance matrix between the rows of a full ranking matrix.

Usage

```
spear_dist(
  rankings,
  rho = NULL,
  subset = NULL,
  diag = FALSE,
```

```
  upper = FALSE,
  plot_dist_mat = FALSE
)
```

Arguments

`rankings`	Integer $N \times n$ matrix or data frame with full rankings in each row.
`rho`	An optional full ranking whose Spearman distance from each row in `rankings` must be computed. Defaults to `NULL`, meaning that the Spearman distance matrix between all pairs of rows in `rankings` must be computed.
`subset`	Optional logical or integer vector specifying the subset of observations, i.e. rows of the `rankings`, to be kept. Missing values are taken as `FALSE`.
`diag`	Logical: whether the diagonal of the Spearman distance matrix must be returned. Used when `rho` = `NULL`. Defaults to `FALSE`.
`upper`	Logical: whether the upper triangle of the Spearman distance matrix must be printed. Used when `rho` = `NULL`. Defaults to `FALSE`.
`plot_dist_mat`	Logical: whether the Spearman distance matrix must be plotted. Used when `rho` = `NULL`. Defaults to `FALSE`.

Details

When `rho` = `NULL`, `spear_dist` recalls the `dist` function from the `base` package to compute the squared Euclidian distance between full rankings; otherwise, it recalls the `compute_rank_distance` routine of the `BayesMallows` package, which implements several metrics for rankings.

Value

When `rho` = `NULL`, an object of class `"dist"` corresponding to the Spearman distance matrix; otherwise, a vector with the Spearman distances between each row in `rankings` and `rho`.

References

Sørensen Ø, Crispino M, Liu Q and Vitelli V (2020). BayesMallows: An R Package for the Bayesian Mallows Model. *The R Journal*, **12**(1), pages 324–342, DOI: 10.32614/RJ-2020-026.

See Also

`plot.dist`, `compute_rank_distance`

Examples

```
## Example 1. Spearman distance between two full rankings.
spear_dist(rankings = c(4, 8, 6, 9, 2, 11, 3, 5, 1, 12, 7, 10), rho = 1:12)

## Example 2. Spearman distance between the Antifragility ranking dataset and the Borda ranking.
r_antifrag <- ranks_antifragility[, 1:7]
borda <- rank(data_description(rankings = r_antifrag)$mean_rank)
spear_dist(rankings = r_antifrag, rho = borda)
```

```
## Example 3. Spearman distance matrix of the Sports ranking dataset.
r_sports <- ranks_sports[, 1:8]
dist_mat <- spear_dist(rankings = r_sports)
dist_mat
# Spearman distance matrix for the subsample of females.
dist_f <- spear_dist(rankings = r_sports, subset = (ranks_sports$Gender == "Female"))
dist_f
```

`spear_dist_distr` *Spearman distance distribution under the uniform ranking model*

Description

Provide (either the exact or the approximate) frequency distribution of the Spearman distance under the uniform (null) ranking model.

Usage

```
spear_dist_distr(n_items, log = TRUE)
```

Arguments

`n_items`	Number of items.
`log`	Logical: whether the frequencies must be reported on the log scale. Defaults to `TRUE`.

Details

When $n \leq 20$, the exact distribution provided by OEIS Foundation Inc. (2023) is returned by relying on a call to the `get_cardinalities` routine of the `BayesMallows` package. When $n > 20$, the approximate distribution introduced by Crispino et al. (2023) is returned. If $n > 170$, the approximation is also restricted over a fixed grid of values for the Spearman distance to limit computational burden.

Value

A list of two named objects:

`distances`	All the possible Spearman distance values (i.e., the support of the distribution).
`logcard`	(Log)-frequencies corresponding to each value in `distances`.

References

OEIS Foundation Inc. (2023). The On-Line Encyclopedia of Integer Sequences, Published electronically at https://oeis.org.

Crispino M, Mollica C, Astuti V and Tardella L (2023). Efficient and accurate inference for mixtures of Mallows models with Spearman distance. *Statistics and Computing*, **33**(98), DOI: 10.1007/s11222-023-10266-8.

Sørensen Ø, Crispino M, Liu Q and Vitelli V (2020). BayesMallows: An R Package for the Bayesian Mallows Model. *The R Journal*, **12**(1), pages 324–342, DOI: 10.32614/RJ-2020-026.

See Also

`spear_dist`, `expected_spear_dist`, `partition_fun_spear`

Examples

```
## Example 1. Exact Spearman distance distribution for n=20 items.
distr <- spear_dist_distr(n_items = 20, log = FALSE)
plot(distr$distances,distr$logcard,type='l',ylab = 'Frequency',xlab='d',
main='Distribution of the Spearman distance\nunder the null model')


## Example 2. Approximate Spearman distance distribution for n=50 items with log-frequencies.
distr <- spear_dist_distr(n_items = 50)
plot(distr$distances,distr$logcard,type='l',ylab = 'Log-frequency',xlab='d',
    main='Log-distribution of the Spearman distance\nunder the null model')
```

`summary.emMSmix` *Summary of the fitted mixture of Mallows models with Spearman distance*

Description

`summary` method for class `"emMSmix"`.

`print` method for class `"summary.emMSmix"`.

Usage

```
## S3 method for class 'emMSmix'
summary(object, digits = 3, ...)

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

Arguments

`object`	An object of class "`emMSmix`" returned by `fitMSmix`.
`digits`	Integer: decimal places for rounding the numerical summaries. Defaults to 3.
`...`	Further arguments passed to or from other methods (not used).
`x`	An object of class "`summary.emMSmix`" returned by `summary.emMSmix`.

Value

A list with the following named components:

`modal_rankings`	Integer matrix with the MLEs of the G component-specific consensus rankings in each row.
`modal_orderings`	Character matrix with the MLEs of the G component-specific consensus orderings in each row.
`theta`	Numeric vector of the MLEs of the G precisions.
`weights`	Numeric vector of the MLEs of the G mixture weights.
`MAP_distr`	Percentage distribution of the component memberships based on the MAP allocation. Returned when `n_clust` > `1`, otherwise `NULL`
`conv_perc`	Percentage of convergence of the EM algorithm over the multiple starting points.
`BIC`	BIC value (based on full or augmented rankings).
`BIC_part`	BIC value (based on partial rankings).
`call`	The matched call.

See Also

`fitMSmix`, `plot.emMSmix`

Examples

```
## Example 1. Fit and summary of a 3-component mixture of Mallow models with Spearman distance
## for the Antifragility dataset.
r_antifrag <- ranks_antifragility[, 1:7]
set.seed(123)
mms_fit <- fitMSmix(rankings = r_antifrag, n_clust = 3, n_start = 10)
summary(mms_fit)
```

`var_spear_dist` *Variance of the Spearman distance*

Description

Compute (either the exact or the approximate) (log-)variance of the Spearman distance under the Mallow model with Spearman distance.

Usage

```
var_spear_dist(theta, n_items, log = TRUE)
```

Arguments

`theta`	Non-negative precision parameter.
`n_items`	Number of items.
`log`	Logical: whether the expected Spearman distance on the log scale must be returned. Defaults to `TRUE`.

Details

When $n \leq 20$, the variance is exactly computed by relying on the Spearman distance distribution provided by OEIS Foundation Inc. (2023). When $n > 20$, it is approximated with the method introduced by Crispino et al. (2023) and, if $n > 170$, the approximation is also restricted over a fixed grid of values for the Spearman distance to limit computational burden.

When $\theta = 0$, this is equal to $\frac{n^2(n-1)(n+1)^2}{36}$, which is the variance of the Spearman distance under the uniform (null) model.

The variance of the Spearman distance is independent of the consensus ranking of the Mallow model with Spearman distance due to the right-invariance of the metric.

Value

Either the exact or the approximate (log-)variance of the Spearman distance under the Mallow model with Spearman distance.

References

Crispino M., Mollica C., Astuti V. and Tardella L. (2023). Efficient and accurate inference for mixtures of Mallows models with Spearman distance. *Statistics and Computing*, **33**(98), DOI: 10.1007/s11222-023-10266-8.

OEIS Foundation Inc. (2023). The On-Line Encyclopedia of Integer Sequences, Published electronically at `https://oeis.org`

Kendall, M. G. (1970). Rank correlation methods. 4th ed. Griffin London.

Examples

```
## Example 1. Variance of the Spearman distance under the uniform (null) model,
## coinciding with n^2(n-1)(n+1)^2/36.
n_items <- 10
var_spear_dist(theta = 0, n_items= n_items, log = FALSE)
n_items^2*(n_items-1)*(n_items+1)^2/36

## Example 2. Variance of the Spearman distance.
var_spear_dist(theta = 0.5, n_items = 10, log = FALSE)

## Example 3. Log-variance of the Spearman distance as a function of theta.
var_spear_dist_vec <- Vectorize(var_spear_dist, vectorize.args = "theta")
curve(var_spear_dist_vec(x, n_items = 10),
  from = 0, to = 0.1, lwd = 2, col = 2,
  xlab = expression(theta), ylab = expression(log(V[theta](D))),
  main = "Log-variance of the Spearman distance")

## Example 4. Log--variance of the Spearman distance for varying number of items
# and values of the concentration parameter.
var_spear_dist_vec <- Vectorize(var_spear_dist, vectorize.args = "theta")
curve(var_spear_dist_vec(x, n_items = 10),
  from = 0, to = 0.1, lwd = 2, col = 2, ylim = c(5, 14),
  xlab = expression(theta), ylab = expression(log(V[theta](D))),
  main = "Log-variance of the Spearman distance")
curve(var_spear_dist_vec(x, n_items = 20), add = TRUE, col = 3, lwd = 2)
curve(var_spear_dist_vec(x, n_items = 30), add = TRUE, col = 4, lwd = 2)
legend("topright", legend = c(expression(n == 10), expression(n == 20), expression(n == 30)),
  col = 2:4, lwd = 2, bty = "n")
```

Index