Package 'kernrank'

April 14, 2016

Title Kernel Methods for Rank Data

Version 1.0.1

URL https://github.com/YunlongJiao/kernrank

Description This package implements kernel functions and kernel methods for analyzing rank data, typically total rankings (namely permutations), interleaving and top-k partial rankings, multivariate rankings. This package is built upon the CRAN package RMallow and serves as a stable update and mostly a significant extension of that package.

Depends R (>= 3.2),

Imports combinat,

Suggests caret,
kernlab,
mvtnorm,
pcaPP,

License GPL-3

LazyData true

RoxygenNote 5.0.1.9000

R topics documented:

`AllKendall` *All Kendall's distances between two sets of total rankings or real-valued vectors*

Description

Calculates all of the Kendall's distances between two sets of total rankings or real-valued vectors.

Usage

```
AllKendall(r, seqs = NULL, data.info = NULL, use.kernel.trick = FALSE,
  kmat = NULL, type = c("type-b", "type-a"), mc = 0.25)
```

Arguments

`r`	A vector or a matrix of m1 sequences in rows and orders of n items in cols.
`seqs`	Another vector or a matrix of m2 sequences in rows and orders of n items in cols. By default "seqs" is set equal to "r".
`data.info`	Optional argument giving the Kendall embedding of "r", that is the result of KendallInfo(r), to facilitate computing Kendall's difference for "r" to "seqs" without exploring the kernel trick.
`use.kernel.trick`	Logical indicating whether the kernel trick is explored. This is particularly interesting when the number of items to be ranked is high and pcaPP::cor.fk() is available. By default (set FALSE), Kendall embedding is explicitly computed; otherwise kernel trick is explored.
`kmat`	Kendall kernel matrix of dimension m1 x m2, correlation type correponding to "type". If given, kernel trick is explored directly.
`type`	A character string indicating the type of Kendall correlation for "kmat".
`mc`	A normalization constant default to 0.25 such that output normalized squared Euclidean distance in the feature space induced by Kendall embedding amounts exactly to Kendall distances.

Value

A matrix of dimension m1 x m2 where entry [i,j] is the distance from sequence "i" in "r" to sequence "j" in "seqs".

Note

Kernel trick is explored in the sense that "r" and "seq" are only used for checking dimensions and getting attributes but not used explicitly to compute the distance. This is particularly interesting when data is high-dimensional in constrast to rather few observations (m1,m2»n). Option "use.kernel.trick" set TRUE or FALSE may give slightly different results due to computation precision.

Author(s)

Yunlong Jiao

References

Kendall rank correlation coefficient: https://en.wikipedia.org/wiki/Kendall_rank_correlation_coefficient

Jiao, Y., & Vert, J.-P. (2016). The Kendall and Mallows Kernels for Permutations. 2016. hal-01279273

Examples

```
#### Ex 1: compute Kendall distance matrix and Mallows kernel matrix
data1 <- do.call("rbind", list(1:5, 5:1, c(3, 2, 1, 4, 5)))
data2 <- do.call("rbind", list(1:5, 5:1))

# Kendall distance matrix
s.K.d.mat <- AllKendall(data1, data2)

# Mallows kernel matrix with dispersion parameter lambda
lambda <- 0.1
M.k.kmat <- exp(-lambda * s.K.d.mat)

#### Ex 2: why kernel trick?
r <- lapply(1:20, function(i) sample.int(1000, replace = TRUE))
r <- do.call('rbind', r)
dim(r)

# I) without kernel trick
pt <- proc.time()
dmat1 <- AllKendall(r, use.kernel.trick = FALSE)
proc.time() - pt

# II) with kernel trick (should be much faster in this setting)
require(pcaPP)
pt <- proc.time()
dmat2 <- AllKendall(r, use.kernel.trick = TRUE)
proc.time() - pt

# NOTE: dmat1 and dmat2 may return slightly different values due to computation precision
isTRUE(all.equal(dmat1, dmat2, check.attributes = FALSE)) # may sometimes output FALSE
isTRUE(max(abs(dmat1 - dmat2)) < 1e-6) # always output TRUE
```

`DistanceDistribution` *Calculate the Kendall distance distribution in N! space.*

Description

This function counts the number of fully-ordered vectors at each distance in N! space.

Usage

```
DistanceDistribution(N = 3)
```

Arguments

N	Integer value, greater than or equal to 3.

Value

Table-like structure, where the names represent the distance from the modal sequence of each sequence in N! space, and the values represent the number of sequences at that distance in the sequence space.

Note

Taken directly from CRAN package RMallow.

Author(s)

Erik Gregory

Examples

```
## Not run:

# DistanceDistribution(10)

## End(Not run)
```

KendallInfo *Kendall embedding of pairwise relative ordering*

Description

Performs between-column comparison on a matrix of sequences denoting sign([,i] - [,j]) for i < j.

Usage

```
KendallInfo(r)
```

Arguments

r	A vector or a matrix of dimension N x n with sequences in rows.

Value

A matrix of dimension N x choose(n,2) with entry values -1/1/0 representing pairwise comparisons of vector values for each row. Specifically, a -1 value denotes that there is an increase between the two columns, 1 a decrease, and 0 indicates that the column values are identical in the same row.

Note

A matrix with one row is returned if the input "r" is a vector.

Author(s)

Yunlong Jiao

References

Jiao, Y., & Vert, J.-P. (2016). The Kendall and Mallows Kernels for Permutations. 2016. hal-01279273

Examples

```
r <- do.call('rbind', combinat::permn(1:5))
KendallInfo(r)
```

---

kendall_partial *Kendall kernel for interleaving partial rankings*

Description

Calculates Kendall kernel between interleaving partial rankings in time O (klogk), where ties (supposed few) are broken by adopting a convolution kernel averaging compatible rankings without ties.

Usage

```
kendall_partial(x, y)
```

Arguments

x	Vector. If x is numeric, the rank vector converted from x indicate that larger values mean being preferred. NAs replace unobserved values.
y	Same as x.

Value

Kendall kernel for interleaving partial rankings defined as kendall tau averaged over all compatible full ranking.

Author(s)

Yunlong Jiao

References

Jiao, Y., & Vert, J.-P. (2016). The Kendall and Mallows Kernels for Permutations. 2016. hal-01279273

Examples

```
x <- c(1.5, 0.1, NA, -4, NA)
y <- c(NA, NA, 0, 3, NA)
kendall_partial(x, y)
```

kendall_top *Kendall kernel for top-k rankings*

Description

Calculates Kendall kernel between top-k rankings in time O (klogk), where ties (supposed few) are broken by adopting a convolution kernel averaging compatible rankings without ties.

Usage

```
kendall_top(x, y)
```

Arguments

x	Vector. If x is numeric, the rank vector converted from x indicate that larger values mean being preferred. NAs replace unobserved values.
y	Same as x.

Value

Kendall kernel for top-k rankings defined as kendall tau averaged over all compatible full ranking.

Author(s)

Yunlong Jiao

References

Jiao, Y., & Vert, J.-P. (2016). The Kendall and Mallows Kernels for Permutations. 2016. hal-01279273

Examples

```
x <- c(1.5, 0.1, NA, -4, NA)
y <- c(NA, NA, 0, 3, NA)
kendall_top(x, y)
```

kendall_total *Kendall kernel for total rankings*

Description

Calculates Kendall kernel between total rankings in time O (nlogn), where ties are dealt with type-b (soft version) of kendall kernel

Usage

```
kendall_total(x, y = NULL)
```

Arguments

x	A vector or a matrix of m1 sequences in rows and orders of n items in cols. If x is numeric, the rank vector converted from x indicate that larger values mean being preferred. NAs are not allowed.
y	Same as x. By default "y" is set equal to "x".

Value

Kendall kernel for total rankings defined as type-b (soft version) of kendall kernel.

References

Kendall rank correlation coefficient: https://en.wikipedia.org/wiki/Kendall_rank_correlation_coefficient

Jiao, Y., & Vert, J.-P. (2016). The Kendall and Mallows Kernels for Permutations. 2016. hal-01279273

Examples

```
x <- c(1.5, 0.1, 0, -4, 0)
y <- c(0, 0, 0, 3, 0)
kendall_total(x, y)
```

LogSumExp *Computing Log-Sum-Exp with a common trick*

Description

Computes log-sum-exp of a series of (typically large in absolute value) numbers with a more accurate computational trick typically useful for small values

Usage

```
LogSumExp(x, byrow = TRUE, bycol = !byrow)
```

Arguments

x	A vector or a matrix of numerics (typicall very small).
byrow	Logical. Computes by rows if a matrix "x" is provided.
bycol	Logical. Computes by cols if a matrix "x" is provided.

Value

Log-Sum-Exp of the numbers in the vector "x", that is log(sum(exp(x))), or row-/column-wise Log-Sum-Exp of the numbers in matrix "x".

Author(s)

Yunlong Jiao

References

Computing Log-Sum-Exp: https://hips.seas.harvard.edu/blog/2013/01/09/computing-log-sum-exp/

Examples

```
x <- c(-1000, -999, -1000)
LogSumExp(x)
```

`Mallows` *Fits a Mallows mixture model to ranking data*

Description

Fits the Mallows mixture model to total rankings, using EM algorithm, for clustering permutations.

Usage

```
Mallows(datas, G, weights = NULL, iter = 100, tol = 0.001,
  logsumexp.trick = TRUE, iterin = iter, key = c("copelandMallows",
  "bruteMallows", "bordaMallows", "kernelMallows", "kernelMallows_Exh",
  "copelandMallows_Eqlam", "bruteMallows_Eqlam", "bordaMallows_Eqlam",
  "kernelMallows_Eqlam", "kernelMallows_Exh_Eqlam", "kernelGaussian",
  "kernelGaussian_Eqlam"), exhkey = "_Exh", eqlamkey = "_Eqlam")
```

Arguments

`datas`	Matrix of dimension N x n with sequences in rows.
`G`	Number of modes, 2 or greater.
`weights`	Numeric vector of length N denoting frequencies of each permutation observed. Each observation is observed once by default. Notably it must not contain 0 and should be of equal length with nrow(datas).
`iter`	Maximum number of iterations for EM algorithm.
`tol`	Stopping precision.
`logsumexp.trick`	Logical. Whether or not to use log-sum-exp trick to compute log-likelihood.
`iterin`	Maximum number of iterations for alternate optimization between centers and lambda. Effective only when performing kernel Mallows with exhaustive optimization.
`key`	A character string defining the type of Mallows mixture model to perform: • *copelandMallows* denotes original Mallows mixture model with cluster centers found by Copeland's method • *bruteMallows* denotes original Mallows mixture model with cluster centers found by brute-force search for optimal Kemeny consensus (not applicable for large n) • *bordaMallows* denotes original Mallows mixture model with cluster centers as Borda count

- *kernelMallows* denotes kernel version of Mallows mixture model with cluster centers as the barycenter in Euclidean space induced by Kendall embedding
- *kernelGaussian* denotes Gaussian mixture model in the Euclidean space induced by Kendall embedding

`exhkey` A character string. If it greps successfully in "key", an alternate optimization between centers and lambda. Effective only when performing *kernelMallows* with exhaustive optimization.

`eqlamkey` A character string. If it greps successfully in "key", the dispersion parameters (lambda) are constrained to be equal for all clusters; otherwise no constraints on lambda.

Value

List.

`key` Character string indicating the type of Mallows mixture model performed

`R` List of length "G" of cluster centers, each entry being a permutation of length n if original Mallows mixture model is performed, or a numeric vector of length choose(n,2) if kernel version is performed

`p` Numeric vector of length "G" representing the proportion probability of each cluster

`lambda` Numeric vector of length "G" representing the dispersion parameters of each cluster

`datas` A copy of "datas" on which the Mallows mixture model is fitted, combined with "weights", fuzzy assignment membership probability "z", distances to centers in "R"

`min.like` Numeric vector of length "iter" representing fitted likelihood values at each iteration

Author(s)

Yunlong Jiao

References

Murphy, T. B., & Martin, D. (2003). Mixtures of distance-based models for ranking data. Computational statistics & data analysis, 41(3), 645-655.

Jiao, Y., & Vert, J.-P. (2016). The Kendall and Mallows Kernels for Permutations. 2016. hal-01279273

Examples

```
datas <- do.call('rbind', combinat::permn(1:5))
G <- 3
weights <- runif(nrow(datas))

# fit Mallows mixture model
model <- Mallows(datas, G, weights, key = 'bordaMallows')
str(model)
```

`MallowsCV` *Compute cross-validated likelihood for Mallows mixture models*

Description

Assess model performance by cross-validated (CV) Mallows likelihood. Do NOT run for large number of ranked alternatives "n".

Usage

```
MallowsCV(datas, G, weights = NULL, ..., seed = 26921332, nfolds = 5,
  nrepeats = 10, ntry = 3, logsumexp.trick = TRUE)
```

Arguments

`datas`	Matrix of dimension N x n with sequences in rows.
`G`	Number of modes, 2 or greater.
`weights`	Integer vector of length N denoting frequencies of each permutation observed. Each observation is observed once by default. Notably it must not contain 0 and should be of equal length with nrow(datas).
`...`	Arguments passed to Mallows().
`seed`	Seed index for reproducible results.
`nfolds`	"nfold"-fold CV created each time.
`nrepeats`	CV repeated "nrepeats" times.
`ntry`	Number of random initializations to restart for each CV run. The best fit returning max likelihood is reported.
`logsumexp.trick`	Logical. Whether or not to use logsumexp trick to compute log-likelihood.

Value

List of length nfolds*nrepeats, each entry being the result on each fold containing:

`...`	See output of Mallows()
`cv.loglik`	Likelihood value assessed against test fold while the mixture model is trained on the training fold

Note

CV split is done by partitioning "weights" so that "weights" must be integers.

Author(s)

Yunlong Jiao

References

Murphy, T. B., & Martin, D. (2003). Mixtures of distance-based models for ranking data. Computational statistics & data analysis, 41(3), 645-655.

Jiao, Y., & Vert, J.-P. (2016). The Kendall and Mallows Kernels for Permutations. 2016. hal-01279273

Examples

```
datas <- do.call('rbind', combinat::permn(1:5))
G <- 3
weights <- rbinom(nrow(datas), 100, 0.5) # positive integers

# cross validate Mallows mixture model
cv.model <- MallowsCV(datas, G, weights, key = 'bordaMallows', nfolds = 3, nrepeats = 1)
# averaged cv.loglik over all CV folds
mean(sapply(cv.model, function(model) model$cv.loglik))
```

RankAggreg *Common ranking aggregation methods for permutations*

Description

Used to update modal sequences of each cluster in the EM algorithm when fitting Mallows mixture models.

Usage

```
RankAggreg(r, z = NULL, infos = NULL, perm = NULL, key = c("borda",
  "copeland", "brute"))
```

Arguments

r	A vector or a matrix of sequences in rows.
z	A vector of weights/frequencies of observations or a matrix of probability of cluster membership for each sequence and each cluster. Set by default a constant vector of 1.
infos	The result of KendallInfo(r). Optional for speeding up computation.
perm	A matrix of full permutations for brute-force search of optimal Kemeny consensus. Only effective for "key" equal to "brute".
key	A character string indicating the ranking aggregation method to find centers. • *borda* denotes the Borda count • *copeland* denotes the Copeland's aggregated ranking • *brute* denotes the optimal Kemeny consensus found by brute-force search

Value

List of length 1 if "z" is a vector, or ncol(z) if "z" is a matrix, each entry being the modal sequence in each cluster.

Author(s)

Yunlong Jiao

Examples

```
r <- do.call("rbind", list(1:5, 5:1, c(2,4,1,5,3)))
RankAggreg(r, key = "borda") # Borda count for sequences in "r"
```

Index