

Package: RMallow (via r-universe)

October 4, 2026

Type Package

Title Fit Multi-Modal Mallows' Models to Ranking Data

Version 1.2.0

Author Erik Gregory [aut, cre]

Maintainer Erik Gregory <egregory2011@gmail.com>

Description An EM algorithm to fit Mallows' mixture models to full or partial rankings, with or without ties. The Mallows model is a distance-based probability model over rankings, where probability decreases exponentially with distance from a central (modal) ranking. This package implements mixture models to handle multi-modal ranking data, as described in Murphy and Martin (2003) <doi:10.1016/S0167-9473(02)00165-2> and Adkins and Fligner (1998) <doi:10.1080/03610929808832223>.

License GPL (>= 2)

URL <https://github.com/erikgregory/RMallow>

BugReports <https://github.com/erikgregory/RMallow/issues>

Depends R (>= 3.5.0)

Imports combinat, grDevices, graphics, stats, Rcpp (>= 1.0.0)

LinkingTo Rcpp

Suggests testthat (>= 3.0.0), knitr, rmarkdown

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Config/testthat/edition 3

NeedsCompilation yes

Repository <https://e-gregory.r-universe.dev>

Date/Publication 2025-12-07 07:33:53 UTC

RemoteUrl <https://github.com/e-gregory/rmallow>

RemoteRef HEAD

RemoteSha 4db25ce8e26e62795b59c252d1c12d8b7d8c7c75

Contents

RMallow-package	2
AllKendall	3
AllSeqDists	4
BestFit	5
C_lam	5
ConstructSeqs	6
datas	7
DistanceDistribution	7
elect	8
EStep	9
FormatOut	10
KendallInfo	11
Lambda	12
Likelihood	12
Mallows	13
NextTable	14
Rgen	15
SeqDistribution	16
SimplifySequences	17
three.mode	18
two.mode	18
two.seq	19
UpdateLambda	20
UpdateP	21
UpdateR	22
Index	23

RMallow-package	<i>RMallow: Fit Multi-Modal Mallows' Models to Ranking Data</i>
-----------------	---

Description

Fits the Mallows' model to ranking data using an EM algorithm. Data can be partially or fully-ranked, with or without ties.

Details

The main functions are:

- [Mallows](#): Fit a Mallows mixture model
- [BestFit](#): Fit multiple models and select the best

Distance computation functions:

- [AllKendall](#): Compute Kendall distances between rankings

- [KendallInfo](#): Compute pairwise comparison information
- [DistanceDistribution](#): Distance distribution in $N!$ space

Utility functions:

- [SimplifySequences](#): Normalize tied rankings
- [ConstructSeqs](#): Reconstruct sequences from preferences

Author(s)

Erik Gregory <egregory2011@gmail.com>

References

- Murphy, T.B. and Martin, D. (2003). Mixtures of distance-based models for ranking data. Computational Statistics & Data Analysis, 41, 645-655. doi:[10.1016/S01679473\(02\)001652](#)
- Beckett, L.A. and Evans, D.A. (1991). Estimating a Population Distribution of Sequences of k Items from Cross-Sectional Data. Journal of the Royal Statistical Society, Series C, 40(1), 31-42.
- Adkins, L. and Fligner, M. (1998). A Non-iterative procedure for maximum likelihood estimation of the parameters of Mallows' Model Based on Partial Rankings. Communications in Statistics - Theory and Methods, 27(9), 2199-2220. doi:[10.1080/03610929808832223](#)

AllKendall

Compute Kendall's distances between two sets of rankings

Description

Calculates all Kendall's distances between each ranking in *r* and each sequence in *seqs*.

Usage

```
AllKendall(r, seqs, data_info = NULL, use_cpp = TRUE)
```

Arguments

<i>r</i>	Matrix of rankings (each row is a ranking).
<i>seqs</i>	Matrix of reference sequences (each row is a sequence).
<i>data_info</i>	Optional precomputed Kendall information matrix for <i>r</i> . If NULL, it will be computed internally. Providing this can improve efficiency when calling the function repeatedly with the same data.
<i>use_cpp</i>	Logical; if TRUE (default), use the fast C++ implementation.

Value

Matrix where entry $[i, j]$ represents the Kendall distance from ranking *i* in *r* to sequence *j* in *seqs*.

Author(s)

Erik Gregory

Examples

```
data1 <- rbind(1:5, 5:1, c(3, 2, 1, 4, 5))
data2 <- rbind(1:5, 5:1)
AllKendall(data1, data2)
```

AllSeqDists

Compute distances from sequences to the canonical ordering

Description

Calculates the Kendall distance from each sequence to the canonical ordering (1, 2, ..., N).

Usage

```
AllSeqDists(seqs, use_cpp = TRUE)
```

Arguments

seqs Matrix or data frame of sequences, where each row is a ranking.

use_cpp Logical; if TRUE (default), use the fast C++ implementation.

Value

Integer vector of distances from each sequence to 1:N.

Author(s)

Erik Gregory

Examples

```
seqs <- rbind(1:5, 5:1, c(1, 3, 2, 4, 5))
AllSeqDists(seqs)
# Returns: 0, 10, 1
```

BestFit*Fit Mallows model multiple times and select best*

Description

The EM algorithm for Mallows' model can converge to local maxima. This function runs the algorithm multiple times with different random initializations and returns the model with the highest likelihood.

Usage

```
BestFit(datas, N, iter, G, verbose = TRUE)
```

Arguments

<code>datas</code>	Matrix of rankings to fit (N x n_items).
<code>N</code>	Number of times to run the model with different initializations.
<code>iter</code>	Maximum number of EM iterations for each run.
<code>G</code>	Number of cluster centers (modal sequences).
<code>verbose</code>	Logical; if TRUE, print progress messages.

Value

The model fit with the highest log-likelihood. See [Mallows](#) for details of the return value.

Author(s)

Erik Gregory

See Also

[Mallows](#)

C_lam*Compute the normalizing constant for Mallows' model*

Description

Calculates the normalizing constant $C(\lambda)$ for Mallows' model in a sequence space of size $N!$.

Usage

```
C_lam(lambda, dists = NULL, dists_table = NULL)
```

Arguments

<code>lambda</code>	Spread parameter for Mallows' model (non-negative).
<code>dists</code>	Optional vector of all distances from each sequence to the modal sequence 1:N. If NULL, <code>dists_table</code> must be provided.
<code>dists_table</code>	Optional table of distance counts. If NULL, computed from <code>dists</code> . This is more efficient for repeated calls.

Value

The normalizing constant $C(\lambda)$.

Author(s)

Erik Gregory

Examples

```
# For a 4-item ranking space
dist_tab <- DistanceDistribution(4)
C_lam(0.5, dists_table = dist_tab)
C_lam(1.0, dists_table = dist_tab)
```

ConstructSeqs

Construct sequences from Kendall preference vectors

Description

Reconstructs ranking sequences from their Kendall information representation. Each fully-ordered sequence has a unique Kendall information vector.

Usage

```
ConstructSeqs(prefs, n_abils, use_cpp = TRUE)
```

Arguments

<code>prefs</code>	Matrix of pairwise preferences, where each row represents one sequence's preferences. A value of 1 indicates a decrease (first element > second), 0 indicates an increase.
<code>n_abils</code>	Number of items in the original ranking.
<code>use_cpp</code>	Logical; if TRUE (default), use the fast C++ implementation.

Value

List of integer vectors, each representing a reconstructed ranking.

Author(s)

Erik Gregory

Examples

```
# Reconstruct a sequence where item 4 is ranked first
# For 4 items, there are  $4*3/2 = 6$  pairwise comparisons
prefs <- matrix(c(1, 1, 1, 0, 0, 0), nrow = 1)
ConstructSeqs(prefs, 4)
# Returns list(c(4, 1, 2, 3))

# Multiple sequences at once
prefs2 <- rbind(c(0, 0, 0, 0, 0, 0), c(1, 1, 1, 1, 1, 1))
ConstructSeqs(prefs2, 4)
# Returns list(c(1,2,3,4), c(4,3,2,1))
```

 datas

Sample synthetic data set

Description

Synthetic data set containing 3 modal sequences in $15!$ space, with some noise added. Useful for testing and demonstrating the package.

Format

A numeric matrix with 1700 rows (observations) and 15 columns (items).

Examples

```
data(datas)
head(datas)
dim(datas)
```

 DistanceDistribution

Compute the Kendall distance distribution in $N!$ space

Description

Efficiently computes the count of permutations at each Kendall distance from the canonical ordering (1, 2, ..., N). Uses a recurrence relation rather than enumeration, making it efficient even for large N.

Usage

```
DistanceDistribution(N = 3, use_cpp = TRUE)
```

Arguments

N Integer value, the length of rankings (must be ≥ 1).

use_cpp Logical; if TRUE (default), use the fast C++ implementation.

Value

Named numeric vector where names are distances (0 to $N*(N-1)/2$) and values are counts of permutations at that distance.

Author(s)

Erik Gregory

See Also

[SeqDistribution](#) for the brute-force approach

Examples

```
# Distribution for 5-item rankings
dist_tab <- DistanceDistribution(5)
dist_tab

# Verify total equals 5!
sum(dist_tab)

# Works efficiently even for large N
dist_tab_20 <- DistanceDistribution(20)
length(dist_tab_20) # Maximum distance is  $20*19/2 = 190$ 
```

elect

1980 APA Presidential Candidate Ranking Data

Description

Pre-processed version of the 1980 American Psychological Association Presidential candidate ranking data. Uninformative rankings have been removed and values have been pre-simplified into partial rankings.

Format

An integer matrix with 1378 rows (voters) and 3 columns (Carter, Reagan, Anderson).

Source

American National Election Studies, <https://electionstudies.org/>

Examples

```
data(elect)
head(elect)
table(elect[, "Carter"]) # Distribution of Carter rankings
```

EStep

*E-Step of the EM algorithm***Description**

Computes the probability that each ranking belongs to each cluster, given the current model parameters.

Usage

```
EStep(R, r, p, lambda, G, N, C, all_dists = NULL, use_cpp = TRUE)
```

Arguments

R	List of current cluster modal sequences.
r	Matrix of rankings (N x n_items).
p	Vector of cluster proportions (length G).
lambda	Vector of spread parameters (length G).
G	Number of clusters.
N	Number of observations.
C	Vector of normalizing constants (length G).
all_dists	Optional precomputed distance matrix (N x G). If NULL, computed from r and R.
use_cpp	Logical; if TRUE (default), use the fast C++ implementation.

Value

Matrix (N x G) where entry $[i, j]$ is the probability that observation i belongs to cluster j .

Author(s)

Erik Gregory

References

Murphy, T.B. and Martin, D. (2003). Mixtures of distance-based models for ranking data. *Computational Statistics & Data Analysis*, 41, 645-655.

FormatOut	<i>Format Mallows model output</i>
-----------	------------------------------------

Description

Formats the results of the Mallows model fitting procedure into a structured list with cluster assignments and diagnostics.

Usage

```
FormatOut(R, p, lambda, z, datas, likelihood)
```

Arguments

R	List of modal sequences (length G).
p	Vector of cluster proportions (length G).
lambda	Vector of spread parameters (length G).
z	Matrix of membership probabilities (N x G).
datas	Original data matrix of rankings (N x n_items).
likelihood	Vector of log-likelihood values at each iteration.

Value

List with components:

R Modal sequences for each cluster

p Proportion of data in each cluster

lambda Spread parameters for each cluster

datas Data frame with original rankings plus cluster assignments, membership probabilities, assigned modal sequence, and distances to each cluster center

min.like Log-likelihood at each iteration

Author(s)

Erik Gregory

KendallInfo*Compute pairwise comparison information for Kendall's distance*

Description

Performs each column-wise comparison on a matrix of sequences. A 0 value denotes an increase between two columns, 1 denotes a decrease, and NA indicates tied values.

Usage

```
KendallInfo(r, inds = NULL, use_cpp = TRUE)
```

Arguments

<code>r</code>	Matrix of sequences, where each row is a ranking.
<code>inds</code>	Optional matrix of column index pairs for comparisons. If NULL, all pairs are computed using <code>combn(ncol(r), 2)</code> .
<code>use_cpp</code>	Logical; if TRUE (default), use the fast C++ implementation.

Value

Matrix of 0s, 1s, and NAs representing pairwise comparisons. Each column corresponds to a pair of positions in the original ranking.

Author(s)

Erik Gregory

References

https://en.wikipedia.org/wiki/Kendall_tau_distance

Examples

```
# Full ranking
r <- matrix(c(1, 2, 3, 4, 5, 5, 4, 3, 2, 1), nrow = 2, byrow = TRUE)
KendallInfo(r)

# Ranking with ties
r_ties <- matrix(c(1, 1, 2, 3, 3), nrow = 1)
KendallInfo(r_ties)
```

Lambda	<i>Objective function for lambda estimation</i>
--------	---

Description

Computes the objective function whose root gives the MLE of lambda in Mallows' model. Used internally by [UpdateLambda](#).

Usage

```
Lambda(lambda, rhs, dists = NULL, dists_table = NULL)
```

Arguments

lambda	Lambda value to evaluate the function at.
rhs	Right-hand side of the estimating equation (weighted mean distance).
dists	Not used (kept for backward compatibility).
dists_table	Table of distance distribution in $N!$ space.

Value

Value of the objective function. The goal is to find lambda where this equals zero.

Author(s)

Erik Gregory

References

Murphy, T.B. and Martin, D. (2003). Mixtures of distance-based models for ranking data. *Computational Statistics & Data Analysis*, 41, 645-655.

Likelihood	<i>Compute log-likelihood of Mallows mixture model</i>
------------	--

Description

Calculates the complete-data log-likelihood of the rankings given the current model parameters.

Usage

```
Likelihood(z, p, C_lam, lambda, all_dists_data)
```

Arguments

<code>z</code>	Matrix of cluster membership probabilities (N x G).
<code>p</code>	Vector of cluster proportions (length G).
<code>C_lam</code>	Vector of normalizing constants (length G).
<code>lambda</code>	Vector of spread parameters (length G).
<code>all_dists_data</code>	Matrix of distances from data to modal sequences (N x G).

Value

Numeric log-likelihood value.

Author(s)

Erik Gregory

References

Murphy, T.B. and Martin, D. (2003). Mixtures of distance-based models for ranking data. *Computational Statistics & Data Analysis*, 41, 645-655.

Mallows

Fit a Multi-Modal Mallows' Model to Ranking Data

Description

Fits a mixture of Mallows' models to partial or full ranking data using an EM algorithm with Kendall's distance metric.

Usage

```
Mallows(datas, G, iter = 100L, hyp = NULL, plot_like = FALSE,
        verbose = TRUE, tol = 1e-06, use_cpp = TRUE)
```

Arguments

<code>datas</code>	Matrix of partial or fully-ranked data (N x n_items).
<code>G</code>	Number of modes (clusters), must be ≥ 1 .
<code>iter</code>	Maximum number of EM iterations (default 100).
<code>hyp</code>	Optional hypothesis sequence to initialize one cluster center.
<code>plot_like</code>	Logical; if TRUE, plot likelihood at each iteration.
<code>verbose</code>	Logical; if TRUE, print progress messages.
<code>tol</code>	Convergence tolerance. Algorithm stops when likelihood change is less than this value.
<code>use_cpp</code>	Logical; if TRUE (default), use the fast C++ implementations.

Value

List with components:

R List of modal sequences for each cluster

p Vector of cluster proportions

lambda Vector of spread parameters

datas Data frame with cluster assignments and diagnostics

min.like Vector of log-likelihood at each iteration

Author(s)

Erik Gregory

References

Murphy, T.B. and Martin, D. (2003). Mixtures of distance-based models for ranking data. *Computational Statistics & Data Analysis*, 41, 645-655.

Adkins, L. and Fligner, M. (1998). A Non-iterative procedure for maximum likelihood estimation of the parameters of Mallows' Model Based on Partial Rankings. *Communications in Statistics - Theory and Methods*, 27:9, 2199-2220.

NextTable

Compute distance distribution for $(N+1)!$ from $N!$ space

Description

Given the Kendall distance distribution in $N!$ space, computes the distribution in $(N+1)!$ space. This uses the recurrence relation for Mahonian numbers (number of permutations with k inversions).

Usage

```
NextTable(last_table, n_last, use_cpp = TRUE)
```

Arguments

last_table Table or numeric vector of distance counts in $N!$ space.

n_last The value of N (current space is $N!$).

use_cpp Logical; if TRUE (default), use the fast C++ implementation.

Value

Numeric vector of distance counts in $(N+1)!$ space.

Author(s)

Erik Gregory

Examples

```
# Get distribution for 5! from 4!
tab4 <- DistanceDistribution(4)
tab5 <- NextTable(tab4, 4)
# Verify: should have 5! = 120 total permutations
sum(tab5)
```

Rgen

Initialize cluster modal sequences

Description

Generates initial modal sequences for the Mallows mixture model clustering. If a hypothesis sequence is provided, it becomes the first cluster center. Remaining cluster centers are initialized randomly.

Usage

```
Rgen(G, hyp = NULL, abils)
```

Arguments

G	Number of cluster centers (modes).
hyp	Optional hypothesis sequence to use as the first cluster center. Must be a permutation of 1:abils.
abils	Number of items being ranked.

Value

List of G integer vectors, each a permutation of 1:abils.

Author(s)

Erik Gregory

Examples

```
# Three random cluster centers for 5-item rankings
set.seed(42)
Rgen(3, abils = 5)

# With a specific hypothesis sequence
Rgen(3, hyp = c(5, 4, 3, 2, 1), abils = 5)
```

SeqDistribution	<i>Compute all Kendall distances in $N!$ space by enumeration</i>
-----------------	--

Description

Computes Kendall's distance from 1:N to each permutation in $N!$ space by enumerating all permutations. This is computationally expensive for $N \geq 8$. For larger N , use [DistanceDistribution](#) instead.

Usage

```
SeqDistribution(N, use_cpp = TRUE)
```

Arguments

N	Length of the ranking. Should be less than 9 for practical use.
use_cpp	Logical; if TRUE (default), use the fast C++ implementation.

Value

Integer vector of Kendall distances from 1:N to each permutation.

Author(s)

Erik Gregory

See Also

[DistanceDistribution](#) for an efficient alternative

Examples

```
# All distances in 4! space
dists <- SeqDistribution(4)
table(dists)
```

SimplifySequences	<i>Simplify tied rankings to consecutive integers</i>
-------------------	---

Description

Transforms rankings so that tie groups use consecutive integers. For example, (1, 1, 2, 4, 4, 5) becomes (1, 1, 2, 3, 3, 4).

Usage

```
SimplifySequences(rankings, use_cpp = TRUE)
```

Arguments

rankings	Matrix of rankings to simplify, where each row is a ranking.
use_cpp	Logical; if TRUE (default), use the fast C++ implementation.

Value

Matrix of simplified rankings with the same dimensions.

Author(s)

Erik Gregory

Examples

```
# Single ranking with gaps
rankings <- matrix(c(1, 1, 2, 4, 4, 5), nrow = 1)
SimplifySequences(rankings)
# Returns: 1 1 2 3 3 4

# Multiple rankings
rankings <- rbind(
  c(1, 3, 3, 5),
  c(2, 2, 4, 6)
)
SimplifySequences(rankings)
```

three.mode	<i>Three-mode Mallows model fit to synthetic data</i>
------------	---

Description

The `datas` data set fitted with 3 modal sequences. Compare to [two.mode](#) to see the effect of model selection.

Format

A list with components:

R List of 3 modal sequences

p Vector of cluster proportions

lambda Vector of spread parameters

datas Data frame with cluster assignments

min.like Log-likelihood trajectory

See Also

[two.mode](#), [datas](#)

Examples

```
data(three.mode)
three.mode$R # Modal sequences
three.mode$p # Cluster proportions
```

two.mode	<i>Two-mode Mallows model fit to synthetic data</i>
----------	---

Description

The `datas` data set fitted with only 2 modal sequences. Since the true data has 3 modes, this shows what happens with model misspecification.

Format

A list with components:

R List of 2 modal sequences

p Vector of cluster proportions

lambda Vector of spread parameters

datas Data frame with cluster assignments

min.like Log-likelihood trajectory

See Also[three.mode](#), [datas](#)**Examples**

```
data(two.mode)
two.mode$R # Only 2 modal sequences
```

two.seq

Two-mode Mallows model fit to APA election data

Description

The elect data fitted with 2 modal sequences. The two modes appear to correspond roughly to Democratic and Republican voting patterns.

Format

A list with components:

R List of 2 modal sequences

p Vector of cluster proportions

lambda Vector of spread parameters

datas Data frame with cluster assignments

min.like Log-likelihood trajectory

Source

American National Election Studies

See Also[elect](#)**Examples**

```
data(two.seq)
two.seq$R # Modal sequences (voting patterns)
two.seq$p # Proportion in each group
```

UpdateLambda	<i>Update spread parameters (M-step)</i>
--------------	--

Description

Updates the lambda (spread) parameters for each cluster to maximize the likelihood given current membership probabilities.

Usage

```
UpdateLambda(r, R, z, G, dists_to_Rg, dists_table, top_bound = 1000)
```

Arguments

<code>r</code>	Matrix of rankings (N x n_items).
<code>R</code>	List of current modal sequences.
<code>z</code>	Matrix of membership probabilities (N x G).
<code>G</code>	Number of clusters.
<code>dists_to_Rg</code>	Matrix of distances from data to modal sequences (N x G).
<code>dists_table</code>	Table of distance distribution in N! space.
<code>top_bound</code>	Maximum allowed value for lambda (default 1000).

Value

Vector of updated lambda parameters (length G).

Author(s)

Erik Gregory

References

Murphy, T.B. and Martin, D. (2003). Mixtures of distance-based models for ranking data. Computational Statistics & Data Analysis, 41, 645-655.

UpdateP	<i>Update cluster proportions (M-step)</i>
---------	--

Description

Updates the proportion of data assigned to each cluster based on current membership probabilities.

Usage

```
UpdateP(z)
```

Arguments

z Matrix of membership probabilities (N x G).

Value

Vector of updated cluster proportions (length G).

Author(s)

Erik Gregory

References

Murphy, T.B. and Martin, D. (2003). Mixtures of distance-based models for ranking data. Computational Statistics & Data Analysis, 41, 645-655.

Examples

```
# Example membership matrix for 10 observations, 3 clusters
z <- matrix(c(
  0.9, 0.05, 0.05,
  0.8, 0.1, 0.1,
  0.1, 0.8, 0.1,
  0.1, 0.1, 0.8,
  0.33, 0.33, 0.34,
  0.9, 0.05, 0.05,
  0.05, 0.9, 0.05,
  0.05, 0.05, 0.9,
  0.7, 0.2, 0.1,
  0.2, 0.3, 0.5
), nrow = 10, byrow = TRUE)
UpdateP(z)
```

`UpdateR`*Update modal sequences (M-step)*

Description

Updates the cluster centers (modal sequences) to maximize the likelihood given current membership probabilities.

Usage

```
UpdateR(r, z, infos = NULL, use_cpp = TRUE)
```

Arguments

<code>r</code>	Matrix of rankings (N x n_items).
<code>z</code>	Matrix of membership probabilities (N x G).
<code>infos</code>	Optional precomputed Kendall information matrix for <code>r</code> .
<code>use_cpp</code>	Logical; if TRUE (default), use the fast C++ implementation.

Value

List of G updated modal sequences.

Author(s)

Erik Gregory

References

Murphy, T.B. and Martin, D. (2003). Mixtures of distance-based models for ranking data. *Computational Statistics & Data Analysis*, 41, 645-655.

Index

- * **BubbleSort**
 - FormatOut, [10](#)
 - * **Distance**
 - AllSeqDists, [4](#)
 - KendallInfo, [11](#)
 - * **Kendall**
 - AllKendall, [3](#)
 - AllSeqDists, [4](#)
 - DistanceDistribution, [7](#)
 - FormatOut, [10](#)
 - KendallInfo, [11](#)
 - NextTable, [14](#)
 - * **Mallows**
 - BestFit, [5](#)
 - RMallow-package, [2](#)
 - * **Mallow**
 - Lambda, [12](#)
 - Likelihood, [12](#)
 - Mallows, [13](#)
 - * **Sequences**
 - ConstructSeqs, [6](#)
 - * **bubblesort**
 - DistanceDistribution, [7](#)
 - NextTable, [14](#)
 - SeqDistribution, [16](#)
 - * **center**
 - UpdateR, [22](#)
 - * **clustering**
 - RMallow-package, [2](#)
 - * **cluster**
 - Mallows, [13](#)
 - Rgen, [15](#)
 - UpdateR, [22](#)
 - * **datasets**
 - datas, [7](#)
 - elect, [8](#)
 - three.mode, [18](#)
 - two.mode, [18](#)
 - two.seq, [19](#)
 - * **distance**
 - AllKendall, [3](#)
 - SeqDistribution, [16](#)
 - * **expectation**
 - EStep, [9](#)
 - * **initialization**
 - Rgen, [15](#)
 - * **lambda**
 - Lambda, [12](#)
 - UpdateLambda, [20](#)
 - * **likelihood**
 - Likelihood, [12](#)
 - * **maximization**
 - EStep, [9](#)
 - UpdateLambda, [20](#)
 - * **mixture**
 - BestFit, [5](#)
 - * **model**
 - BestFit, [5](#)
 - * **normalize**
 - C_lam, [5](#)
 - * **package**
 - RMallow-package, [2](#)
 - * **proportion**
 - UpdateP, [21](#)
 - * **ranking**
 - RMallow-package, [2](#)
 - * **sequence**
 - SimplifySequences, [17](#)
 - * **simplify**
 - SimplifySequences, [17](#)
- AllKendall, [2](#), [3](#)
AllSeqDists, [4](#)

BestFit, [2](#), [5](#)

C_lam, [5](#)
ConstructSeqs, [3](#), [6](#)

datas, [7](#), [18](#), [19](#)

DistanceDistribution, [3](#), [7](#), [16](#)

elect, [8](#), [19](#)

EStep, [9](#)

FormatOut, [10](#)

KendallInfo, [3](#), [11](#)

Lambda, [12](#)

Likelihood, [12](#)

Mallows, [2](#), [5](#), [13](#)

NextTable, [14](#)

Rgen, [15](#)

RMallow (RMallow-package), [2](#)

RMallow-package, [2](#)

SeqDistribution, [8](#), [16](#)

SimplifySequences, [3](#), [17](#)

three.mode, [18](#), [19](#)

two.mode, [18](#), [18](#)

two.seq, [19](#)

UpdateLambda, [12](#), [20](#)

UpdateP, [21](#)

UpdateR, [22](#)