Package 'MultFourier'

March 12, 2025

Type Package

Title Calculation of p-values for a Multinomial Distribution

Version 1.0

Author Charles Thraves, Pablo Rivas, Felipe Subiabre

Maintainer Pablo Rivas <pablo.rivas.g@ug.uchile.cl>

Description Computes both approximated and exact p-values for multinomial goodness-of-fit tests based on multiple test statistics, including the logP test, Pearson's chi-square, and the log-likelihood ratio statistic. The exact p-value is computed using an exhaustive algorithm recommended for small datasets, while an approximation is provided using a Fourier series algorithm for larger datasets.

License GPL (>= 2)

RoxygenNote 7.3.2

Encoding UTF-8

Contents

`MultFourier-package` *MultFourier: Efficient Computation of Multinomial Test p-values*

Description

Computes both approximated and exact p-values for multinomial goodness-of-fit tests based on multiple test statistics, including the logP test, Pearson's chi-square, and the log-likelihood ratio statistic. The exact p-value is computed using an exhaustive algorithm recommended for small datasets, while an approximation is provided using a Fourier series algorithm for larger datasets.

Details

The `MultFourier` package provides efficient methods for computing p-values in multinomial tests, including both exact and approximate approaches. The exact p-value is computed using an exhaustive algorithm, which is recommended for small datasets. For larger datasets, an approximation is provided using a Fourier series algorithm, which is computationally efficient and provides control over precision through parameters such as `max_terms`, `rel_eps`, and `undersampling`.

The package includes the following methods:

- `pval_exhaustive`: Computes the exact p-value by enumerating all possible outcomes. Suitable for small datasets.
- `pval_fourier`: Approximates the p-value using a Fourier series expansion. Efficient for large datasets.
- `pval_flexible`: Automatically selects between the exhaustive and Fourier series methods based on the dataset size.

Key features:

- Control over precision through parameters such as `max_terms`, `rel_eps`, and `undersampling`.
- Support for multiple test statistics, including logP, Pearson's chi-square, and log-likelihood ratio.
- Verbose output for debugging and analysis.

References

- Thraves, C. and Subiabre, F. et al. (Year): "Title of the Paper". Journal Name, Volume(Issue), Pages.

See Also

- Report bugs at https://github.com/yourusername/MultFourier/issues

---

`pval_exhaustive` *Compute p-value using the Exhaustive Method*

Description

This function calculates the p-value for a multinomial test using the exhaustive method, which evaluates all possible outcomes to compute the exact p-value. This method is computationally intensive but provides precise results for small datasets.

Usage

```
pval_exhaustive(
  x,
  p,
  stat = "prob",
  lambda = 0,
  max_time = 600,
  verbose = FALSE
)
```

Arguments

x	An Integer vector with realizations for each category.
p	A Numeric vector with the probabilities for each category. These should be non-negative and sum to one. It should be the same size as ‘x‘.
stat	String with the name of the statistic to compute. If ‘"prob"‘, the exact Multinomial p-value is computed. If ‘"pearson"‘, the Pearson's Chi-square p-value is computed. If ‘"llr"‘, the log-likelihood ratio p-value is computed. If ‘"power_div"‘, a Power Divergence p-value is computed, in which case a ‘lambda‘ parameter must be given. The default value is ‘"prob"‘.
lambda	A Numeric with the lambda value of the Power Divergence statistic. Only works if ‘stat = "power_div"‘, otherwise is ignored.
max_time	A Numeric with the maximum time limit in seconds. The default is 600.
verbose	Boolean. If ‘TRUE‘, it prints information on the run time. If ‘FALSE‘, it does not print. The default is ‘FALSE‘.

Value

Returns a ‘MultF‘ object with the following attributes:

- ‘x‘: The input vector of the observed realizations for each category.
- ‘p‘: The input vector of the probabilities for each category.
- ‘pval‘: The p-value computed.
- ‘time‘: The total execution time of the algorithm in seconds.
- ‘p0‘: Probability mass function in ‘x‘.
- ‘status‘: The final status ID of the algorithm upon completion:
  - ‘0‘: Successful computation.
  - ‘1‘: Maximum time reached.
- ‘message‘: The finishing status displayed as a message.
- ‘method‘: A String with value ‘"exhaustive"‘.

Examples

```
# Example 1: Compute p-value using the exhaustive method
probs <- c(0.00040161, 0.00080321, 0.00200803, 0.00401606, 0.00682731,
           0.01044177, 0.01485944, 0.02008032, 0.02610442, 0.03293173,
           0.04056225, 0.04899598, 0.05823293, 0.06827309, 0.07911647,
           0.09076305, 0.10321285, 0.11646586, 0.13052209, 0.14538153)

x0 <- rep(10, length(probs))

result <- pval_exhaustive(x0, probs, verbose = TRUE)
print(result)

# Example 2: Error case (mismatched lengths)
## Not run:
x0_invalid <- c(10, 10, 10)
probs_invalid <- c(0.2, 0.3)  # Different lengths, should raise an error
result_invalid <- pval_exhaustive(x0_invalid, probs_invalid)
print(result_invalid)
## End(Not run)
```

`pval_flexible` *Compute p-value for a Multinomial Test*

Description

This function calculates the p-value based on the multinomial distribution using a C function for efficient computation. The p-value is determined by comparing observed counts (‘x‘) against expected probabilities (‘p‘) for a multinomial test.

Usage

```
pval_flexible(
  x,
  p,
  stat = "prob",
  lambda = 0,
  max_time = 600,
  max_terms = 300,
  rel_eps = 0.001,
  undersampling = 1,
  verbose = FALSE
)
```

Arguments

`x`	An Integer vector with realizations for each category.
`p`	A Numeric vector with the probabilities for each category. These should be non-negative and sum to one. It should be the same size as ‘x‘.
`stat`	String with the name of the statistic to compute. If ‘"prob"‘, the exact Multinomial p-value is computed. If ‘"pearson"‘, the Pearson’s Chi-square p-value is computed. If ‘"llr"‘, the log-likelihood ratio p-value is computed. If ‘"power_div"‘, a Power Divergence p-value is computed, in which case a ‘lambda‘ parameter must be given. The default value is ‘"prob"‘.
`lambda`	A Numeric with the lambda value of the Power Divergence statistic. Only works if ‘stat = "power_div"‘, otherwise is ignored.
`max_time`	A Numeric with the maximum time limit in seconds. The default is 600.
`max_terms`	An Integer indicating the number of terms to add in the Fourier series. The default is 300.
`rel_eps`	A Numeric with the relative error tolerance. The default is 0.001.
`undersampling`	An Integer with the undersampling value to use. The default and recommended value is 1. Values greater than one will speed up calculations but will sacrifice precision.
`verbose`	Boolean. If ‘TRUE‘, it prints intermediate results every 10 terms. If ‘FALSE‘, it does not print intermediate computations. The default is ‘FALSE‘.

Value

Returns a ‘MultF‘ object with the following attributes:

- ‘x‘: The input vector of the observed realizations for each category.
- ‘p‘: The input vector of the probabilities for each category.
- ‘pval‘: The p-value computed.
- ‘gamma‘: The optimal gamma obtained in the first part of the method.
- ‘n_terms‘: The number of terms of the Fourier sum.
- ‘time‘: The total execution time of the algorithm in seconds.
- ‘p0‘: Probability mass function in ‘x‘.
- ‘status‘: The final status ID of the algorithm upon completion:
  - ‘0‘: Converged.
  - ‘1‘: Maximum time reached.
  - ‘2‘: Maximum number of terms reached.
  - ‘3‘: Could not solve the optimization of gamma.
- ‘message‘: The finishing status displayed as a message.
- ‘method‘: A String with value ‘"fourier"‘ or ‘"exhaustive"‘, depending on the method used.

Examples

```
# Example 1: Compute p-value using the exact multinomial statistic
probs <- c(0.00040161, 0.00080321, 0.00200803, 0.00401606, 0.00682731,
           0.01044177, 0.01485944, 0.02008032, 0.02610442, 0.03293173,
           0.04056225, 0.04899598, 0.05823293, 0.06827309, 0.07911647,
           0.09076305, 0.10321285, 0.11646586, 0.13052209, 0.14538153)

x0 <- rep(10, length(probs))

result <- pval_flexible(x0, probs, max_terms = 300, verbose = TRUE)
print(result)

# Example 2: Error case (mismatched lengths)
## Not run:
x0_invalid <- c(10, 10, 10)
probs_invalid <- c(0.2, 0.3)  # Different lengths, should raise an error
result_invalid <- pval_flexible(x0_invalid, probs_invalid)
print(result_invalid)
## End(Not run)
```

`pval_fourier` *Compute p-value using the Fourier Method*

Description

This function calculates the p-value for a multinomial test using the Fourier series method, which approximates the p-value using a Fourier series expansion. This method is more efficient for large datasets but may introduce some approximation error.

Usage

```
pval_fourier(
  x,
  p,
  stat = "prob",
  lambda = 0,
  max_time = 600,
  max_terms = 300,
  rel_eps = 0.001,
  undersampling = 1,
  verbose = FALSE
)
```

Arguments

x	An Integer vector with realizations for each category.
p	A Numeric vector with the probabilities for each category. These should be non-negative and sum to one. It should be the same size as ‘x‘.
stat	String with the name of the statistic to compute. If ‘"prob"‘, the exact Multinomial p-value is computed. If ‘"pearson"‘, the Pearson's Chi-square p-value is computed. If ‘"llr"‘, the log-likelihood ratio p-value is computed. If ‘"power_div"‘, a Power Divergence p-value is computed, in which case a ‘lambda‘ parameter must be given. The default value is ‘"prob"‘.
lambda	A Numeric with the lambda value of the Power Divergence statistic. Only works if ‘stat = "power_div"‘, otherwise is ignored.
max_time	A Numeric with the maximum time limit in seconds. The default is 600.
max_terms	An Integer indicating the number of terms to add in the Fourier series. The default is 300.
rel_eps	A Numeric with the relative error tolerance. The default is 0.001.
undersampling	An Integer with the undersampling value to use. The default and recommended value is 1. Values greater than one will speed up calculations but will sacrifice precision.
verbose	Boolean. If ‘TRUE‘, it prints intermediate results every 10 terms. If ‘FALSE‘, it does not print intermediate computations. The default is ‘FALSE‘.

Value

Returns a ‘MultF‘ object with the following attributes:

- ‘x‘: The input vector of the observed realizations for each category.
- ‘p‘: The input vector of the probabilities for each category.
- ‘pval‘: The p-value computed.
- ‘gamma‘: The optimal gamma obtained in the first part of the method.
- ‘n_terms‘: The number of terms of the Fourier sum.
- ‘time‘: The total execution time of the algorithm in seconds.
- ‘p0‘: Probability mass function in ‘x‘.
- ‘status‘: The final status ID of the algorithm upon completion:
  - ‘0‘: Converged.

  - '1': Maximum time reached.
  - '2': Maximum number of terms reached.
  - '3': Could not solve the optimization of gamma.
- 'message': The finishing status displayed as a message.
- 'method': A String with value '"fourier"'.

Examples

```
# Example 1: Compute p-value using the Fourier method
probs <- c(0.00040161, 0.00080321, 0.00200803, 0.00401606, 0.00682731,
           0.01044177, 0.01485944, 0.02008032, 0.02610442, 0.03293173,
           0.04056225, 0.04899598, 0.05823293, 0.06827309, 0.07911647,
           0.09076305, 0.10321285, 0.11646586, 0.13052209, 0.14538153)

x0 <- rep(10, length(probs))

result <- pval_fourier(x0, probs, max_terms = 300, verbose = TRUE)
print(result)

# Example 2: Error case (mismatched lengths)
## Not run:
x0_invalid <- c(10, 10, 10)
probs_invalid <- c(0.2, 0.3)  # Different lengths, should raise an error
result_invalid <- pval_fourier(x0_invalid, probs_invalid)
print(result_invalid)
## End(Not run)
```

Index