Package 'MultinomFourier'

November 12, 2026

Type Package

Title p-Values of Multinomial Goodness-of-Fit Tests by Fourier Series Inversion

Version 1.1.0

Author Charles Thraves [aut] (ORCID: <https://orcid.org/0000-0001-7491-7477>), Pablo Rivas [aut, cre], Felipe Subiabre [aut]

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

Description Computes p-values of multinomial goodness-of-fit tests based on an additive statistic: the log-likelihood ratio, Pearson's chi-squared, the Cressie-Read power divergence, and the exact multinomial test. The main function, pval_fourier(), writes the p-value as a Fourier series in the moment-generating function of the statistic and sums it until an adaptive stopping rule is met, which makes small p-values of large instances computationally feasible. The function pval_exact() computes the exact p-value by branch pruning with bounds from a dynamic program, and pval_flexible() chooses between the two according to the size of the instance.

License GPL (>= 2)

Encoding UTF-8

Imports parallel

Suggests knitr, rmarkdown, ggplot2, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.1.0

NeedsCompilation yes

Repository CRAN

Date/Publication 2026-11-12 20:50:02 UTC

Contents

MultinomFourier-package 2

print.multinomfourier 3
pval_exact 4
pval_flexible 6
pval_fourier 9

Index **14**

`MultinomFourier-package`
MultinomFourier: p-values of multinomial goodness-of-fit tests

---

Description

Computes p-values of multinomial goodness-of-fit tests based on an additive statistic: the log-likelihood ratio, Pearson's chi-squared, the Cressie-Read power divergence, and the exact multinomial test. The main function, pval_fourier(), writes the p-value as a Fourier series in the moment-generating function of the statistic and sums it until an adaptive stopping rule is met, which makes small p-values of large instances computationally feasible. The function pval_exact() computes the exact p-value by branch pruning with bounds from a dynamic program, and pval_flexible() chooses between the two according to the size of the instance.

Details

`MultinomFourier` computes the p-value $P\{T(Y) \geq T(x)\}$ of a multinomial goodness-of-fit test, where x is the observation (the vector of observed counts in the K categories, with $N = \sum_k x_k$ trials), $Y \sim \text{Multinomial}(N, p)$ under the null hypothesis with category probabilities p, and T is an additive statistic: the log-likelihood ratio G^2, Pearson's $\mathcal{X}^2$, the Cressie–Read power divergence, or the probability mass function of the multinomial (exact multinomial test). It is aimed at small p-values, far in the tail, and at instances where the support of the multinomial is too large to sum over.

The package provides three functions:

- `pval_fourier`: writes the p-value as a Fourier series whose terms are values of the moment-generating function of the statistic, and sums it until an adaptive stopping rule is met. Its cost grows slowly with N and K; its accuracy depends on the statistic (fast convergence for G^2 and the probability mass function, possibly none for $\mathcal{X}^2$).
- `pval_exact`: exact p-value by branch pruning with bounds from a dynamic program, without enumerating the support. Feasible for a moderate number of categories and trials.
- `pval_flexible`: chooses between the two according to the size of the support.

All three return an object of class `"multinomfourier"`; check its `status` component before using the p-value of `pval_fourier`.

Author(s)

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

Authors:

- Pablo Rivas <pablo.rivas.g@ug.uchile.cl>
- Charles Thraves (ORCID)
- Felipe Subiabre

References

Subiabre, F. and Thraves, C. Efficient computation of p-values for multinomial tests. Manuscript.

See Also

Report bugs at https://github.com/PabRvss/MultinomFourier/issues.

`print.multinomfourier` *Print method for multinomfourier objects*

Description

Prints the method, statistic, p-value, status and total runtime and, for the Fourier method, also the tilt `gamma`, the time spent choosing it, the period half-width `W` and the number of terms.

Usage

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

Arguments

`x`	Object of class `"multinomfourier"` returned by `pval_fourier`, `pval_exact` or `pval_flexible`.
`...`	Additional arguments (currently unused).

Value

Invisibly returns `x`.

Examples

```
# Note: Reported p-values in comments are illustrative and may vary slightly
# across versions or due to changes in numerical criteria.
p <- c(0.1, 0.2, 0.3, 0.4)
x <- c(35, 30, 60, 75)

# Fourier method: also shows gamma, W and the number of terms
fit_fourier <- pval_fourier(x, p, stat = "llr")
fit_fourier              # same as print(fit_fourier)

# Exact method
fit_exact <- pval_exact(x, p, stat = "llr")
fit_exact
```

`pval_exact` *Exact multinomial test p-value*

Description

Computes the exact p-value $P\{T(Y) \geq T(x)\}$, $Y \sim \text{Multinomial}(N, p)$, of a multinomial goodness-of-fit test, up to floating-point rounding. Instead of enumerating the support, it prunes whole groups of outcomes using bounds on the statistic (see Details). `pval_exhaustive()` is a deprecated alias.

Usage

```
pval_exact(
  x,
  p = rep(1/length(x), length(x)),
  stat = c("llr", "pd", "chi2", "pmf"),
  lambda = NULL,
  rescale_p = FALSE,
  n_threads = min(2L, parallel::detectCores()),
  max_time = NULL,
  verbose = FALSE,
  ...
)

pval_exhaustive(...)
```

Arguments

`x`	Vector of observed counts, one per category (non-negative integers, without missing values). Their sum is the number of trials N.
`p`	Vector of category probabilities under the null hypothesis, all strictly positive. They must sum to 1 (up to a tolerance of `sqrt(.Machine$double.eps)`), unless `rescale_p = TRUE`. Default: equiprobable categories.

`stat`	Test statistic: `"llr"` (log-likelihood ratio G^2, the default), `"chi2"` (Pearson's $\mathcal{X}^2$), `"pd"` (Cressie–Read power divergence with parameter `lambda`) or `"pmf"` (probability mass function of the multinomial, i.e. the exact multinomial test). See Details.
`lambda`	Parameter λ of the power-divergence statistic; used only when `stat = "pd"`, and then required (there is no default). Must be a single finite real number strictly greater than -1 ($\lambda > -1$); values $\lambda \leq -1$ produce an error. A warning is issued if it is given for another statistic.
`rescale_p`	Logical; if `TRUE`, `p` is rescaled to `p / sum(p)`, so it can be given as any vector of positive weights. If `FALSE` (default) and `p` does not sum to 1, an error is given. Same argument and default as `rescale.p` in `chisq.test`.
`n_threads`	Number of threads (positive integer, capped at the number of available cores). Default `min(2L, parallel::detectCores())`.
`max_time`	Optional time limit in seconds. `NULL` (default) means no limit. The limit is only checked between blocks of values of the first category, so a run can exceed it by a wide margin; if it is reached, `pval` is `NA` and `status` is 1.
`verbose`	Logical; if `TRUE`, prints progress information. Default `FALSE`.
`...`	Advanced options passed to the C engine. Unknown names produce a warning.

Details

The statistics are those of `pval_fourier`. Because the statistic is additive, a dynamic program over the $(N+1)K$ states (category, trials left) first computes the exact minimum and maximum that the remaining categories can add to it, at cost $O(KN^2)$. Counts are then assigned one category at a time, in decreasing order of probability; whenever the bounds show that every completion of a partial assignment is (or none is) at least as extreme as the observed data, its whole probability is added (or it is discarded) without visiting it. Only partial assignments near the boundary $T(y) = T(x)$ are expanded.

The cost still grows quickly with the number of categories K. As a rough guide, in our tests with the log-likelihood ratio on 8 threads, instances with $K \leq 6$ and $N \leq 1000$ took under five minutes, while $K \geq 8$ with $N \geq 500$ did not finish within an hour. For large instances use `pval_fourier`.

Value

An object of class `"multinomfourier"`: a list with components

x, p The input counts and null probabilities (`p` after rescaling, if `rescale_p = TRUE`).

stat_value The observed value of the test statistic $T(x)$.

pval The exact p-value, up to floating-point rounding; `NA` if `max_time` was reached or if the p-value is smaller than 5×10^{-324}.

log_pval Natural logarithm of the exact p-value. Preserves the exact magnitude when `pval` is smaller than 5×10^{-324} and reported as `NA`.

stat Name of the test statistic, e.g. `"Log-likelihood ratio"` or `"power-divergence (lambda = 0.6667)"`.

method `"exact"`.

status `0`: the computation finished; `1`: `max_time` was reached before it finished; `4`: the accumulated probability came out zero or not finite, so the p-value would be wrong; both `pval` and `log_pval` are then `NA`.

message Human-readable version of `status`. For `status = 0` it reads `"Converged"`, a label shared with `pval_fourier`.

time Total wall-clock time in seconds.

The components that describe the Fourier series are also present, so that results of both methods have the same structure, but are always `NA`: `Poissonized`, `gamma`, `time_gamma`, `W`, `n_terms`, `Q`, `D_Q`, `err_rel_ext`, `err_rel_est`, `n_eff` and `nu_max_ratio`.

See Also

`pval_fourier`, `pval_flexible`.

Examples

```
# Note: Reported p-values in comments are illustrative and may vary slightly
# across versions or due to changes in numerical criteria.
p <- c(0.1, 0.2, 0.3, 0.4)
x <- c(35, 30, 60, 75)   # N = 200

# Log-likelihood ratio
fit <- pval_exact(x, p, stat = "llr")
fit
fit$pval                 # 0.00684
fit$status               # 0: computation finished

# Probability mass function (exact multinomial test)
fit_pmf <- pval_exact(x, p, stat = "pmf")
fit_pmf$pval             # 0.00578

# Pearson's chi-square
fit_chi2 <- pval_exact(x, p, stat = "chi2")
fit_chi2$pval            # 0.00294
```

`pval_flexible` *Multinomial test p-value, choosing the method automatically*

Description

Computes the p-value with `pval_exact` when $\log_{10}$ of the support size is strictly less than `enum_cutoff`, and with `pval_fourier` when it is greater than or equal to `enum_cutoff`.

Usage

```
pval_flexible(
  x,
  p = rep(1/length(x), length(x)),
  stat = c("llr", "pd", "chi2", "pmf"),
  lambda = NULL,
  rescale_p = FALSE,
  enum_cutoff = 10,
  undersampling = 1,
  rel_eps = 0.001,
  B = 50,
  max_terms = 10000,
  max_time = NULL,
  exact_terms = FALSE,
  n_threads = min(2L, parallel::detectCores()),
  precompute = TRUE,
  verbose = FALSE,
  ...
)
```

Arguments

x	Vector of observed counts, one per category (non-negative integers, without missing values). Their sum is the number of trials N.
p	Vector of category probabilities under the null hypothesis, all strictly positive. They must sum to 1 (up to a tolerance of `sqrt(.Machine$double.eps)`), unless `rescale_p = TRUE`. Default: equiprobable categories.
stat	Test statistic: `"llr"` (log-likelihood ratio G^2, the default), `"chi2"` (Pearson's $\mathcal{X}^2$), `"pd"` (Cressie–Read power divergence with parameter `lambda`) or `"pmf"` (probability mass function of the multinomial, i.e. the exact multinomial test). See Details.
lambda	Parameter λ of the power-divergence statistic; used only when `stat = "pd"`, and then required (there is no default). Must be a single finite real number strictly greater than -1 ($\lambda > -1$); values $\lambda \leq -1$ produce an error. A warning is issued if it is given for another statistic.
rescale_p	Logical; if `TRUE`, `p` is rescaled to `p / sum(p)`, so it can be given as any vector of positive weights. If `FALSE` (default) and `p` does not sum to 1, an error is given. Same argument and default as `rescale.p` in `chisq.test`.
enum_cutoff	Threshold on $\log_{10}$ of the support size $\binom{N+K-1}{K-1}$ (the number of possible count vectors). Below it ($<$ enum_cutoff) the exact method is used; at or above it ($\geq$ enum_cutoff) the Fourier method is used. Default 10. See Details.
undersampling	Positive factor dividing the half-width W of the Fourier period. The default 1 uses the smallest W for which the series is exact. Values below 1 enlarge W (still exact, but more terms are needed); values above 1 shrink W below that minimum, so the series converges to a wrong value (aliasing). Intended for experimentation only.

`rel_eps` Relative tolerance of the stopping rule (≥ 0). Default 0.001. It controls how stable the partial sums must be before the series is truncated; it is *not* a bound on the error of the returned p-value.

`B` Non-negative integer used by the term-size stopping rule: the series stops after `B` consecutive terms each smaller, relative to the partial sum, than `rel_eps / B`. Default 50; `B = 0` uses an internal value.

`max_terms` Maximum number of series terms (positive integer). Default 10000. If it is reached before the stopping rule is met, `status` is 2.

`max_time` Optional time limit in seconds, passed to the chosen method.

`exact_terms` Logical; if `FALSE` (default), evaluates the Fourier terms using the fast Poissonized Cauchy-integral evaluator. If `TRUE`, evaluates each term exactly via polynomial convolution.

`n_threads` Number of threads, passed to whichever method is chosen. Default `min(2L, parallel::detectCores())`.

`precompute` Logical; if `TRUE` (default), precomputes the FFT plan when evaluating terms via polynomial convolution (`exact_terms = TRUE`).

`verbose` Logical; if `TRUE`, prints the partial sums and diagnostic information while running. Default `FALSE`.

`...` Advanced options passed to the C engine, for example `print_freq` (how often partial sums are printed when `verbose = TRUE`; default 100) or `extrapolate` (0: no extrapolation; 1: report the extrapolated value; 2: also stop on it; 3: report the median of the deep extrapolation table over the final window, the default). Unknown names produce a warning.

Details

The cost of `pval_exact` depends strongly on the observed counts, not only on the support size, and it is highest far in the tail, where small p-values are computed. With counts drawn from the null hypothesis it takes about one second on 2 threads at a support of 10^{12} to 10^{13} outcomes (log-likelihood ratio, $K = 5, 6, 7$); with counts far in the tail it can take tens of seconds at a support of 10^{11} (for example $K = 6$, $N = 400$, $p \approx 10^{-33}$). The default `enum_cutoff = 10` is a simple, conservative value that keeps the exact method for instances where it is fast in both situations.

When the Fourier method is chosen and does not meet its stopping rule (`status != 0`), the result is returned as is; there is no automatic fallback to the exact method.

Value

An object of class `"multinomfourier"`, as described in `pval_fourier`; its `method` component (`"exact"` or `"fourier"`) shows which method was used.

See Also

`pval_fourier`, `pval_exact`.

Examples

```
# Note: Reported p-values in comments are illustrative and may vary slightly
# across versions or due to changes in numerical criteria.
# Small instance (K = 4, N = 200, about 10^6 outcomes): exact method
p <- c(0.1, 0.2, 0.3, 0.4)
x <- c(35, 30, 60, 75)
fit <- pval_flexible(x, p, stat = "llr")
fit
fit$method              # "exact"
fit$pval                # 0.00684
fit$status              # 0: computation finished

# Large instance (K = 8, N = 2000, about 10^19 outcomes): Fourier method
x_large <- c(240, 260, 250, 270, 230, 250, 245, 255)
fit_large <- pval_flexible(x_large, stat = "llr")
fit_large$method        # "fourier"
fit_large$pval          # 0.761
fit_large$status        # 0: stopping rule met
```

`pval_fourier` *Multinomial test p-value by Fourier series inversion*

Description

Computes the p-value of a multinomial goodness-of-fit test,

$$p = P\{T(Y) \geq T(x)\}, \qquad Y \sim \text{Multinomial}(N, p), \quad N = \sum_k x_k,$$

for an additive test statistic T, by summing a truncated Fourier series whose terms are values of the moment-generating function of the statistic. The series is truncated by an adaptive stopping rule, so the result is an approximation whose accuracy depends on the statistic and the instance; always check the returned `status`.

Usage

```
pval_fourier(
  x,
  p = rep(1/length(x), length(x)),
  stat = c("llr", "pd", "chi2", "pmf"),
  lambda = NULL,
  rescale_p = FALSE,
  undersampling = 1,
  rel_eps = 0.001,
  B = 50,
  max_terms = 10000,
  max_time = NULL,
  n_threads = min(2L, parallel::detectCores()),
```

```
  exact_terms = FALSE,
  precompute = TRUE,
  verbose = FALSE,
  ...
)
```

Arguments

x	Vector of observed counts, one per category (non-negative integers, without missing values). Their sum is the number of trials N.
p	Vector of category probabilities under the null hypothesis, all strictly positive. They must sum to 1 (up to a tolerance of `sqrt(.Machine$double.eps)`), unless `rescale_p = TRUE`. Default: equiprobable categories.
stat	Test statistic: `"llr"` (log-likelihood ratio G^2, the default), `"chi2"` (Pearson's $\mathcal{X}^2$), `"pd"` (Cressie–Read power divergence with parameter `lambda`) or `"pmf"` (probability mass function of the multinomial, i.e. the exact multinomial test). See Details.
lambda	Parameter λ of the power-divergence statistic; used only when `stat = "pd"`, and then required (there is no default). Must be a single finite real number strictly greater than -1 ($\lambda > -1$); values $\lambda \leq -1$ produce an error. A warning is issued if it is given for another statistic.
rescale_p	Logical; if `TRUE`, `p` is rescaled to `p / sum(p)`, so it can be given as any vector of positive weights. If `FALSE` (default) and `p` does not sum to 1, an error is given. Same argument and default as `rescale.p` in `chisq.test`.
undersampling	Positive factor dividing the half-width W of the Fourier period. The default 1 uses the smallest W for which the series is exact. Values below 1 enlarge W (still exact, but more terms are needed); values above 1 shrink W below that minimum, so the series converges to a wrong value (aliasing). Intended for experimentation only.
rel_eps	Relative tolerance of the stopping rule (≥ 0). Default 0.001. It controls how stable the partial sums must be before the series is truncated; it is *not* a bound on the error of the returned p-value.
B	Non-negative integer used by the term-size stopping rule: the series stops after `B` consecutive terms each smaller, relative to the partial sum, than `rel_eps / B`. Default 50; `B = 0` uses an internal value.
max_terms	Maximum number of series terms (positive integer). Default 10000. If it is reached before the stopping rule is met, `status` is 2.
max_time	Optional time limit in seconds. `NULL` (default) means no limit. When it is reached, `status` is 1.
n_threads	Number of threads used to evaluate the series terms (positive integer, capped at the number of available cores). Default `min(2L, parallel::detectCores())`.
exact_terms	Logical; if `FALSE` (default), evaluates the Fourier terms using the fast Poissonized Cauchy-integral evaluator. If `TRUE`, evaluates each term exactly via polynomial convolution.
precompute	Logical; if `TRUE` (default), precomputes the FFT plan when evaluating terms via polynomial convolution (`exact_terms = TRUE`).

verbose	Logical; if TRUE, prints the partial sums and diagnostic information while running. Default FALSE.
...	Advanced options passed to the C engine, for example print_freq (how often partial sums are printed when verbose = TRUE; default 100) or extrapolate (0: no extrapolation; 1: report the extrapolated value; 2: also stop on it; 3: report the median of the deep extrapolation table over the final window, the default). Unknown names produce a warning.

Details

Statistics. All supported statistics are additive, $T(y) = \sum_k f_k(y_k)$, which is what the method requires. With $E_k = Np_k$:

- "llr": $G^2 = 2\sum_k y_k \log(y_k/E_k)$.
- "chi2": $\mathcal{X}^2 = \sum_k (y_k - E_k)^2/E_k$.
- "pd": $2I^\lambda = \frac{2}{\lambda(\lambda+1)} \sum_k y_k[(y_k/E_k)^\lambda - 1]$ (Cressie and Read, 1984); $\lambda \to 0$ gives G^2 and $\lambda = 1$ gives $\mathcal{X}^2$.
- "pmf": the probability mass function $P(y)$ of the multinomial under the null hypothesis (exact multinomial test). Small values are the extreme ones, so the p-value is $P\{P(Y) \le P(x)\}$, the total probability of the outcomes that are at most as likely as the observed one.

Which statistics converge. The p-value is written exactly as a Fourier series for every additive statistic, but the number of terms needed for a given accuracy depends on the statistic. For "llr" and "pmf" the series typically converges in a few hundred to a few thousand terms. For "chi2", and for "pd" as λ moves away from 0, the values of the statistic lie on or near a lattice and the terms decay much more slowly, so the series often needs many more terms and can fail to converge within max_terms (status = 2) or even return a negative value. In that case use pval_exact when it is feasible.

Stopping rule and reported value. The partial sums are accelerated with Wynn's epsilon algorithm (in the safeguarded form of QUADPACK's DQELG) and smoothed with a running median of five. The series stops as soon as either (i) B consecutive terms are each smaller than rel_eps / B relative to the partial sum, or (ii) the smoothed extrapolated values vary by at most a relative rel_eps over a trailing window covering the last half of the terms. Two epsilon tables are kept in parallel: a shallow one, which drives the stopping rule above, and a deep one. With extrapolate = 3 (the default) the reported p-value is the median of the deep table over the last quarter of the terms, which is markedly more accurate for lattice-like statistics such as "chi2"; with extrapolate = 2 it is the last smoothed value of the shallow table. The stopping point is the same either way.

Term evaluation. With exact_terms = FALSE (default), terms are evaluated via a Poissonized Cauchy-integral evaluator, which is significantly faster on large instances. With exact_terms = TRUE, terms are evaluated via exact polynomial convolution.

Underflow below double precision. In standard IEEE 754 64-bit floating-point arithmetic, positive numbers smaller than approximately 5×10^{-324} underflow to exact zero. Because multinomial p-values are mathematically strictly positive, whenever the true p-value falls below 5×10^{-324}, pval is returned as NA and an informative console note is issued. The exact magnitude is retained in log_pval, which can be exported or evaluated via arbitrary-precision software (e.g. exp(log_pval)). The series is summed relative to a common magnitude and the p-value is assembled in logarithms, so log_pval stays finite far below the range of the floating-point type, on every platform (it does not need __float128).

Value

An object of class "multinomfourier": a list with components

x, p The input counts and null probabilities (p after rescaling, if rescale_p = TRUE).

stat_value The observed value of the test statistic $T(x)$.

pval The computed p-value, or NA if underflow occurs (when the p-value is smaller than 5×10^{-324}, below the 64-bit double precision limit) or if the algorithm did not converge.

log_pval Natural logarithm of the p-value. When the p-value is smaller than 5×10^{-324}, pval is reported as NA to indicate that it cannot be stored in 64-bit floating-point format without collapsing to zero, but log_pval preserves the exact magnitude without underflow.

stat Name of the test statistic, e.g. "Log-likelihood ratio" or "power-divergence (lambda = 0.6667)".

method "fourier".

Poissonized Logical; TRUE if the fast Poissonized Cauchy-integral evaluator was used (exact_terms = FALSE, the default), or FALSE if exact polynomial convolution was used (exact_terms = TRUE).

time Total wall-clock time in seconds.

gamma The exponential tilt γ applied to the statistic so that the series terms stay within floating-point range (a saddlepoint choice).

time_gamma Time in seconds spent choosing gamma.

W Half-width of the Fourier period, $\max(s_{\max}, -s_{\min})$ over the range $[s_{\min}, s_{\max}]$ of the statistic minus its observed value, divided by undersampling.

n_terms Number of series terms summed.

status 0: the stopping rule was met (the message reads "Converged"); 1: max_time was reached; 2: max_terms was reached without meeting the stopping rule, so pval is unreliable; 3: the search for gamma failed; 4: numerical overflow in the evaluation of the terms; 5: the first term of the series underflowed, so no p-value is returned (n_terms is 0). With a status other than 0, pval is the last value reached or NA.

message Human-readable version of status.

err_rel_ext Relative spread of the smoothed extrapolated values over the final window, i.e. the quantity that rule (ii) compares with rel_eps. When the series stopped by rule (i) it can be larger than rel_eps.

err_rel_est Estimate of the relative truncation error of the raw partial sum (not of the extrapolated value that is reported), based on a random-phase argument; in our tests it tends to underestimate the error.

n_eff Effective number of support points implied by the size of the late terms; used by err_rel_est.

nu_max_ratio Largest size of the late transform values relative to the first one; values near 1 mean the terms are not decaying (a lattice-like statistic).

Q Number of equispaced points on the saddlepoint circle used to extract the multinomial coefficient in each term of the series (a Q-point trapezoid rule for Cauchy's integral). It is chosen automatically, by doubling from a power of two until the aliasing diagnostic D_Q falls below 10^{-10}. NA with exact_terms = TRUE.

D_Q Aliasing diagnostic $D(Q)$. $D(Q)$ is a computable lower bound on $\alpha(Q)$, which provides a uniform upper bound on the aliasing error of each Fourier-series term. For the probability-mass statistic (`"pmf"`), $D(Q)$ approximates $\alpha(Q)$ to first order. For the log-likelihood-ratio statistic (`"llr"`), $D(Q)$ remains a lower bound on $\alpha(Q)$, but no guarantee of closeness is available. Stopping the doubling of Q when $D(Q)$ falls below a threshold does not by itself guarantee that the aliasing error is strictly below that threshold. `D_Q` measures an error separate from the truncation of the series (reported by `err_rel_ext` and `err_rel_est`). It is `0` when `Q` is large enough for the Cauchy formula to be exact (no aliasing possible), and `NA` with `exact_terms = TRUE`.

The last six components are diagnostics of the series. The four about truncation (`err_rel_ext`, `err_rel_est`, `n_eff` and `nu_max_ratio`) are heuristics, not error bounds. The names `Gamma` and `Time_gamma` used by earlier versions still work, with a deprecation warning.

References

Subiabre, F. and Thraves, C. Efficient computation of p-values for multinomial tests. Manuscript.

Cressie, N. and Read, T. R. C. (1984). Multinomial goodness-of-fit tests. *Journal of the Royal Statistical Society, Series B*, 46(3), 440–464.

See Also

`pval_exact` for the exact p-value, `pval_flexible` to choose between the two automatically.

Examples

```
# Note: Reported p-values and term counts in comments are illustrative and
# may vary slightly across versions or due to changes in numerical criteria.
p <- c(0.1, 0.2, 0.3, 0.4)
x <- c(35, 30, 60, 75)   # N = 200

# Log-likelihood ratio
fit <- pval_fourier(x, p, stat = "llr")
fit
fit$pval                 # 0.00688
fit$status               # 0: stopping rule met
fit$n_terms              # 1208

# Probability mass function (exact multinomial test)
fit_pmf <- pval_fourier(x, p, stat = "pmf")
fit_pmf$pval             # 0.00577
fit_pmf$status           # 0: stopping rule met

# Power divergence: lambda is required
fit_pd <- pval_fourier(x, p, stat = "pd", lambda = 2/3)
fit_pd$pval              # 0.00384
fit_pd$status            # 0: stopping rule met
```

Index

∗ **package**
MultinomFourier-package, 2

chisq.test, *5*, *7*, *10*

MultinomFourier
(MultinomFourier-package), 2
MultinomFourier-package, 2

print.multinomfourier, 3
pval_exact, *2, 3*, 4, *6*, *8*, *11*, *13*
pval_exhaustive (pval_exact), 4
pval_flexible, *2, 3*, *6*, 6, *13*
pval_fourier, *2, 3*, *5, 6*, *8*, 9