Package ‘finity’

December 20, 2023

Type Package
Title Test for Finiteness of Moments in a Distribution
Version 0.1.5
Date 2023-12-20
Author Torsten Heinrich [aut, cre],
Julian Winkler [aut]
Maintainer Torsten Heinrich <torsten.heinrich@posteo.net>
Description The purpose of this package is to test whether a given
moment of the distribution of a given sample is finite or not. For
heavy-tailed distributions with tail exponent \( b \), only moments of
order smaller than \( b \) are finite. Tail exponent and heavy-
tailedness are notoriously difficult to ascertain. But the
finiteness of moments (including fractional moments) can be tested
directly. This package does that following the test suggested by
License GPL (>= 2)
Imports Rcpp (>= 1.0.3), stabledist (>= 0.7)
LinkingTo Rcpp, RcppArmadillo, BH
RoxygenNote 7.2.3
Encoding UTF-8
NeedsCompilation yes
Repository CRAN
Date/Publication 2023-12-20 19:20:12 UTC

\textbf{R topics documented:}

finity-package .................................................. 2
compute_absolute_moment ..................................... 3
finite_moment_test ............................................. 4
get_chisq1_percentile ......................................... 5

Index 6
Description

The purpose of this package is to test whether a given moment of the distribution of a given sample is finite or not. For heavy-tailed distributions with tail exponent \(b\), only moments of order smaller than \(b\) are finite. Tail exponent and heavy-tailedness are notoriously difficult to ascertain. But the finiteness of moments (including fractional moments) can be tested directly. This package does that following the test suggested by Trapani (2016) <doi:10.1016/j.jeconom.2015.08.006>.

Details

The main function of the package, which performs the test for the finiteness of moments, is `finite_moment_test()`.

The test assumes a sample drawn from an unknown distribution \(F\). Given this sample, the finiteness or infiniteness of any moment of order \(k\) of distribution \(F\) can be ascertained. For this, the test follows a randomised testing procedure with artificial randomness. The absolute sample moment \(\mu_k\) of the desired order \(k\) of the sample is transformed into a test statistic, which follows a \(\chi^2\) distribution with one degree of freedom exactly if the moment of the same order \(k\) of \(F\) is not finite. The null hypothesis in the test is that the moment is infinite; the alternative is that it is finite.

It should be noted that while the moment of order \(k\) of \(F\) may be infinite, the sample moment \(\mu_k\) is always finite because the sample is of finite size. The sample moment will, however, diverge with growing sample size if the moment of the same order \(k\) of the original distribution \(F\) is not finite.

The test works as follows: A standard normal distribution is rescaled with \(\sqrt{\exp(\mu_k)}\), yielding a normal distribution with mean 0 and either finite or infinite variance, depending on whether the hypothesis holds. For every observation of the resulting distribution, it is then tested, if the observation is located within an interval \([-u, u]\). The resulting binary quantity \(\zeta\) (0, or 1, true or false) follows a Bernoulli distribution with mean 1/2 exactly if the \(k\)th moment of \(F\) is infinite. Otherwise the mean is not 1/2. Sampling a number of different intervals characterized by different bounds \(u\) drawn from a distribution with finite support, the test aggregates over quantities \(\zeta\) such that the resulting test statistic follows a \(\chi^2\) exactly if \(E(\zeta) = 1/2\), i.e., if the \(k\)th moment of \(F\) is infinite.

Trapani (2016) offers some insights into the performance of the test and the impact its parameters have. These parameters are optional arguments of the testing function `finite_moment_test()`.

Author(s)

Torsten Heinrich [aut, cre], Julian Winkler [aut]

Maintainer: Torsten Heinrich <torsten.heinrich@posteo.net>

References

**compute_absolute_moment**

**See Also**

https://github.com/x0range/finity

**Examples**

```r
# Generate sample
rvs <- stabledist::rstable(100000, 1.9, 0.5, 1, 0, pm = 0)
# Perform test
result <- finite_moment_test(rvs, 2)
# Print results
message(paste("Test statistic:", result[1], "p-value:", result[2], "\n\n"))

# More examples are included in https://github.com/x0range/finity/examples
```

---

**compute_absolute_moment**

*Absolute Moment of Order k*

**Description**

Computes the absolute moment of order k of a sample of observations.

**Usage**

```r
compute_absolute_moment(obs, k)
```

**Arguments**

- `obs` : Observations (type: armadillo numeric vector).
- `k` : Moment order (type: double)

**Value**

Moment value (type: double)

**Examples**

```r
rvs <- stabledist::rstable(100000, 1.9, 0.5, 1, 0, pm = 0)
absolute_moment <- compute_absolute_moment(rvs, 2)
```
finite_moment_test  

Finite Moment Test

Description

Computes Trapani’s (2016) finite moment test for moment of order k of the distribution of a given the sample of observations obs. Knowledge of the identity of the distribution is not required. The null hypothesis is that the moment is infinite; the alternative is that it is finite. The function takes parameters of the test as optional arguments; some insights into the impact the choice of parameter values has are given in Trapani (2016).

Usage

finite_moment_test(
  obs,
  k,
  r = 0L,
  psi = 2,
  u = 1,
  force_random_variate_sample = 0L,
  ignore_errors = 0L,
  verbose = 0L,
  random_salting = 0L
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obs</td>
<td>Observations (type: armadillo numeric vector).</td>
</tr>
<tr>
<td>k</td>
<td>Moment order (type: double)</td>
</tr>
<tr>
<td>r</td>
<td>Artificial sample size (type: int). Default is $N^{0.8}$.</td>
</tr>
<tr>
<td>psi</td>
<td>Pescaling moment (type: double). Must be &lt;k. Default is 2.0.</td>
</tr>
<tr>
<td>u</td>
<td>Sampling range width for sampling range [-u, u] (type: double) Default is 1.0.</td>
</tr>
<tr>
<td>force_random_variate_sample</td>
<td>If True, draw random variates for xi and u_series. If False, use quantile function values from a regular percentile space grid. This represents the density function better. Default is False.</td>
</tr>
<tr>
<td>ignore_errors</td>
<td>Ignore errors caused by Inf and NaN results for too large absolute moments. If True, it will return test statistic=NA, pvalue=1. If False, it will stop with an error. Default is False. But normally this will indicate an infinite moment.</td>
</tr>
<tr>
<td>verbose</td>
<td>If True, print detailed output for debugging. Default is False.</td>
</tr>
<tr>
<td>random_salting</td>
<td>Salt number to be added to the random seed (type: int). This prevents identical random variate series if multiple instances are started and run in parallel. Default is 0.</td>
</tr>
</tbody>
</table>
get_chisq1_percentile

Value
Trapani’s Theta test statistic (type: double).
Corresponding p-value (Chi^2(1) percentile) (type: double).

Examples
rvs <- stabledist::rstable(100000L, 1.9, 0.5, 1, 0, pm = 0)
result <- finite_moment_test(rvs, 2)

generate 100,000 samples from a stable distribution with shape 1.9, skewness 0.5,
location 1, and scale 0.
Result after running the finite moment test with shape 2.

Description
Returns the Chi^2(1) percentile for the test statistic.

Usage
get_chisq1_percentile(value)

Arguments
value, Chi^2(1) value (type: double).

Value
Chi^2(1) percentile (type: double).

Examples
get_chisq1_percentile(20.0)
Index

* package
   finity-package, 2

compute_absolute_moment, 3

finite_moment_test, 4
finity(finity-package), 2
finity-package, 2

get_chisq1_percentile, 5