Package ‘OwenQ’

September 20, 2021

Type Package
Title Owen Q-Function
Version 1.0.4
Author Stéphane Laurent
Maintainer Stéphane Laurent <laurent_step@outlook.fr>
Description Evaluates the Owen Q-function for an integer value of the degrees of freedom, by applying Owen's algorithm (1965) <doi:10.1093/biomet/52.3-4.437>. It is useful for the calculation of the power of equivalence tests.
License GPL-3
Imports Rcpp (>= 0.12.10), stats
LinkingTo Rcpp, BH, RcppNumerical, RcppEigen
Suggests testthat, knitr, rmarkdown, mvtnorm
Encoding UTF-8
SystemRequirements C++11
RoxygenNote 7.0.2
VignetteBuilder knitr
URL https://github.com/stla/OwenQ
BugReports https://github.com/stla/OwenQ/issues
NeedsCompilation yes
Repository CRAN
Date/Publication 2021-09-20 06:50:09 UTC

R topics documented:

OwenQ1 ......................................................... 2
OwenQ2 ......................................................... 3
OwenT ......................................................... 4
powen ......................................................... 4
psbt ......................................................... 6
ptOwen ....................................................... 7
spowen2 ..................................................... 8
**Index**

OwenQ1  
*First Owen Q-function*

---

**Description**

Evaluates the first Owen Q-function (integral from 0 to \(R\)) for an integer value of the degrees of freedom.

**Usage**

\[
\text{OwenQ1}(\nu, t, \delta, R, \text{algo} = 2)
\]

**Arguments**

- \(\nu\): integer greater than 1, the number of degrees of freedom  
- \(t\): number, positive or negative, possibly infinite  
- \(\delta\): vector of finite numbers, with the same length as \(R\)  
- \(R\): (upper bound of the integral) vector of finite positive numbers, with the same length as \(\delta\)  
- \(\text{algo}\): the algorithm, 1 or 2

**Value**

A vector of numbers between 0 and 1, the values of the integral from 0 to \(R\).

**Note**

When the number of degrees of freedom is odd, the procedure resorts to the Owen T-function (OwenT).

**References**


**Examples**

# As R goes to Inf, OwenQ1(\(\nu\), \(t\), \(\delta\), \(R\)) goes to pt(\(t\), \(\nu\), \(\delta\)):
OwenQ1(\(\nu=5\), \(t=3\), \(\delta=2\), \(R=100\))
pt(q=3, df=5, ncp=2)
**OwenQ2**

---

**Second Owen Q-function**

---

**Description**

Evaluates the second Owen Q-function (integral from $R$ to $\infty$) for an integer value of the degrees of freedom.

**Usage**

```r
OwenQ2(nu, t, delta, R, algo = 2)
```

**Arguments**

- **nu**: integer greater than 1, the number of degrees of freedom
- **t**: number, positive or negative, possibly infinite
- **delta**: vector of finite numbers, with the same length as R
- **R**: (lower bound of the integral) vector of finite positive numbers, with the same length as delta
- **algo**: the algorithm used, 1 or 2

**Value**

A vector of numbers between 0 and 1, the values of the integral from $R$ to $\infty$.

**Note**

When the number of degrees of freedom is odd, the procedure resorts to the Owen T-function ($OwenT$).

**References**


**Examples**

```r
# OwenQ1(nu, t, delta, R) + OwenQ2(nu, t, delta, R) equals pt(t, nu, delta):
OwenQ1(nu=5, t=3, delta=2, R=1) + OwenQ2(nu=5, t=3, delta=2, R=1)
pt(q=3, df=5, ncp=2)
```
**OwenT**  

*Owen T-function*

---

**Description**

Evaluates the Owen T-function.

**Usage**

\[
\text{OwenT}(h, a)
\]

**Arguments**

- **h** numeric scalar
- **a** numeric scalar

**Details**

This is a port of the function `owens_t` of the `boost` collection of C++ libraries.

**Value**

A number between 0 and 0.25.

**References**


**Examples**

- \[
  \text{integrate(function(x) pnorm(1+2*x)^2*dnorm(x), lower=-Inf, upper=Inf)}
\]
- \[
  \text{pnorm(1/sqrt(5)) - 2*OwenT(1/sqrt(5), 1/3)}
\]

---

**powen**  

*Owen distribution functions when \(\delta_1 > \delta_2\)*

---

**Description**

Evaluates the Owen distribution functions when the noncentrality parameters satisfy \(\delta_1 > \delta_2\) and the number of degrees of freedom is integer.

- **powen1** evaluates \(P(T_1 \leq t_1, T_2 \leq t_2)\) (Owen’s equality 8)
- **powen2** evaluates \(P(T_1 \leq t_1, T_2 > t_2)\) (Owen’s equality 9)
- **powen3** evaluates \(P(T_1 > t_1, T_2 > t_2)\) (Owen’s equality 10)
- **powen4** evaluates \(P(T_1 > t_1, T_2 \leq t_2)\) (Owen’s equality 11)
Usage

powen1(nu, t1, t2, delta1, delta2, algo = 2)
powen2(nu, t1, t2, delta1, delta2, algo = 2)
powen3(nu, t1, t2, delta1, delta2, algo = 2)
powen4(nu, t1, t2, delta1, delta2, algo = 2)

Arguments

nu integer greater than 1, the number of degrees of freedom; infinite allowed
t1, t2 two numbers, positive or negative, possible infinite
delta1, delta2 two vectors of possibly infinite numbers with the same length, the noncentrality parameters; must satisfy delta1>delta2
algo the algorithm used, 1 or 2

Value

A vector of numbers between 0 and 1, possibly containing some NaN.

Note

When the number of degrees of freedom is odd, the procedure resorts to the Owen T-function (OwenT).

References


See Also

Use psbt for general values of delta1 and delta2.

Examples

nu=5; t1=2; t2=1; delta1=3; delta2=2
# Wolfram integration gives 0.1394458271284726
( p1 <- powen1(nu, t1, t2, delta1, delta2) )
# Wolfram integration gives 0.0353568969628651
( p2 <- powen2(nu, t1, t2, delta1, delta2) )
# Wolfram integration gives 0.806507459306199
( p3 <- powen3(nu, t1, t2, delta1, delta2) )
# Wolfram integration gives 0.018689824158
( p4 <- powen4(nu, t1, t2, delta1, delta2) )
# the sum should be 1
p1+p2+p3+p4
Owen distribution functions

Description

Evaluates the Owen cumulative distribution function for an integer number of degrees of freedom.

- `psbt1` evaluates \( P(T_1 \leq t_1, T_2 \leq t_2) \)
- `psbt2` evaluates \( P(T_1 \leq t_1, T_2 > t_2) \)
- `psbt3` evaluates \( P(T_1 > t_1, T_2 > t_2) \)
- `psbt4` evaluates \( P(T_1 > t_1, T_2 \leq t_2) \)

Usage

```r
psbt1(nu, t1, t2, delta1, delta2, algo = 2)
psbt2(nu, t1, t2, delta1, delta2, algo = 2)
psbt3(nu, t1, t2, delta1, delta2, algo = 2)
psbt4(nu, t1, t2, delta1, delta2, algo = 2)
```

Arguments

- `nu` integer greater than 1, the number of degrees of freedom; infinite allowed
- `t1, t2` two numbers, positive or negative, possibly infinite
- `delta1, delta2` two vectors of possibly infinite numbers with the same length, the noncentrality parameters
- `algo` the algorithm used, 1 or 2

Value

A vector of numbers between 0 and 1, possibly containing some NaN.

Note

When the number of degrees of freedom is odd, the procedure resorts to the Owen T-function (`OwenT`).

References


See Also

It is better to use `powen` if `delta1`>`delta2`. 
Examples

\[
\begin{align*}
\text{nu} & = 5; \ t1 = 1; \ t2 = 2; \ \text{delta1} = 2; \ \text{delta2} = 3 \\
(\ p1 \leftarrow \text{psbt1}(\text{nu}, \ t1, \ t2, \ \text{delta1}, \ \text{delta2}) \ ) \\
(\ p2 \leftarrow \text{psbt2}(\text{nu}, \ t1, \ t2, \ \text{delta1}, \ \text{delta2}) \ ) \\
(\ p3 \leftarrow \text{psbt3}(\text{nu}, \ t1, \ t2, \ \text{delta1}, \ \text{delta2}) \ ) \\
(\ p4 \leftarrow \text{psbt4}(\text{nu}, \ t1, \ t2, \ \text{delta1}, \ \text{delta2}) \ ) \\
\# \text{ the sum should be 1} \\
\ p1 + p2 + p3 + p4
\end{align*}
\]

ptOwen

**ptOwen**  
*Student CDF with integer number of degrees of freedom*

**Description**
Cumulative distribution function of the noncentral Student distribution with an integer number of degrees of freedom.

**Usage**

\[
\text{ptOwen}(q, \ \text{nu}, \ \text{delta} = 0)
\]

**Arguments**

- **q**: quantile, a finite number
- **nu**: integer greater than 1, the number of degrees of freedom; possibly infinite
- **delta**: numeric vector of noncentrality parameters; possibly infinite

**Value**
Numeric vector, the CDF evaluated at \( q \).

**Note**
The results are theoretically exact when the number of degrees of freedom is even. When odd, the procedure resorts to the Owen T-function.

**References**

**Examples**

\[
\begin{align*}
\text{ptOwen}(2, 3) & - \text{pt}(2, 3) \\
\text{ptOwen}(2, 3, \ \text{delta}=1) & - \text{pt}(2, 3, \ ncp=1)
\end{align*}
\]
spowen2

![Special case of second Owen distribution function](image)

**Description**

Evaluation of the second Owen distribution function in a special case (see details).

**Usage**

```r
spowen2(nu, t, delta, algo = 2)
```

**Arguments**

- `nu`: positive integer, possibly infinite
- `t`: positive number
- `delta`: vector of positive numbers
- `algo`: the algorithm used, 1 or 2

**Details**

The value of `spowen2(nu, t, delta)` is the same as the value of `powen2(nu, t, -t, delta, -delta)`, but it is evaluated more efficiently.

**Value**

A vector of numbers between 0 and 1.

**See Also**

`powen2`

**Examples**

```r
spowen2(4, 1, 2) == powen2(4, 1, -1, 2, -2)
```
Index

OwenQ1, 2
OwenQ2, 3
OwenT, 2, 3, 4, 5, 6

powen, 4, 6
powen1 (powen), 4
powen2, 8
powen2 (powen), 4
powen3 (powen), 4
powen4 (powen), 4
psbt, 5, 6
psbt1 (psbt), 6
psbt2 (psbt), 6
psbt3 (psbt), 6
psbt4 (psbt), 6
ptOwen, 7

spowen2, 8