Package 'visualize'

May 8, 2026

Type Package

Title Graph Probability Distributions with User Supplied Parameters and Statistics

Version 4.5.0

Depends R (>= 4.0.0)

Description Graphs the pdf or pmf and highlights what area or probability is present in user defined locations. Visualize is able to provide lower tail, bounded, upper tail, and two tail calculations. Supports strict and equal to inequalities. Also provided on the graph is the mean and variance of the distribution.

License MIT + file LICENSE

URL https://github.com/coatless-rpkg/visualize, https://thecoatlessprofessor.com/projects/visualize/, https://r-pkg.thecoatlessprofessor.com/visualize/

BugReports https://github.com/coatless-rpkg/visualize/issues

Encoding UTF-8

RoxygenNote 7.2.3

NeedsCompilation no

Author James Balamuta [aut, cph, cre] (ORCID: <https://orcid.org/0000-0003-2826-8458>)

Maintainer James Balamuta <james.balamuta@gmail.com>

Repository CRAN

Date/Publication 2023-11-13 09:40:02 UTC

Contents

`visualize.beta` *Visualize Beta Distribution*

Description

Generates a plot of the Beta distribution with user specified parameters.

Usage

```
visualize.beta(stat = 1, alpha = 3, beta = 2, section = "lower")
```

Arguments

`stat`	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as `stat = c(lower_bound, upper_bound)`. Otherwise, a simple `stat = desired_point` will suffice.
`alpha`	`alpha` is considered to be *shape1* by R's implementation of the beta distribution. `alpha` must be greater than 0.
`beta`	`beta` is considered to be *shape2* by R's implementation of the beta distribution. `beta` must be greater than 0.
`section`	Select how you want the statistic(s) evaluated via `section=` either `"lower"`,`"bounded"`, `"upper"`, or`"tails"`.

Value

Returns a plot of the distribution according to the conditions supplied.

Author(s)

James Balamuta

See Also

visualize.it(), dbeta().

Examples

```
# Evaluates lower tail.
visualize.beta(stat = 1, alpha = 2, beta = 3, section = "lower")

# Evaluates bounded region.
visualize.beta(stat = c(.5,1), alpha = 4, beta = 3, section = "bounded")

# Evaluates upper tail.
visualize.beta(stat = 1, alpha = 2, beta = 3, section = "upper")
```

---

visualize.binom *Visualize Binomial Distribution*

Description

Generates a plot of the Binomial distribution with user specified parameters.

Usage

```
visualize.binom(
  stat = 1,
  size = 3,
  prob = 0.5,
  section = "lower",
  strict = FALSE
)
```

Arguments

stat	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as stat = c(lower_bound, upper_bound). Otherwise, a simple stat = desired_point will suffice.
size	size of sample.
prob	probability of picking object.
section	Select how you want the statistic(s) evaluated via section= either "lower","bounded", "upper", or"tails".
strict	Determines whether the probability will be generated as a strict (<, >) or equal to (<=, >=) inequality. strict= requires either values = 0 or =FALSE for equal to OR values =1 or =TRUE for strict. For bounded condition use: strict=c(0,1) or strict=c(FALSE,TRUE).

Author(s)

James Balamuta

See Also

`visualize.it()` , `dbinom()`.

Examples

```
# Evaluates lower tail with equal to inequality.
visualize.binom(stat = 1, size = 3, prob = 0.5, section = "lower", strict = FALSE)

# Evaluates bounded region with lower bound equal to and upper bound strict inequality.
visualize.binom(stat = c(1,2), size = 5, prob = 0.35, section = "bounded", strict = c(0,1))

# Evaluates upper tail with strict inequality.
visualize.binom(stat = 1, size = 3, prob = 0.5, section = "upper", strict = TRUE)
```

`visualize.cauchy` *Visualize Cauchy Distribution*

Description

Generates a plot of the Cauchy distribution with user specified parameters.

Usage

```
visualize.cauchy(stat = 1, location = 2, scale = 1, section = "lower")
```

Arguments

`stat`	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as `stat = c(lower_bound, upper_bound)`. Otherwise, a simple `stat = desired_point` will suffice.
`location`	location parameter
`scale`	scale parameter
`section`	Select how you want the statistic(s) evaluated via `section=` either `"lower"`,`"bounded"`, `"upper"`, or`"tails"`.

Value

Returns a plot of the distribution according to the conditions supplied.

Author(s)

James Balamuta

See Also

visualize.it(), dcauchy().

Examples

```
# Evaluates lower tail.
visualize.cauchy(stat = 1, location = 4, scale = 2, section = "lower")

# Evaluates bounded region.
visualize.cauchy(stat = c(3,5), location = 5, scale = 3, section = "bounded")

# Evaluates upper tail.
visualize.cauchy(stat = 1, location = 4, scale = 2, section = "upper")
```

---

visualize.chisq *Visualize Chi-squared Distribution*

Description

Generates a plot of the Chi-squared distribution with user specified parameters.

Usage

```
visualize.chisq(stat = 1, df = 3, section = "lower")
```

Arguments

stat	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as stat = c(lower_bound, upper_bound). Otherwise, a simple stat = desired_point will suffice.
df	degrees of freedom of Chi-squared distribution.
section	Select how you want the statistic(s) evaluated via section= either "lower","bounded", "upper", or"tails".

Value

Returns a plot of the distribution according to the conditions supplied.

Author(s)

James Balamuta

See Also

visualize.it(), dchisq().

Examples

```
# Evaluates lower tail.
visualize.chisq(stat = 1, df = 3, section = "lower")
# Evaluates bounded region.
visualize.chisq(stat = c(1,2), df = 6, section = "bounded")
# Evaluates upper tail.
visualize.chisq(stat = 1, df = 3, section = "upper")
```

`visualize.continuous` *Graphing function for Continuous Distributions.*

Description

Handles how continuous distributions are graphed. Users should not use this function. Instead, users should use `visualize.it()`.

Usage

```
visualize.continuous(dist, stat = c(0, 1), params, section = "lower")
```

Arguments

`dist`	contains a supported continuos distribution shortname.
`stat`	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as `stat = c(lower_bound, upper_bound)`. Otherwise, a simple `stat = desired_point` will suffice.
`params`	A list that must contain the necessary parameters for each distribution. For example, `params = list(mu = 1, sd = 1)` would be for a normal distribution with mean 1 and standard deviation 1. If you are not aware of the parameters for the distribution, consider using the `visualize.`*dist_name* functions listed under the "See Also" section.
`section`	Select how you want the statistic(s) evaluated via `section=` either `"lower"`,`"bounded"`, `"upper"`, or`"tails"`

Author(s)

James Balamuta

See Also

`visualize.it()`, `visualize.beta()`, `visualize.chisq()`, `visualize.exp()`, `visualize.gamma()`, `visualize.norm()`, `visualize.unif()`, `visualize.cauchy()`, `visualize.f()`, `visualize.lnorm()`, `visualize.t()`, `visualize.wilcox()`, `visualize.logis()`.

Examples

```
# Function does not have dist look up, must go through visualize.it
visualize.it(dist='norm', stat = c(0,1), params = list(mu = 1, sd = 1), section = "bounded")
```

`visualize.discrete` *Graphing function for Discrete Distributions.*

Description

Handles how discrete distributions are graphed. Users should not use this function. Instead, users should use `link{visualize.it}`.

Usage

```
visualize.discrete(dist, stat = c(0, 1), params, section = "lower", strict)
```

Arguments

`dist`	contains the distribution from `link{visualize.distributions}`.
`stat`	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as `stat = c(lower_bound, upper_bound)`. Otherwise, a simple `stat = desired_point` will suffice.
`params`	A list that must contain the necessary parameters for each distribution. For example, `params = list(n = 5, prob = .25)` would be for a binomial distribution with size 5 and probability .75. If you are not aware of the parameters for the distribution, consider using the `visualize.`*dist_name* functions listed under the "See Also" section.
`section`	Select how you want the statistic(s) evaluated via `section=` either `"lower"`,`"bounded"`, `"upper"`, or`"tails"`.
`strict`	Determines whether the probability will be generated as a strict (<, >) or equal to (<=, >=) inequality. `strict=` requires either values = 0 or =FALSE for equal to OR values =1 or =TRUE for strict. For bounded condition use: `strict=c(0,1)` or `strict=c(FALSE,TRUE)`.

Author(s)

James Balamuta

See Also

`visualize.it()`, `visualize.binom()`, `visualize.geom()`, `visualize.hyper()`, `visualize.nbinom()`, `visualize.pois()`.

Examples

```
# Function does not have dist look up, must go through visualize.it
visualize.it(dist='geom', stat = c(2,4), params = list(prob = .75), section = "bounded",
          strict = c(0,1))
```

`visualize.exp` *Visualize Exponential Distribution*

Description

Generates a plot of the Exponential distribution with user specified parameters.

Usage

```
visualize.exp(stat = 1, theta = 1, section = "lower")
```

Arguments

`stat`	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as `stat = c(lower_bound, upper_bound)`. Otherwise, a simple `stat = desired_point` will suffice.
`theta`	vector of rates
`section`	Select how you want the statistic(s) evaluated via `section=` either `"lower"`,`"bounded"`, `"upper"`, or`"tails"`.

Value

Returns a plot of the distribution according to the conditions supplied.

Author(s)

James Balamuta

See Also

`visualize.it()`, `dexp()`.

Examples

```
# Evaluates lower tail.
visualize.exp(stat = .5, theta = 3, section = "lower")

# Evaluates bounded region.
visualize.exp(stat = c(1,2), theta = 3, section = "bounded")

# Evaluates upper tail.
visualize.exp(stat = .5, theta = 3, section = "upper")
```

`visualize.f` *Visualize F distribution*

Description

Generates a plot of the F distribution with user specified parameters.

Usage

```
visualize.f(stat = 1, df1 = 5, df2 = 4, section = "lower")
```

Arguments

`stat`	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as `stat = c(lower_bound, upper_bound)`. Otherwise, a simple `stat = desired_point` will suffice.
`df1`	First Degrees of Freedom
`df2`	Second Degrees of Freedom
`section`	Select how you want the statistic(s) evaluated via `section=` either `"lower"`,`"bounded"`, `"upper"`, or`"tails"`.

Value

Returns a plot of the distribution according to the conditions supplied.

Author(s)

James Balamuta

See Also

`visualize.it()`, `df()`.

Examples

```
# Evaluates lower tail.
visualize.f(stat = 1, df1 = 5, df2 = 4, section = "lower")

# Evaluates bounded region.
visualize.f(stat = c(3,5), df1 = 6, df2 = 3, section = "bounded")

# Evaluates upper tail.
visualize.f(stat = 1, df1 = 5, df2 = 4, section = "upper")
```

visualize.gamma *Visualize Gamma Distribution*

Description

Generates a plot of the Gamma distribution with user specified parameters.

Usage

```
visualize.gamma(stat = 1, alpha = 1, theta = 1, section = "lower")
```

Arguments

stat	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as stat = c(lower_bound, upper_bound). Otherwise, a simple stat = desired_point will suffice.
alpha	alpha is considered to be *shape* by R's implementation of the gamma distribution. alpha must be greater than 0.
theta	theta is considered to be *rate* by R's implementation of the gamma distribution. theta must be greater than 0.
section	Select how you want the statistic(s) evaluated via section= either "lower","bounded", "upper", or"tails".

Author(s)

James Balamuta

See Also

visualize.it(), dgamma().

Examples

```
# Evaluate lower tail.
visualize.gamma(stat = 1, alpha = 3, theta = 1, section = "lower")

# Evaluate bounded section.
visualize.gamma(stat = c(0.75,1), alpha = 3, theta = 1, section = "bounded")

# Evaluate upper tail.
visualize.gamma(stat = 1, alpha = 3, theta = 1, section = "upper")
```

visualize.geom *Visualize Geometric Distribution*

Description

Generates a plot of the Geometric distribution with user specified parameters.

Usage

```
visualize.geom(stat = 1, prob = 0.3, section = "lower", strict = FALSE)
```

Arguments

stat	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as stat = c(lower_bound, upper_bound). Otherwise, a simple stat = desired_point will suffice.
prob	probability of picking object.
section	Select how you want the statistic(s) evaluated via section= either "lower","bounded", "upper", or"tails".
strict	Determines whether the probability will be generated as a strict (<, >) or equal to (<=, >=) inequality. strict= requires either values = 0 or =FALSE for equal to OR values =1 or =TRUE for strict. For bounded condition use: strict=c(0,1) or strict=c(FALSE,TRUE).

Author(s)

James Balamuta

See Also

visualize.it() , dgeom().

Examples

```
# Evaluates lower tail.
visualize.geom(stat = 1, prob = 0.5, section = "lower", strict = FALSE)

# Evaluates bounded region.
visualize.geom(stat = c(1,3), prob = 0.35, section = "bounded", strict = c(0,1))

# Evaluates upper tail.
visualize.geom(stat = 1, prob = 0.5, section = "upper", strict = 1)
```

`visualize.hyper` *Visualize Hypergeometric Distribution*

Description

Generates a plot of the Hypergeometric distribution with user specified parameters.

Usage

```
visualize.hyper(
  stat = 1,
  m = 5,
  n = 4,
  k = 2,
  section = "lower",
  strict = FALSE
)
```

Arguments

`stat`	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as `stat = c(lower_bound, upper_bound)`. Otherwise, a simple `stat = desired_point` will suffice.
`m`	`m` white balls. `m` must be greater than 0.
`n`	`n` black balls. `n` must be greater than 0.
`k`	draw `k` balls without replacement.
`section`	Select how you want the statistic(s) evaluated via `section=` either `"lower"`,`"bounded"`, `"upper"`, or`"tails"`.
`strict`	Determines whether the probability will be generated as a strict (<, >) or equal to (<=, >=) inequality. `strict=` requires either values = 0 or =FALSE for equal to OR values =1 or =TRUE for strict. For bounded condition use: `strict=c(0,1)` or `strict=c(FALSE,TRUE)`.

Author(s)

James Balamuta

See Also

`visualize.it()` , `dhyper()`.

Examples

```
# Evaluates lower tail.
visualize.hyper(stat = 1, m=4, n=5, k=3, section = "lower", strict = 0)

# Evaluates bounded region.
visualize.hyper(stat = c(2,4), m=14, n=5, k=2, section = "bounded", strict = c(0,1))

# Evaluates upper tail.
visualize.hyper(stat = 1, m=4, n=5, k=3, section = "upper", strict = 1)
```

`visualize.it` *Visualize's Processing Function*

Description

Acts as a director of traffic and first line of error handling regarding submitted visualization requests. This function should only be used by advanced users.

Usage

```
visualize.it(
  dist = "norm",
  stat = c(0, 1),
  params = list(mu = 0, sd = 1),
  section = "lower",
  strict = c(0, 1)
)
```

Arguments

`dist`	a string that should be contain a supported probability distributions name in R. Supported continuous distributions: `"beta"`, `"chisq"`, `"exp"`, `"gamma"`, `"norm"`, and `"unif"`. Supported discrete distributions: `"binom"`, `"geom"`, `"hyper"`, `"nbinom"`, and `"pois"`.
`stat`	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as `stat = c(lower_bound, upper_bound)`. Otherwise, a simple `stat = desired_point` will suffice.
`params`	A list that must contain the necessary parameters for each distribution. For example, `params = list(mu = 1, sd = 1)` would be for a normal distribution with mean 1 and standard deviation 1. If you are not aware of the parameters for the distribution, consider using the `visualize.dist` functions listed under the "See Also" section.
`section`	Select how you want the statistic(s) evaluated via `section=` either `"lower"`,`"bounded"`, `"upper"`, or`"tails"`.

`strict`	Determines whether the probability will be generated as a strict (<, >) or equal to (<=, >=) inequality. `strict=` requires either values = 0 or =FALSE for strict OR values =1 or =TRUE for equal to. For bounded condition use: `strict=c(0,1)` or `strict=c(FALSE,TRUE)`.

Value

Returns a plot of the distribution according to the conditions supplied.

Author(s)

James Balamuta

References

http://cran.r-project.org/web/views/Distributions.html

See Also

`visualize.beta()`, `visualize.chisq()`, `visualize.exp()`, `visualize.gamma()`, `visualize.norm()`, `visualize.unif()`, `visualize.binom()`, `visualize.geom()`, `visualize.hyper()`, `visualize.nbinom()`, `visualize.pois()`.

Examples

```
# Defaults to lower tail evaluation
visualize.it(dist = 'norm', stat = 1, list(mu = 3 , sd = 2), section = "lower")

# Set to evaluate the upper tail.
visualize.it(dist = 'norm', stat = 1, list(mu=3,sd=2),section="upper")

# Set to shade inbetween a bounded region.
visualize.it(dist = 'norm', stat = c(-1,1), list(mu=0,sd=1), section="bounded")

# Gamma distribution evaluated at upper tail.
visualize.it(dist = 'gamma', stat = 2, params = list(alpha=2,beta=1),section="upper")

# Binomial distribution evaluated at lower tail.
visualize.it('binom', stat = 2, params = list(n=4,p=.5))
```

`visualize.lnorm` *Visualize Log Normal Distribution*

Description

Generates a plot of the Log Normal distribution with user specified parameters.

Usage

```
visualize.lnorm(stat = 1, meanlog = 3, sdlog = 1, section = "lower")
```

Arguments

stat	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as stat = c(lower_bound, upper_bound). Otherwise, a simple stat = desired_point will suffice.
meanlog	Mean of the distribution
sdlog	Standard deviation of the distribution
section	Select how you want the statistic(s) evaluated via section= either "lower","bounded", "upper", or"tails".

Value

Returns a plot of the distribution according to the conditions supplied.

Author(s)

James Balamuta

See Also

visualize.it(), dlnorm().

Examples

```
# Evaluates lower tail.
visualize.lnorm(stat = 1, meanlog = 3, sdlog = 1, section = "lower")

# Evaluates bounded region.
visualize.lnorm(stat = c(3,5), meanlog = 3, sdlog = 3, section = "bounded")

# Evaluates upper tail.
visualize.lnorm(stat = 1, meanlog = 3, sdlog = 1, section = "upper")
```

---

visualize.logis *Visualize Logistic distribution*

Description

Generates a plot of the Logistic distribution with user specified parameters.

Usage

```
visualize.logis(stat = 1, location = 3, scale = 1, section = "lower")
```

Arguments

stat	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as stat = c(lower_bound, upper_bound). Otherwise, a simple stat = desired_point will suffice.
location	Location of the distribution.
scale	Scale of the distribution.
section	Select how you want the statistic(s) evaluated via section= either "lower","bounded", "upper", or"tails".

Value

Returns a plot of the distribution according to the conditions supplied.

Author(s)

James Balamuta

See Also

visualize.it(), dlogis().

Examples

```
# Evaluates lower tail.
visualize.logis(stat = 1, location = 4, scale = 2, section = "lower")

# Evaluates bounded region.
visualize.logis(stat = c(3,5), location = 4, scale = 2, section = "bounded")

# Evaluates upper tail.
visualize.logis(stat = 1, location = 4, scale = 2, section = "upper")
```

---

visualize.nbinom *Visualize Negative Binomial Distribution*

Description

Generates a plot of the Negative Binomial distribution with user specified parameters.

Usage

```
visualize.nbinom(
  stat = 1,
  size = 6,
  prob = 0.5,
  section = "lower",
  strict = FALSE
)
```

Arguments

stat	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as stat = c(lower_bound, upper_bound). Otherwise, a simple stat = desired_point will suffice.
size	number of objects.
prob	probability of picking object.
section	Select how you want the statistic(s) evaluated via section= either "lower","bounded", "upper", or"tails".
strict	Determines whether the probability will be generated as a strict (<, >) or equal to (<=, >=) inequality. strict= requires either values = 0 or = FALSE for equal to OR values =1 or =TRUE for strict. For bounded condition use: strict=c(0,1) or strict=c(FALSE,TRUE).

Author(s)

James Balamuta

See Also

visualize.it() , dnbinom().

Examples

```
# Evaluates lower tail.
visualize.nbinom(stat = 1, size = 5, prob = 0.5, section = "lower", strict = 0)

# Evaluates bounded region.
visualize.nbinom(stat = c(1,3), size = 10, prob = 0.35, section = "bounded",
                    strict = c(TRUE, FALSE))

# Evaluates upper tail.
visualize.nbinom(stat = 1, size = 5, prob = 0.5, section = "upper", strict = 1)
```

---

visualize.norm *Visualize Normal Distribution*

Description

Generates a plot of the Normal distribution with user specified parameters.

Usage

```
visualize.norm(stat = 1, mu = 0, sd = 1, section = "lower")
```

Arguments

stat	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as stat = c(lower_bound, upper_bound). Otherwise, a simple stat = desired_point will suffice.
mu	mean of the Normal Distribution.
sd	standard deviation of the Normal Distribution.
section	Select how you want the statistic(s) evaluated via section= either "lower","bounded", "upper", or"tails".

See Also

visualize.it() , dnorm().

Examples

```
# Evaluates lower tail.
visualize.norm(stat = 1, mu = 4, sd = 5, section = "lower")

# Evaluates bounded region.
visualize.norm(stat = c(3,6), mu = 5, sd = 3, section = "bounded")

# Evaluates upper tail.
visualize.norm(stat = 1, mu = 3, sd = 2, section = "upper")
```

---

visualize.pois *Visualize Poisson Distribution*

Description

Generates a plot of the Poisson distribution with user specified parameters.

Usage

```
visualize.pois(stat = 1, lambda = 3.5, section = "lower", strict = FALSE)
```

Arguments

stat	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as stat = c(lower_bound, upper_bound). Otherwise, a simple stat = desired_point will suffice.
lambda	lambda value of the Poisson Distribution.
section	Select how you want the statistic(s) evaluated via section= either "lower","bounded", "upper", or"tails".
strict	Determines whether the probability will be generated as a strict (<, >) or equal to (<=, >=) inequality. strict= requires either values = 0 or =FALSE for equal to OR values =1 or =TRUE for strict. For bounded condition use: strict=c(0,1) or strict=c(FALSE,TRUE).

Author(s)

James Balamuta

See Also

visualize.it() , dpois().

Examples

```
# Evaluates lower tail.
visualize.pois(stat = 1, lambda = 2, section = "lower", strict = FALSE)

# Evaluates bounded region.
visualize.pois(stat = c(1,3), lambda = 3, section = "bounded", strict = c(0,1))

# Evaluates upper tail.
visualize.pois(stat = 1, lambda = 2, section = "upper", strict = 1)
```

---

visualize.t *Visualize Student's t distribution*

Description

Generates a plot of the Student's t distribution with user specified parameters.

Usage

```
visualize.t(stat = 1, df = 3, section = "lower")
```

Arguments

stat	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as stat = c(lower_bound, upper_bound). Otherwise, a simple stat = desired_point will suffice.
df	Degrees of freedom
section	Select how you want the statistic(s) evaluated via section= either "lower","bounded", "upper", or"tails".

Value

Returns a plot of the distribution according to the conditions supplied.

Author(s)

James Balamuta

See Also

visualize.it(), dt().

Examples

```
# Evaluates lower tail.
visualize.t(stat = 1, df = 4, section = "lower")

# Evaluates bounded region.
visualize.t(stat = c(3,5), df = 6, section = "bounded")

# Evaluates upper tail.
visualize.t(stat = 1, df = 4, section = "upper")
```

---

visualize.unif *Visualize Uniform Distribution*

Description

Generates a plot of the Uniform distribution with user specified parameters.

Usage

```
visualize.unif(stat = 1, a = 0, b = 1, section = "lower")
```

Arguments

stat	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as stat = c(lower_bound, upper_bound). Otherwise, a simple stat = desired_point will suffice.
a	starting point. Note: a<b
b	end point. Note: b > a
section	Select how you want the statistic(s) evaluated via section= either "lower","bounded", "upper", or"tails".

Author(s)

James Balamuta

See Also

visualize.it() , dunif().

Examples

```
# Evaluates lower tail.
visualize.unif(stat = 8.75, a = 7, b = 10, section = "lower")

# Evaluates bounded region.
visualize.unif(stat = c(3,6), a = 1, b = 7, section = "bounded")

# Evaluates upper tail.
visualize.unif(stat = 2, a = 1, b = 5, section = "upper")
```

---

visualize.wilcox *Visualize Cauchy Distribution*

Description

Generates a plot of the Wilcoxon Rank Sum distribution with user specified parameters.

Usage

```
visualize.wilcox(stat = 1, m = 7, n = 3, section = "lower")
```

Arguments

`stat`	a statistic to obtain the probability from. When using the "bounded" condition, you must supply the parameter as `stat = c(lower_bound, upper_bound)`. Otherwise, a simple `stat = desired_point` will suffice.
`m`	Sample size from group 1.
`n`	Sample size from group 2.
`section`	Select how you want the statistic(s) evaluated via `section=` either `"lower"`,`"bounded"`, `"upper"`, or`"tails"`.

Value

Returns a plot of the distribution according to the conditions supplied.

Author(s)

James Balamuta

See Also

`visualize.it()`, `dwilcox()`.

Examples

```
# Evaluates lower tail.
visualize.wilcox(stat = 1, m = 7, n = 3, section = "lower")

# Evaluates bounded region.
visualize.wilcox(stat = c(2,3), m = 5, n = 4, section = "bounded")

# Evaluates upper tail.
visualize.wilcox(stat = 1, m = 7, n = 3, section = "upper")
```

Index