Package 'fastqq'

October 25, 2022

Type Package

Title Faster Generation of Quantile Quantile Plots with Large Samples

Version 0.1.3

Maintainer Gudmundur Einarsson <gudmundur.einarsson.phd@gmail.com>

Description New and faster implementations for quantile quantile plots.
The package also includes a function to prune data for quantile quantile
plots. This can drastically reduce the running time for large samples,
for 100 million samples, you can expect a factor 80X speedup.

URL https://github.com/gumeo/fastqq

BugReports https://github.com/gumeo/fastqq/issues

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.2.1

Imports Rcpp

LinkingTo Rcpp

NeedsCompilation yes

Author Gudmundur Einarsson [aut, cre],
Hafsteinn Einarsson [aut]

Repository CRAN

Date/Publication 2022-10-25 06:05:10 UTC

R topics documented:

| drop_dense | *Internal function to prune quantiles of non-important values for visu-alization.* |

Description

This function is not exposed, since we want to hard-code the parameters for simplicity of usage.

Usage

```
drop_dense(x, y, N_hard = 10000)
```

Arguments

x	A numeric vector of sample/theoretical points.
y	A numeric vector of theoretical/sample points.
N_hard	Desired upper bound on the number of points to plot.

Value

data.frame with o and e pruned as columns.

| qq | *Creates a Q-Q plot* |

Description

Creates a quantile-quantile plot from p-values from an association study, e.g. a genome wide association study (GWAS). We compare the data quantile with a theoretical quantile from a uniform distribution. This code is mostly adapted from the qqman package, but improved for speed. A graph with a hundred million points should only take a few seconds to generate.

Usage

```
qq(pvector, ...)
```

Arguments

pvector	A numeric vector of p-values.
...	Other arguments passed to `plot()`

Value

No return value, called for plotting side effects.

Examples

```
qq(stats::runif(1e6))
```

| qqnorm | *Creates a Q-Q plot for comparing with normal quantiles* |

Description

Faster alternative to `stats::qqnorm()`. For more than 1e5 points we remove excess points, that would not be visible in the plot, since the points are so close. Otherwise this should work exactly the same, and the code is mostly adapted from `stats::qqnorm()`. This code produces more lightweight plots for excessive amounts of data.

Usage

```
qqnorm(
  y,
  ylim,
  main = "Normal Q-Q Plot",
  xlab = "Theoretical Quantiles",
  ylab = "Sample Quantiles",
  plot.it = TRUE,
  datax = FALSE,
  ...
)
```

Arguments

y	sample, to compare to normal quantiles.
ylim	graphical limits.
main	Plot title.
xlab	X label.
ylab	Y label.
plot.it	Should the plot be created.
datax	logical. Should data values be on x-axis?
...	Other arguments passed to `plot()`

Value

`data.frame` with sorted sample and normal quantiles, NA values are excluded.

Examples

```
qqnorm(stats::rnorm(1e6))
```

---

qqplot	*Creates a Q-Q plot*

Description

Faster alternative to `stats::qqplot()`. For more than 1e5 points we remove excess points, that would not be visible in the plot, since the points are so close.

Usage

```
qqplot(
  x,
  y,
  plot.it = TRUE,
  xlab = deparse1(substitute(x)),
  ylab = deparse1(substitute(y)),
  ...
)
```

Arguments

x	First sample for `qqplot`.
y	Second sample for `qqplot`.
plot.it	Should the plot be created.
xlab	x label for plot.
ylab	y label for plot.
...	Other arguments passed to `plot()`

Value

list with sorted samples, interpolated to be same size.

Examples

```
qqplot(stats::runif(1e6),stats::runif(1e6))
```

Index