Package 'qqplotr'

September 5, 2025

Type Package

Version 0.0.7

Title Quantile-Quantile Plot Extensions for 'ggplot2'

Description Extensions of 'ggplot2' Q-Q plot functionalities.

URL https://github.com/aloy/qqplotr

BugReports https://github.com/aloy/qqplotr/issues

License GPL-3 | file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Collate 'data.R' 'geom_qq_band.R' 'qqplotr.R' 'runShinyExample.R'
'stat_pp_band.R' 'stat_pp_line.R' 'stat_pp_point.R'
'stat_qq_line.R' 'stat_qq_band.R' 'stat_qq_point.R'

VignetteBuilder knitr

Depends R (>= 3.1), ggplot2 (>= 2.2)

Imports dplyr, robustbase, MASS, opdisDownsampling, qqconf (>= 1.3.1)

Suggests shiny, devtools, lattice, shinyBS, knitr, rmarkdown

NeedsCompilation no

Author Alexandre Almeida [aut],
Adam Loy [aut, cre],
Heike Hofmann [aut]

Maintainer Adam Loy <loyad01@gmail.com>

Repository CRAN

Date/Publication 2025-09-05 17:00:02 UTC

Contents

`geom_qq_band` *Quantile-quantile confidence bands*

Description

Draws quantile-quantile confidence bands, with an additional detrend option.

Usage

```
geom_qq_band(
  mapping = NULL,
  data = NULL,
  stat = "qq_band",
  position = "identity",
  na.rm = TRUE,
  show.legend = NA,
  inherit.aes = TRUE,
  distribution = "norm",
  dparams = list(),
  detrend = FALSE,
  identity = FALSE,
  qtype = 7,
  qprobs = c(0.25, 0.75),
  bandType = "pointwise",
  B = 1000,
  conf = 0.95,
  mu = NULL,
  sigma = NULL,
  ...
)

stat_qq_band(
  mapping = NULL,
  data = NULL,
```

```
  geom = "qq_band",
  position = "identity",
  na.rm = TRUE,
  show.legend = NA,
  inherit.aes = TRUE,
  distribution = "norm",
  dparams = list(),
  detrend = FALSE,
  identity = FALSE,
  qtype = 7,
  qprobs = c(0.25, 0.75),
  bandType = "pointwise",
  B = 1000,
  conf = 0.95,
  mu = NULL,
  sigma = NULL,
  ...
)
```

Arguments

mapping	Set of aesthetic mappings created by aes(). If specified and inherit.aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data	The data to be displayed in this layer. There are three options: If NULL, the default, the data is inherited from the plot data as specified in the call to ggplot(). A data.frame, or other object, will override the plot data. All objects will be fortified to produce a data frame. See fortify() for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a data.frame, and will be used as the layer data. A function can be created from a formula (e.g. ~ head(.x, 10)).
stat	statistic to use to calculate confidence bands. Should be 'qq_band'.
position	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The position argument accepts the following: • The result of calling a position function, such as position_jitter(). This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the position_ prefix. For example, to use position_jitter(), give the position as "jitter". • For more information and other ways to specify the position, see the layer position documentation.
na.rm	If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.

show.legend — logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use TRUE. If NA, all levels are shown in legend, but unobserved levels are omitted.

inherit.aes — If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification

distribution — Character. Theoretical probability distribution function to use. Do not provide the full distribution function name (e.g., "dnorm"). Instead, just provide its shortened name (e.g., "norm"). If you wish to provide a custom distribution, you may do so by first creating the density, quantile, and random functions following the standard nomenclature from the stats package (i.e., for "custom", create the dcustom, pcustom, qcustom, and rcustom functions).

dparams — List of additional parameters passed on to the previously chosen distribution function. If an empty list is provided (default) then the distributional parameters are estimated via MLE. MLE for custom distributions is currently not supported, so you must provide the appropriate dparams in that case.

detrend — Logical. Should the plot objects be detrended? If TRUE, the objects will be detrended according to the reference Q-Q line. This procedure was described by Thode (2002), and may help reducing visual bias caused by the orthogonal distances from Q-Q points to the reference line.

identity — Logical. Should an identity line be used as the reference line used to construct the confidence bands? If TRUE, the identity line is used. If FALSE (default), the commonly-used Q-Q line that intercepts two data quantiles specified in qprobs is used. Please notice that the chosen reference line will also be used for the detrending procedure, if detrend = TRUE.

qtype — Integer between 1 and 9. Type of the quantile algorithm to be used by the quantile function to construct the Q-Q line.

qprobs — Numeric vector of length two. Represents the quantiles used by the quantile function to construct the Q-Q line.

bandType — Character. Either "pointwise", "boot", "ks" or "ts", or "ell". "pointwise" constructs pointwise confidence bands based on Normal confidence intervals. "boot" creates pointwise confidence bands based on a parametric bootstrap; parameters are estimated with MLEs. "ks" constructs simultaneous confidence bands based on the Kolmogorov-Smirnov test. "ts" constructs tail-sensitive confidence bands, as described by Aldor-Noiman et al. (2013) (also, see 'Note' for limitations). Finally, "ell" constructs simultaenous bands using the equal local levels test describe by Weine et al. (2021).

B — Integer. If bandType = "boot", then B is the number of bootstrap replicates. If bandType = "ts", then B is the number of simulated samples.

conf — Numerical. Confidence level of the bands.

mu — Numerical. Only used if bandType = "ts". Center distributional parameter used to construct the simulated tail-sensitive confidence bands. If either mu or sigma are NULL, then those parameters are estimated using Qn and s_Qn, respectively.

`sigma` Numerical. Only used if `bandType = "ts"`. Scale distributional parameter used to construct the simulated tail-sensitive confidence bands. If either `mu` or `sigma` are `NULL`, then those parameters are estimated using robust estimates from the **stats** package.

`...` Other arguments passed on to `layer()`'s `params` argument. These arguments broadly fall into one of 4 categories below. Notably, further arguments to the `position` argument, or aesthetics that are required can *not* be passed through `....` Unknown arguments that are not part of the 4 categories below are ignored.

- Static aesthetics that are not mapped to a scale, but are at a fixed value and apply to the layer as a whole. For example, `colour = "red"` or `linewidth = 3`. The geom's documentation has an **Aesthetics** section that lists the available options. The 'required' aesthetics cannot be passed on to the `params`. Please note that while passing unmapped aesthetics as vectors is technically possible, the order and required length is not guaranteed to be parallel to the input data.
- When constructing a layer using a `stat_*()` function, the `...` argument can be used to pass on parameters to the `geom` part of the layer. An example of this is `stat_density(geom = "area", outline.type = "both")`. The geom's documentation lists which parameters it can accept.
- Inversely, when constructing a layer using a `geom_*()` function, the `...` argument can be used to pass on parameters to the `stat` part of the layer. An example of this is `geom_area(stat = "density", adjust = 0.5)`. The stat's documentation lists which parameters it can accept.
- The `key_glyph` argument of `layer()` may also be passed on through `....` This can be one of the functions described as key glyphs, to change the display of the layer in the legend.

`geom` The geometric object to use to display the data for this layer. When using a `stat_*()` function to construct a layer, the `geom` argument can be used to override the default coupling between stats and geoms. The `geom` argument accepts the following:

- A `Geom` ggproto subclass, for example `GeomPoint`.
- A string naming the geom. To give the geom as a string, strip the function name of the `geom_` prefix. For example, to use `geom_point()`, give the geom as `"point"`.
- For more information and other ways to specify the geom, see the layer geom documentation.

Note

- Tail-sensitive confidence bands are only implemented for Normal Q-Q plots. As a future update, we intend to generalize to other distributions.
- Bootstrap bands are constructed based on a MLE parametric bootstrap. Hence, it is not possible to construct such bands if the sample and theoretical distributions present mismatching supports.

References

- Thode, H. (2002), Testing for Normality. CRC Press, 1st Ed.
- doi:10.1080/00031305.2013.847865
- Weine, E. et al. (2021). Application of Equal Local Levels to Improve Q-Q Plot Testing Bands with R Package qqconf.

Examples

```
# generate random Normal data
set.seed(0)
smp <- data.frame(norm = rnorm(100))

# Normal Q-Q plot of Normal data
gg <- ggplot(data = smp, mapping = aes(sample = norm)) +
 stat_qq_band() +
 stat_qq_line() +
 stat_qq_point()
gg + labs(x = "Theoretical Quantiles", y = "Sample Quantiles")

# Normal Q-Q plot of Normal data with equal local levels (ell) bands
bt <- "ell"
gg <- ggplot(data = smp, mapping = aes(sample = norm)) +
 stat_qq_band(bandType = bt) +
 stat_qq_line() +
 stat_qq_point() +
 labs(x = "Theoretical Quantiles", y = "Sample Quantiles")
gg

# Exponential Q-Q plot of mean ozone levels (airquality dataset)
di <- "exp"
dp <- list(rate = 1)
gg <- ggplot(data = airquality, mapping = aes(sample = Ozone)) +
 stat_qq_band(distribution = di, dparams = dp) +
 stat_qq_line(distribution = di, dparams = dp) +
 stat_qq_point(distribution = di, dparams = dp) +
 labs(x = "Theoretical Quantiles", y = "Sample Quantiles")
gg

# Detrended Exponential Q-Q plot of mean ozone levels
di <- "exp"
dp <- list(rate = 1)
de <- TRUE
gg <- ggplot(data = airquality, mapping = aes(sample = Ozone)) +
 stat_qq_band(distribution = di, detrend = de) +
 stat_qq_line(distribution = di, detrend = de) +
 stat_qq_point(distribution = di, detrend = de) +
 labs(x = "Theoretical Quantiles", y = "Sample Quantiles")
gg

## Not run:
# Normal Q-Q plot of Normal data with bootstrap confidence bands
```

```
bt <- "boot"
gg <- ggplot(data = smp, mapping = aes(sample = norm)) +
 stat_qq_band(bandType = bt) +
 stat_qq_line() +
 stat_qq_point() +
 labs(x = "Theoretical Quantiles", y = "Sample Quantiles")
gg

# Normal Q-Q plot of Normal data with tail-sensitive confidence bands
bt <- "ts"
gg <- ggplot(data = smp, mapping = aes(sample = norm)) +
 stat_qq_band(bandType = bt) +
 stat_qq_line() +
 stat_qq_point() +
 labs(x = "Theoretical Quantiles", y = "Sample Quantiles")
gg

## End(Not run)
```

---

iowa *2012 BRFSS sample for the state of Iowa*

Description

2012 BRFSS sample for the state of Iowa

Usage

```
data(iowa)
```

Format

A data frame with 7166 observations on 3 variables:

SEX Gender

WTKG3 Weight in kg

HTIN4 Height in inch

Source

https://www.cdc.gov/brfss/annual_data/annual_2012.html

longjump *Men's Olympic Long Jump Qualifiers 2012*

Description

Men's Olympic Long Jump Qualifiers 2012

Usage

```
data(longjump)
```

Format

A data frame with 42 observations on the following 4 variables:

rank Athlete's rank at the qualifying event

name Athlete's name

country Athlete's country of origin

distance Result in meters

Source

https://www.olympics.com/en/olympic-games/london-2012/results/athletics/long-jump-men

qqplotr *Q-Q and P-P plot extensions for 'ggplot2'*

Description

This package extends some `ggplot2` functionalities by permitting the drawing of both quantile-quantile (Q-Q) and probability-probability (P-P) points, lines, and confidence bands. The functions of this package also allow the detrend adjustment, proposed by Thode (2002), which helps reduce visual bias when assessing those plots.

Details

The functions of this package, presented as `ggplot2` Stats, are divided into two groups: Q-Q and P-P related.

Each of the groups is composed of three Stats: `point`, `line`, and `band`. Those Stats, while independent, complement each other when plotted together.

Author(s)

Maintainer: Adam Loy <loyad01@gmail.com>

Authors:

- Alexandre Almeida <almeida.xan@gmail.com>
- Heike Hofmann

See Also

Useful links:

- https://github.com/aloy/qqplotr
- Report bugs at https://github.com/aloy/qqplotr/issues

`stat_pp_band` *Probability-probability confidence bands*

Description

Draws probability-probability confidence bands.

Usage

```
stat_pp_band(
  mapping = NULL,
  data = NULL,
  geom = "ribbon",
  position = "identity",
  na.rm = TRUE,
  show.legend = NA,
  inherit.aes = TRUE,
  distribution = "norm",
  dparams = list(),
  bandType = "boot",
  B = 1000,
  conf = 0.95,
  detrend = FALSE,
  ...
)
```

Arguments

`mapping`	Set of aesthetic mappings created by `aes()`. If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.

`data` The data to be displayed in this layer. There are three options:

If `NULL`, the default, the data is inherited from the plot data as specified in the call to `ggplot()`.

A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created.

A `function` will be called with a single argument, the plot data. The return value must be a `data.frame`, and will be used as the layer data. A `function` can be created from a `formula` (e.g. `~ head(.x, 10)`).

`geom` The geometric object to use to display the data for this layer. When using a `stat_*()` function to construct a layer, the `geom` argument can be used to override the default coupling between stats and geoms. The `geom` argument accepts the following:

- A `Geom` ggproto subclass, for example `GeomPoint`.
- A string naming the geom. To give the geom as a string, strip the function name of the `geom_` prefix. For example, to use `geom_point()`, give the geom as `"point"`.
- For more information and other ways to specify the geom, see the layer geom documentation.

`position` A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The `position` argument accepts the following:

- The result of calling a position function, such as `position_jitter()`. This method allows for passing extra arguments to the position.
- A string naming the position adjustment. To give the position as a string, strip the function name of the `position_` prefix. For example, to use `position_jitter()`, give the position as `"jitter"`.
- For more information and other ways to specify the position, see the layer position documentation.

`na.rm` If `FALSE`, the default, missing values are removed with a warning. If `TRUE`, missing values are silently removed.

`show.legend` logical. Should this layer be included in the legends? `NA`, the default, includes if any aesthetics are mapped. `FALSE` never includes, and `TRUE` always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use `TRUE`. If `NA`, all levels are shown in legend, but unobserved levels are omitted.

`inherit.aes` If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification

`distribution` Character. Theoretical probability distribution function to use. Do not provide the full distribution function name (e.g., `"dnorm"`). Instead, just provide its shortened name (e.g., `"norm"`). If you wish to provide a custom distribution, you may do so by first creating the density, quantile, and random functions following the standard nomenclature from the `stats` package (i.e., for `"custom"`, create the `dcustom`, `pcustom`, `qcustom`, and `rcustom` functions).

dparams — List of additional parameters passed on to the previously chosen `distribution` function. If an empty list is provided (default) then the distributional parameters are estimated via MLE. MLE for custom distributions is currently not supported, so you must provide the appropriate `dparams` in that case.

bandType — Character. Only `"boot"` and `"ell"` are available for now. `"boot"` creates pointwise confidence bands based on a bootstrap. `"ell"` constructs simultaenous bands using the equal local levels test.

B — Integer. If `bandType = "boot"`, then `B` is the number of bootstrap replicates.

conf — Numerical. Confidence level of the bands.

detrend — Logical. Should the plot objects be detrended? If `TRUE`, the objects will be detrended according to the default identity P-P line. This procedure was described by Thode (2002), and may help reducing visual bias caused by the orthogonal distances from P-P points to the reference line.

... — Other arguments passed on to `layer()`'s `params` argument. These arguments broadly fall into one of 4 categories below. Notably, further arguments to the `position` argument, or aesthetics that are required can *not* be passed through `....` Unknown arguments that are not part of the 4 categories below are ignored.

- Static aesthetics that are not mapped to a scale, but are at a fixed value and apply to the layer as a whole. For example, `colour = "red"` or `linewidth = 3`. The geom's documentation has an **Aesthetics** section that lists the available options. The 'required' aesthetics cannot be passed on to the `params`. Please note that while passing unmapped aesthetics as vectors is technically possible, the order and required length is not guaranteed to be parallel to the input data.
- When constructing a layer using a `stat_*()` function, the `...` argument can be used to pass on parameters to the `geom` part of the layer. An example of this is `stat_density(geom = "area", outline.type = "both")`. The geom's documentation lists which parameters it can accept.
- Inversely, when constructing a layer using a `geom_*()` function, the `...` argument can be used to pass on parameters to the `stat` part of the layer. An example of this is `geom_area(stat = "density", adjust = 0.5)`. The stat's documentation lists which parameters it can accept.
- The `key_glyph` argument of `layer()` may also be passed on through `....` This can be one of the functions described as key glyphs, to change the display of the layer in the legend.

References

- Thode, H. (2002), Testing for Normality. CRC Press, 1st Ed.
- doi:10.1080/00031305.2013.847865
- Weine, E. et al. (2021). Application of Equal Local Levels to Improve Q-Q Plot Testing Bands with R Package qqconf.

Examples

```
# generate random Normal data
```

```
set.seed(0)
smp <- data.frame(norm = rnorm(100), exp = rexp(100))

# Normal P-P plot of Normal data
gg <- ggplot(data = smp, mapping = aes(sample = norm)) +
 stat_pp_band() +
 stat_pp_line() +
 stat_pp_point() +
 labs(x = "Probability Points", y = "Cumulative Probability")
gg

# Shifted Normal P-P plot of Normal data
dp <- list(mean = 1.5)
gg <- ggplot(data = smp, mapping = aes(sample = norm)) +
 stat_pp_band(dparams = dp, bandType = "ell") +
 stat_pp_line() +
 stat_pp_point(dparams = dp) +
 labs(x = "Probability Points", y = "Cumulative Probability")
gg

# Exponential P-P plot of Exponential data
di <- "exp"
gg <- ggplot(data = smp, mapping = aes(sample = exp)) +
 stat_pp_band(distribution = di, bandType = "ell") +
 stat_pp_line() +
 stat_pp_point(distribution = di) +
 labs(x = "Probability Points", y = "Cumulative Probability")
gg

## Not run:
# Normal P-P plot of mean ozone levels (airquality dataset)
dp <- list(mean = 38, sd = 27)
gg <- ggplot(data = airquality, mapping = aes(sample = Ozone)) +
 stat_pp_band(dparams = dp) +
 stat_pp_line() +
stat_pp_point(dparams = dp) +
 labs(x = "Probability Points", y = "Cumulative Probability")
gg

## End(Not run)
```

`stat_pp_line` *Probability-probability lines*

Description

Draws a probability-probability line.

Usage

```
stat_pp_line(
  mapping = NULL,
  data = NULL,
  geom = "path",
  position = "identity",
  na.rm = TRUE,
  show.legend = NA,
  inherit.aes = TRUE,
  ab = c(0, 1),
  detrend = FALSE,
  ...
)
```

Arguments

mapping	Set of aesthetic mappings created by aes(). If specified and inherit.aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data	The data to be displayed in this layer. There are three options: If NULL, the default, the data is inherited from the plot data as specified in the call to ggplot(). A data.frame, or other object, will override the plot data. All objects will be fortified to produce a data frame. See fortify() for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a data.frame, and will be used as the layer data. A function can be created from a formula (e.g. ~ head(.x, 10)).
geom	The geometric object to use to display the data for this layer. When using a stat_*() function to construct a layer, the geom argument can be used to override the default coupling between stats and geoms. The geom argument accepts the following: • A Geom ggproto subclass, for example GeomPoint. • A string naming the geom. To give the geom as a string, strip the function name of the geom_ prefix. For example, to use geom_point(), give the geom as "point". • For more information and other ways to specify the geom, see the layer geom documentation.
position	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The position argument accepts the following: • The result of calling a position function, such as position_jitter(). This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the position_ prefix. For example, to use position_jitter(), give the position as "jitter".

- For more information and other ways to specify the position, see the layer position documentation.

`na.rm` If `FALSE`, the default, missing values are removed with a warning. If `TRUE`, missing values are silently removed.

`show.legend` logical. Should this layer be included in the legends? `NA`, the default, includes if any aesthetics are mapped. `FALSE` never includes, and `TRUE` always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use `TRUE`. If `NA`, all levels are shown in legend, but unobserved levels are omitted.

`inherit.aes` If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification

`ab` Numeric vector of length two. The intercept (`a`) and slope (`b`) of the P-P line. Defaults to the identity line (`a = 0, b = 1`).

`detrend` Logical. Should the plot objects be detrended? If `TRUE`, the objects will be detrended according to the default identity P-P line. This procedure was described by Thode (2002), and may help reducing visual bias caused by the orthogonal distances from P-P points to the reference line.

`...` Other arguments passed on to `layer()`'s `params` argument. These arguments broadly fall into one of 4 categories below. Notably, further arguments to the `position` argument, or aesthetics that are required can *not* be passed through `...`. Unknown arguments that are not part of the 4 categories below are ignored.

- Static aesthetics that are not mapped to a scale, but are at a fixed value and apply to the layer as a whole. For example, `colour = "red"` or `linewidth = 3`. The geom's documentation has an **Aesthetics** section that lists the available options. The 'required' aesthetics cannot be passed on to the `params`. Please note that while passing unmapped aesthetics as vectors is technically possible, the order and required length is not guaranteed to be parallel to the input data.
- When constructing a layer using a `stat_*()` function, the `...` argument can be used to pass on parameters to the `geom` part of the layer. An example of this is `stat_density(geom = "area", outline.type = "both")`. The geom's documentation lists which parameters it can accept.
- Inversely, when constructing a layer using a `geom_*()` function, the `...` argument can be used to pass on parameters to the `stat` part of the layer. An example of this is `geom_area(stat = "density", adjust = 0.5)`. The stat's documentation lists which parameters it can accept.
- The `key_glyph` argument of `layer()` may also be passed on through `...`. This can be one of the functions described as key glyphs, to change the display of the layer in the legend.

Examples

```
# generate random Normal data
set.seed(0)
smp <- data.frame(norm = rnorm(100))
```

```
# Normal P-P plot of Normal data
gg <- ggplot(data = smp, mapping = aes(sample = norm)) +
 stat_pp_line() +
 stat_pp_point() +
 labs(x = "Probability Points", y = "Cumulative Probability")
gg

# Shifted Normal P-P plot of Normal data
dp <- list(mean = 1.5)
gg <- ggplot(data = smp, mapping = aes(sample = norm)) +
 stat_pp_line() +
 stat_pp_point(dparams = dp) +
 labs(x = "Probability Points", y = "Cumulative Probability")
gg

# Normal P-P plot of mean ozone levels (airquality dataset)
dp <- list(mean = 38, sd = 27)
gg <- ggplot(data = airquality, mapping = aes(sample = Ozone)) +
 stat_pp_line() +
stat_pp_point(dparams = dp) +
 labs(x = "Probability Points", y = "Cumulative Probability")
gg
```

`stat_pp_point` *Probability-probability points*

Description

Draws probability-probability points.

Usage

```
stat_pp_point(
  mapping = NULL,
  data = NULL,
  geom = "point",
  position = "identity",
  na.rm = TRUE,
  show.legend = NA,
  inherit.aes = TRUE,
  distribution = "norm",
  dparams = list(),
  detrend = FALSE,
  down.sample = NULL,
  ...
)
```

Arguments

mapping	Set of aesthetic mappings created by aes(). If specified and inherit.aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data	The data to be displayed in this layer. There are three options: If NULL, the default, the data is inherited from the plot data as specified in the call to ggplot(). A data.frame, or other object, will override the plot data. All objects will be fortified to produce a data frame. See fortify() for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a data.frame, and will be used as the layer data. A function can be created from a formula (e.g. ~ head(.x, 10)).
geom	The geometric object to use to display the data for this layer. When using a stat_*() function to construct a layer, the geom argument can be used to override the default coupling between stats and geoms. The geom argument accepts the following: • A Geom ggproto subclass, for example GeomPoint. • A string naming the geom. To give the geom as a string, strip the function name of the geom_ prefix. For example, to use geom_point(), give the geom as "point". • For more information and other ways to specify the geom, see the layer geom documentation.
position	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The position argument accepts the following: • The result of calling a position function, such as position_jitter(). This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the position_ prefix. For example, to use position_jitter(), give the position as "jitter". • For more information and other ways to specify the position, see the layer position documentation.
na.rm	If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.
show.legend	logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use TRUE. If NA, all levels are shown in legend, but unobserved levels are omitted.
inherit.aes	If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification

`distribution` Character. Theoretical probability distribution function to use. Do not provide the full distribution function name (e.g., `"dnorm"`). Instead, just provide its shortened name (e.g., `"norm"`). If you wish to provide a custom distribution, you may do so by first creating the density, quantile, and random functions following the standard nomenclature from the `stats` package (i.e., for `"custom"`, create the `dcustom`, `pcustom`, `qcustom`, and `rcustom` functions).

`dparams` List of additional parameters passed on to the previously chosen `distribution` function. If an empty list is provided (default) then the distributional parameters are estimated via MLE. MLE for custom distributions is currently not supported, so you must provide the appropriate `dparams` in that case.

`detrend` Logical. Should the plot objects be detrended? If `TRUE`, the objects will be detrended according to the default identity P-P line. This procedure was described by Thode (2002), and may help reducing visual bias caused by the orthogonal distances from P-P points to the reference line.

`down.sample` Integer specifying how many points you want to sample in a reduced sample (i.e., a down sample). The default value is `NULL` indicating no downsampling.

`...` Other arguments passed on to `layer()`'s `params` argument. These arguments broadly fall into one of 4 categories below. Notably, further arguments to the `position` argument, or aesthetics that are required can *not* be passed through `....` Unknown arguments that are not part of the 4 categories below are ignored.

- Static aesthetics that are not mapped to a scale, but are at a fixed value and apply to the layer as a whole. For example, `colour = "red"` or `linewidth = 3`. The geom's documentation has an **Aesthetics** section that lists the available options. The 'required' aesthetics cannot be passed on to the `params`. Please note that while passing unmapped aesthetics as vectors is technically possible, the order and required length is not guaranteed to be parallel to the input data.
- When constructing a layer using a `stat_*()` function, the `...` argument can be used to pass on parameters to the `geom` part of the layer. An example of this is `stat_density(geom = "area", outline.type = "both")`. The geom's documentation lists which parameters it can accept.
- Inversely, when constructing a layer using a `geom_*()` function, the `...` argument can be used to pass on parameters to the `stat` part of the layer. An example of this is `geom_area(stat = "density", adjust = 0.5)`. The stat's documentation lists which parameters it can accept.
- The `key_glyph` argument of `layer()` may also be passed on through `....` This can be one of the functions described as key glyphs, to change the display of the layer in the legend.

References

- Thode, H. (2002), Testing for Normality. CRC Press, 1st Ed.

Examples

```
# generate random Normal data
set.seed(0)
```

```
smp <- data.frame(norm = rnorm(100))

# Normal P-P plot of Normal data
gg <- ggplot(data = smp, mapping = aes(sample = norm)) +
 stat_pp_point() +
 labs(x = "Probability Points", y = "Cumulative Probability")
gg

# Shifted Normal P-P plot of Normal data
dp <- list(mean = 1.5)
gg <- ggplot(data = smp, mapping = aes(sample = norm)) +
 stat_pp_point(dparams = dp) +
 labs(x = "Probability Points", y = "Cumulative Probability")
gg

# Normal P-P plot of mean ozone levels (airquality dataset)
dp <- list(mean = 38, sd = 27)
gg <- ggplot(data = airquality, mapping = aes(sample = Ozone)) +
stat_pp_point(dparams = dp) +
 labs(x = "Probability Points", y = "Cumulative Probability")
gg
```

`stat_qq_line` *Quantile-quantile lines*

Description

Draws a quantile-quantile line, with an additional detrend option.

Usage

```
stat_qq_line(
  mapping = NULL,
  data = NULL,
  geom = "path",
  position = "identity",
  na.rm = TRUE,
  show.legend = NA,
  inherit.aes = TRUE,
  distribution = "norm",
  dparams = list(),
  detrend = FALSE,
  identity = FALSE,
  qtype = 7,
  qprobs = c(0.25, 0.75),
  ...
)
```

Arguments

mapping	Set of aesthetic mappings created by aes(). If specified and inherit.aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data	The data to be displayed in this layer. There are three options: If NULL, the default, the data is inherited from the plot data as specified in the call to ggplot(). A data.frame, or other object, will override the plot data. All objects will be fortified to produce a data frame. See fortify() for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a data.frame, and will be used as the layer data. A function can be created from a formula (e.g. ~ head(.x, 10)).
geom	The geometric object to use to display the data for this layer. When using a stat_*() function to construct a layer, the geom argument can be used to override the default coupling between stats and geoms. The geom argument accepts the following: • A Geom ggproto subclass, for example GeomPoint. • A string naming the geom. To give the geom as a string, strip the function name of the geom_ prefix. For example, to use geom_point(), give the geom as "point". • For more information and other ways to specify the geom, see the layer geom documentation.
position	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The position argument accepts the following: • The result of calling a position function, such as position_jitter(). This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the position_ prefix. For example, to use position_jitter(), give the position as "jitter". • For more information and other ways to specify the position, see the layer position documentation.
na.rm	If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.
show.legend	logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use TRUE. If NA, all levels are shown in legend, but unobserved levels are omitted.
inherit.aes	If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification

`distribution` Character. Theoretical probability distribution function to use. Do not provide the full distribution function name (e.g., `"dnorm"`). Instead, just provide its shortened name (e.g., `"norm"`). If you wish to provide a custom distribution, you may do so by first creating the density, quantile, and random functions following the standard nomenclature from the `stats` package (i.e., for `"custom"`, create the `dcustom`, `pcustom`, `qcustom`, and `rcustom` functions).

`dparams` List of additional parameters passed on to the previously chosen `distribution` function. If an empty list is provided (default) then the distributional parameters are estimated via MLE. MLE for custom distributions is currently not supported, so you must provide the appropriate `dparams` in that case.

`detrend` Logical. Should the plot objects be detrended? If `TRUE`, the objects will be detrended according to the reference Q-Q line. This procedure was described by Thode (2002), and may help reducing visual bias caused by the orthogonal distances from Q-Q points to the reference line.

`identity` Logical. Should an identity line be used as the reference line? If `TRUE`, the identity line is used. If `FALSE` (default), the commonly-used Q-Q line that intercepts two data quantiles specified in `qprobs` is used. Please notice that the chosen reference line will also be used for the detrending procedure, if `detrend = TRUE`.

`qtype` Integer between 1 and 9. Only used if `detrend = TRUE` and `identity = FALSE`. Type of the quantile algorithm to be used by the `quantile` function to construct the Q-Q line.

`qprobs` Numeric vector of length two. Only used if `detrend = TRUE` and `identity = FALSE`. Represents the quantiles used by the `quantile` function to construct the Q-Q line.

`...` Other arguments passed on to `layer()`'s `params` argument. These arguments broadly fall into one of 4 categories below. Notably, further arguments to the `position` argument, or aesthetics that are required can *not* be passed through `....` Unknown arguments that are not part of the 4 categories below are ignored.

- Static aesthetics that are not mapped to a scale, but are at a fixed value and apply to the layer as a whole. For example, `colour = "red"` or `linewidth = 3`. The geom's documentation has an **Aesthetics** section that lists the available options. The 'required' aesthetics cannot be passed on to the `params`. Please note that while passing unmapped aesthetics as vectors is technically possible, the order and required length is not guaranteed to be parallel to the input data.
- When constructing a layer using a `stat_*()` function, the `...` argument can be used to pass on parameters to the `geom` part of the layer. An example of this is `stat_density(geom = "area", outline.type = "both")`. The geom's documentation lists which parameters it can accept.
- Inversely, when constructing a layer using a `geom_*()` function, the `...` argument can be used to pass on parameters to the `stat` part of the layer. An example of this is `geom_area(stat = "density", adjust = 0.5)`. The stat's documentation lists which parameters it can accept.
- The `key_glyph` argument of `layer()` may also be passed on through `....` This can be one of the functions described as key glyphs, to change the display of the layer in the legend.

References

- Thode, H. (2002), Testing for Normality. CRC Press, 1st Ed.

Examples

```
# generate random Normal data
set.seed(0)
smp <- data.frame(norm = rnorm(100))

# Normal Q-Q plot of Normal data
gg <- ggplot(data = smp, mapping = aes(sample = norm)) +
 stat_qq_line() +
 stat_qq_point() +
 labs(x = "Theoretical Quantiles", y = "Sample Quantiles")
gg

# Exponential Q-Q plot of mean ozone levels (airquality dataset)
di <- "exp"
dp <- list(rate = 1)
gg <- ggplot(data = airquality, mapping = aes(sample = Ozone)) +
 stat_qq_line(distribution = di, dparams = dp) +
 stat_qq_point(distribution = di, dparams = dp) +
 labs(x = "Theoretical Quantiles", y = "Sample Quantiles")
gg

# Detrended Exponential Q-Q plot of mean ozone levels
di <- "exp"
dp <- list(rate = 1)
de <- TRUE
gg <- ggplot(data = airquality, mapping = aes(sample = Ozone)) +
 stat_qq_line(distribution = di, detrend = de) +
 stat_qq_point(distribution = di, detrend = de) +
 labs(x = "Theoretical Quantiles", y = "Sample Quantiles")
gg
```

`stat_qq_point` *Quantile-quantile points*

Description

Draws quantile-quantile points, with an additional detrend option.

Usage

```
stat_qq_point(
  mapping = NULL,
  data = NULL,
  geom = "point",
```

```
  position = "identity",
  na.rm = TRUE,
  show.legend = NA,
  inherit.aes = TRUE,
  distribution = "norm",
  dparams = list(),
  detrend = FALSE,
  identity = FALSE,
  qtype = 7,
  qprobs = c(0.25, 0.75),
  down.sample = NULL,
  ...
)
```

Arguments

mapping	Set of aesthetic mappings created by aes(). If specified and inherit.aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
data	The data to be displayed in this layer. There are three options: If NULL, the default, the data is inherited from the plot data as specified in the call to ggplot(). A data.frame, or other object, will override the plot data. All objects will be fortified to produce a data frame. See fortify() for which variables will be created. A function will be called with a single argument, the plot data. The return value must be a data.frame, and will be used as the layer data. A function can be created from a formula (e.g. ~ head(.x, 10)).
geom	The geometric object to use to display the data for this layer. When using a stat_*() function to construct a layer, the geom argument can be used to override the default coupling between stats and geoms. The geom argument accepts the following: • A Geom ggproto subclass, for example GeomPoint. • A string naming the geom. To give the geom as a string, strip the function name of the geom_ prefix. For example, to use geom_point(), give the geom as "point". • For more information and other ways to specify the geom, see the layer geom documentation.
position	A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The position argument accepts the following: • The result of calling a position function, such as position_jitter(). This method allows for passing extra arguments to the position. • A string naming the position adjustment. To give the position as a string, strip the function name of the position_ prefix. For example, to use position_jitter(), give the position as "jitter".

- For more information and other ways to specify the position, see the layer position documentation.

`na.rm` If `FALSE`, the default, missing values are removed with a warning. If `TRUE`, missing values are silently removed.

`show.legend` logical. Should this layer be included in the legends? `NA`, the default, includes if any aesthetics are mapped. `FALSE` never includes, and `TRUE` always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use `TRUE`. If `NA`, all levels are shown in legend, but unobserved levels are omitted.

`inherit.aes` If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification

`distribution` Character. Theoretical probability distribution function to use. Do not provide the full distribution function name (e.g., `"dnorm"`). Instead, just provide its shortened name (e.g., `"norm"`). If you wish to provide a custom distribution, you may do so by first creating the density, quantile, and random functions following the standard nomenclature from the `stats` package (i.e., for `"custom"`, create the `dcustom`, `pcustom`, `qcustom`, and `rcustom` functions).

`dparams` List of additional parameters passed on to the previously chosen `distribution` function. If an empty list is provided (default) then the distributional parameters are estimated via MLE. MLE for custom distributions is currently not supported, so you must provide the appropriate `dparams` in that case.

`detrend` Logical. Should the plot objects be detrended? If `TRUE`, the objects will be detrended according to the reference Q-Q line. This procedure was described by Thode (2002), and may help reducing visual bias caused by the orthogonal distances from Q-Q points to the reference line.

`identity` Logical. Only used if `detrend = TRUE`. Should an identity line be used as the reference line for the plot detrending? If `TRUE`, the points will be detrended according to the reference identity line. If `FALSE` (default), the commonly-used Q-Q line that intercepts two data quantiles specified in `qprobs` is used.

`qtype` Integer between 1 and 9. Only used if `detrend = TRUE` and `identity = FALSE`. Type of the quantile algorithm to be used by the `quantile` function to construct the Q-Q line.

`qprobs` Numeric vector of length two. Only used if `detrend = TRUE` and `identity = FALSE`. Represents the quantiles used by the `quantile` function to construct the Q-Q line.

`down.sample` Integer specifying how many points you want to sample in a reduced sample (i.e., a down sample). The default value is `NULL` indicating no downsampling.

`...` Other arguments passed on to `layer()`'s `params` argument. These arguments broadly fall into one of 4 categories below. Notably, further arguments to the `position` argument, or aesthetics that are required can *not* be passed through `....` Unknown arguments that are not part of the 4 categories below are ignored.

- Static aesthetics that are not mapped to a scale, but are at a fixed value and apply to the layer as a whole. For example, `colour = "red"` or `linewidth = 3`. The geom's documentation has an **Aesthetics** section that lists the

available options. The 'required' aesthetics cannot be passed on to the `params`. Please note that while passing unmapped aesthetics as vectors is technically possible, the order and required length is not guaranteed to be parallel to the input data.

- When constructing a layer using a `stat_*()` function, the `...` argument can be used to pass on parameters to the `geom` part of the layer. An example of this is `stat_density(geom = "area", outline.type = "both")`. The geom's documentation lists which parameters it can accept.
- Inversely, when constructing a layer using a `geom_*()` function, the `...` argument can be used to pass on parameters to the `stat` part of the layer. An example of this is `geom_area(stat = "density", adjust = 0.5)`. The stat's documentation lists which parameters it can accept.
- The `key_glyph` argument of `layer()` may also be passed on through `....` This can be one of the functions described as key glyphs, to change the display of the layer in the legend.

References

- Thode, H. (2002), Testing for Normality. CRC Press, 1st Ed.

Examples

```
# generate random Normal data
set.seed(0)
smp <- data.frame(norm = rnorm(100))

# Normal Q-Q plot of simulated Normal data
gg <- ggplot(data = smp, mapping = aes(sample = norm)) +
 stat_qq_point() +
 labs(x = "Theoretical Quantiles", y = "Sample Quantiles")
gg

# Exponential Q-Q plot of mean ozone levels (airquality dataset)
di <- "exp"
dp <- list(rate = 1)
gg <- ggplot(data = airquality, mapping = aes(sample = Ozone)) +
 stat_qq_point(distribution = di, dparams = dp) +
 labs(x = "Theoretical Quantiles", y = "Sample Quantiles")
gg
```

Index