Package: inferit (via r-universe)

August 28, 2024

Type Package

Version 0.3.1

Title Hypothesis Tests and Statistical Distributions for 'SciViews::R'

Description Statistical distributions (including their visual representation) and hypothesis tests with rich-formatted tabular outputs for the 'SciViews::R' dialect.

Maintainer Philippe Grosjean <phgrosjean@sciviews.org>

Depends R (>= 4.2.0)

Imports chart (>= 1.5.0), distributional (>= 0.3.2), flextable (>= 0.9.1), ggplot2 (>= 3.4.2), knitr (>= 1.42), rlang (>= 1.1.1), stats (>= 4.2.0), tabularise (>= 0.6.0)

Suggests equatags (>= 0.2.0), equatiomatic (>= 0.3.0), rmarkdown (>= 2.21), spelling (>= 2.2.1), testthat (>= 3.0.0)

Remotes SciViews/chart, SciViews/tabularise

License MIT + file LICENSE

URL https://github.com/SciViews/inferit, https://www.sciviews.org/inferit/

BugReports https://github.com/SciViews/inferit/issues

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

VignetteBuilder knitr

Encoding UTF-8

Language en-US

ByteCompile yes

Config/testthat/edition 3

Repository https://sciviews.r-universe.dev

RemoteUrl https://github.com/SciViews/inferit

RemoteRef HEAD

RemoteSha 6256cd24a42b1c4ed7b50d102e63474209c8abed

Contents

`inferit-package` *Hypothesis Tests and Statistical Distributions for 'SciViews::R'*

Description

Statistical distributions and hypothesis tests objects with rich-formatted charts and tables.

Important functions

- `tabularise()` methods for **htest** objects.
- `stddev()` for **distribution** objects.
- `dfun()`, `cdfun()` density function for **distribution** objects.
- `chart()` method for **distribution** objects.
- `geom_funfill()` fills a part of a distribution density function.

`dfun` *Create and plot density functions for distribution objects*

Description

The **distribution** objects represent one or more statistical distributions. The functions `dfun()` and `geom_funfill()`, together with `chart()` allow to plot them.

Usage

```
dfun(object, i = 1)

cdfun(object, i = 1)

## S3 method for class 'distribution'
autoplot(
  object,
  n = 500,
  xlim = NULL,
  size = 99.5,
  xlab = "Quantile",
```

```
  ylab = if (type == "density") "Probability density" else
    "Cumulative probability density",
  plot.it = TRUE,
  use.chart = FALSE,
  ...,
  type = "density",
  theme = NULL
)

## S3 method for class 'distribution'
chart(data, ..., type = "density", env = parent.frame())

geom_funfill(
  mapping = NULL,
  data = NULL,
  fun,
  from,
  to,
  geom = "area",
  fill = "salmon",
  alpha = 0.5,
  ...
)
```

Arguments

object	A **distribution** object, as from the {distributional} package.
i	The distribution to use from the list (first one by default)
n	The number of points to use to draw the density functions (500 by default) of continuous distributions.
xlim	Two numbers that limit the X axis.
size	If `xlim=` is not provided, it is automatically calculated using the size of the CI between 0 and 100 (99.5 by default) for continuous distributions.
xlab	The label of the X axis ("Quantile" by default).
ylab	The label of the Y axis ("Probability density" or "Cumulative probability density" by default).
plot.it	Should the densities be plotted for all the distributions (`TRUE` by default)?
use.chart	Should `chart()` be used (`TRUE` by default)? Otherwise, `ggplot()` is used.
...	Further arguments to `stat_function()`.
type	The type of plot ("density" by default, or "cumulative").
theme	The theme for the plot (ignored for now).
data	The data frame to use (`NULL` by default).
env	The environment to use to evaluate expressions.
mapping	the mapping to use (`NULL` by default.

fun	The function to use (could be dfun(distribution_object)).
from	The first quantile to delimit the filled area.
to	The second quantile to delimit the filled area.
geom	The geom to use ("area" by default).
fill	The color to fill the area ("salmon" by default).
alpha	The alpha transparency to apply, 0.5 by default.

Value

Either a function or a ggplot object.

Examples

```
library(distributional)
library(chart)
di1 <- dist_normal(mu = 1, sigma = 1.5)
chart(di1) +
  geom_funfill(fun = dfun(di1), from = -5, to = 1)

# With two distributions
di2 <- c(dist_normal(10, 1), dist_student_t(df = 3, 13, 1))
chart(di2) +
  geom_funfill(fun = dfun(di2, 1), from = -5, to = 0) +
  geom_funfill(fun = dfun(di2, 2), from = 2, to = 6, fill = "turquoise3")
chart$cumulative(di2)
# A discrete distribution
di3 <- dist_binomial(size = 7, prob = 0.5)
chart(di3)
chart$cumulative(di3)
# A continuous together with a discrete distribution
di4 <- c(dist_normal(mu = 4, sigma = 2), dist_binomial(size = 8, prob = 0.5))
chart(di4)
chart$cumulative(di4)
```

---

stddev *Get standard deviation for a distribution objects*

Description

The **distribution** objects represent one or more statistical distributions. The generic functions stddev() returns the standard deviation for these distributions.

Usage

```
stddev(x, ...)

## Default S3 method:
```

```
stddev(x, ...)

## S3 method for class 'distribution'
stddev(x, ...)
```

Arguments

x	A **distribution** object, as from the {distributional} package.
...	Further arguments (not used yet).

Value

A numeric vector with one or more standard deviations.

Examples

```
library(distributional)
n1 <- dist_normal(mu = 1, sigma = 1.5)
n1
class(n1)
family(n1)
mean(n1)
variance(n1)
stddev(n1)
```

---

tabularise_default.htest

Create a rich-formatted table from an htest object

Description

`tabularise()` an **htest** object (into a a **flextable**) that can be further post-edited..

Usage

```
## S3 method for class 'htest'
tabularise_default(
  data,
  header = TRUE,
  title = NULL,
  lang = getOption("data.io_lang", "en"),
  show.signif.stars = getOption("show.signif.stars", TRUE),
  ...,
  kind = "ft",
  env = parent.frame()
)
```

Arguments

`data`	An **htest** object
`header`	If `TRUE` (by default), add a header to the table
`title`	If `TRUE`, add a title to the table header. Default to the same value than header, except outside of a chunk where it is `FALSE` if a table caption is detected (`tbl-cap` YAML entry).
`lang`	The natural language to use. The default value can be set with, e.g., `options(data.io_lang = "fr")` for French.
`show.signif.stars`	If `TRUE`, add the significance stars to the table. The default value is obtained from `getOption("show.signif.stars"")`.
`...`	Additional arguments (unused for now).
`kind`	The kind of table to produce: "tt" for tinytable, or "ft" for flextable (default).
`env`	The environment where to evaluate lazyeval expressions (unused for now).

Value

A **flextable** object you can print in different forms or rearrange with the {flextable} functions.

Examples

```
data(iris)
iris_cor <- cor.test(iris$Sepal.Length, iris$Sepal.Width)
tabularise::tabularise(iris_cor)

tabularise::tabularise(t.test(x = 1:10, y = 7:20), lang = "fr")
```

Index