## Package ‘edfun’

August 29, 2016

**Type** Package  
**Title** Creating Empirical Distribution Functions  
**Version** 0.2.0  
**Date** 2016-08-27  
**Depends** R (>= 3.0.0)  
**Imports** stats  
**Suggests** knitr, rmarkdown  
**Description** Easily creating empirical distribution functions from data: 'dfun', 'pfun', 'qfun' and 'rfun'.

**VignetteBuilder** knitr  
**License** GPL-2 | GPL-3  
**URL**  
  - https://cran.r-project.org/package=edfun,  
  - https://github.com/talgalili/edfun/,  
  - https://www.r-statistics.com/tag/edfun/  
**BugReports** https://github.com/talgalili/edfun/issues  
**LazyData** TRUE  
**RoxygenNote** 5.0.1  
**NeedsCompilation** no  
**Author** Tal Galili [aut, cre, cph] (https://www.r-statistics.com)  
**Maintainer** Tal Galili <tal.galili@gmail.com>  
**Repository** CRAN  
**Date/Publication** 2016-08-27 14:24:35

### R topics documented:

- edfun

---

Index

1
edfun

Creating Empirical Distribution Functions

Description

A function for creating a set of (one dimensional) empirical distribution functions (density, CDF, inv-CDF, and random number generator). This is either based on a vector of observations from the distribution, or a density function.

Usage

edfun(x, support = range(x), dfun, qfun_method = NULL, ...)

Arguments

x
numeric vector of data or (in case density is not NULL) a sequence of values for which to evaluate the density function for creating the inv-CDF. Also, the rfun will be based on the inverse CDF on uniform distribution (inv-CDF(U[0,1]) - which is "better" than using sample, if we have the density).

support
a 2d numeric vector giving the boundaries of the distribution. Default is the range of x. This is used in qfun to decide how to work with extreme cases of q->0|1.

dfun
a density function. If supplied, this creates a different pfun (which now relies on integrate) and rfun (which will now rely on inv-CDF(U[0,1])). If missing, then it is created using density. If NULL then it is not created.

qfun_method
can get a quantile function to use (for example "quantile"), with the first parameter accepts the data (x) and the second accepts probs (numeric vector of probabilities with values in [0,1]). If it is NULL (the default) then the quantiles are estimated using approxfun from predicting the x values from the pfun(x) values.

... ignored

Value

A list with 4+ components: dfun, pfun, qfun and rfun. The 5th component is pfun_integrate_dfun which is NULL if dfun is not supplied. If it is supplied, it returns a function that relies on integrate of dfun for returning pfun. Since this method is VERY slow, it is not returned within pfun. Instead, pfun will pre-compute pfun_integrate_dfun on all values of x.

Each component is a function to perform the usual tasks of distributions.

Examples

set.seed(2016-08-18)
x <- rnorm(100)
x_funs <- edfun(x)
x_funs$qfun(0) # -2.6

# for extreme cases, we can add the support vector
x_funs <- edfun(x, support = c(-Inf, Inf))
x_funs$qfun(0) # -Inf

f <- x_funs$dfun
curve(f, -2,2)

f <- x_funs$pfun
curve(f, -2,2)

f <- x_funs$qfun
curve(f, 0,1)

f <- x_funs$rfunc
hist(f(1000))
Index

approxfun, 2
density, 2
defun, 2
integrate, 2
sample, 2