Package 'DistributionFitR'

October 12, 2022

Type Package

Title Fitting Multiple Parametric Distributions

Version 0.1

Date 2020-02-20

Author
Benedikt Geier [aut], Borui Niklas Zhu [cre, aut], Moritz Kern [aut], Manuel J. Hentschel [aut], Kiril Dik [aut], Moritz Lauffner [aut], Adrian Heppeler [aut], Nadine Tampe [aut], Leonardo Vela [aut], Till Freihaut [aut], Niclas Lietzow [aut], Helene Peter [aut], Martin Schlather [aut, cre], Yiqi Li [ctb]

Maintainer Borui Niklas Zhu <bzhu@mail.uni-mannheim.de>

Description Given an univariate dataset, returns the best fitting parameter families, as defined in Shao (2003) <doi:10.1007/B97553>, including their parameter estimates via maximum likelihood.

License GPL (>= 3)

Imports utils, stats, stringr, datasets, graphics, grDevices, methods, parallel, doParallel, foreach

Depends R (>= 3.5)

Encoding UTF-8

LazyData true

NeedsCompilation no

Repository CRAN

Date/Publication 2020-03-04 14:00:08 UTC

R topics documented:

`DistributionFitR-package` *Fitting Multiple Parametric Distributions*

Description

Parametric densities (or count densities) will be fitted to user-given univariate data via maximum likelihood. The user more or less only enters the data. The program automatically searches for parametric distributions and parameters thereof that best describe the data. It then returns the ten best parameter families including the fitted parameters.

DistributionFitR comes in with a standard search list of 408 parametric distribution families as given by R-packages on CRAN.

The package contains the following functions:

- `globalfit` Given some univariate data, determines the best fitting parametric distributions from R-packages
- `install.packages_DistributionFitR` Installs R-packages that contribute to DistributionFitR's standard search list
- `getFamilies` Browsing for distribution families
- `getFamily` Find all distributions within a package
- `getParams` For a single distribution family, parameters are determined; upper/lower limits and other chraracteristics are extracted

Note

The most relevant function is `globalfit`, with an S4 object as return value. See the examples on how to display the results, such as using `summary` or `hist`. Read more on the return value in `globalfit` and `globalfitSummary`.

For exotic packages used frequently or where parameter extraction is time-consuming, users may be interested to do the latter once with `getFamilies` and save the results for subsequent usage as argument in `globalfit`. The functions `getFamily` and `getParams` are lower-level functions invoked by `getFamilies`, and may be of usage for other purposes.

Author(s)

Borui Niklas Zhu, Benedikt Geier, Moritz Kern, Kiril Dik, Moritz Lauff, Manuel J. Hentschel, Adrian Heppeler, Niclas Lietzow, Till Freihaut, Tim Glockner, Nadine Tampe, Leonardo Vela, Helene Peter, Martin Schlather, Yiqi Li

Maintainer: Borui Niklas Zhu, <bzhu@mail.uni-mannheim.de>

References

- Zucchini, W. (2000). An introduction to model selection. *Journal of Mathematical Psychology*, **44**(1), 41-61.
- Shao, J. (2003) *Mathematical Statistics* New York: Springer, ISBN 978-0-387-21718-5.

See Also

globalfit, install.packages_DistributionFitR, getFamilies, getFamily, getParams

Examples

```
# example for globalfit
data <- rnorm(n = 100, mean = 70, sd = 4)
r <- globalfit(data, cores = if(interactive()) NULL else 2)
summary(r)

# example for getFamily
str(getFamily("stats"))

# example for getParams
getParams("beta", package = "stats")
```

---

getFamilies *Lists (all) distribution families with their parameters*

Description

This function extracts distribution families from R-packages along with their parameters and characteristics, and scans specific packages if desired.

Usage

```
getFamilies(all.packages)
```

Arguments

all.packages	character vector or missing or logical; package names in which to scan: **character** the distribution families given TRUE all installed packages FALSE base packages of R If missing, a list of recognized families is returned.

Details

This function retrieves distribution families from R-packages as specified in 'arguments'. A distribution family is defined as follows:
Take the functions dnorm, pnorm, qnorm and rnorm from the package 'stats'. These functions are related to sampling, density etc. of the normal distribution. The family name is the part of the function name without the prefixes 'd', 'p', 'q', or 'r' , in this case: 'norm'.

Value

A list of lists. Each sublist pertains to exactly one distribution family (such as 'norm' from 'stats') and contains:

`package` character; name of the package containing the family

`family` character; name of the family, as defined under "Details"

`family_info`

- `lower`: named numeric vector; lower bounds for distribution parameters
- `upper`: named numeric vector; upper bounds for distribution parameters
- `accepts_float`: named logical vector; whether each parameter can be any real number (`TRUE`) or only integers are valid inputs (`FALSE`)
- `defaults`: names numeric vector; set of default parameters that jointly constitute valid input. Used for optimisation.
- `log`: logical; whether log values are provided by the density function
- `discrete`: logical; whether the distribution itself (not the parameters) takes on discrete values only
- `support_min`: numeric; lower bound of the support, i.e. lowest value of x where $f(x) > 0$ numerically for any valid parameter set, f the density function
- `support_max`: numeric; upper bound of the support
- `support_max_depends_on`: named logical vector; whether `support_max` depends on the value of each parameter
- `support_min_depends_on`: named logical vector; whether `support_min` depends on the value of each parameter

Author(s)

Tim Glockner, Adrian Heppeler, Borui Niklas Zhu

Examples

```
str(getFamilies())
```

---

getFamily *Find all distributions in a package*

Description

Given the name of a package, this function finds all distribution families that are provided in the package. Distributions are identified by scanning all function names in a package for the pattern r***, p***, q***, d***, (*** representing at least one character)

Usage

```
getFamily(pkg)
```

Arguments

pkg	character string; name of the package

Value

A list of of lists, the latter with two named elements:

package	The name of the package as provided to the function
family	The name of the function that belongs to a distribution family, referred to as *** above.

Note

The function make do with at least two functions that have the same remainder of the function name.

Author(s)

Manuel Hentschel, Valentin von Trotha

Examples

```
str(getFamily("stats"))
```

---

getParams *Get parameters and their properties for a Distribution Family*

Description

Given a distribution family, this function attempts to retrieve the distribution parameters and various characteristics, such as: valid parameter ranges, whether they accept non-integer values and the support of the distribution.

Usage

```
getParams(fam, package)
```

Arguments

fam	character or list. If fam is a list or a named vector, then it has the two entries "package" and "family", e.g. list(package="stats", family="beta")
package	character. Optional argument if fam is given. In most cases the package can be determined by getParams itself.

Details

The family name is defined as the part of the function name that follows “d”, “p”, “q” and “r”. So in case of the continuous uniform the family name is “unif”.
The values returned by `getParams` are included in the DistributionFitR-package and updated with each package update. Users may use it to update the library of parameter characteristics themselves or find the function useful to use the parameter characteristics retrieved for other purposes.

Value

For `getParams` a list with components, each of them a named vector: The names are the arguments of the distribution family as specified in e.g. the “r<distributionFamilyName>”-function, the value is described below:

`lower` named vector; values: numeric, lower bound of the respective parameter value

`upper` named vector; values: numeric, upper bound of the respective parameter value. Length and names must coincide with entry `lower`.

`accepts_float` named vector; values: boolean, `TRUE` if respective parameter value is continuous, `FALSE` if only integers are accepted. Length and names must coincide with entry `lower`.

`defaults` named vector; values: numeric, default values for the respective parameter (needed for optimisation to work). Length and names must coincide with entry `lower`.

`log` single boolean; `TRUE` if log(probability) is provided by d[family](), `FALSE` if they are not. Generating log-probabilities oneself may be numerically unstable.

`discrete` single boolean; `TRUE` if only integers are taken as values, `FALSE` otherwise.

`support_min` single numeric; left bound of support of the distributions density, i.e. minimum value where the density is not zero.

`support_max` single numeric; right bound of support of the distributions density, i.e. maximum value where the density is not zero.

`supp_max_depends_on` named vector; value: booleans, `TRUE` if right support bound depends on the repective parameter, `FALSE` if not. Note that dependency is currently implemented as equality between bound and parameter, linear relationships may be implemented in the future. Length and names must coincide with entry `lower`.

`supp_max_depends_on` named vector; value: booleans, `TRUE` if right support bound depends on the repective parameter, `FALSE` if not. See note in entry `supp_max_depends_on`. Length and names must coincide with entry `lower`.

Author(s)

Benedikt Geier, Borui Niklas Zhu

See Also

See also `getFamilies` for a convenient wrapper available to the user where distributions are extracted from whole packages.

Examples

```
getParams("beta", package = "stats")
getParams("unif", package = "stats" )
```

`globalfit` *Detect continuity and fit multiple distributions to given data*

Description

Given a numerical data vector, this function fits multiple distributions with the maximum likelihood method and returns an object containing the best fitted parameters and information criteria. Refer to the "Examples" section or the result class `globalfit` on how to sort and output the results with e.g. `summary`.

Usage

```
globalfit(data, continuity = NULL, method = "MLE",
  verbose = TRUE, packages = "stats",
  append_packages = FALSE, cores = NULL,
  max_dim_discrete = Inf, sanity = 1,
  timeout = 5
  )
```

Arguments

`data`	numeric vector of data points.
`continuity`	logical; if `TRUE`, the data is fitted with continuous distributions. If no input is given, the data will be tested for continuity.
`method`	character; method for parameter estimation. So far only Maximum-Likelihood is implemented, thus this argument must be `"MLE"`.
`verbose`	logical; if `TRUE`, show progress and packages from where to fit distributions.
`packages`	either a character vector with names of packages; or a list such as those returned by `getFamilies` or `NULL`, i.e. all families known by this package (recommended). default: "stats".
`append_packages`	logical; if `TRUE` (default) appends packages specified in the argument `packages` to the standard search list, if `FALSE` `globalfit` will use only those packages and ignore the standard search list.
`max_dim_discrete`	non-negative integer; distributions with more non-continuous parameters than `max_dim_discrete` will not be considered. Manual setting is recommended if calculation speed has to be cut down.

cores	integer; number of CPU cores to be used in the calculations of best fitted parameters and information criteria.
sanity	either a positive numeric or logical; if it is a positive numeric, it controls a sanity check where obviously bad fits are filtered out. The smaller the number, the stricter the check will be executed and the more potential distributions will be rejected. If sanity = FALSE a sanity check is not carried out. (DistributionFitR generally depends on other packages to supply reasonable distribution functions.) Default is 1.
timeout	logical or numeric. if it is a positive numeric, it gives the seconds until timeout for the underlying optimiser optim. If timeout = FALSE no timeout is performed.

Details

If there is no continuity input given, this function first tests via multiple criteria whether the data is continuously or discretely distributed. Given that information, the related distributions from getFamilies() are fitted to the data via maximum likelihood method and information criteria are calculated. For discrete data not in the form of integers only, an appropriate linear transformation is applied to ensure stable optimization.

Since DistributionFitR technically allows for comparing over all distributions in any R-package, computation speed is likely to be an issue. The following may help:

- using argument packages with append_packages = FALSE to restrict the search to certain packages
- discarding distributions with too many discrete parameters using argument max_dim_discrete
- specifying timeout, which affects the maximum time spent on each distribution (not overall!). The value in timeout will not be translated directly to the actual maximum time due to differing number of times optim is run under different algorithms.

Value

globalfit returns an object of class globalfit.

Author(s)

Moritz Lauff, Kiril Dik, Nadine Tampe, Borui Niklas Zhu, Benedikt Geier, Moritz Kern

Examples

```
# Example 1
data <- rnorm(n = 100, mean = 70, sd = 4)
r <- globalfit(data, cores = if(interactive()) NULL else 2)
summary(r)

# continuous or discrete
```

```
# Example 2
# Alternatively, it is possible to input whether the data is
globalfit(data, continuity = TRUE)

# Example 3
# fit over all distribution in the standard search list
globalfit(data, packages = NULL)
```

`globalfit-class` *Class "globalfit"*

Description

The class `globalfit` handles return objects from globalfit. It contains for some given data a list of fitted distributions, their estimated parameters and supplementary information.

Objects from the Class

Objects can be created by calls of the form `new("globalfit", data, continuity, method, fits)`. More comfortably, you may use the function `globalfit`. The result of these calls is a globalfit object.

Slots

call the call, which created this object

data vector of data points

continuity logical; if `TRUE`, indicating that the data points come from a continuous distribution; if `FALSE`, indicating that they come from a discrete distribution

method character; the method used for the fit.

fits list of S4-objects of class `optimParams`

Methods

summary `signature(object = "globalfit")`: summarizes the object and creates an object of `globalfitSummary`. Specify argument `ic` to choose how the results are to be sorted (as in method `sort`.

hist `signature(x = "globalfit")`: computes a histogram of the given data points and plots it together with the density of the estimated best fit. Specify argument `which` to choose which fitted density to overlay: the number of the fit as returned by `summary`; i.e. `which = 1` for the best fit, `which = 2` for the second-best etc.
Default is `1`.

print `signature(x = "globalfit")`: applies the method `summary` and prints the result.

AIC `signature(x = "globalfit")`: shows the AIC value of the fits. Specify argument `n` to display AIC for the `n` best fits according to this criterion.

BIC `signature(x = "globalfit")`: shows the BIC value of the fits. Specify argument `n` to display BIC for the `n` best fits according to this criterion.

sort `signature(x = "globalfit")`: sorts the results in slot `fits` by the information criterium selected. in argument `ic`. Available options are `"AIC"`, `"BIC"` or `"AICc"`.

Author(s)

Moritz Lauff, Kiril Dik, Nadine Tampe, Borui Niklas Zhu, Benedikt Geier, Moritz Kern

See Also

`globalfitSummary`
`optimParams`
`globalfit`

Examples

```
data <- rnorm(n = 100, mean = 10, sd = 1)
r <- globalfit(data, cores = if(interactive()) NULL else 2)

sort(r, ic = 'BIC')

print(r)

summary(r)
summary(r, ic = 'AICc', n = 7)

hist(r, ic = 'BIC', which = 4)

AIC(r, n = 2)
BIC(r)
```

---

globalfitSummary-class

Class "globalfitSummary"

Description

The `globalfitSummary`-class is the class to handle the entries of the class `globalfit` and assemble them together for summary.

Objects from the Class

Objects can be created by calls of the form `new("globalfitSummary", data, continuity, method, fits, ic)`. More comfortably, you may use the method `summary` on an object of class `globalfit` - its result is a `globalfitSummary`-object.

Slots

call the call, which created the globalfit, where this object originated.

data vector of data points

continuity logical; if TRUE, indicates that the data points come from a continuous distribution; if FALSE, indicates that they come from a discrete distribution

method character; the method used for the fit.

fits data frame; sorted by the ic selected in summary or the constructor call, with the columns: family, package and ic.

ic character; indicates by which crietrion fits was sorted.

Methods

show signature(x = "globalfitSummary"): display the object

print signature(x = "globalfitSummary"): calls show

Author(s)

Moritz Kern

See Also

globalfitSummary
globalfit
optimParams

Examples

```
data <- rnorm(n=100, mean=10, sd= 1)
r <- globalfit(data, cores= if (interactive()) NULL else 1,
               packages="stats", append_packages=FALSE)
summary(r)
```

---

install.packages_DistributionFitR

Installs all packages from DistributionFitR's standard search list.

Description

DistributionFitR comes with an extensive list of distribution families on CRAN together with their characteristics, referred to in our manuals as "standard search list". This function is a helper to install all the packages which contribute to this list. Usage is exactly like install.packages in base R, but without the need to specify which packages to install. If some packages fail to install, the others will not be affected (except dependencies).

Usage

```
install.packages_DistributionFitR(...)
```

Arguments

`...`	any argument that can be passed to base R's `install.packages`, except the argument `pkgs`.

Value

invisible `NULL`.

Author(s)

Borui Niklas Zhu

Examples

```
## Not run:
 # running the next command will run several minutes
 # and install many packages
 install.packages_DistributionFitR()

## End(Not run)
```

---

optimParams-class *Class "optimParams"*

Description

The class `optimParams` handles the slot `fits` of the class `globalfit`. It contains for some given data the optimization results and their characteristics.

Objects from the Class

Objects can be created by calls of the form `new("optimParams", family, package, estimatedValues, AIC, BIC, AICc, sanity)`. More comfortably, you may use the function `globalfit`. The list elements in the `fits`-slot of the result is a `optimParams`-object.

Slots

family character string; indicating the family name in the package, e.g. "norm".

package character string; indicating the package where the family was found, e.g. "stats".

estimatedValues named numeric vector; the estimated parameters of the family.

AIC numeric; Akaike Information Criterion.

BIC numeric; Bayes Information Criterion.

AICc numeric; small sample-corrected Akaike Information Criterion.

sanity named list; with numeric items `hist_check` und `int_check` and boolean item `good` as a result of the sanity check.

Author(s)

Moritz Kern

See Also

globalfitSummary, globalfit

Examples

```
data <- rnorm(n=100, mean=10, sd= 1)
r <- globalfit(data, cores= if (interactive()) NULL else 1,
               packages="stats", append_packages=FALSE)

# optimParams of best fit
r@fits[[1]]
```

Index