

Package ‘drob’

June 20, 2024

Title Compute robust estimates of dose-response model parameters

Description Compute robust estimates of potentially nonlinear and heteroscedastic dose-response model parameters using a 3-step algorithm in the spirit of MM-estimation.

Version 0.0.1

URL <https://github.com/memeplex/drob>

BugReports <https://github.com/memeplex/drob/issues>

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Imports drc (>= 3.0.1),
DEoptimR (>= 1.1.3),
MASS

R topics documented:

bisquare	1
drob	2
fpl	5
m_scale	6

Index	7
--------------	----------

bisquare	<i>Return bisquare and its derivatives</i>
----------	--

Description

This computes bisquare (aka Tukey’s biweight) function for a given cutoff point. It also computes its first two derivatives. All three functions are returned as elements of a list with names rho, psi and dps, as required by the step_1 parameter of the drob function.

Usage

bisquare(k)

Arguments

k The cutoff point below and above which the bisquare function evaluates to 1.

Value

A list with three elements:

- **rho**: The bisquare function.
- **psi**: The first derivative of the bisquare function.
- **dpsi**: The second derivative of the bisquare function.

Examples

```
x <- seq(-3, 3, length.out = 100)
bi2 <- bisquare(2)
plot(x, bi2$rho(x), type = "l", ylim = c(-1.5, 1.5))
lines(x, bi2$psi(x), col = 2)
lines(x, bi2$dpsi(x), col = 3)
```

drob

Compute robust estimates of dose-response model parameters

Description

drob computes an M-estimate of location from data x and y , given a potentially nonlinear and heteroscedastic model. It implements a 3-step procedure in the spirit of MM-estimation:

- Step 1: heuristically computes an initial location estimate t_0 .
- Step 2: computes scale estimates s for each dose, based on the residuals with respect to t_0 .
- Step 3: starting from t_0 and scaling by s a final location M-estimate t is found iteratively.

Each step may be fine-tuned using the parameters described below.

Usage

```
drob(
  x,
  y,
  model = "fpl",
  step_1 = "sbi",
  step_2 = "sbi",
  step_3 = "mbi",
  lts_q = 0.5,
  mbi_k = 3.44,
  sbi_k = 1.548,
  sl1_k = 1.48,
  de_args = identity,
  qn_args = identity,
  qn_gr = FALSE,
  ms_extend = 5,
  init_extend = 3
)
```

Arguments

- x** A vector of doses. It is expected that each dose is repeated enough times so as to be able to compute a good estimate of scale conditional to the dose during step 2.
- y** A vector of responses with the same length than x.
- model** The string "fpl" for the predefined 4PL model or, more generally, a list describing a model with at least two mandatory elements:
- **fun**: the model function, which takes doses x and parameters t as its two only arguments.
 - **init**: an initialization function that is able to produce lower and upper search bounds for step 1. It takes x and y as arguments, as well as an optional extend argument that controls the extension of the search region.
- A third optional element can also be included in the list:
- **grad**: the gradient of fun, which has the same signature. When grad is present it will be used both for computing a gradient in step 3 (if `qn_gr` is TRUE) and for computing the standard errors of the returned estimates.
- step_1** The loss function used for computing t_0 . It may be a function that takes a vector of residuals as its only argument or one of the following predefined strings:
- "lts": lts_q-upper-trimmed mean of squared residuals.
 - "lms": median of squared residuals.
 - "ml1": mean of absolute residuals. This implements an M-estimate with $\rho(r) = |r|$, i.e. L1-regression.
 - "sl1": median of absolute residuals. This implements an S-estimate with $\rho(r) = I(|r| > 1)$.
 - "mbi": loss for bisquare M-estimate with cutoff point `mbi_k`, scaled by $1 / \text{mad}(y)$ so as to make it scale-equivariant.
 - "sbi": (the default) loss for bisquare S-estimate with cutoff point `sbi_k`. The root finding search interval will be extended according to `ms_extend` (see the documentation for `m_scale`).
- $\text{loss}(y - \text{model}\$fun(x, t))$ will be minimized with respect to t using a differential evolution routine provided by the `DEoptimR` package. Lower and upper bounds for the search come from `model$init` as documented for the `model` parameter. Other arguments passed to `JDEoptim` may be overridden by passing a `de_args` function. If the optimizer fails to converge to a solution, the entire `drob` function execution is aborted with an error.
- step_2** The function used to compute a scale estimate for each dose. It may be a function taking a vector of residuals (relative to t_0 , as computed in step 1) and returning a scale estimate, or one of the following predefined strings:
- "sl1": median of absolute residuals. This is an M-estimate of scale with $\rho(r) = I(|r| > 1)$. It is scaled by `sl1_k`.
 - "sbi": (the default) bisquare M-estimate of scale with cutoff point `sbi_k`.
- The default values for `sl1_k` and `sbi_k` make the estimates equal to 1 under the standard normal distribution. The function is called once for each different dose, passing it a vector of residuals for such dose.
- step_3** The loss function used to compute t. It may be the string "mbi" (the default) for a bisquare loss defining an M-estimate with cutoff point `mbi_k` or, more generally, a list containing at the least the following mandatory element:

- rho: a rho-function taking a vector of scaled residuals.

The list may also contain two optional elements:

- psi: the first derivative of rho.
- dpsi: the second derivative of rho.

When psi and dpsi are present (as well as model\$grad) standard errors for the returned estimates will be computed. Moreover, when psi is present (as well as model\$grad and also qn_gr is TRUE) a gradient function is composed and passed as the argument for the parameter gr of optim.

This step minimizes $\text{loss}((y - \text{model}\$fun(x, t)) / s)$ with respect to t. It uses the standard optim routine with initial parameter t0 (also used for parameter scaling, see the parscale control parameter of optim) and sequentially attempting the following three methods in order:

1. Quasi-Newton L-BFGS-B with the same bounds than in step 1.
2. Quasi-Newton BFGS.
3. Conjugate gradient CG.

The result of the first successful optimizer is kept. If all optimizers fail to converge to a solution, the entire drob function execution is aborted with an error. Other arguments passed to optim may be overridden by passing a qn_args function.

lts_q	The proportion of data to be removed by the upper-trimmed mean loss (used when step_1 is "lts"). By default 0.5 to achieve a high breakdown point.
mbi_k	The cutoff point used for bisquare M-estimates (argument "mbi" to step_1 or step_3). By default 3.44 in order to achieve about 85% asymptotic efficiency under the normal distribution. Other typical values and their asymptotic efficiencies are: 3.14 (80%), 3.88 (90%), 4.68 (95%).
sbi_k	The cutoff point used for bisquare M-estimates of scale and, consequently, also for bisquare S-estimates (argument "sbi" to step_1 or step_2). By default 1.548 in order to achieve a breakdown point of about 0.5.
sl1_k	Scaling factor for the median of absolute residuals (argument "sl1" to step_2). By default 1.48 in order to get 1 under the standard normal distribution.
de_args	A function that takes a list with the arguments to be passed to JDEoptim in step 1 and returns an arbitrarily modified list of arguments. By default it returns its argument unmodified.
qn_args	A function that takes a list with the arguments to be passed to optim in step 3 and returns an arbitrarily modified list of arguments. By default it returns its argument unmodified.
qn_gr	A flag that indicates if a gradient function is to be built and passed as the argument to the parameter gr of the optim routine in step 3. Besides qn_gr = TRUE, for this to be the case both model\$grad and loss\$psi must be provided. Otherwise, a finite-difference approximation will be used as per usual. By default FALSE.
ms_extend	When computing bisquare M-estimates of scale and, consequently, also when computing bisquare S-estimates, ms_extend will be passed to m_scale in order to extend the root finding interval (for further details, refer to the documentation of m_scale). By default 5.
init_extend	Passed to model\$init in order to extend the search region defined by the lower and bounds employed both for the differential evolution routine in step 1 and for the quasi-Newton L-BFGS-B routine in step 3. By default 3.

Value

A list containing the following elements:

- `t`: a vector with the final location estimate, produced by step 3.
- `t0`: a vector with the initial location estimate, produced by step 1.
- `s`: a vector with a scale estimate for each dose, produced by step 2. `names(s)` will then give the corresponding doses.
- `init`: a list with the results of the initialization phase (`model$init`). It usually includes a non-robust location estimate and its standard error estimates, besides lower and upper search bounds.
- `loss`: the value of the loss function for step 3 evaluated at `t`.

If `model$grad`, `loss$psi` and `loss$dpsi` are all provided, the resulting list will also include a further element:

- `se`: asymptotic standard error estimates for `t`.

Examples

```
set.seed(0)
t <- c(10, 10, 40, 20)
x <- rep(seq(1, 100, 10), each = 30)
n <- length(x)
cont <- runif(n) > 0.9
e <- rnorm(n, ifelse(cont, 5, 0), (x / 100) * ifelse(cont, 3, 1))
y <- fpl$fun(x, t) + e
est <- drob(x, y)
plot(x, y)
lines(x, fpl$fun(x, t), lwd = 2)
lines(x, fpl$fun(x, est$init$t), col = 2)
lines(x, fpl$fun(x, est$t), col = 3)
legend("topleft", legend = c("t", "t_ls", "t_m"), lty = rep(1, 3), col = 1:3)
```

fpl

The 4-parameter logistic model (4PL)

Description

`fpl` is a list containing three elements related to the 4-parameter logistic model, as required by the `model` parameter of the `drob` function:

- `fun`: the 4PL-function itself. It takes as arguments a vector of values `x` and a vector of 4 parameters `t`.
- `grad`: the gradient of the 4PL-function, also expressed as a function of `x` and `t`.
- `init`: a function that computes initial parameters and search bounds.

Usage

`fpl`

Format

An object of class `list` of length 3.

Examples

```
t <- c(10, 10, 40, 20)
x <- 1:100
eta <- fpl$fun(x, t)
y <- eta + rnorm(100)
est <- fpl$init(x, y)
plot(x, y)
lines(x, eta)
lines(x, fpl$fun(x, est$t), lty = "dashed")
```

m_scale	<i>Compute M-estimate of scale</i>
---------	------------------------------------

Description

m_scale computes an M-estimate of scale for a given rho-function using a one-dimensional root finding routine.

Usage

```
m_scale(r, rho, extend = 5)
```

Arguments

r	A vector of values (typically residuals) whose scale is to be computed.
rho	The rho-function that defines the M-estimate of scale.
extend	The interval for root finding will extend from 0 to extend times the median absolute value of r. Defaults to 5.

Value

An M-estimate of scale for r based on function rho.

Examples

```
r <- rnorm(1000, 0, 3)
rho <- bisquare(1.55)$rho
m_scale(r, rho)
```

Index

* **datasets**

fpl, [5](#)

bisquare, [1](#)

drob, [2](#)

fpl, [5](#)

m_scale, [6](#)