

Package ‘minpack.lm’

January 2, 2012

Version 1.1-5

Title R interface to the Levenberg-Marquardt nonlinear least-squares algorithm found in MINPACK

Author Timur V. Elzhov, Katharine M. Mullen, Ben Bolker

Maintainer Katharine M. Mullen <katharine.mullen@nist.gov>

Description Provides R interface to `lmdr` and `lmdif` from the MINPACK library, for solving nonlinear least-squares problems by a modification of the Levenberg-Marquardt algorithm.

License file LICENSE

Repository CRAN

Date/Publication 2010-09-29 08:00:49

R topics documented:

<code>nls.lm</code>	1
<code>nls.lm.control</code>	6

Index	9
--------------	----------

<code>nls.lm</code>	<i>Addresses NLS problems with the Levenberg-Marquardt algorithm</i>
---------------------	--

Description

The purpose of `nls.lm` is to minimize the sum square of the vector returned by the function `fn`, by a modification of the Levenberg-Marquardt algorithm. The user may also provide a function `jac` which calculates the Jacobian.

Usage

```
nls.lm(par, fn, jac = NULL, control = nls.lm.control(), ...)
```

Arguments

<code>par</code>	A list or numeric vector of starting estimates. If <code>par</code> is a list, then each element must be of length 1.
<code>fn</code>	A function that returns a vector of residuals, the sum square of which is to be minimized. The first argument of <code>fn</code> must be <code>par</code> .
<code>jac</code>	A function to return the Jacobian for the <code>fn</code> function.
<code>control</code>	An optional list of control settings. See nls.lm.control for the names of the settable control values and their effect.
<code>...</code>	Further arguments to be passed to <code>fn</code> and <code>jac</code> .

Details

Both functions `fn` and `jac` (if provided) must return numeric vectors. Length of the vector returned by `fn` must not be lower than the length of `par`. The vector returned by `jac` must have length equal to $length(fn(par, \dots)) \cdot length(par)$.

The `control` argument is a list; see [nls.lm.control](#) for details.

Successful completion.

The accuracy of `nls.lm` is controlled by the convergence parameters `ftol`, `ptol`, and `gtol`. These parameters are used in tests which make three types of comparisons between the approximation `par` and a solution `par0`. `nls.lm` terminates when any of the tests is satisfied. If any of the convergence parameters is less than the machine precision, then `nls.lm` only attempts to satisfy the test defined by the machine precision. Further progress is not usually possible.

The tests assume that `fn` as well as `jac` are reasonably well behaved. If this condition is not satisfied, then `nls.lm` may incorrectly indicate convergence. The validity of the answer can be checked, for example, by rerunning `nls.lm` with tighter tolerances.

First convergence test.

If $|z|$ denotes the Euclidean norm of a vector z , then this test attempts to guarantee that

$$|fvec| < (1 + ftol) |fvec_0|,$$

where $fvec_0$ denotes the result of `fn` function evaluated at `par0`. If this condition is satisfied with $ftol \simeq 10^{-k}$, then the final residual norm $|fvec|$ has k significant decimal digits and `info` is set to 1 (or to 3 if the second test is also satisfied). Unless high precision solutions are required, the recommended value for `ftol` is the square root of the machine precision.

Second convergence test.

If D is the diagonal matrix whose entries are defined by the array `diag`, then this test attempt to guarantee that

$$|D(par - par_0)| < ptol |D par_0|,$$

If this condition is satisfied with $ptol \simeq 10^{-k}$, then the larger components of $(D par)$ have k significant decimal digits and `info` is set to 2 (or to 3 if the first test is also satisfied). There is a danger that the smaller components of $(D par)$ may have large relative errors, but if `diag` is internally set, then the accuracy of the components of `par` is usually related to their sensitivity. Unless high precision solutions are required, the recommended value for `ptol` is the square root of the machine

precision.

Third convergence test.

This test is satisfied when the cosine of the angle between the result of fn evaluation *fvec* and any column of the Jacobian at *par* is at most *gtol* in absolute value. There is no clear relationship between this test and the accuracy of *nls.lm*, and furthermore, the test is equally well satisfied at other critical points, namely maximizers and saddle points. Therefore, termination caused by this test (*info* = 4) should be examined carefully. The recommended value for *gtol* is zero.

Unsuccessful completion.

Unsuccessful termination of *nls.lm* can be due to improper input parameters, arithmetic interrupts, an excessive number of function evaluations, or an excessive number of iterations.

Improper input parameters.

info is set to 0 if $length(par) = 0$, or $length(fvec) < length(par)$, or $ftol < 0$, or $ptol < 0$, or $gtol < 0$, or $maxfev \leq 0$, or $factor \leq 0$.

Arithmetic interrupts.

If these interrupts occur in the *fn* function during an early stage of the computation, they may be caused by an unacceptable choice of *par* by *nls.lm*. In this case, it may be possible to remedy the situation by rerunning *nls.lm* with a smaller value of *factor*.

Excessive number of function evaluations.

A reasonable value for *maxfev* is $100 \cdot (length(par) + 1)$. If the number of calls to *fn* reaches *maxfev*, then this indicates that the routine is converging very slowly as measured by the progress of *fvec* and *info* is set to 5. In this case, it may be helpful to force *diag* to be internally set.

Excessive number of function iterations.

The allowed number of iterations defaults to 50, can be increased if desired.

The list returned by *nls.lm* has methods for the generic functions [coef](#), [deviance](#), [df.residual](#), [print](#), [residuals](#), [summary](#), [confint](#), and [vcov](#).

Value

A list with components:

<i>par</i>	The best set of parameters found.
<i>hessian</i>	A symmetric matrix giving an estimate of the Hessian at the solution found.
<i>fvec</i>	The result of the last <i>fn</i> evaluation; that is, the residuals.
<i>info</i>	<i>info</i> is an integer code indicating the reason for termination. <ul style="list-style-type: none"> 0 Improper input parameters. 1 Both actual and predicted relative reductions in the sum of squares are at most <i>ftol</i>. 2 Relative error between two consecutive iterates is at most <i>ptol</i>. 3 Conditions for <i>info</i> = 1 and <i>info</i> = 2 both hold.

	<p>4 The cosine of the angle between <code>fvec</code> and any column of the Jacobian is at most <code>gtol</code> in absolute value.</p> <p>5 Number of calls to <code>fn</code> has reached <code>maxfev</code>.</p> <p>6 <code>ftol</code> is too small. No further reduction in the sum of squares is possible.</p> <p>7 <code>ptol</code> is too small. No further improvement in the approximate solution <code>par</code> is possible.</p> <p>8 <code>gtol</code> is too small. <code>fvec</code> is orthogonal to the columns of the Jacobian to machine precision.</p> <p>9 The number of iterations has reached <code>maxiter</code>.</p>
<code>message</code>	character string indicating reason for termination
<code>.</code>	
<code>diag</code>	The result list of <code>diag</code> . See Details .
<code>niter</code>	The number of iterations completed before termination.
<code>rsstrace</code>	The residual sum of squares at each iteration. Can be used to check the progress each iteration.
<code>deviance</code>	The sum of the squared residual vector.

Note

The public domain FORTRAN sources of MINPACK package by J.J. Moré, implementing the Levenberg-Marquardt algorithm were downloaded from <http://ftp.netlib.org/minpack>, and left unchanged. The contents of this manual page are largely extracted from the comments of MINPACK sources.

References

J.J. Moré, "The Levenberg-Marquardt algorithm: implementation and theory," in *Lecture Notes in Mathematics* **630**: Numerical Analysis, G.A. Watson (Ed.), Springer-Verlag: Berlin, 1978, pp. 105-116.

See Also

[optim](#), [nls](#), [nls.lm.control](#)

Examples

```
##### example 1

## values over which to simulate data
x <- seq(0,5,length=100)

## model based on a list of parameters
getPred <- function(parS, xx) parS$a * exp(xx * parS$b) + parS$c

## parameter values used to simulate data
pp <- list(a=9,b=-1, c=6)
```

```

## simulated data, with noise
simDNoisy <- getPred(pp,x) + rnorm(length(x),sd=.1)

## plot data
plot(x,simDNoisy, main="data")

## residual function
residFun <- function(p, observed, xx) observed - getPred(p,xx)

## starting values for parameters
parStart <- list(a=3,b=-.001, c=1)

## perform fit
nls.out <- nls.lm(par=parStart, fn = residFun, observed = simDNoisy,
xx = x, control = nls.lm.control(nprint=1))

## plot model evaluated at final parameter estimates
lines(x,getPred(as.list(coef(nls.out)), x), col=2, lwd=2)

## summary information on parameter estimates
summary(nls.out)

##### example 2

## function to simulate data
f <- function(TT, tau, N0, a, f0) {
  expr <- expression(N0*exp(-TT/tau)*(1 + a*cos(f0*TT)))
  eval(expr)
}

## helper function for an analytical gradient
j <- function(TT, tau, N0, a, f0) {
  expr <- expression(N0*exp(-TT/tau)*(1 + a*cos(f0*TT)))
  c(eval(D(expr, "tau")), eval(D(expr, "N0" )),
    eval(D(expr, "a" )), eval(D(expr, "f0" )))
}

## values over which to simulate data
TT <- seq(0, 8, length=501)

## parameter values underlying simulated data
p <- c(tau = 2.2, N0 = 1000, a = 0.25, f0 = 8)

## get data
Ndet <- do.call("f", c(list(TT = TT), as.list(p)))
## with noise
N <- Ndet + rnorm(length(Ndet), mean=Ndet, sd=.01*max(Ndet))

## plot the data to fit
par(mfrow=c(2,1), mar = c(3,5,2,1))
plot(TT, N, bg = "black", cex = 0.5, main="data")

```

```

## define a residual function
fcn    <- function(p, TT, N, fcall, jcall)
      (N - do.call("fcall", c(list(TT = TT), as.list(p))))

## define analytical expression for the gradient
fcn.jac <- function(p, TT, N, fcall, jcall)
      -do.call("jcall", c(list(TT = TT), as.list(p)))

## starting values
guess <- c(tau = 2.2, N0 = 1500, a = 0.25, f0 = 10)

## to use an analytical expression for the gradient found in fcn.jac
## uncomment jac = fcn.jac
out <- nls.lm(par = guess, fn = fcn, jac = fcn.jac,
             fcall = f, jcall = j,
             TT = TT, N = N, control = nls.lm.control(nprint=1))

## get the fitted values
N1 <- do.call("f", c(list(TT = TT), out$par))

## add a blue line representing the fitting values to the plot of data
lines(TT, N1, col="blue", lwd=2)

## add a plot of the log residual sum of squares as it is made to
## decrease each iteration; note that the RSS at the starting parameter
## values is also stored
plot(1:(out$niter+1), log(out$rsstrace), type="b",
     main="log residual sum of squares vs. iteration number",
     xlab="iteration", ylab="log residual sum of squares", pch=21,bg=2)

## get information regarding standard errors
summary(out)

```

nls.lm.control

Control various aspects of the Levenberg-Marquardt algorithm

Description

Allow the user to set some characteristics Levenberg-Marquardt nonlinear least squares algorithm implemented in `nls.lm`.

Usage

```

nls.lm.control(ftol = sqrt(.Machine$double.eps),
              ptol = sqrt(.Machine$double.eps), gtol = 0, diag = list(), epsfcn = 0,
              factor = 100, maxfev = integer(), maxiter = 50, nprint = 0)

```

Arguments

ftol	non-negative numeric. Termination occurs when both the actual and predicted relative reductions in the sum of squares are at most ftol. Therefore, ftol measures the relative error desired in the sum of squares.
ptol	non-negative numeric. Termination occurs when the relative error between two consecutive iterates is at most ptol. Therefore, ptol measures the relative error desired in the approximate solution.
gtol	non-negative numeric. Termination occurs when the cosine of the angle between result of fn evaluation <i>fvec</i> and any column of the Jacobian is at most gtol in absolute value. Therefore, gtol measures the orthogonality desired between the function vector and the columns of the Jacobian.
diag	a list or numeric vector containing positive entries that serve as multiplicative scale factors for the parameters. Length of diag should be equal to that of par. If not, user-provided diag is ignored and diag is internally set.
epsfcn	(used if jac is not provided) is a numeric used in determining a suitable step for the forward-difference approximation. This approximation assumes that the relative errors in the functions are of the order of epsfcn. If epsfcn is less than the machine precision, it is assumed that the relative errors in the functions are of the order of the machine precision.
factor	positive numeric, used in determining the initial step bound. This bound is set to the product of factor and the $ \text{diag} * \text{par} $ if nonzero, or else to factor itself. In most cases factor should lie in the interval (0.1,100). 100 is a generally recommended value.
maxfev	integer; termination occurs when the number of calls to fn has reached maxfev. Note that nls.lm sets the value of maxfev to $100 * (\text{length}(\text{par}) + 1)$ if maxfev = integer(), where par is the list or vector of parameters to be optimized.
maxiter	positive integer. Termination occurs when the number of iterations reaches maxiter.
nprint	is an integer; set nprint to be positive to enable printing of iterates

Value

A list with exactly nine components:

```
ftol
ptol
gtol
diag
epsfcn
factor
maxfev
nprint
```

with meanings as explained under 'Arguments'.

References

J.J. Moré, "The Levenberg-Marquardt algorithm: implementation and theory," in *Lecture Notes in Mathematics* **630**: Numerical Analysis, G.A. Watson (Ed.), Springer-Verlag: Berlin, 1978, pp. 105-116.

See Also

[nls.lm](#)

Examples

```
nls.lm.control(maxiter = 4)
```

Index

- *Topic **nonlinear**
 - nls.lm, 1
 - nls.lm.control, 6
- *Topic **optimize**
 - nls.lm, 1
 - nls.lm.control, 6
- *Topic **regression**
 - nls.lm, 1
 - nls.lm.control, 6

- coef, 3
- confint, 3

- deviance, 3
- df.residual, 3

- nls, 4
- nls.lm, 1, 8
- nls.lm.control, 2, 4, 6

- optim, 4

- print, 3

- residuals, 3

- summary, 3

- vcov, 3