Package 'Rvmmin'

February 19, 2015

Type Package

Title Variable Metric Nonlinear Function Minimization

Version 2013-11.12

Date 2013-11-12

Maintainer John C. Nash <nashjc@uottawa.ca>

Description Variable metric nonlinear function minimization with bounds constraints.

Depends optextras

LazyLoad Yes

License GPL (>= 2)

Author John C. Nash [aut, cre]

NeedsCompilation no

Repository CRAN

Date/Publication 2014-12-06 15:11:25

R topics documented:

`Rvmmin` *Variable metric nonlinear function minimization, driver.*

Description

A driver to call the unconstrained and bounds constrained versions of an R implementation of a variable metric method for minimization of nonlinear functions, possibly subject to bounds (box) constraints and masks (fixed parameters). The algorithm is based on Nash (1979) Algorithm 21 for main structure, which is itself drawn from Fletcher's (1970) variable metric code. This is also the basis of optim() method 'BFGS' which, however, does not deal with bounds or masks. In the present method, an approximation to the inverse Hessian (B) is used to generate a search direction t = - B &*& g, a simple backtracking line search is used until an acceptable point is found, and the matrix B is updated using a BFGS formula. If no acceptable point can be found, we reset B to the identity i.e., the search direction becomes the negative gradient. If the search along the negative gradient is unsuccessful, the method terminates.

This set of codes is entirely in R to allow users to explore and understand the method. It also allows bounds (or box) constraints and masks (equality constraints) to be imposed on parameters.

Usage

```
Rvmmin(par, fn, gr, lower, upper, bdmsk, control = list(), ...)
```

Arguments

`par`	A numeric vector of starting estimates.
`fn`	A function that returns the value of the objective at the supplied set of parameters `par` using auxiliary data in .... The first argument of `fn` must be `par`.
`gr`	A function that returns the gradient of the objective at the supplied set of parameters `par` using auxiliary data in .... The first argument of `fn` must be `par`. This function returns the gradient as a numeric vector. Note that a gradient function must generally be provided. However, to ensure compatibility with other optimizers, if `gr` is NULL, the forward gradient approximation from routine `grfwd` will be used. The use of numerical gradients for Rvmmin is discouraged. First, the termination test uses a size measure on the gradient, and numerical gradient approximations can sometimes give results that are too large. Second, if there are bounds constraints, the step(s) taken to calculate the approximation to the derivative are NOT checked to see if they are out of bounds, and the function may be undefined at the evaluation point. There is also the option of using the routines `grfwd`, `grback`, `grcentral` or `grnd` from package `optextras`. The last of these calls the `grad()` function from package numDeriv. These are called by putting the name of the (numerical) gradient function in quotation marks, e.g., gr="grfwd" to use the standard forward difference numerical approximation. Note that all but the `grnd` routine use a stepsize parameter that can be redefined in a special scratchpad storage variable `deps`. See package `optextras`. The default is `deps = 1e-07`. However, redefining this is discouraged unless you understand what you are doing.
`lower`	A vector of lower bounds on the parameters.

upper	A vector of upper bounds on the parameters.
bdmsk	An indicator vector, having 1 for each parameter that is "free" or unconstrained, and 0 for any parameter that is fixed or MASKED for the duration of the optimization.
control	An optional list of control settings.
...	Further arguments to be passed to fn.

Details

Functions fn must return a numeric value. The control argument is a list. Successful completion. The source code Rvmmin for R is still a work in progress, so users should watch the console output.

The control argument is a list.

maxit A limit on the number of iterations (default 500). This is the maximum number of gradient evaluations allowed.

maxfevals A limit on the number of function evaluations allowed (default 3000).

trace Set 0 (default) for no output, >0 for trace output (larger values imply more output).

dowarn = TRUE if we want warnings generated by optimx. Default is TRUE.

checkgrad = TRUE if we wish analytic gradient code checked against the approximations computed by numDeriv. Default is TRUE.

checkbounds = TRUE if we wish parameters and bounds to be checked for an admissible and feasible start. Default is TRUE.

keepinputpar = TRUE if we want bounds check to stop program when parameters are out of bounds. Else when FALSE, moves parameter values to nearest bound. Default is FALSE.

maximize To maximize user_function, supply a function that computes (-1)*user_function. An alternative is to call Rvmmin via the package optimx.

As of 2011-11-21 the following controls have been REMOVED

usenumDeriv There is now a choice of numerical gradient routines. See argument gr.

Value

A list with components:

par	The best set of parameters found.
value	The value of the objective at the best set of parameters found.
counts	A vector of two integers giving the number of function and gradient evaluations.
convergence	An integer indicating the situation on termination of the function. 0 indicates that the method believes it has succeeded. Other values: 1 indicates that the iteration limit maxit had been reached. 20 indicates that the initial set of parameters is inadmissible, that is, that the function cannot be computed or returns an infinite, NULL, or NA value. 21 indicates that an intermediate set of parameters is inadmissible.
message	A description of the situation on termination of the function.

bdmsk — Returned index describing the status of bounds and masks at the proposed solution. Parameters for which bdmsk are 1 are unconstrained or "free", those with bdmsk 0 are masked i.e., fixed. For historical reasons, we indicate a parameter is at a lower bound using -3 or upper bound using -1.

References

??? Fletcher, R (1970) ...

Nash, J C (1979, 1990) ...

To be added.

See Also

optim

Examples

```
######################
## All examples are in this .Rd file
##
## Rosenbrock Banana function
fr <- function(x) {
    x1 <- x[1]
    x2 <- x[2]
    100 * (x2 - x1 * x1)^2 + (1 - x1)^2
}
ansrosenbrock <- Rvmmin(fn=fr,gr="grfwd", par=c(1,2))
print(ansrosenbrock) # use print to allow copy to separate file that
cat("No gr specified as a test\n")
ansrosenbrock0 <- Rvmmin(fn=fr, par=c(1,2))
print(ansrosenbrock0) # use print to allow copy to separate file that
#    can be called using source()
######################
# Simple bounds and masks test
bt.f<-function(x){
 sum(x*x)
}

bt.g<-function(x){
  gg<-2.0*x
}

n<-10
xx<-rep(0,n)
lower<-rep(0,n)
upper<-lower # to get arrays set
bdmsk<-rep(1,n)
bdmsk[(trunc(n/2)+1)]<-0
for (i in 1:n) {
   lower[i]<-1.0*(i-1)*(n-1)/n
   upper[i]<-1.0*i*(n+1)/n
```

```
}
xx<-0.5*(lower+upper)
ansbt<-Rvmmin(xx, bt.f, bt.g, lower, upper, bdmsk, control=list(trace=1))

print(ansbt)

####################
genrose.f<- function(x, gs=NULL){ # objective function
## One generalization of the Rosenbrock banana valley function (n parameters)
n <- length(x)
        if(is.null(gs)) { gs=100.0 }
fval<-1.0 + sum (gs*(x[1:(n-1)]^2 - x[2:n])^2 + (x[2:n] - 1)^2)
        return(fval)
}
genrose.g <- function(x, gs=NULL){
# vectorized gradient for genrose.f
# Ravi Varadhan 2009-04-03
n <- length(x)
        if(is.null(gs)) { gs=100.0 }
gg <- as.vector(rep(0, n))
tn <- 2:n
tn1 <- tn - 1
z1 <- x[tn] - x[tn1]^2
z2 <- 1 - x[tn]
gg[tn] <- 2 * (gs * z1 - z2)
gg[tn1] <- gg[tn1] - 4 * gs * x[tn1] * z1
gg
}

# analytic gradient test
xx<-rep(pi,10)
lower<-NULL
upper<-NULL
bdmsk<-NULL
genrosea<-Rvmmin(xx,genrose.f, genrose.g, gs=10)
genrosenf<-Rvmmin(xx,genrose.f, gr="grfwd", gs=10) # use local numerical gradient
genrosenullgr<-Rvmmin(xx,genrose.f, gs=10) # no gradient specified
cat("genrosea uses analytic gradient\n")
print(genrosea)
cat("genrosenf uses grfwd standard numerical gradient\n")
print(genrosenf)
cat("genrosenullgr has no gradient specified\n")
print(genrosenullgr)
cat("If optextras is loaded, then other numerical gradients can be used.\n")

cat("timings B vs U\n")
lo<-rep(-100,10)
up<-rep(100,10)
bdmsk<-rep(1,10)
tb<-system.time(ab<-Rvmminb(xx,genrose.f, genrose.g, lower=lo, upper=up, bdmsk=bdmsk))[1]
tu<-system.time(au<-Rvmminu(xx,genrose.f, genrose.g))[1]
cat("times U=",tu,"   B=",tb,"\n")
cat("solution Rvmminu\n")
```

```
print(au)
cat("solution Rvmminb\n")
print(ab)
cat("diff fu-fb=",au$value-ab$value,"\n")
cat("max abs parameter diff = ", max(abs(au$par-ab$par)),"\n")



maxfn<-function(x) {
       n<-length(x)
ss<-seq(1,n)
f<-10-(crossprod(x-ss))^2
f<-as.numeric(f)
return(f)
}


negmaxfn<-function(x) {
f<-(-1)*maxfn(x)
return(f)
}

cat("test that maximize=TRUE works correctly\n")

n<-6
xx<-rep(1,n)
ansmax<-Rvmmin(xx,maxfn, gr="grfwd", control=list(maximize=TRUE,trace=1))
print(ansmax)

cat("using the negmax function should give same parameters\n")
ansnegmax<-Rvmmin(xx,negmaxfn, gr="grfwd", control=list(trace=1))
print(ansnegmax)


#####################
cat("test bounds and masks\n")
nn<-4
startx<-rep(pi,nn)
lo<-rep(2,nn)
up<-rep(10,nn)
grbds1<-Rvmmin(startx,genrose.f, genrose.g, lower=lo,upper=up)
print(grbds1)

cat("test lower bound only\n")
nn<-4
startx<-rep(pi,nn)
lo<-rep(2,nn)
grbds2<-Rvmmin(startx,genrose.f, genrose.g, lower=lo)
print(grbds2)

cat("test lower bound single value only\n")
nn<-4
startx<-rep(pi,nn)
```

```
lo<-2
up<-rep(10,nn)
grbds3<-Rvmmin(startx,genrose.f, genrose.g, lower=lo)
print(grbds3)

cat("test upper bound only\n")
nn<-4
startx<-rep(pi,nn)
lo<-rep(2,nn)
up<-rep(10,nn)
grbds4<-Rvmmin(startx,genrose.f, genrose.g, upper=up)
print(grbds4)

cat("test upper bound single value only\n")
nn<-4
startx<-rep(pi,nn)
grbds5<-Rvmmin(startx,genrose.f, genrose.g, upper=10)
print(grbds5)



cat("test masks only\n")
nn<-6
bd<-c(1,1,0,0,1,1)
startx<-rep(pi,nn)
grbds6<-Rvmmin(startx,genrose.f, genrose.g, bdmsk=bd)
print(grbds6)

cat("test upper bound on first two elements only\n")
nn<-4
startx<-rep(pi,nn)
upper<-c(10,8, Inf, Inf)
grbds7<-Rvmmin(startx,genrose.f, genrose.g, upper=upper)
print(grbds7)


cat("test lower bound on first two elements only\n")
nn<-4
startx<-rep(0,nn)
lower<-c(0,1.1, -Inf, -Inf)
grbds8<-Rvmmin(startx,genrose.f,genrose.g,lower=lower, control=list(maxit=2000))
print(grbds8)

cat("test n=1 problem using simple squares of parameter\n")

sqtst<-function(xx) {
   res<-sum((xx-2)*(xx-2))
}

nn<-1
startx<-rep(0,nn)
onepar<-Rvmmin(startx,sqtst, gr="grfwd", control=list(trace=1))
print(onepar)
```

```
cat("Suppress warnings\n")
oneparnw<-Rvmmin(startx,sqtst, gr="grfwd", control=list(dowarn=FALSE,trace=1))
print(oneparnw)
```

Rvmminb *Variable metric nonlinear function minimization with bounds constraints*

Description

A bounds-constarined R implementation of a variable metric method for minimization of nonlinear functions subject to bounds (box) constraints and masks (fixed parameters).

See manual Rvmmin.Rd for more details and examples.

Usage

```
Rvmminb(par, fn, gr, lower, upper, bdmsk, control = list(), ...)
```

Arguments

par	A numeric vector of starting estimates.
fn	A function that returns the value of the objective at the supplied set of parameters par using auxiliary data in .... The first argument of fn must be par.
gr	A function that returns the gradient of the objective at the supplied set of parameters par using auxiliary data in .... The first argument of fn must be par. This function returns the gradient as a numeric vector. Note that a gradient function MUST be provided. See the manual for Rvmmin, which is the usual way Rvmminb is called. The user must take responsibility for errors if Rvmminb is called directly.
lower	A vector of lower bounds on the parameters.
upper	A vector of upper bounds on the parameters.
bdmsk	An indicator vector, having 1 for each parameter that is "free" or unconstrained, and 0 for any parameter that is fixed or MASKED for the duration of the optimization.
control	An optional list of control settings.
...	Further arguments to be passed to fn.

Details

This routine is intended to be called from Rvmmin, which will, if necessary, supply a gradient approximation. However, some users will want to avoid the extra overhead, in which case it is important to provide an appropriate and high-accuracy gradient routine.

Note that bounds checking, if it is carried out, is done by Rvmmin.

Functions fn must return a numeric value. The control argument is a list. Successful completion. The source code Rvmmin for R is still a work in progress, so users should watch the console output.

The control argument is a list.

maxit A limit on the number of iterations (default 500). This is the maximum number of gradient evaluations allowed.

maxfevals A limit on the number of function evaluations allowed (default 3000).

trace Set 0 (default) for no output, >0 for trace output (larger values imply more output).

dowarn = TRUE if we want warnings generated by optimx. Default is TRUE.

maximize To maximize user_function, supply a function that computes (-1)*user_function. An alternative is to call Rvmmin via the package optimx.

acctol To adjust the acceptable point tolerance (default 0.0001) in the test f <= fmin + gradproj * steplength * acctol

As of 2011-11-21 the following controls have been REMOVED

usenumDeriv There is now a choice of numerical gradient routines. See argument gr.

Value

A list with components:

par	The best set of parameters found.
value	The value of the objective at the best set of parameters found.
counts	A vector of two integers giving the number of function and gradient evaluations.
convergence	An integer indicating the situation on termination of the function. 0 indicates that the method believes it has succeeded. Other values: 1 indicates that the iteration limit maxit had been reached. 20 indicates that the initial set of parameters is inadmissible, that is, that the function cannot be computed or returns an infinite, NULL, or NA value. 21 indicates that an intermediate set of parameters is inadmissible.
message	A description of the situation on termination of the function.
bdmsk	Returned index describing the status of bounds and masks at the proposed solution. Parameters for which bdmsk are 1 are unconstrained or "free", those with bdmsk 0 are masked i.e., fixed. For historical reasons, we indicate a parameter is at a lower bound using -3 or upper bound using -1.

See Also

optim

Examples

```
## See Rvmmin.Rd
```

Rvmminu *Variable metric nonlinear function minimization, unconstrained*

Description

An R implementation of a variable metric method for minimization of unconstrained nonlinear functions.

See the manual Rvmmin.Rd for details.

Usage

```
Rvmminu(par, fn, gr, control = list(), ...)
```

Arguments

par	A numeric vector of starting estimates.
fn	A function that returns the value of the objective at the supplied set of parameters par using auxiliary data in .... The first argument of fn must be par.
gr	A function that returns the gradient of the objective at the supplied set of parameters par using auxiliary data in .... The first argument of fn must be par. This function returns the gradient as a numeric vector. Note that a gradient function MUST be provided. See the manual for Rvmmin, which is the usual way Rvmminb is called. The user must take responsibility for errors if Rvmminb is called directly.
control	An optional list of control settings.
...	Further arguments to be passed to fn.

Details

This routine is intended to be called from Rvmmin, which will, if necessary, supply a gradient approximation. However, some users will want to avoid the extra overhead, in which case it is important to provide an appropriate and high-accuracy gradient routine.

Functions fn must return a numeric value. The control argument is a list. Successful completion. The source code Rvmmin for R is still a work in progress, so users should watch the console output.

The control argument is a list.

maxit A limit on the number of iterations (default 500). This is the maximum number of gradient evaluations allowed.

maxfevals A limit on the number of function evaluations allowed (default 3000).

trace Set 0 (default) for no output, >0 for trace output (larger values imply more output).

dowarn = TRUE if we want warnings generated by optimx. Default is TRUE.

maximize To maximize user_function, supply a function that computes (-1)*user_function. An alternative is to call Rvmmin via the package optimx.

acctol To adjust the acceptable point tolerance (default 0.0001) in the test f <= fmin + gradproj * steplength * acctol

As of 2011-11-21 the following controls have been REMOVED

usenumDeriv There is now a choice of numerical gradient routines. See argument gr.

Value

A list with components:

par	The best set of parameters found.
value	The value of the objective at the best set of parameters found.
counts	A vector of two integers giving the number of function and gradient evaluations.
convergence	An integer indicating the situation on termination of the function. 0 indicates that the method believes it has succeeded. Other values: 1 indicates that the iteration limit maxit had been reached. 20 indicates that the initial set of parameters is inadmissible, that is, that the function cannot be computed or returns an infinite, NULL, or NA value. 21 indicates that an intermediate set of parameters is inadmissible.
message	A description of the situation on termination of the function.

See Also

optim

Examples

```
####in Rvmmin.Rd ####
```

Index