Package 'ucminf'

July 2, 2014

Version 1.1-3

Date 2012-01-06

Title General-purpose unconstrained non-linear optimization

Author Hans Bruun Nielsen and Stig Bousgaard Mortensen

Maintainer Stig Bousgaard Mortensen <stigbm@gmail.com>

Description An algorithm for general-purpose unconstrained non-linear optimization.
The algorithm is of quasi-Newton type with BFGS updating of the inverse
Hessian and soft line search with a trust region type monitoring of the
input to the line search algorithm. The interface of 'ucminf' is
designed for easy interchange with 'optim'.

Suggests numDeriv

License GPL (>= 2)

Repository CRAN

Date/Publication 2012-01-07 15:20:48

NeedsCompilation yes

R topics documented:

| ucminf | *General-purpose unconstrained non-linear optimization* |

Description

An algorithm for general-purpose unconstrained non-linear optimization. The algorithm is of quasi-Newton type with BFGS updating of the inverse Hessian and soft line search with a trust region type monitoring of the input to the line search algorithm. The interface of 'ucminf' is designed for easy interchange with 'optim'.

Usage

```
ucminf(par, fn, gr = NULL, ..., control = list(), hessian=0)
```

Arguments

par	Initial estimate of minimum for fn.
fn	Objective function to be minimized.
gr	Gradient of objective function. If NULL a finite difference approximation is used.
...	Optional arguments passed to the objective and gradient functions.
control	A list of control parameters. See 'Details'.
hessian	Integer value:

> **0** No hessian approximation is returned.
>
> **1** Returns a numerical approximation of the Hessian using 'hessian' in the package 'numDeriv'.
>
> **2** Returns final approximation of the inverse Hessian based on the series of BFGS updates during optimization.
>
> **3** Same at 2, but will also return the Hessian (the inverse of 2).
>
> If a TRUE or FALSE value is given it will switch between option 1 or 0.

Details

The algorithm is documented in (Nielsen, 2000) (see References below) together with a comparison to the Fortran subroutine 'MINF' and the Matlab function 'fminunc'. The implementation of 'ucminf' in R uses the original Fortran version of the algorithm.

The interface in R is designed so that it is very easy to switch between using 'ucminf' and 'optim'. The arguments par, fn, gr, and hessian are all the same (with a few extra options for hessian in 'ucminf'). The difference is that there is no method argument in 'ucminf' and that some of the components in the control argument are different due to differences in the algorithms.

The algorithm can be given an initial estimate of the Hessian for the optimization and it is possible to get the final approximation of the Hessian based on the series of BFGS updates. This extra functionality may be useful for optimization in a series of related problems.

The functions fn and gr can return Inf or NaN if the functions cannot be evaluated at the supplied value, but the functions must be computable at the initial value. The functions are not allowed to

return NA. Any names given to `par` will be copied to the vectors passed to `fn` and `gr`. No other attributes of `par` are copied over.

The `control` argument is a list that can supply any of the following components:

trace If trace is positive then detailed tracing information is printed for each iteration.

grtol The algorithm stops when $||F'(x)||_\infty \leq$ grtol, that is when the largest absolute value of the gradient is less than grtol. Default value is `grtol = 1e-6`.

xtol The algorithm stops when $||x - x_p||_2 \leq$ xtol $\cdot$ (xtol $+ ||x||_2$), where x_p and x are the previous and current estimate of the minimizer. Thus the algorithm stops when the last relative step length is sufficiently small. Default value is `xtol = 1e-12`.

stepmax Initial maximal allowed step length (radius of trust-region). The value is updated during the optimization. Default value is `stepmax = 1`.

maxeval The maximum number of function evaluations. A function evaluation is counted as one evaluation of the objective function and of the gradient function. Default value is `maxeval = 500`.

grad Either 'forward' or 'central'. Controls the type of finite difference approximation to be used for the gradient if no gradient function is given in the input argument 'gr'. Default value is `grad = 'forward'`.

gradstep Vector of length 2. The step length in finite difference approximation for the gradient. Step length is $|x_i| \cdot$ gradstep[1]+gradstep[2]. Default value is `gradstep = c(1e-6, 1e-8)`.

invhessian.lt A vector with an initial approximation to the lower triangle of the inverse Hessian. If not given, the inverse Hessian is initialized as the identity matrix. If H0 is the initial hessian matrix then the lower triangle of the inverse of H0 can be found as `invhessian.lt = solve(H0)[lower.tri(H0,diag=`

Value

par	Computed minimizer.
value	Objective function value at computed minimizer.
convergence	Flag for reason of termination:

   1 Stopped by small gradient (grtol).
   2 Stopped by small step (xtol).
   3 Stopped by function evaluation limit (maxeval).
   4 Stopped by zero step from line search
   -2 Computation did not start: length(par) = 0.
   -4 Computation did not start: stepmax is too small.
   -5 Computation did not start: grtol or xtol <= 0.
   -6 Computation did not start: maxeval <= 0.
   -7 Computation did not start: given Hessian not pos. definite.

message	String with reason of termination.
hessian, invhessian	Estimate of (inv.) Hessian at computed minimizer. The type of estimate is given by the input argument 'hessian'.
invhessian.lt	The lower triangle of the final approximation to the inverse Hessian based on the series of BFGS updates during optimization.

info   Information about the search:

> **maxgradient** $||F'(x)||_\infty$, the largest element in the absolute value of the gradient at the computed minimizer.
>
> **laststep** Length of last step.
>
> **stepmax** Final maximal allowed step length.
>
> **neval** Number of calls to both objective and gradient function.

Author(s)

'UCMINF' algorithm design and Fortran code by Hans Bruun Nielsen.

Implementation in R by Stig B. Mortensen, <stigbm@gmail.com>.

Modifications by Douglas Bates <bates@stat.wisc.edu>, Nov. 2010, to support nested optimization and correct issues with printing on Windows.

References

Nielsen, H. B. (2000) 'UCMINF - An Algorithm For Unconstrained, Nonlinear Optimization', Report IMM-REP-2000-18, Department of Mathematical Modelling, Technical University of Denmark. http://www2.imm.dtu.dk/~hbn/publ/TR0019.ps or http://orbit.dtu.dk/recid/200975.

The original Fortran source code can be found at http://www2.imm.dtu.dk/~hbn/Software/ucminf.f. The code has been slightly modified in this package to be suitable for use with R.

The general structure of the implementation in R is based on the package 'FortranCallsR' by Diethelm Wuertz.

See Also

optim, nlminb, nlm.

Examples

```
## Rosenbrock Banana function
fR <- function(x) (1 - x[1])^2 + 100 * (x[2] - x[1]^2)^2
gR <- function(x) c(-400 * x[1] * (x[2] - x[1] * x[1]) - 2 * (1 - x[1]),
                     200 * (x[2] - x[1] * x[1]))

# Find minimum and show trace
ucminf(par = c(2,.5), fn = fR, gr = gR, control = list(trace = 1))
```

Index