Package ‘Rtnmin’

July 7, 2016

Type Package
Title Truncated Newton Function Minimization with Bounds Constraints
Version 2016-7.7
Date 2016-07-07
Maintainer John C Nash <nashjc@uottawa.ca>
Description Truncated Newton function minimization with bounds constraints
based on the 'Matlab'/'Octave' codes of Stephen Nash.
License GPL (>= 2)
NeedsCompilation no
Author John C Nash [aut, cre, cph],
Stephen G Nash [aut, cph]
Repository CRAN
Date/Publication 2016-07-07 23:56:42

R topics documented:

\begin{verbatim}
tn .................................................. 1
tnbc ............................................... 4
\end{verbatim}

Index 6

\begin{verbatim}
tn Truncated Newton minimization of an unconstrained function.
\end{verbatim}

Description

An R implementation of the Truncated Newton method of Stephen Nash for driver to call the
unconstrained function minimization. The algorithm is based on Nash (1979)

This set of codes is entirely in R to allow users to explore and understand the method.
Usage

    tn(x, fgfun, trace, ...)

Arguments

    x      A numeric vector of starting estimates.
    fgfun  A function that returns the value of the objective at the supplied set of parameters
           par using auxiliary data in ... . The gradient is returned as attribute "gradient".
           The first argument of fgfun must be par.
    trace  TRUE if progress output is to be presented. (Not yet verified.)
           Further arguments to be passed to fn.

Details

    Function fgfun must return a numeric value in list item f and a numeric vector in list item g.

Value

    A list with components:

    xstar  The best set of parameters found.
    f      The value of the objective at the best set of parameters found.
    g      The gradient of the objective at the best set of parameters found.
    ierror An integer indicating the situation on termination. 0 indicates that the method
            believes it has succeeded; 2 that more than maxfun (default 150*n, where there
            are n parameters); 3 if the line search appears to have failed (which may not be
            serious); and -1 if there appears to be an error in the input parameters.
    nfngr  A number giving a measure of how many conjugate gradient solutions were used
            during the minimization process.

References

    Analysis, vol. 21, no. 4, pages 770-788.
    For Matlab code, see http://www.netlib.org/opt/tn

See Also

    optim

Examples

    # All examples are in this .Rd file
    #
    # Rosenbrock Banana function
    fr <- function(x) {
        x1 <- x[1]
        ...
```r
x2 <- x[2]
100 * (x2 - x1 * x1)^2 + (1 - x1)^2
}

gr <- function(x) {
x1 <- x[1]
x2 <- x[2]
g1 <- -400 * (x2 - x1*x1) * x1 - 2*x(1-x1)
g2 <- 200*x(x2 - x1*x1)
gg <- c(g1, g2)
}

rosefg<--function(x){
f<-fr(x)
g<-gr(x)
attr(f, "gradient") <- g
f
}

x<-c(-1,1)
ansrosenbrock <- tn(x, rosefg)
print(ansrosenbrock) # use print to allow copy to separate file that
cat("Compare to optim\n")
ansoptrose <- optim(x, fr, gr)
print(ansoptrose)

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
}

rosefg<--function(x, gs=100.0) {
f<-genrose.f(x, gs)
g<-genrose.g(x, gs)
attr(f, "gradient") <- g
```

```r
f
)

n <- 100
x <- (1:100)/20
groseu <- tn(x, grosefg, gs=10)
print(groseu)

groseuo <- optim(x, fn=genrose.f, gr=genrose.g, method="BFGS",
control=list(maxit=1000), gs=10)
cat("compare optim BFGS\n")
print(groseuo)

lower<-1+(1:n)/100
upper<-5-(1:n)/100
xmid<-0.5*(lower+upper)

grosec <- tnbc(xmid, grosefg, lower, upper)
print(grosec)

cat("compare L-BFGS-B\n")
grosec1 <- optim(par=xmid, fn=genrose.f, gr=genrose.g,
   lower=lower, upper=upper, method="L-BFGS-B")
print(grosec1)
```

---

**tnbc**

*Truncated Newton function minimization with bounds constraints*

---

**Description**

A bounds-constrained R implementation of a truncated Newton method for minimization of non-linear functions subject to bounds (box) constraints.

**Usage**

```
tnbc(x, fgfun, lower, upper, trace=FALSE, ...)
```

**Arguments**

- **x**: A numeric vector of starting estimates.
- **fgfun**: A function that returns the value of the objective at the supplied set of parameters `par` using auxiliary data in `...`. The gradient is returned as attribute "gradient".
- **lower**: A vector of lower bounds on the parameters.
- **upper**: A vector of upper bounds on the parameters.
- **trace**: Set TRUE to cause intermediate output to allow progress to be followed.
- **...**: Further arguments to be passed to `fn`.  

---
Details

Function fgfun must return a numeric value in list item f and a numeric vector in list item g.

Value

A list with components:

- **xstar**: The best set of parameters found.
- **f**: The value of the objective at the best set of parameters found.
- **g**: The gradient of the objective at the best set of parameters found.
- **ierror**: An integer indicating the situation on termination. 0 indicates that the method believes it has succeeded; 2 that more than maxfun (default 150*n, where there are n parameters); 3 if the line search appears to have failed (which may not be serious); and -1 if there appears to be an error in the input parameters.
- **nfngr**: A number giving a measure of how many conjugate gradient solutions were used during the minimization process.

References


For Matlab code, see [http://www.netlib.org/opt/tn](http://www.netlib.org/opt/tn)

See Also

- optim

Examples

```r
## See tn.Rd
```
Index

*Topic nonlinear
   tn, 1
   tmbc, 4

*Topic optimize
   tn, 1
   tmbc, 4

optim, 2, 5

tn, 1
tnbc, 4