Package ‘GenSA’

December 12, 2023

Type Package
Title R Functions for Generalized Simulated Annealing
Version 1.1.12
Date 2023-12-05
Author Sylvain Gubian, Yang Xiang, Brian Suomela, Julia Hoeng, PMP SA.
Maintainer Sylvain Gubian <DL.RSupport@pmi.com>
Depends R (>= 2.12.0)
Description Performs search for global minimum of a very complex non-linear objective function with a very large number of optima.
License GPL-2
LazyLoad yes
NeedsCompilation yes
Repository CRAN
RoxygenNote 7.2.3
Date/Publication 2023-12-12 16:50:02 UTC

R topics documented:

GenSA ................................................................. 1

Index 5

------------------------------------------------------------------------
GenSA  Generalized Simulated Annealing Function
------------------------------------------------------------------------

Description

This function searches for global minimum of a very complex non-linear objective function with a very large number of optima.
Usage

GenSA(par = NULL, fn, lower, upper, control = list(), ...)

Arguments

- **par**: Vector. Initial values for the components to be optimized. Default is NULL, in which case, default values will be generated automatically.
- **fn**: A function to be minimized, with first argument the vector of parameters over which minimization is to take place. It should return a scalar result.
- **lower**: Vector with length of par. Lower bounds for components.
- **upper**: Vector with length of par. Upper bounds for components.
- **control**: The argument is a list that can be used to control the behavior of the algorithm.
  - **maxit**: Integer. Maximum number of iterations of the algorithm.
  - **threshold.stop**: Numeric. The program will stop when the expected objective function value threshold.stop is reached. Default value is NULL.
  - **nb.stop.improvement**: Integer. The program will stop when there is no any improvement in nb.stop.improvement steps.
  - **smooth**: Logical. TRUE when the objective function is smooth, or differentiable almost everywhere in the region of par, FALSE otherwise. Default value is TRUE.
  - **max.call**: Integer. Maximum number of call of the objective function. Default is set to 1e7.
  - **max.time**: Numeric. Maximum running time in seconds.
  - **temperature**: Numeric. Initial value for temperature.
  - **visiting.param**: Numeric. Parameter for visiting distribution.
  - **acceptance.param**: Numeric. Parameter for acceptance distribution.
  - **verbose**: Logical. TRUE means that messages from the algorithm are shown. Default is FALSE.
  - **simple.function**: Logical. FALSE means that the objective function has only a few local minima. Default is FALSE which means that the objective function is complicated with many local minima.
  - **trace.mat**: Logical. Default is TRUE which means that the trace matrix will be available in the returned value of GenSA call.
  - **seed**: Integer. Negative integer value that can be set to initialize the internal random generator.

Details

The default values of the control components are set for a complex optimization problem. For usual optimization problem with medium complexity, GenSA can find a reasonable solution quickly so the user is recommended to let GenSA stop earlier by setting threshold.stop. If threshold.stop is the expected function value, or by setting max.time. If the user just want to run GenSA for max.time seconds, or by setting max.call. If the user just want to run GenSA within max.call function calls. Please refer to the examples below. For very complex optimization problems, the user is recommended to increase maxit and temp.
Value

The returned value is a list with the following fields:

- **value**: Numeric. The value of fn corresponding to par.
- **par**: Vector. The best set of parameters found.
- **trace.mat**: A matrix which contains the history of the algorithm. (By columns: Step number, temperature, current objective function value, current minimal objective function value).
- **counts**: Integer. Total number of calls of the objective function.

Author(s)

Yang Xiang, Sylvain Gubian, Brian Suomela, Julia Hoeng, PMP SA. (Y. Xiang and S. Gubian are equal contributors)

References


Examples

```r
library(GenSA)
# Try Rastrigin function (The objective function value for global minimum
# is 0 with all components of par are 0.)
Rastrigin <- function(x) {
  sum(x^2 - 10 * cos(2 * pi * x)) + 10 * length(x)
}
# Perform the search on a 30 dimensions rastrigin function. Rastrigin
# function with dimension 30 is known as the most
# "Evolutionary Programming Made Faster." 
# IEEE Transactions on Evolutionary Computation, 3(2), 82-102.
# GenSA will stop after finding the targeted function value 0 with
# absolute tolerance 1e-13
set.seed(1234) # The user can use any seed.
dimension <- 30
```
global.min <- 0
tol <- 1e-13
lower <- rep(-5.12, dimension)
upper <- rep(5.12, dimension)
out <- GenSA(lower = lower, upper = upper, fn = Rastrigin,
            control=list(threshold.stop=global.min+tol,verbose=TRUE))
out[c("value","par","counts")]

# GenSA will stop after running for about 2 seconds
# Note: The time for solving this problem by GenSA may vary
# depending on the computer used.
set.seed(1234) # The user can use any seed.
dimension <- 30
global.min <- 0
tol <- 1e-13
lower <- rep(-5.12, dimension)
upper <- rep(5.12, dimension)
out <- GenSA(lower = lower, upper = upper, fn = Rastrigin,
            control=list(max.time=2))
out[c("value","par","counts")]

Index

GenSA, 1