

Package ‘optimParallel’

February 25, 2019

Type Package

Title Parallel Version of the L-BFGS-B Optimization Method

Version 0.8

Date 2019-02-25

Author Florian Gerber

Maintainer Florian Gerber <florian.gerber@math.uzh.ch>

Description Provides a parallel version of the L-BFGS-B method of `optim()`. The main function of the package is `optimParallel()`, which has the same usage and output as `optim()`. Using `optimParallel()` can significantly reduce the optimization time.

License GPL (>= 2)

URL <https://git.math.uzh.ch/florian.gerber/optimParallel>

BugReports <https://git.math.uzh.ch/florian.gerber/optimParallel/issues>

Depends R (>= 3.5), stats, parallel

Suggests R.rsp, roxygen2, spam, microbenchmark, testthat, ggplot2, numDeriv, lbfgsb3, lbfgsb3c

VignetteBuilder R.rsp

RoxygenNote 6.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2019-02-25 21:40:04 UTC

R topics documented:

`optimParallel` 2

Index 5

optimParallel *parallel version of the L-BFGS-B method of [optim](#)*

Description

The function provides a parallel version of the L-BFGS-B method of [optim](#). If the evaluation time of the objective function `fn` is more than 0.1 seconds, `optimParallel` can significantly reduce the optimization time. For a p -parameter optimization the speed increase is about factor $1 + 2p$ when no analytic gradient is specified and $1 + 2p$ processor cores are available.

Usage

```
optimParallel(par, fn, gr = NULL, ..., lower = -Inf, upper = Inf,
             control = list(), hessian = FALSE, parallel = list())
```

Arguments

<code>par</code>	see the documentation of optim .
<code>fn</code>	see the documentation of optim .
<code>gr</code>	see the documentation of optim .
<code>...</code>	see the documentation of optim . All R object required by <code>fn</code> and <code>gr</code> have to be added here.
<code>lower</code>	see the documentation of optim .
<code>upper</code>	see the documentation of optim .
<code>control</code>	see the documentation of optim .
<code>hessian</code>	see the documentation of optim .
<code>parallel</code>	is a list of additional control parameters and can supply any of the following components: <ul style="list-style-type: none"> <code>c1</code> an object of class "cluster" specifying the cluster to be used for parallel execution. See makeCluster for more information. If the argument is not specified or NULL, the default cluster is used. See setDefaultCluster for information on how to set up a default cluster. <code>forward</code> logical vector of length 1. If FALSE (default when loading the package), a numeric central difference approximation of the gradient defined as $(fn(x + \epsilon) - fn(x - \epsilon))/(2\epsilon)$ is used, which corresponds to the gradient approximation used in optim. If TRUE, a numeric forward difference approximation of the gradient essentially defined as $(fn(x + \epsilon) - fn(x))/\epsilon$ is used. This reduces the number of function calls from $1 + 2p$ to $1 + p$ and can be useful if the number of available cores is smaller than $1 + 2p$ or if the memory limit is reached. Note that the numeric central difference approximation is more accurate than the numeric forward difference approximation. <code>loginfo</code> logical vector of length 1 with default value FALSE when loading the package. If TRUE, additional log information containing the evaluated parameters as well as return values of <code>fn</code> and <code>gr</code> is returned.

Details

optimParallel is a wrapper to `optim` and relies on the lexical scoping mechanism of R and the R package `parallel` to evaluate `fn` and its (approximate) gradient in parallel.

Some default values of the argument `parallel` can be set via `options("optimParallel.forward", "optimParallel.loginfo")`.

Value

Same as the return value of `optim`. See the documentation thereof for more information. If `parallel=list(loginfo=TRUE)`, additional log information containing the evaluated parameters as well as the return values of `fn` and `gr` is returned.

Issues and bug report

A list of known issues of `optimParallel` can be found at <https://git.math.uzh.ch/florian.gerber/optimParallel/issues>. Please report issues not listed there to <florian.gerber@math.uzh.ch>. Do not forget to include an R script reproducing the issue and the output of `sessionInfo()`.

Author(s)

Florian Gerber, <florian.gerber@math.uzh.ch>, <https://user.math.uzh.ch/gerber>.

References

F. Gerber, R. Furrer (2018) `optimParallel`: an R Package Providing Parallel Versions of the Gradient-Based Optimization Methods of `optim()`. ArXiv e-prints. URL <http://arxiv.org/abs/1804.11058>. Also available as vignette of this package `vignette("optimParallel")`.

See Also

[optim](#), [makeCluster](#), [setDefaultCluster](#), [stopCluster](#), [detectCores](#).

Examples

```
negll <- function(par, x, sleep=0, verbose=TRUE){
  if(verbose)
    cat(par, "\n")
  Sys.sleep(sleep)
  -sum(dnorm(x=x, mean=par[1], sd=par[2], log=TRUE))
}
set.seed(13); x <- rnorm(1000, 5, 2)

cl <- makeCluster(2) # set the number of processor cores
setDefaultCluster(cl=cl) # set 'cl' as default cluster

optimParallel(par=c(1,1), fn=negll, x=x, lower=c(-Inf, .0001))

optimParallel(par=c(1,1), fn=negll, x=x, sleep=0, verbose=TRUE,
              lower=c(-Inf, .0001), parallel=list(loginfo=TRUE))
```

```
setDefaultCluster(cl=NULL); stopCluster(cl)

## default values of the argument 'parallel':
options("optimParallel.forward", "optimParallel.loginfo")

## Not run:
## - use all available processor cores
## - return cat() output to R prompt
## (may have issues on Windows)
if(tolower(.Platform$OS.type) != "windows"){
  cl <- makeCluster(spec=detectCores(), type="FORK", outfile="")
} else
  cl <- makeCluster(spec=detectCores(), outfile="")
setDefaultCluster(cl=cl)

## return log information
options(optimParallel.loginfo=TRUE)

## stop if change of f(x) is smaller than 0.01
control <- list(factr=.01/.Machine$double.eps)

optimParallel(par=c(1,1), fn=negl1, x=x, sleep=.5, verbose=TRUE,
              verbose=TRUE, lower=c(-Inf, .0001), control=control)
## each step invokes 5 parallel calls to negl1()

optimParallel(par=c(1,1), fn=negl1, x=x, sleep=.5, verbose=TRUE,
              lower=c(-Inf, .0001), control=control,
              parallel=list(forward=TRUE))
## each step invokes 3 parallel calls to negl1()

setDefaultCluster(cl=NULL); stopCluster(cl)
## End(Not run)
```

Index

*Topic **package**

optimParallel, 2

detectCores, 3

makeCluster, 2, 3

optim, 2, 3

optimParallel, 2

optimparallel (optimParallel), 2

OptimParallel-Package (optimParallel), 2

OptimParallel-package (optimParallel), 2

optimParallel-Package (optimParallel), 2

optimParallel-package (optimParallel), 2

optimparallel-Package (optimParallel), 2

optimparallel-package (optimParallel), 2

setDefaultCluster, 2, 3

stopCluster, 3