Package 'mixopt'

September 15, 2024

Type Package

Title Mixed Variable Optimization

Version 0.1.3

Maintainer Collin Erickson <collinberickson@gmail.com>

Description Mixed variable optimization for non-linear functions.
Can optimize function whose inputs are a combination of continuous,
ordered, and unordered variables.

Depends dplyr, ggplot2, splitfngr

Suggests ContourFunctions, gridExtra, lhs, testthat (>= 3.0.0)

License LGPL (>= 3)

Encoding UTF-8

URL https://github.com/CollinErickson/mixopt

BugReports https://github.com/CollinErickson/mixopt/issues

RoxygenNote 7.3.1

Config/testthat/edition 3

NeedsCompilation no

Author Collin Erickson [aut, cre]

Repository CRAN

Date/Publication 2024-09-15 00:20:02 UTC

Contents

`as.mixopt_list` *Coerce to a mixopt_list*

Description

Coerce to a mixopt_list

Usage

```
as.mixopt_list(x, simplifyifpossible = FALSE)
```

Arguments

`x`	Object
`simplifyifpossible`	If possible, should the class be simplified to numeric or character?

Value

Object of class mixopt_list

`c_mixopt_list` *Combines mixopt_list objects*

Description

Combines mixopt_list objects

Usage

```
c_mixopt_list(x, ...)
```

Arguments

`x`	Object
`...`	Additional objects

Value

A combined mixopt_list

Examples

```
c_mixopt_list(NULL, as.mixopt_list(1:5), NULL, as.mixopt_list(letters[1:5]))
c_mixopt_list(as.mixopt_list(1:3), NULL)
```

`full_index_line_search` *Optimize over array using line search*

Description

Optimize over array using line search

Usage

```
full_index_line_search(
  f,
  xarray,
  startind,
  plot = "none",
  ystart = NULL,
  verbose = 0
)
```

Arguments

`f`	Function
`xarray`	Array of values
`startind`	Starting index
`plot`	Should plots be made?
`ystart`	Value of f when evaluated on element of xarray at index startind, aka f(xarray[startind])
`verbose`	Level of info to print

Value

List

Examples

```
full_index_line_search(function(x) {(x-50)^2}, 3:12, 5)
full_index_line_search(function(x) {(x-50)^2}, 3, 1)
full_index_line_search(function(x) {(x-50)^2}, 3:4, 1)
full_index_line_search(function(x) {(x-50)^2}, 3:5, 1)
full_index_line_search(function(x) {(x+50)^2}, 3, 1)
full_index_line_search(function(x) {(x+50)^2}, 3:4, 1)
full_index_line_search(function(x) {(x+50)^2}, 3:5, 1)
full_index_line_search(function(x) {(x-50)^2}, 12:3, 8)
```

```
full_index_line_search(function(x) {(x-50)^2}, 0:1000, 8)
full_index_line_search(function(x) {(x-50)^2}, 0:1000, 999)
full_index_line_search(function(x) {sin(x/30)}, 0:1000, 999)
```

`index_line_search` *Line search over indexed array in one direction*

Description

Line search over indexed array in one direction

Usage

```
index_line_search(f, xarray, y1 = NULL, plot = "none", verbose = 0)
```

Arguments

`f`	f
`xarray`	xarray
`y1`	y1
`plot`	plot
`verbose`	Level to print

Value

List

Examples

```
index_line_search(function(x) {(x-100)^2}, 1:290)
index_line_search(function(x) {(-x-100)^2}, -(1:290)^.92, plot="ind")
index_line_search(function(x) {(-x-100)^2}, -(1:290)^.92, plot="x")
xx <- sort(runif(1e2, -250, -30))
index_line_search(function(x) {(-x-100)^2}, xx, plot="ind")
index_line_search(function(x) {(-x-100)^2}, xx, plot="x")
```

is.mixopt_list *Checks if object is mixopt_list*

Description

Checks if object is mixopt_list

Usage

```
is.mixopt_list(x)
```

Arguments

x	Object

Value

TRUE if x has class "mixopt_list"

mixopt *Mixed variable optimization using coordinate descent*

Description

Mixed variable optimization using coordinate descent

Usage

```
mixopt(
  par,
  fn,
  gr = NULL,
  global = "multistart",
  local = "coorddesc",
  ...,
  method,
  verbose = 0,
  track
)

mixopt_blockcd(
  par,
  fn,
  gr = NULL,
  ...,
  control = list(),
```

```
  maxblocksize = NULL,
  method,
  fngr = NULL,
  maxiter = 100,
  maxeval = NULL,
  maxtime = NULL,
  verbose = 0,
  track = FALSE
)

mixopt_coorddesc(
  par,
  fn,
  gr = NULL,
  ...,
  method,
  maxiter = 100,
  maxeval = NULL,
  maxtime = NULL,
  verbose = 0,
  track = FALSE
)

mixopt_multistart(
  par,
  fn,
  gr = NULL,
  ...,
  method,
  fngr = NULL,
  n0 = 20,
  n1 = 2,
  maxiter = 100,
  maxeval = NULL,
  verbose = 0,
  groupeval = FALSE,
  track = FALSE
)
```

Arguments

par	List of parameters
fn	Function to evaluate
gr	Gradient of fn
global	Global optimization algorithm to use. 'FALSE' if you only want local optimization.
local	Local optimization algorithm to use.

`...`	Additional args
`method`	Optimization method
`verbose`	How much to print. 0 is none, 1 is standard, 2 is some, 3 is a lot, etc.
`track`	Should it track the parameters evaluated and value?
`control`	Parameters for optimizing.
`maxblocksize`	The maximum number of continuous dimensions that should be placed into a single block.
`fngr`	Function that returns the function and gradient value for the given input as a list with names "fn" and "gr".
`maxiter`	Maximum number of outer iterations. For coordinate descent, one iteration is a loop over each parameter.
`maxeval`	Maximum number of function evaluations. It may go over this number while in an inner optimization loop, but will exit after that.
`maxtime`	Maximum time to run in seconds. Not an exact limit, only checks occasionally.
`n0`	For multistart, number of random initial points to evaluate.
`n1`	For multistart, number of best starts to optimize with. You should have ‘n0‘ less than ‘n1‘, potentially by a large factor. gradient descent.
`groupeval`	Can multiple inputs be evaluated at once? This can speed up greatly for certain circumstances. Use "matrix" to have it give a set of points as rows of a matrix to all be evaluated at once.

Value

List

References

https://en.wikipedia.org/wiki/Coordinate_descent

https://en.wikipedia.org/wiki/Coordinate_descent

https://www.uv.es/rmarti/paper/docs/multi2.pdf

Examples

```
# Simple 1D example
mixopt_blockcd(par=list(mopar_cts(2,8)), fn=function(x) {(4.5-x[1])^2})
# With gradient (isn't faster)
mixopt_blockcd(par=list(mopar_cts(2,8)), fn=function(x) {(4.5-x[1])^2},
               gr=function(x) {-2*(4.5-x[1])})

# 1D discrete ordered
mixopt_blockcd(par=list(mopar_ordered(100:10000)),
                 fn=function(x) {(x[1] - 500.3)^2})

# 2D: one continuous, one factor
mixopt_blockcd(par=list(mopar_cts(2,8), mopar_unordered(letters[1:6])),
                 fn=function(x) {ifelse(x[2] == 'b', -1, 0) +
```

```
                                (4.5-x[1])^2})
# Simple 1D example
mixopt_coorddesc(par=list(mopar_cts(2,8)), fn=function(x) {(4.5-x[1])^2})

# 1D discrete ordered
mixopt_coorddesc(par=list(mopar_ordered(100:10000)),
                 fn=function(x) {(x[1] - 500.3)^2})

# 2D: one continuous, one factor
mixopt_coorddesc(par=list(mopar_cts(2,8), mopar_unordered(letters[1:6])),
                 fn=function(x) {ifelse(x[2] == 'b', -1, 0) +
                                (4.5-x[1])^2})
# 2D
library(ggplot2)
library(dplyr)
f6 <- function(x) {-(-x[1]*.5*sin(.5*x[1])*1 - 1e-2*x[2]^2 +
                       .2*x[1] - .3*x[2])}

if (requireNamespace("ContourFunctions", quietly = TRUE)) {
  ContourFunctions::cf_func(f6, xlim=c(0,100), ylim=c(-100,100))
}
m6 <- mixopt_coorddesc(par=list(mopar_cts(0,100), mopar_cts(-100,100)),
                       fn=f6, track = TRUE)
plot_track(m6)
ms6 <- mixopt_multistart(par=list(mopar_cts(0,100), mopar_cts(-100,100)),
                         fn=f6, track = TRUE)
plot_track(ms6)
if (requireNamespace("ContourFunctions", quietly = TRUE)) {
  ContourFunctions::cf_func(f6, xlim=c(0,100), ylim=c(-100,100),
                            gg = TRUE) +
    geom_point(data=as.data.frame(matrix(unlist(ms6$track$par),
                                         ncol=2, byrow=TRUE)) %>%
                 bind_cols(newbest=ms6$track$newbest),
               aes(V1, V2, color=newbest), alpha=.5)
}
```

`mopar_cts` *Continuous variable*

Description

Continuous variable

Usage

```
mopar_cts(lower, upper, start = NULL)
```

Arguments

`lower`	Lower

upper	Upper
start	Start. Defaults to midpoint if not given.

Value

mixopt_par list

Examples

```
mopar_cts(2,8)
mopar_cts(2,8,7)
```

---

mopar_ordered *Ordered variable parameter*

Description

Ordered variable parameter

Usage

```
mopar_ordered(values, start = NULL)
```

Arguments

values	Values the parameter can take, in order
start	Start parameter for optimization

Value

mixopt_par list

Examples

```
mopar_ordered(c(1,3,5))
mopar_ordered(c('a','c'))
mopar_ordered(1:4)
mopar_ordered(4:1)
mopar_ordered(list('a', 2, 'c', sin))
```

mopar_unordered *Unordered factor parameter*

Description

Unordered factor parameter

Usage

```
mopar_unordered(values, start = NULL)
```

Arguments

values	Values the variable can take
start	Start value. Chosen randomly if not given.

Value

mixopt_par list

Examples

```
mopar_unordered(c(1,3,9))
mopar_unordered(letters)
```

plot_track *Plot the tracked parameters from an optimization*

Description

Plot the tracked parameters from an optimization

Usage

```
plot_track(out)
```

Arguments

out	Output from mixopt

Value

Plot

Examples

```
f8 <- function(x) {-(x[[1]]+x[[2]]) + .1*(x[[1]] - x[[2]])^2}
if (requireNamespace("ContourFunctions", quietly = TRUE)) {
  ContourFunctions::cf_func(f8, xlim=c(0,100), ylim=c(0,100))
}
m8 <- mixopt_coorddesc(par=list(mopar_ordered(0:100), mopar_ordered(0:100)),
                       fn=f8, track = TRUE)
plot_track(m8)

library(ggplot2)
library(dplyr)
if (requireNamespace("ContourFunctions", quietly = TRUE)) {
  ContourFunctions::cf_func(f8, xlim=c(0,100), ylim=c(0,100),
                            gg = TRUE) +
    geom_point(data=as.data.frame(matrix(unlist(m8$track$par),
                                  ncol=2, byrow=TRUE)) %>%
                      bind_cols(newbest=m8$track$newbest),
               aes(V1, V2, color=newbest))
}
```

`verify_par` *Verify parameters*

Description

Verify parameters

Usage

```
verify_par(par)
```

Arguments

`par`	List of parameters

Value

Nothing, raises error if not valid

Examples

```
verify_par(
  list(
    mopar_cts(2, 8, 6)
  )
)
```

`[.mixopt_list` *Index mixopt_list*

Description

Avoid standard list indexing which returns list for single index.

Usage

```
## S3 method for class 'mixopt_list'
x[i, value]
```

Arguments

x	x
i	i
value	value

Value

value at index

Examples

```
a <- list(1,4,'c', 'g')
class(a) <- "mixopt_list"
a
a[3]
a[2:3]
a[-(2:3)]
as.data.frame(a)

b <- as.mixopt_list(c(1,2,3,4,5))
sum(b)
b^2
b+b
b-b
b*b
b/b
c(b)
c(b, b)
c(b, 1)
c(1, b)
c(a, b, a)
c_mixopt_list(0, 1, 2, 3, 4, a, 5, 6, 7, 8, b, 9)
c_mixopt_list(NULL, 3, NULL, a, NULL, 66666, NULL, b)
```

Index