Package 'optimizeR'

March 25, 2023

Title Unified Framework for Numerical Optimizers

Version 0.3.2

Description Provides a unified framework for numerical optimizers in R, particularly for their inputs and outputs.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.2.3

Suggests knitr, rmarkdown, testthat (>= 3.0.0), pracma, R.utils, covr

Config/testthat/edition 3

URL https://github.com/loelschlaeger/optimizeR

BugReports https://github.com/loelschlaeger/optimizeR/issues

Depends R (>= 4.0.0)

Imports cli, glue

NeedsCompilation no

Author Lennart Oelschläger [aut, cre] (<https://orcid.org/0000-0001-5421-9313>), Marius Ötting [ctb] (<https://orcid.org/0000-0002-9373-0365>)

Maintainer Lennart Oelschläger <oelschlaeger.lennart@gmail.com>

Repository CRAN

Date/Publication 2023-03-25 12:50:02 UTC

R topics documented:

apply_optimizer *Apply* optimizer *object*

Description

This function performs numerical optimization using an optimizer object.

Usage

```
apply_optimizer(optimizer = optimizer_nlm(), objective, initial, ...)
```

Arguments

optimizer	An object of class optimizer.
objective	The function to be optimized, returning a single numeric. Its first argument must be a numeric vector of the length of initial, followed by any other arguments specified by the ... argument.
initial	A numeric vector with starting parameter values for the optimization.
...	Additional arguments to be passed to objective.

Value

A named list, containing at least three elements:

value A numeric, the value of the estimated optimum of objective.

parameter A numeric vector, the parameter vector where the optimum of objective is obtained. secondsA numeric, the total optimization time in seconds. initialA numeric, the initial parameter values.

Additional output elements of the optimizer (if not excluded by the output_ignore element via define_optimizer) are appended.

See Also

define_optimizer() for specifying an optimizer object.

Examples

```
apply_optimizer(optimizer_nlm(), function(x) x^4 + 3*x - 5, 2)
```

`define_optimizer` *Specify numerical optimizer*

Description

This function specifies the framework for a numerical optimizer.

Two wrappers for common optimizer are available:

1. `optimizer_nlm()` for the `nlm` optimizer
2. `optimizer_optim()` for the `optim` optimizer

Usage

```
define_optimizer(
  optimizer,
  objective,
  initial,
  value,
  parameter,
  ...,
  output_ignore = character(0),
  validate = FALSE,
  validation_settings = list(objective_test = function(x) {
     stopifnot(is.numeric(x),
    length(x) == 2)
     -20 * exp(-0.2 * sqrt(0.5 * (x[1]^2 + x[2]^2))) - exp(0.5 *
    (cos(2 * pi * x[1]) + cos(2 * pi * x[2]))) + exp(1) + 20
 }, objective_add = list(),
    initial = round(stats::rnorm(2), 2), check_seconds = 10)
)

optimizer_nlm(
  ...,
  output_ignore = character(0),
  validate = FALSE,
  validation_settings = list()
)

optimizer_optim(
  ...,
  output_ignore = character(0),
  validate = FALSE,
  validation_settings = list()
)
```

Arguments

`optimizer` A `function`, a numerical optimizer. Four conditions must be met:

1. It must have an input named `"objective"` for a `function`, the objective function which is optimized over its first argument.
2. It must have an input named `"initial"` for a `numerical` vector, the initial parameter vector.
3. It must have a `...` argument for additional parameters to the objective function.
4. The output must be a named `list`, including the optimal function value and the optimal parameter vector.

`objective` A `character`, the name of the function input of `optimizer`.

`initial` A `character`, the name of the starting parameter values input of `optimizer`.

`value` A `character`, the name of the optimal function value in the output list of `optimizer`.

`parameter` A `character`, the name of the optimal parameter vector in the output list of `optimizer`.

`...` Additional arguments to be passed to the optimizer. Without specifications, the default values of the optimizer are used.

`output_ignore` A `character` vector of element names in the output of `optimizer` that are not saved. The elements `value` and `parameter` are added automatically to `output_ignore`, because they are saved separately, see the output documentation of `apply_optimizer`.

`validate` A `logical`, set to `TRUE` (`FALSE`) to (not) validate the `optimizer` object. By default, `validate = FALSE`.

`validation_settings`

Ignored if `valdiate = FALSE`. Otherwise, a `list` of validation settings:

objective_test A `function`, the test function to be optimized. By default, it is the Ackley function.

objective_add A `list` of additional arguments to `objective_test` (if any). By default, `objective_add = list()`, because the default function for `objective_test` does not have additional arguments.

initial A `numeric` vector, the initial values for the optimization of `objective_test`. By default, `initial = round(stats::rnorm(2), 2)`.

check_seconds An `integer`, the maximum number of seconds before the test is aborted. The test call is considered to be successful if no error occurred within `check_seconds` seconds. By default, `check_seconds = 10`.

Value

An `optimizer` object.

Format

An `optimizer` object is a `list` of five elements:

optimizer A `function`, the optimization function `optimizer`.

optimizer_name A `character`, the name of `optimizer`.

optimizer_add A named `list`, where each element is an additional function argument for `optimizer`.

argument_names A named `list` of four `character`:

- **objective** the name of the function input of `optimizer`
- **initial** the name of the starting parameter values input of `optimizer`
- **value** the name of the optimal function value in the output list of `optimizer`
- **parameter** the name of the optimal parameter vector in the output list of `optimizer`.

output_ignore A `character` vector of element names in the output `list` of `optimizer` that are ignored. The elements `value` and `parameter` are added automatically to `output_ignore`, because they are saved separately, see the output documentation of `apply_optimizer`.

See Also

Use `apply_optimizer()` to apply an `optimizer` object for numerical optimization.

Examples

```
define_optimizer(
  optimizer = pracma::nelder_mead,
  objective = "fn",
  initial = "x0",
  value = "fmin",
  parameter = "xmin",
  output_ignore = c("fcount", "restarts", "errmess"), # ignore some outputs
  tol = 1e-6, # an additional argument for pracma::nelder_mead()
  validate = TRUE # validate the framework
)
```

Index