Package: optimCheck (via r-universe)

September 5, 2024

Type Package

Title Graphical and Numerical Checks for Mode-Finding Routines

Version 1.0.1

Date 2024-09-04

Description Tools for checking that the output of an optimization algorithm is indeed at a local mode of the objective function. This is accomplished graphically by calculating all one-dimensional ``projection plots'' of the objective function, i.e., varying each input variable one at a time with all other elements of the potential solution being fixed. The numerical values in these plots can be readily extracted for the purpose of automated and systematic unit-testing of optimization routines.

URL https://github.com/mlysy/optimCheck, https://mlysy.github.io/optimCheck/

BugReports https://github.com/mlysy/optimCheck/issues

License GPL-3

Imports stats, graphics

RoxygenNote 7.3.1

Encoding UTF-8

Suggests testthat, quantreg, mclust, knitr, rmarkdown

VignetteBuilder knitr

Roxygen list(markdown = TRUE)

Repository https://mlysy.r-universe.dev

RemoteUrl https://github.com/mlysy/optimcheck

RemoteRef HEAD

RemoteSha 2bc3dc7f90c78ac4753fddc147aedcf30370eb29

Contents

`optimCheck-package` *Graphical and numerical checks for mode-finding routines.*

Description

Tools for checking that the output of an optimization algorithm is indeed at a local mode of the objective function. This is accomplished graphically by calculating all one-dimensional "projection plots" of the objective function, i.e., varying each input variable one at a time with all other elements of the potential solution being fixed. The numerical values in these plots can be readily extracted for the purpose of automated and systematic unit-testing of optimization routines.

Author(s)

Maintainer: Martin Lysy <mlysy@uwaterloo.ca>

See Also

Useful links:

- https://github.com/mlysy/optimCheck
- Report bugs at https://github.com/mlysy/optimCheck/issues

Examples

```
# example: logistic regression
ilogit <- binomial()$linkinv

# generate data
p <- sample(2:10,1) # number of parameters
n <- sample(1000:2000,1) # number of observations
X <- matrix(rnorm(n*p),n,p) # design matrix
beta0 <- rnorm(p, sd = .1) # true parameter values
y <- rbinom(n, size = 1, prob = ilogit(X %*% beta0)) # response

# fit logistic regression
bhat <- coef(glm(y ~ X - 1, family = binomial))
```

```
# check convergence

# likelihood function
loglik <- function(beta, y, X) {
  sum(dbinom(y, size = 1, prob = ilogit(X %*% beta), log = TRUE))
}

# projection plots
bnames <- parse(text = paste0("beta[", 1:p, "]"))
system.time({
  oproj <- optim_proj(xsol = bhat,
                      fun = function(beta) loglik(beta, y, X),
                      xnames = bnames,
                      xlab = "Coefficient", ylab = "Loglikelihood")
})

# numerical summary
oproj # see ?summary.optproj for more information

# elementwise differences between potential and optimal solution
diff(oproj) # same as summary(oproj)$xdiff

# refit general purpose optimizer starting from bhat
# often faster than optim_proj, but less stable
system.time({
  orefit <- optim_refit(xsol = bhat,
                        fun = function(beta) loglik(beta, y, X))
})
orefit
```

`diff.optcheck` *Elementwise difference between potential and optimal solutions.*

Description

Elementwise difference between potential and optimal solutions.

Usage

```
## S3 method for class 'optcheck'
diff(x, ...)

## S3 method for class 'summary.optcheck'
diff(x, ...)
```

Arguments

`x`	Object of class `optcheck` or `summary.optcheck`, currently returned by `optim_proj()`, `optim_refit()`, or a summary of either of those calls.
`...`	Further arguments to be passed to or from other methods.

Details

This function is simply a wrapper to `summary(x)$xdiff` and `x$xdiff`, for `optcheck` and `summary.optcheck` objects respectively.

Value

A two-column matrix consisting of the absolute and relative differences between the potential and optimal solutions (`xsol` and `xopt`).

`optim_proj` *Projection plot test.*

Description

Given the objective function of an optimization problem and a potential solution, calculates "projection plots" along each coordinate of the solution vector, with all other coordinates being fixed at the input values.

Usage

```
optim_proj(
  xsol,
  fun,
  maximize = TRUE,
  xrng = 0.1,
  npts = 100,
  plot = TRUE,
  ...
)
```

Arguments

`xsol`	Potential solution vector of length `nx`.
`fun`	Objective function to be maximized (or minimized), with first argument the length-`nx` parameter vector over which optimization is to take place. Should return a scalar result.
`maximize`	Logical, whether a maximum or a minimum of the objective function is sought.
`xrng`	Optional specification of the range of each projection plot. Can be: (i) a `2 x nx` matrix giving the endpoints of the range, (ii) a scalar or vector of length `nx`, such that the range in each plot is `theta +/- xrange * abs(theta)`.
`npts`	Number of points in each projection plot.
`plot`	Logical, whether or not to display the projection plots or just return their contents.
`...`	Further arguments to pass to the `plot` method (see `plot.optproj()`).

Value

An object of class `optproj` inheriting from `optcheck` (returned invisibly if `plot = TRUE`, with elements:

`xsol` The potential solution.

`ysol` The value of `fun(xsol)`.

`maximize` Logical indicating whether the potential solution should maximize or minimize the objective function.

`xproj` An `npts x nx` matrix where each column is the x-axis of the projection plot along the given component of `theta`.

`yproj` An `npts x nx` matrix where each column is the y-axis of the corresponding projection plot.

See Also

`plot`, `summary`, `print`, and `diff` methods for projection plots are available; see `plot.optproj()`, `summary.optproj()`, `print.optproj()`, and `diff.optproj()`.

`optim_refit` *Refined optimization test.*

Description

If the potential solution is indeed a local optimum of the objective function, and if it is used to initialize a second optimization, then original and "refined" solutions ought to be close.

Usage

```
optim_refit(xsol, fun, maximize = TRUE, maxit = 5000, reltol = 1e-08, xopt)
```

Arguments

`xsol`	Potential solution vector of length `nx`.
`fun`	Objective function to be maximized (or minimized), with first argument the length-`nx` parameter vector over which optimization is to take place. Should return a scalar result.
`maximize`	Logical, whether a maximum or a minimum of the objective function is sought.
`maxit`	Maximum number of iterations for `stats::optim()` refit (see Details).
`reltol`	Relative tolerance for convergence of `stats::optim()` refit (see Details).
`xopt`	Optional refit solution calculated externally from an optimization algorithm of choice (see Details).

Details

By default, a so-called **refi**ned op(**t**)imization (or refit) test is performed by running the default Nelder-Mead simplex method provided by `stats::optim()`, initialized by the potential solution `xsol`. Only a simplified interface to `stats::optim()`'s control parameters are provided here.

Alternatively, the refit test can be performed with any optimization algorithm of choice. This is done externally, with the refined solution passed to `optim_refit()` via the argument `xopt`.

Value

An object of class `optrefit` inheriting from `optcheck`, with elements:

`xsol` The potential solution.

`ysol` The value of `fun(xsol)`.

`maximize` Logical indicating whether the potential solution should maximize or minimize the objective function.

`xopt` The solution found by the general-purpose optimizer.

`yopt` The function value at the optimal solution, i.e., `fun(xopt)`.

See Also

`summary`, `print`, and `diff` for `optrefit` objects are available; see `summary.optrefit()`, `print.optrefit()`, and `diff.optrefit()`.

`plot.optproj` *Projection plots for optimization routines.*

Description

Projection plots for optimization routines.

Usage

```
## S3 method for class 'optproj'
plot(x, xnames, xind, equalize = FALSE, layout, xlab, ylab, ...)
```

Arguments

`x`	An `optproj` object, i.e., output from function `optim_proj()`.
`xnames`	Optional vector of names for the plot titles.
`xind`	Integer or logical vector of indices indicating which projections should be plotted. Defaults to all projection plots.
`equalize`	If `TRUE`, narrows the range in each projection plot such that the y-value is more or less the same at either endpoint.
`layout`	Optional vector giving the number of rows and columns in the plot layout. For `nx` plots, defaults to `c(nr, nc)`, where `nr = floor(nx)` and `nc = ceiling(nx/nr)`.
`xlab, ylab`	Outer x-axis and y-axis labels.
`...`	Further arguments to be passed to or from other methods.

Value

A grid of projection plots, with vertical lines at the potential solution.

`print.optcheck` *Print method for* `optcheck` *and* `summary.optcheck` *objects.*

Description

Print method for `optcheck` and `summary.optcheck` objects.

Usage

```
## S3 method for class 'optcheck'
print(x, digits = max(3L, getOption("digits") - 3L), n = 5L, ...)

## S3 method for class 'summary.optcheck'
print(x, digits = max(3L, getOption("digits") - 3L), n = 5L, ...)
```

Arguments

`x`	Object of class `optcheck` or `summary.optcheck`, currently returned by `optim_proj()`, `optim_refit()`, or a summary of either of those calls.
`digits`	Number of digits to display.
`n`	Number of elements of solution vector to display (see Details).
`...`	Further arguments to be passed to or from other methods.

Details

The `print` methods for `optcheck` and `summary.optcheck` objects both display three-column matrix, consisting of the potential solution (`xsol`), the absolute difference between it and the optimal solution (`xopt`) return by either `optim_proj()` and `optim_refit()`, and the relative difference (`R = (xopt - xsol)/|xsol|`). Only the elemnts corresponding to the top-`n` relative differences are displayed.

Value

Invisibly `x` itself.

summary.optproj summary *method for projection plots.*

Description

summary method for projection plots.

Usage

```
## S3 method for class 'optproj'
summary(object, xnames, ...)
```

Arguments

object	An optproj object, i.e., output from the function optim_proj().
xnames	Optional vector of names for the elements of the potential solution.
...	Further arguments to be passed to or from other methods.

Details

The print methods for summary.optproj and optproj objects themselves both return a three-column matrix, consisting of the potential solution (xsol), the optimal solution in each projection plot (xopt), and the relative difference between the two (R = (xopt - xsol)/|xsol|).

Value

An object of class summary.optproj inheriting from summary.optcheck, with elements:

xsol The potential solution vector.

ysol The value of the objective function at xsol.

maximize Logical indicating whether the potential solution should maximize or minimize the objective function.

xopt A vector containing the argmax/argmin in each projection plot.

yopt A vector containing the max/min in each projection plot.

xdiff A two-column matrix containing the differences between xsol and xopt. The first column is the absolute difference D = xopt - xsol, the second is the relative difference R = D/|xsol|.

ydiff Same thing, but between ysol and yopt.

See Also

print.summary.optproj() for print method.

summary.optrefit summary *method for* optrefit *objects.*

Description

summary method for optrefit objects.

Usage

```
## S3 method for class 'optrefit'
summary(object, xnames, ...)
```

Arguments

object	An optrefit object, i.e., output from the function optim_refit().
xnames	Optional vector of names for the elements of the potential solution.
...	Further arguments to be passed to or from other methods.

Value

An object of class summary.optrefit inheriting from summary.optcheck, with elements:

xsol The potential solution vector.

ysol The value of the objective function at xsol.

maximize Logical indicating whether the potential solution should maximize or minimize the objective function.

xopt A vector containing the argmax/argmin in each projection plot.

yopt The scalar value of the max/min found by optim_refit.

xdiff A two-column matrix containing the differences between xsol and xopt. The first column is the absolute difference D = xopt - xsol, the second is the relative difference R = D/|xsol|.

ydiff A length-two vector containing the absolute and relative difference between ysol and yopt.

See Also

print.summary.optcheck() for print method.

Index