Package 'reval'

May 26, 2015

Title Repeated Function Evaluation for Sensitivity Analysis

Version 2.0.0

Date 2015-05-25

Author Michael C Koohafkan [aut, cre]

Maintainer Michael C Koohafkan <michael.koohafkan@gmail.com>

Description Simplified scenario testing and sensitivity analysis with R via a generalized function for one-factor-at-a-time (OFAT) sensitivity analysis, evaluation of parameter sets and (sampled) parameter permutations. Options for formatting output and parallel processing are also provided.

URL https://github.com/mkoohafkan/reval

BugReports https://github.com/mkoohafkan/reval/issues

License GPL (>= 3)

Depends R (>= 3.1.3)

Imports doParallel (>= 1.0.8), foreach (>= 1.4.2)

Suggests knitr, ggplot2, dplyr, rivr

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2015-05-26 08:24:24

R topics documented:

reval-package *This package provides the function* evalmany *to simplify scenario testing and sensitivity analysis with R.*

Description

This package provides the function evalmany to simplify scenario testing and sensitivity analysis with R.

evalmany *Repeated evaluations*

Description

Evaluate a function repeatedly across argument sets or permutations.

Usage

```
evalmany(fun, ..., method = c("ofat", "permute", "set"), sample = 0L,
  default.args = list(), collate = TRUE, collate.id = c("single",
  "multi"), collate.prepend = "", collate.fun = identity, clusters = 1L,
  packages = NULL)
```

Arguments

fun	The function to be evaluated.
...	Arguments to be varied when evaluating fun, where each argument in '...' is a (named) vector or list of values. Lists of multi-value objects (e.g. data.frames) should be named explicitly and may otherwise produce unexpected or incorrect names.
method	The sensitivity analysis method to be used. Can be either one-factor-at-a-time ("ofat") evaluation, evaluation of parameter sets ("set"), or (sampled) permutations of parameter sets ("permute"). When method = "ofat", the first element of each argument in '...' is assumed to be the "default" value of that argument.
sample	If method = "permute", the number of parameter permutations to evaluate (sampling without replacement). If sample < 1 (the default) then all possible permutations are evaluated.
default.args	The default values of arguments passed to fun.
collate	Whether to collate the results or not. If TRUE, output elements will be coerced into data.frames using as.data.frame. Otherwise, the raw outputs will be returned as a named list.
collate.id	If collate = TRUE, the method used to store the evaluation identifiers. If collate.id = "single", a single column named 'id' is used. If collate.id = "multi", one column is created for each argument in '...', e.g. 'arg1', 'arg2', etc.

collate.prepend
: A character string prepended to the identifier column. If `collate.id = "single"`, the identifier column will be named `<collate.prepend>id`. If `collate.id = "multi"`, identifier columns will be named as `<collate.prepend><arg>` where `arg` is an element of `...`.

collate.fun
: If `collate = TRUE`, an optional function for reshaping the output of each evaluation prior to coercing and collating the outputs.

clusters
: Number of clusters to use for parallel processing. Default is 1 (serial computation).

packages
: For parallel processing. Character vector of packages that 'fun' depends on.

Value

If `collate = TRUE`, a data.frame. Otherwise, a named list.

Examples

```
myfun = function(n, mean=0, sd = 1){
  x = rnorm(n, mean, sd)
  data.frame(sample.mean = mean(x), sample.sd = sd(x))
}
evalmany(myfun, mean = c(5, 9), sd = c(2, 3), default.args = list(n = 1e6))
evalmany(myfun, mean = seq(20), sd = seq(1, 4, by = 0.1),
  default.args = list(n = 1e6), method = "permute", sample = 10,
  collate.id = "multi", collate.prepend = "arg.")

# vector recycling
evalmany(myfun, mean = c(0, 3, 5), sd = c(1, 10),
  default.args = list(n = 1e6), method = "set", collate.id = "multi")

# Parallel processing
evalmany(myfun, mean = seq(0, 50, by = 10), sd = seq(1, 10, by = 1.5),
  default.args = list(n = 1e5), method = "permute", collate.id = "multi",
  clusters = 2)

## Not run:
# Complex objects
formulas = list(y ~ 1, y ~ x, y ~ x + z)
datasets = list(
  A = data.frame(x = seq(0, 99), y = seq(0, 99) + rnorm(100)),
  B = data.frame(x = seq(0, 99), y = seq(0, 99) + rnorm(100, mean = 5)),
  C = data.frame(x = seq(0, 99), y = seq(0, 99) + rlnorm(100, meanlog = 1),
    z = seq(0, 99) - rlnorm(100, meanlog = -1))
)
# raw output
evalmany(lm, formula = formulas, data = datasets, method = "set",
  collate = FALSE)
# data extraction and error handling
evalmany(lm, formula = formulas, data = datasets, method = "permute",
  collate.id = "multi", collate.fun = function(x)
    data.frame(param = names(x$coefficients), value = x$coefficients,
```

```
    row.names=NULL))

## End(Not run)
```

Index