Package ‘hesim’

December 3, 2017

Type Package

Title Health-Economic Simulation Modeling and Decision Analysis

Version 0.1.0

Description Functionality for developing and analyzing the output of health-economic simulation models. Contains random sampling functions for conducting probabilistic sensitivity analyses (Claxton et al. 2005) <doi:10.1002/hec.985> and individual patient simulations (Brennan et al. 2006) <doi:10.1002/hec.1148>. Individualized cost-effectiveness analysis (Basu and Meltzer 2007, Ioannidis and Garber 2011) <doi:10.1177/0272989X06297393>, <doi:10.1371/journal.pmed.1001058> can be performed on simulation output and used to summarize a probabilistic sensitivity analysis at the subgroup or individual level. Core functions are written in C++ to facilitate computationally intensive modeling.

URL https://github.com/InnovationValueInitiative/hesim

BugReports https://github.com/InnovationValueInitiative/hesim/issues

License GPL-3

LazyData true

LinkingTo Rcpp, RcppArmadillo

Depends R (>= 3.2.3)

Imports data.table, Rcpp (>= 0.12.9), stats

Suggests covr, ggplot2, knitr, msm, rmarkdown, scales, testthat, truncnorm

VignetteBuilder knitr

RoxygenNote 6.0.1

NeedsCompilation yes

Author Devin Incerti [aut, cre]

Maintainer Devin Incerti <devin.incerti@gmail.com>

Repository CRAN

Date/Publication 2017-12-03 12:16:20 UTC

R topics documented:

`custom_table` *Custom CEA summary table*

Description

Custom table summarizing outcomes from probabilistic sensitivity analysis.

Usage

```
custom_table(x, strategy, grp, custom_vars, FUN = NULL)
```

Arguments

`x`	Matrix containing information on outcome variables for each simulation and strategy.
`strategy`	Name of column denoting treatment strategy.
`grp`	Name of column denoting subgroup
`custom_vars`	Name of custom variables to calculate summary statistic for.
`FUN`	summary statistic function.

Value

A data.table of summary statistics for each variable specified in `custom_vars`. By default, returns the mean, 2.5%, and 97.5% quantile of each variable. Different summary statistics can be calculated using FUN. This function is used in `icea` and `icea_pw` to create the `custom.table` output.

Examples

```
# simulation output
nsims <- 100
sim <- data.frame(sim = rep(seq(nsims), 4),
             c = c(rlnorm(nsims, 5, .1), rlnorm(nsims, 5, .1),
                    rlnorm(nsims, 11, .1), rlnorm(nsims, 11, .1)),
             e = c(rnorm(nsims, 8, .2), rnorm(nsims, 8.5, .1),
                   rnorm(nsims, 11, .6), rnorm(nsims, 11.5, .6)),
             strategy = rep(paste0("Strategy ", seq(1, 2)),
```

```
                      each = nsims * 2),
             grp = rep(rep(c("Group 1", "Group 2"),
                      each = nsims), 2)
)

# Custom summary table
custom.fun <- function(x) list(mean = mean(x), median = median(x),
                               quantile(x, c(.025, .975)))
custom_table(sim, strategy = "strategy", grp = "grp",
           custom_vars = "e", FUN = custom.fun)
```

hesim — *hesim: an R package for implement and analyzing health-economic simulation models*

Description

hesim: an R package for implement and analyzing health-economic simulation models

icea — *Individualized cost-effectiveness analysis*

Description

Conduct Bayesian cost-effectiveness analysis (e.g. summarize a probabilistic sensitivity analysis) by subgroup.

Usage

```
icea(x, k, sim = "sim", strategy = "strategy", grp = "grp", e = "e",
  c = "c", custom_vars = NULL, custom_fun = NULL)

icea_pw(x, k, comparator, sim = "sim", strategy = "strategy", grp = "grp",
  e = "e", c = "c", custom_vars = NULL, custom_fun = NULL)
```

Arguments

x	Matrix containing information on mean costs and effectiveness for each simulation. Should be in long form with unit of observation as the simulation and treatment strategy. Should have the following columns (sim = simulation number, strategy = treatment strategy, c = summary of cost for each simulation and treatment strategy e = summary of clinical effectiveness for each simulation and treatment strategy)
k	Vector of willingness to pay values
sim	Name of column denoting simulation number. Default is "sim".

`strategy`	Name of column denoting treatment strategy. Default is "strategy".
`grp`	Name of column denoting subgroup. Default is "grp".
`e`	Name of column denoting clinical effectiveness. Default is "e".
`c`	Name of column denoting costs. Default is "c".
`custom_vars`	Character vector of variable names to use in creating a custom summary table. Table will contain means and 95% credible intervals for each variable. Can contain e and c.
`custom_fun`	Function to apply to custom_vars to make custom table. If `custom_vars` is not NULL and `custom_fun` is NULL, then returns the mean, 2.5% quantile, and 97.5% quantile for each variable in `custom_vars`.
`comparator`	Name of the comparator strategy in x.

Value

`icea` returns a list containing four data.tables:

summary A data.table of the mean, 2.5% quantile, and 97.5% quantile by strategy and group for clinical effectiveness and costs.

mce The probability that each strategy is the most effective treatment for each group for the range of specified willingness to pay values.

evpi The expected value of perfect information by group for the range of specified willingness to pay values.

nmb The mean, 2.5% quantile, and 97.5% quantile of (monetary) net benefits for the range of specified willingness to pay values.

In addition, if `custom_vars` is not NULL, `icea` returns `custom.table`, which is a data.table containing summary statistics for each variable in `custom_vars` by strategy and group.

`icea_pw` also returns a list containing four data.tables:

summary A data.table of the mean, 2.5% quantile, and 97.5% quantile by strategy and group for clinical effectiveness and costs.

delta Incremental effectiveness and incremental cost for each simulated parameter set by strategy and group. Can be used to plot a cost-effectiveness plane. Also returns the difference between each treatment strategy and the comparator for each variable in `custom_vars` if `custom_vars` is not NULL.

ceac Values needed to plot a cost-effectiveness acceptability curve by group. In other words, the probability that each strategy is more cost-effective than the comparator for the specified willingness to pay values.

inmb The mean, 2.5% quantile, and 97.5% quantile of (monetary) incremental net benefits for the range of specified willingness to pay values.

If `custom_vars` is not NULL, also returns `custom.table`, which is a data.table containing summary statistics for the values of each variable returned in `delta` by strategy and group.

Examples

```
# simulation output
nsims <- 100
sim <- data.frame(sim = rep(seq(nsims), 4),
               c = c(rlnorm(nsims, 5, .1), rlnorm(nsims, 5, .1),
                      rlnorm(nsims, 11, .1), rlnorm(nsims, 11, .1)),
               e = c(rnorm(nsims, 8, .2), rnorm(nsims, 8.5, .1),
                     rnorm(nsims, 11, .6), rnorm(nsims, 11.5, .6)),
               strategy = rep(paste0("Strategy ", seq(1, 2)),
                             each = nsims * 2),
               grp = rep(rep(c("Group 1", "Group 2"),
                             each = nsims), 2)
)

# icea
icea.dt <- icea(sim, k = seq(0, 200000, 500), sim = "sim", strategy = "strategy",
 grp = "grp", e = "e", c = "c")
names(icea.dt)
# the probability that each strategy is the most cost-effective
# in each group with a willingness to pay of 20,000
library("data.table")
icea.dt$mce[k == 20000]

# icea_pw
icea.pw.dt <-  icea_pw(sim,  k = seq(0, 200000, 500), comparator = "Strategy 1",
                      sim = "sim", strategy = "strategy", e = "e", c = "c")
names(icea.pw.dt)
# cost-effectiveness acceptability curve
library("ggplot2")
ggplot2::ggplot(icea.pw.dt$ceac[k < 50000], aes_string(x = "k", y = "prob")) +
 geom_line() + facet_wrap(~grp) + xlab("Willingess to pay") +
 ylab("Probability Strategy 2 is more cost-effective than Strategy 1") +
 theme(legend.position = "bottom")
```

`incr_effect` *Incremental treatment effect*

Description

Calculate incremental effect of all treatment strategies on outcome variables from probabilistic sensitivity analysis relative to comparator.

Usage

```
incr_effect(x, comparator, sim, strategy, grp, outcomes)
```

Arguments

x	Matrix containing information on outcome variables for each simulation and strategy.
comparator	Name of comparator strategy.
sim	Name of column denoting simulation number.
strategy	Name of column denoting treatment strategy.
grp	Name of column denoting subgroup.
outcomes	Name of columns to calculate incremental changes for.

Value

A data.table containing the differences in the values of each variable specified in outcomes between each treatment strategy and the comparator. It is the same output generated in `delta` from `icea_pw`.

Examples

```
# simulation output
nsims <- 100
sim <- data.frame(sim = rep(seq(nsims), 4),
             c = c(rlnorm(nsims, 5, .1), rlnorm(nsims, 5, .1),
                    rlnorm(nsims, 11, .1), rlnorm(nsims, 11, .1)),
             e = c(rnorm(nsims, 8, .2), rnorm(nsims, 8.5, .1),
                   rnorm(nsims, 11, .6), rnorm(nsims, 11.5, .6)),
             strategy = rep(paste0("Strategy ", seq(1, 2)),
                            each = nsims * 2),
             grp = rep(rep(c("Group 1", "Group 2"),
                            each = nsims), 2)
)
# calculate incremental effect of Strategy 2 relative to Strategy 1 by group
incr.effect <- incr_effect(sim, comparator = "Strategy 1", sim = "sim",
                           strategy = "strategy", grp = "grp", outcomes = c("c", "e"))
head(incr.effect)
```

---

rcat — *Random generation for categorical distribution*

Description

Draw random samples from a categorical distribution given a matrix of probabilities. `rcat` is vectorized and written in C++ for speed.

Usage

```
rcat(n, prob)
```

Arguments

n	Number of random observations to draw.
prob	A matrix of probabilities where rows correspond to observations and columns correspond to categories.

Value

A vector of random samples from the categorical distribution. The length of the sample is determined by n. The numerical arguments other than n are recycled so that the number of samples is equal to n.

Examples

```
p <- c(.2, .5, .3)
n <- 10000
pmat <- matrix(rep(p, n), nrow = n, ncol = length(p), byrow = TRUE)

# rcat
set.seed(100)
ptm <- proc.time()
samp1 <- rcat(n, pmat)
proc.time() - ptm
prop.table(table(samp1))

# rmultinom from base R
set.seed(100)
ptm <- proc.time()
samp2 <- t(apply(pmat, 1, rmultinom, n = 1, size = 1))
samp2 <- apply(samp2, 1, function(x) which(x == 1))
proc.time() - ptm
prop.table(table(samp2))
```

---

rdirichlet_mat *Random generation for multiple Dirichlet distributions*

Description

Draw random samples from multiple Dirichlet distributions. `rdirichlet_mat` is vectorized and written in C++ for speed.

Usage

```
rdirichlet_mat(n, alpha)
```

Arguments

n	Number of samples to draw.
alpha	A matrix where each row is a separate vector of shape parameters.

Details

This function is particularly useful for representing the distribution of transition probabilities in a transition matrix.

Value

An array of matrices where each row of each matrix is a sample from the Dirichlet distribution.

Examples

```
alpha <- matrix(c(100, 200, 500, 50, 70, 75), ncol = 3, nrow = 2, byrow = TRUE)
samp <- rdirichlet_mat(100, alpha)
print(samp[, , 1:2])
```

`rpwexp` *Random generation for piecewise exponential distribution*

Description

Draw random samples from an exponential distribution with piecewise rates. `rpwexp` is vectorized and written in C++ for speed.

Usage

```
rpwexp(n, rate = 1, time = 0)
```

Arguments

`n`	Number of random observations to draw.
`rate`	A matrix of rates where rows correspond to observations and columns correspond to rates during specified time intervals.
`time`	A vector equal to the number of columns in `rate` giving the times at which the rate changes

Value

A vector of random samples from the piecewise exponential distribution. The length of the sample is determined by n. The numerical arguments other than n are recycled so that the number of samples is equal to n.

Examples

```
rate <- c(.6, 1.2, 1.3)
n <- 100000
ratemat <- matrix(rep(rate, n/2), nrow = n,
                  ncol = 3, byrow = TRUE)
t <- c(0, 10, 15)
ptm <- proc.time()
samp <- rpwexp(n, ratemat, t)
proc.time() - ptm
summary(samp)
```

Index