Package ‘idem’

May 24, 2020

Title Inference in Randomized Controlled Trials with Death and Missingness

Version 5.0

Author Chenguang Wang [aut, cre],
        Andrew Leroux [aut, cre],
        Elizabeth Colantuoni [aut],
        Daniel O Scharfstein [aut],

Maintainer Chenguang Wang <cwang68@jhmi.edu>

Description In randomized studies involving severely ill patients, functional outcomes are often unobserved due to missed clinic visits, premature withdrawal or death. It is well known that if these unobserved functional outcomes are not handled properly, biased treatment comparisons can be produced. In this package, we implement a procedure for comparing treatments that is based on the composite endpoint of both the functional outcome and survival. The procedure was proposed in Wang et al. (2016) <DOI:10.1111/biom.12594> and Wang et al. (2020) <DOI:10.18637/jss.v093.i12>. It considers missing data imputation with different sensitivity analysis strategies to handle the unobserved functional outcomes not due to death.

Depends R (>= 3.4.0), Rcpp (>= 0.12.0), methods

License GPL (>= 3)

LinkingTo StanHeaders (>= 2.18.0), rstan (>= 2.18.1), BH (>= 1.66.0),
        Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0)

RcppModules stan_fit4idem_mod

Imports rstan (>= 2.18.1), rstantools (>= 1.5.0), sqldf (>= 0.4),
        survival (>= 2.38), mice (>= 3.9.0), parallel (>= 3.2),

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

Suggests knitr, shiny, rmarkdown, pander, DT, shinythemes

VignetteBuilder knitr
SystemRequirements GNU make
NeedsCompilation yes
Repository CRAN
Date/Publication 2020-05-23 23:50:02 UTC

R topics documented:

idem-package
abc
imData
imFitModel
imImpAll
imImpAll_mice
imImpSingle
imInfer
imShiny
plot.IDEMDATA
plot.IDEMFIT
plot.IDEMIMP
plot.IDEMINFER
plot.IDEMSINGLE
plot.summary.IDEMINFER
print.IDEMDATA
print.IDEMERROR
print.IDEMFIT
print.IDEMIMP
print.IDEMINFER
print.IDEMSINGLE
print.summary.IDEMINFER
summary.IDEMDATA
summary.IDEMINFER

Index

Description

This package contains the functions for drawing inference in randomized clinical trials with death and intermittent missingness.

Notation

Consider a two-arm randomized study. Let $Y_k$ denote outcome measured at time $t_k$ and $Z$ denote a functional endpoint that is a function of $Y$. Let $L$ denote the survival time. Let $X$ denote the baseline covariates and $T$ denote the treatment assignment.
Ranking

If two subjects were both alive at the end of the study, they are ranked based on functional outcome $Z$. If at least one subject was dead at the end of the study, they are ranked based on survival time $L$.

Treatment effect, $\theta$ is defined as the probability that the outcome for a random individual randomized to treatment $T = 0$ is less than the outcome for a random individual randomized to treatment $T = 1$ minus the probability that the outcome for a random individual randomized to treatment $T = 0$ is greater than the outcome of a random individual randomized to treatment $T = 1$.

Missingness

In order to estimate $\theta$ in the presence of missing data, we need to impute $Z$ for subjects alive at the end of the study with $Y_k$ missing for some $k$.

The benchmark assumption we consider for the imputation is the complete case missing value (CCMV) restrictions. We then consider exponential tilting models for introducing sensitivity parameters for evaluating the robustness of the findings with regards to different missing data mechanism assumptions. The models are as follows:

$$f(Y_{mis}^{(s)}|Y_{obs}^{(s)}, Y_0, X, T, S = s) \propto \exp(\beta_T Z) f(Y_{mis}^{(s)}|Y_{obs}^{(s)}, Y_0, X, T, S = 1)$$

where $S$ denotes the missingness patterns, $S = 1$ denotes the completers and $\beta_T$ denotes the sensitivity parameter for arm $T$.

Graphical user interface (GUI)

This package provides a web-based GUI. See imShiny for details.

References


Example dataset

The Awakening and Breathing Controlled (ABC) trial randomized critically ill patients receiving mechanical ventilation 1:1 within each study site to management with a paired sedation plus ventilator weaning protocol involving daily interruption of sedative through spontaneous awakening trials (SATs) and spontaneous breathing trials (SBTs) or sedation per usual care (UC) and SBTs.

The example dataset is from a single site substudy in ABC. The researchers assessed differences in cognitive, psychological and functional outcomes at 3 and 12 months after randomization.
Format

A dataframe with 5 variables:

AGE  Age
TRT  Treatment assignment. 0: UC + SBT, 1: SAT + SBT
SURV Survival days
Y2  Cognitive score at 12 months
Y1  Cognitive score at 3 months

References


---

**imData**

*Create data for IDEM analysis*

**Description**

Create a class IDEMDATA object for IDEM analysis

**Usage**

```r
imData(
  data, 
  trt = NULL, 
  surv = NULL, 
  outcome = NULL, 
  endfml = NULL, 
  y0 = NULL, 
  cov = NULL, 
  duration = 9999, 
  bounds = NULL, 
  trt.label = NULL, 
  unitTime = "days", 
  err.terminate = TRUE, 
  ...
)
```

Arguments

data | Original dataset  
trt | Variable name for the Control (0) and Intervention (1) treatment assignments in the dataset  
surv | Variable name for the survival (time to event) variable in the dataset  
outcome | Chronologically ordered vector of variable names for clinical outcomes in the dataset excluding baseline  
endfml | R expression indicating the user-specified final outcome of interest. This is the function for $Z$ of one or more of $Y_k$'s  
y0 | Variable name of the baseline clinical outcome  
cov | Vector of variable names for the covariates used in the imputation procedure for missing clinical outcomes  
duration | Length of the study. This is the time at which subjects' are assumed to be censored  
bounds | Numeric vector of lower and upper bounds for subjects' imputed clinical outcomes  
trt.label | label of the treatment arms  
unitTime | Unit of time measurement for survival and function outcome time points  
err.terminate | When there is error in the specification, the program should be stopped with an error message if err.terminate is true. Otherwise, the error message will be returned and the program will continue.  

Details

When there are errors in the specification, i.e. trt is not a column of data, a class IDEMERROR object will be returned. The detailed errors can be checked by calling print of the IDEMERROR object.

Value

When the specifications are correct, a class IDEMDATA list will be returned. The list contains

- **data** | Original dataset  
- **lst.var** | List of the specifications

Examples

```r
rst.data <- imData(abc, trt="TRT", surv="SURV", outcome=c("Y1","Y2"),
y0=NULL, endfml="Y2",
trt.label = c("UC+SBT", "SAT+SBT"),
cov=c("AGE"), duration=365, bounds=c(0,100));
```
imFitModel

Imputation model fitting

Description

Fit linear imputation models to the observed data from complete survivors for each treatment arm at each time point.

Usage

imFitModel(im.data)

Arguments

im.data A class IDEMDATA object generated by \texttt{imData}

Value

A class IDEMFIT list of modeling fitting results with the following items:

\begin{itemize}
\item \texttt{im.data} Original class IDEMDATA object
\item \texttt{rst.mdl} A list of modeling fitting results for each model with
  \begin{itemize}
  \item \texttt{lm} results from function \texttt{lm}
  \item \texttt{formula} model formula
  \item \texttt{coef} model coefficients
  \item \texttt{res} residuals
  \item \texttt{h} bandwidth of residuals for kernel density estimation
  \end{itemize}
\end{itemize}

See Also

\texttt{imData}, \texttt{idem-package}

Examples

\begin{verbatim}
im.abc <- imData(abc, trt="TRT", surv="SURV", outcome=c("Y1","Y2"), y0=NULL, endfml="Y2", trt.label = c("UC+SBT", "SAT+SBT"), cov=c("AGE"), duration=365, bounds=c(0,100));
im.fit <- imFitModel(im.abc);
\end{verbatim}
**imImpAll**  

*Impute missing data*

**Description**

Conduct imputation under benchmark assumptions or for sensitivity analysis for a given set of subjects using the model fitting results.

**Usage**

```r
imImpAll(
  fit.rst,
  data.all = NULL,
  deltas = 0,
  normal = TRUE,
  n.imp = 5,
  endponly = TRUE,
  update.progress = NULL,
  imputeNone = FALSE,
  ..., 
  seed = NULL
)
```

**Arguments**

- `fit.rst`: A class IDEMFIT results generated by `imFitModel`.
- `data.all`: A dataframe containing subjects with missing data. The default value is NULL, in which case the function will impute missing data for subjects in the original dataset in the class IDEMFIT object `fit.rst`.
- `deltas`: Vector of imputation sensitivity parameters.
- `normal`: Logical variable indicating whether normality assumption should be made for the residuals.
- `n.imp`: Number of complete datasets required.
- `endponly`: Logical variable that indicates whether clinical outcomes not used in calculating the functional outcome are considered as missing and should be imputed. The default is `FALSE`, indicating that all missing clinical outcomes will be imputed sequentially.
- `update.progress`: Parameter reserved for run `idem` in GUI mode.
- `imputeNone`: If TRUE, no imputation will be conducted. The data from subjects that do not need imputation will be returned.
- `...`: options to call STAN sampling. These options include `chains`, `iter`, `warmup`, `thin`, `algorithm`. See `rstan::sampling` for details.
- `seed`: Random seed.
imImpAll_mice

Impute missing data by mice

Value

If `imputeNone` is TRUE, return a dataset with the original data for the subset of subjects who died at the end of the study or had no missing outcomes. Otherwise, return a class IDEMIMP list with components

- **lst.var**: List of parameters
- **complete**: A dataset with the original data for the subset of subjects who died at the end of the study or had no missing outcomes and the `n.imp` imputed missing outcomes for subjects who need missing value imputation.
- **n.imp**: Number of imputed complete datasets
- **deltas**: Imputation sensitivity parameters
- **org.data**: Original dataset
- **normal**: Normal assumption for the imputation
- **stan.par**: STAN options

Examples

```r
## Not run:
rst.abc <- imData(abc, trt="TRT", surv="SURV", outcome=c("Y1","Y2"),
y0=NULL, endfml="Y2",
trt.label = c("UC+SBT", "SAT+SBT"),
cov=c("AGE"), duration=365, bounds=c(0,100));
 rst.fit <- imFitModel(rst.abc);
 rst.imp <- imImpAll(rst.fit, deltas=c(-0.25,0,0.25),
 normal=TRUE, chains = 2, iter = 2000, warmup = 1000);
## End(Not run)
```

---

### Description

Conduct imputation using the NARFCS model implemented in the mice package

### Usage

```r
imImpAll_mice(
  im.data,
  deltas = 0,
  n.imp = 5,
  endponly = TRUE,
  seed = NULL,
  ...
)
```
Arguments

**im.data** A class IDEMDATA object generated by `imData`

**deltas** Vector of imputation sensitivity parameters

**n.imp** Number of complete datasets required

**endponly** Logical variable that indicates whether clinical outcomes not used in calculating the functional outcome are considered as missing and should be imputed. The default is `FALSE`, indicating that all missing clinical outcomes will be imputed sequentially

**seed** Random seed

... Parameters for `mice`

Value

If `imputeNone` is `TRUE`, return a dataset with the original data for the subset of subjects who died at the end of the study or had no missing outcomes.

Otherwise, return a class IDEMIMP list with components

- **lst.var** List of parameters
- **complete** A dataset with the original data for the subset of subjects who died at the end of the study or had no missing outcomes and the `n.imp` imputed missing outcomes for subjects who need missing value imputation.
- **n.imp** Number of imputed complete datasets
- **deltas** Imputation sensitivity parameters
- **org.data** Original dataset

Examples

```r
## Not run:
rst.abc <- imData(abc, trt="TRT", surv="SURV", outcome=c("Y1","Y2"),
y0=NULL, endfml="Y2",
trt.label = c("UC+SBT", "SAT+SBT"),
cov=c("AGE"), duration=365, bounds=c(0,100));
rst.imp <- imImpAll_mice(rst.abc, deltas=-0.25,0,0.25);
## End(Not run)
```

---

**imImpSingle** Impute missing data for MCMC convergence checking

Description

Call STAN model to impute missing data for an individual subject under benchmark assumption for MCMC convergence checking
Usage

```r
imImpSingle(
  dsub, 
  fit.rst, 
  normal = TRUE, 
  chains = 4, 
  iter = 5000, 
  warmup = 1000, 
  control = list(adapt_delta = 0.95), 
  ..., 
  seed = NULL 
)
```

Arguments

dsub original individual subject data
fit.rst A class IDEMFIT results generated by `imFitModel`.
normal Logical variable indicating whether normality assumption should be made for the residuals
chains STAN parameter. Number of Markov chains
iter STAN parameter. Number of iterations
warmup STAN parameter. Number of burnin.
control STAN parameter. See `rstan::stan` for details.
... other options to call STAN sampling such as `thin`, `algorithm`. See `rstan::sampling` for details.
seed Random seed

Value

NULL if there is no missing data in `dsub`
Otherwise, return a class IDEMSINGLE object that contains a list with components

- **dsub** original data of the subject
- **rst.stan** A stan.fit class result returned from `rstan::sampling`
- **complete** A dataframe with complete data for the selected subject

Examples

```r
im.abc <- imData(abc, trt="TRT", surv="SURV", outcome=c("Y1","Y2"), 
y0=NULL, endfml="Y2", 
trt.label = c("UC+SBT", "SAT+SBT"), 
cov=c("AGE"), duration=365, bounds=c(0,100));
im.fit <- imFitModel(im.abc);
im.imp <- imImpSingle(abc[1,], im.fit, chains = 4, iter = 200, warmup = 100);
```
**imInfer**

*Treatment effect estimation and hypothesis testing*

**Description**

Estimate treatment effect size. Estimate variation and conduct hypothesis testing by bootstrap analysis.

**Usage**

```r
imInfer(
  imp.rst,  # IDEMIMP object returned from imImpAll
  n.boot = 0,  # Number of bootstrap samples
  n.cores = 1,  # Number of cores for parallel computation. Fixed at 1 for Windows.
  update.progress = NULL,  # Parameter reserved for run idem in GUI mode
  effect.quantiles = c(0.25, 0.5, 0.75),  # Composite quantiles of interest for measuring treatment effect
  quant.ci = c(0.025, 0.975),  # Quantiles for extracting bootstrap confidence intervals
  ...  # Extra options for ranking subjects using the composite endpoint that include
  cut.z: Clinically meaningful difference in the functional outcome
  cut.surv: Clinically meaningful difference in survival time
  seed = NULL  # Random seed
)
```

**Arguments**

- `imp.rst`: A class IDEMIMP object returned from `imImpAll`
- `n.boot`: Number of bootstrap samples
- `n.cores`: Number of cores for parallel computation. Fixed at 1 for Windows.
- `update.progress`: Parameter reserved for run idem in GUI mode
- `effect.quantiles`: Composite quantiles of interest for measuring treatment effect
- `quant.ci`: Quantiles for extracting bootstrap confidence intervals
- `...`: Extra options for ranking subjects using the composite endpoint that include
  - `cut.z`: Clinically meaningful difference in the functional outcome
  - `cut.surv`: Clinically meaningful difference in survival time
- `seed`: Random seed

**Details**

If `n.boot=0`, bootstrap analysis will not be conducted. Instead, only the treatment effect size will be estimated using the imputed data.
Value

A class IDEMTEST list containing

**lst.var** List of specification parameters

**deltas** Vector of sensitivity parameters

**theta** A data frame with columns

- **Delta0**: Sensitivity parameter for control arm
- **Delta1**: Sensitivity parameter for intervention arm
- **Theta**: Estimated $\theta$
- **SD**: Standard deviation (when n.boot > 0)
- **PValue**: p-value (when n.boot > 0)

**effect.quantiles** A data frame with columns

- **Delta**: Sensitivity parameter
- **TRT**: Treatment arm
- **Q**: Quantiles of the composite endpoint to be estimated
- **QuantY**: Estimated quantiles if the quantiles correspond to functional outcome (when n.boot > 0)
- **QuantSurv**: Estimated quantiles if the quantiles correspond to survival days (when n.boot > 0)
- **Q**: Bootstrap quantiles for the QuantY (when n.boot > 0)
- **QSurv**: Bootstrap quantiles for the QuantSurv (when n.boot > 0)

**bootstrap** A list with length n.boot. The $i$th item is the class IDEMEST list corresponding to the $i$th bootstrap sample

Examples

```r
## Not run:
rst.abc <- imData(abc, trt="TRT", surv="SURV", outcome=c("Y1","Y2"),
y0=NULL, endfml="Y2",
  trt.label = c("UC+SBT", "SAT+SBT"),
  cov=c("AGE"), duration=365, bounds=c(0,100));
rst.fit <- imFitModel(rst.abc);
rst.imp <- imImpAll(rst.fit, deltas=c(-0.25,0,0.25),
  normal=TRUE, chains = 2, iter = 2000, warmup = 1000);
rst.est <- imInfer(rst.imp, n.boot = 0, effect.quantiles = c(0.25,0.5,0.75));
rst.test <- imInfer(rst.imp, n.boot = 100, effect.quantiles = c(0.25,0.5,0.75));
## End(Not run)
```
**imShiny**

**Run Web-Based idem application**

**Description**

Call Shiny to run idem as a web-based application.

**Usage**

```r
imShiny()
```

**Details**

A web browser will be brought up for users to access the GUI of idem.

**Examples**

```r
## Not run:
run.idem()
## End(Not run)
```

---

**plot.IDEMDATA**

**Plot of IDEMDATA object**

**Description**

Generate different types of plots for class IDEMDATA objects.

**Usage**

```r
## S3 method for class 'IDEMDATA'
plot(
  x,
  opt = c("survivor", "missing", "KM"),
  cols = c("black", "blue"),
  fname = NULL,
  ...
)
```
Arguments

x A class IDEMDATA object generated by \texttt{imData}

opt Types of the plot
- survivor: Spaghetti plot for subjects alive at the end of the study
- missing: Plot the missing patterns of the observed data
- KM: Plot Kaplan-Meier survival curves

cols Curve colors of the treatment and control arm for survival plot or colors of the observed and missing data for missingness plot.

fname File name of the result pdf file. If \texttt{fname} is null, result pdf file will not be generated

... Extra arguments for \texttt{plot}

See Also

\texttt{imData}

Examples

\begin{verbatim}
rst.data <- imData(abc, trt="TRT", surv="SURV", outcome=c("Y1","Y2"),
y0=NULL, endfml="Y2",
  trt.label = c("UC+SBT", "SAT+SBT"),
  cov=c("AGE"), duration=365, bounds=c(0,100));
plot(rst.data, opt = "survivor");
plot(rst.data, opt = "missing", cols = c("blue", "gray"));
plot(rst.data, opt = "KM");
\end{verbatim}

---

\texttt{plot.IDEMFIT} \hspace{1cm} \textit{Plot model fitting results}

Description

Plot method of the class IDEMFIT to generate model fitting diagnosis plots

Usage

\begin{verbatim}
## S3 method for class 'IDEMFIT'
plot(x, trt = NULL, mfrow = NULL, ...)
\end{verbatim}

Arguments

x A class IDEMFIT object generated by \texttt{imFitModel}

trt Treatment arm selected for the diagnostic plots. If NULL, all treatment arms are included

mfrow Plot option

... Additional arguments
See Also

imFitModel

Examples

```r
im.abc <- imData(abc, trt="TRT", surv="SURV", outcome=c("Y1","Y2"),
y0=NULL, endfml="Y2",
trt.label = c("UC+SBT", "SAT+SBT"),
cov=c("AGE"), duration=365, bounds=c(0,100));
im.fit <- imFitModel(im.abc);
plot(im.fit, mfrow=c(2,4));
```

Description

Generate different types of plots for class IDEMIMP objects generated by `imImpAll`

Usage

```r
## S3 method for class 'IDEMIMP'
plot(x, opt = c("imputed", "composite"), fname = NULL, ...)
```

Arguments

- `x` A class IDEMIMP object returned from `imImpAll`
- `opt` Types of the plot
  - `imputed`: Plot density of imputed values and the density of the observed outcomes
  - `composite`: Generate cumulative plot of the composite survival and functional outcome
- `fname` File name of the result pdf file. If `fname` is null, result pdf file will not be generated
- `...` Options for generating the plots.
  - `type = imputed`
    - `deltas`: Imputation sensitivity parameter for which to generate the results
    - `endp`: If `TRUE`, plot the densities of the imputed functional outcomes. Otherwise, plot the densities of the imputed outcomes
    - `adjdensity` estimation option
    - `colsplot` option for colors
    - `ltysplot` options for line types
    - `xlimplot` options
• ylimplot options
• mfrowplot options

**type = composite**

• *at.surv*: Sets the range of the survival times to plot in the cumulative distribution function. By default the range is the range of survival values up to the duration of the study
• *at.z*: Sets the range of the functional outcome to plot in the cumulative distribution function. By default this is the range of the functional outcomes plus the buffer amount to improve visibility in the transition from survival to functional outcome
• *p.death*: Proportion of the plot width devoted to Survival. By default the cumulative distribution will devote horizontal space to the survival portion that is proportional to the number of subjects who die prior to duration
• *buffer*: Small horizontal gap used to better visually distinguish the transition from survival to functional outcome
• *delta*: Imputation sensitivity parameter for which to generate the results
• *seg.lab*: Labels for the two components of the composite outcome
• *main*: plot options

**See Also**

imImpAll

**Examples**

```r
## Not run:
im.abc <- imData(abc, trt="TRT", surv="SURV", outcome=c("Y1","Y2"),
y0=NULL, endfml="Y2",
  trt.label = c("UC+SBT", "SAT+SBT"),
  cov=c("AGE"), duration=365, bounds=c(0,100));
rst.fit <- imFitModel(im.abc);
rst.imp <- imImpAll(rst.fit, deltas=c(-0.25,0,0.25),
  normal=TRUE, chains = 2, iter = 2000, warmup = 1000);
plot(rst.imp, opt = "imputed"),
plot(rst.imp, opt = "composite")
## End(Not run)
```

---

**plot.IDEMINFER**  
Plot hypothesis testing results  

**Description**

Generate contour plot of p-values or treatment effect theta for sensitivity analysis results
Usage

## S3 method for class 'IDEMINFER'
plot(x, con.v = 0.05, nlevels = 30, opt = c("pvalue", "effect"), ...)

Arguments

x A class IDEMINFER list generated by `imInfer`
con.v Levels of contour plot
nlevels Levels of color scale
opt contour plots of pvalue or effect
... Options for `filled.contour`

Details

The plot will only be generated when bootstrap analysis has been conducted.

Examples

## Not run:
rst.abc <- imData(abc, trt="TRT", surv="SURV", outcome=c("Y1","Y2"),
y0=NULL, endfml="Y2",
trt.label = c("UC+SBT", "SAT+SBT"),
cov=c("AGE"), duration=365, bounds=c(0,100));
rst.fit <- imFitModel(rst.abc);
rst.imp <- imImpAll(rst.fit, deltas=c(-0.25,0,0.25),
  normal=TRUE, chains = 2, iter = 2000, warmup = 1000);
rst.est <- imInfer(rst.imp, n.boot = 100);
plot(rst.est);
## End(Not run)

plot.IDEMSINGLE  

Plot MCMC mixing results

Description

Plot method of the class IDEMSINGLE to generate traceplot of the imputed missing outcomes

Usage

## S3 method for class 'IDEMSINGLE'
plot(x, ...)

Arguments

x A class IDEMSINGLE object returned from `imImpSingle`
... Additional arguments
See Also

imImpSingle

Examples

im.abc <- imData(abc, trt="TRT", surv="SURV", outcome=c("Y1","Y2"),
y0=NULL, endfml="Y2",
trt.label = c("UC+SBT", "SAT+SBT"),
cov=c("AGE"), duration=365, bounds=c(0,100));
im.fit <- imFitModel(im.abc);
im.imp.single <- imImpSingle(abc[1,], im.fit,
chairs = 4, iter = 200, warmup = 100);
plot(im.imp.single);

plot.summary.IDEMINFER

Plot survivors only and SACE analysis results

Description

Generate a plot of survivor only and survivor average causal effect values

Usage

## S3 method for class 'summary.IDEMINFER'
plot(
  x,
  opt = c("pvalue", "effect"),
  by.sace = TRUE,
  delta0 = 0,
  delta1 = 0,
  sace.delta = NULL,
  ...
)

Arguments

x A class summary.IDEMSACE object generated by summary of IDEMINFER
opt contour plots of pvalue or effect
by.sace Logical value. If True, create a contour plot for given SACE sensitivity parameter. Otherwise, create a plot for treatment effect for given imputation sensitivity parameters
delta0 Selected treatment arm 0 sensitivity parameters
delta1 Selected treatment arm 1 sensitivity parameters
sace.delta Single SACE sensitivity parameter
... Options for plot
Details

The plot function will only generate the contour plot of p-values or treatment effects on functional outcomes for survivors only analyses.

For SACE analysis, the plot function generates contour plot of line plot based on the value of by.sace.

Examples

```r
## Not run:
rst.abc <- imData(abc, trt="TRT", surv="SURV", outcome=c("Y1","Y2"),
                y0=NULL, endfml="Y2",
                trt.label = c("UC+SBT", "SAT+SBT"),
                cov=c("AGE"), duration=365, bounds=c(0,100));
rst.fit <- imFitModel(rst.abc);
rst.imp <- imImpAll(rst.fit, deltas=c(-0.25,0,0.25),
                 normal=TRUE, chains = 2, iter = 2000, warmup = 1000);
rst.infer <- imInfer(rst.imp, n.boot = 100, effect.quantiles = c(0.25,0.5,0.75));
rst.survivors <- summary(rst.infer, opt="survivor");
plot(rst.survivors);
## End(Not run)
```

print.IDEMDATA

Print IDEMDATA object

Description

Print the specification details of class IDEMDATA objects generated by `imData`

Usage

```r
## S3 method for class 'IDEMDATA'
print(x, ...)  
```

Arguments

- `x` A class IDEMDATA object generated by `imData`
- `...` Additional arguments

See Also

`imData`
print.IDEMERROR  

Print error messages

Description

Print error messages in the parameter specifications generated by \texttt{imData}

Usage

```r
## S3 method for class 'IDEMERROR'
print(x, html = FALSE, ...)
```

Arguments

- \texttt{x}  
  A class \texttt{IDEMERROR} object returned by \texttt{imData} when there are misspecifications
- \texttt{html}  
  Logical indicator for the format of the error messages. When \texttt{TRUE}, the error messages are formatted in HTML format
- \texttt{...}  
  Additional arguments

See Also

\texttt{imData}

Examples

```r
## Not run:
rst.data <- imData(abc, trt="TRT", outcome=c("Y1","Y2"), y0=NULL,
endfml="Y3", bounds=c(10,20), duration=365,
err.terminate=FALSE);
print(rst.data);
## End(Not run)
```

print.IDEMFIT  

Print model fitting results

Description

Print method of the class IDEMFIT generated by \texttt{imFitModel}

Usage

```r
## S3 method for class 'IDEMFIT'
print(x, ...)
```
print.IDEMIMP

Arguments

x  A class IDEMFIT object generated by \texttt{imFitModel}
...

Details

Print the results from \texttt{lm} for all the models

See Also

\texttt{imFitModel}

\section*{Description}

Print method for class IDEMIMP objects generated by \texttt{imImpAll}

\section*{Usage}

\begin{verbatim}
## S3 method for class 'IDEMIMP'
print(x, ...)
\end{verbatim}

Arguments

x  A class IDEMIMP object returned from \texttt{imImpAll}
...

Extra arguments

See Also

\texttt{imImpAll}

\section*{print.IDEMINFER}

\section*{Description}

Print method of class IDEMINFER for treatment effect estimation and hypothesis testing results

\section*{Usage}

\begin{verbatim}
## S3 method for class 'IDEMINFER'
print(x, delta0 = NULL, delta1 = NULL, ...)
\end{verbatim}
Arguments

- `x`: A class IDEMINFER list generated by `imInfer`
- `delta0`: Selected treatment arm 0 sensitivity parameters
- `delta1`: Selected treatment arm 1 sensitivity parameters
- `...`: Extra arguments

Examples

```r
## Not run:
rst.abc <- imData(abc, trt="TRT", surv="SURV", outcome=c("Y1","Y2"),
y0=NULL, endfml="Y2",
trt.label = c("UC+SBT", "SAT+SBT"),
cov=c("AGE"), duration=365, bounds=c(0,100));
rst.fit <- imFitModel(rst.abc);
rst.imp <- imImpAll(rst.fit, deltas=c(-0.25,0,0.25),
                   normal=TRUE, chains = 2, iter = 2000, warmup = 1000);
rst.test <- imInfer(rst.imp, n.boot = 100);
print(rst.test, delta0 = 0, delta1 = 0.15)
## End(Not run)
```

Description

Print method for class IDEMSINGLE objects generated by `imImpSingle`

Usage

```r
## S3 method for class 'IDEMSINGLE'
print(x, ...)
```

Arguments

- `x`: A class IDEMSINGLE object returned from `imImpSingle`
- `...`: Additional arguments

See Also

- `imImpSingle`
print.summary.IDEMINFER

Print survivors only or SACE analysis results

Description

Print survivors only or SACE analysis results

Usage

```r
## S3 method for class 'summary.IDEMINFER'
print(x, delta0 = NULL, delta1 = NULL, ...)
```

Arguments

- `x`: A class IDEMINFER list generated by `imInfer`
- `delta0`: Selected treatment arm 0 sensitivity parameters
- `delta1`: Selected treatment arm 1 sensitivity parameters
- `...`: Extra arguments

summary.IDEMDATA

Summary of IDEMDATA object

Description

Summarize the missing data information for class IDEMDATA objects generated by `imData`.

Usage

```r
## S3 method for class 'IDEMDATA'
summary(object, opt = c("misstable", "missid", "trt"), endponly = FALSE, ...)
```

Arguments

- `object`: A class IDEMDATA object returned by `imData`
- `opt`: Types of the summarization
  - `misstable`: Summarize the frequencies of each missing pattern
  - `missid`: Summarize the indices of subjects that need imputation, i.e. survivors with functional endpoint missing
  - `trt`: Treatment arms
- `endponly`: Logical variable that indicates whether clinical outcomes not used in calculating the functional outcome are considered as missing and should be imputed. The default is `FALSE`, indicating that all missing clinical outcomes will be imputed sequentially
- `...`: Extra arguments
Value

A dataframe with frequencies of each missing pattern when opt is misstable.
A vector of indices of subjects that need imputation when opt is missid.
A vector of treatment arms in the data when opt is trt.

See Also

imData

summary.IDEMINFER

Summary of the inference results

Description

Summarize survivors only or Survivor Averaged Causal Effect (SACE) based on the imputation and bootstrap analysis

Usage

## S3 method for class 'IDEMINFER'

summary(object, opt = c("survivor", "SACE"), sace.deltas = NULL, ...)

Arguments

object A class IDEMINFER list generated by imInfer
opt Types of the summary
  • survivor: Survivors only analysis
  • SACE: Survivor Averaged Causal Effect
sace.deltas Vector of sensitivity parameters for SACE estimation. If NULL, the values will be generated based on the standard deviations of the estimated differences in the functional outcomes between the treatment and control arms
...

Optional arguments for summary

Details

For SACE, the default sensitivity parameters will be determined by the standard deviation of the treatment effect size on the functional outcomes.
A class `summary.IDEMINFER` list containing:

- **deltas**: imputation sensitivity parameters
- **n.boot**: number of bootstrap samples in bootstrap analysis
- **sace.deltas**: SACE sensitivity parameters when `opt = SACE`

A data frame with columns:

- **Delta0**: Imputation sensitivity parameter for control arm,
- **Delta1**: Imputation sensitivity parameter for intervention arm
- **SACE_Delta**: SACE sensitivity parameter when `opt = SACE`
- **Effect**: SACE estimate
- **LB**: Lower bound of the 95
- **UB**: Upper bound of the 95
- **PValue**: p-value when when `n.boot > 0` in the IDEMINFER object

References


Examples

```r
## Not run:
rst.abc <- imData(abc, trt="TRT", surv="SURV", outcome=c("Y1","Y2"),
y0=NULL, endfml="Y2",
trt.label = c("UC+SBT", "SAT+SBT"),
cov=c("AGE"), duration=365, bounds=c(0,100));
rst.fit <- imFitModel(rst.abc);
rst.imp <- imImpAll(rst.fit, deltas=c(-0.25,0,0.25),
                 normal=TRUE, chains = 2, iter = 2000, warmup = 1000);
rst.infer <- imInfer(rst.imp, n.boot = 100, effect.quantiles = c(0.25,0.5,0.75));
rst.sace <- summary(rst.infer, opt = "SACE")
## End(Not run)
```
Index

abc, 3
idem (idem-package), 2
idem-package, 2
imData, 4, 6, 9, 14, 19, 20, 23, 24
imFitModel, 6, 7, 10, 14, 15, 20, 21
imImpAll, 7, 11, 15, 16, 21
imImpAll_mice, 8
imImpSingle, 9, 17, 18, 22
imInfer, 11, 17, 22–24
imShiny, 3, 13
plot.IDEMDATA, 13
plot.IDEMFIT, 14
plot.IDEMIMP, 15
plot.IDEMINFER, 16
plot.IDEMSINGLE, 17
plot.summary.IDEMINFER, 18
print.IDEMDATA, 19
print.IDEMERROR, 20
print.IDEMFIT, 20
print.IDEMIMP, 21
print.IDEMINFER, 21
print.IDEMSINGLE, 22
print.summary.IDEMINFER, 23
summary.IDEMDATA, 23
summary.IDEMINFER, 24