Package ‘IrregLong’

January 24, 2020

Type Package

Title Analysis of Longitudinal Data with Irregular Observation Times

Version 0.2.0

Date 2020-01-23

Author Eleanor Pullenayegum

Maintainer Eleanor Pullenayegum <eleanor.pullenayegum@sickkids.ca>


Depends R (>= 2.10)

Imports survival, geepack, frailtypack, data.table

License GPL-3

RoxygenNote 7.0.2

Suggests knitr, rmarkdown, nlme, MEMSS

VignetteBuilder knitr

LazyData true

Language en-GB

NeedsCompilation no

Repository CRAN

Date/Publication 2020-01-24 05:10:02 UTC

R topics documented:

abacus.plot ............................................................... 2
addcensoredrows ....................................................... 3
iiw ................................................................. 4
iiw.weights ............................................................. 5
iiwgee ................................................................. 8
abacus.plot

Create an abacus plot

Create an abacus plot, depicting visits per subject over time

Usage

abacus.plot(
  n,
  time,
  id,
  data,
  tmin,
  tmax,
  xlab.abacus = "Time",
  ylab.abacus = "Subject",
  pch.abacus = 16,
  col.abacus = 1
)

Arguments

n the number of subjects to randomly sample. Subjects are sampled without replacement and therefore n must be smaller than the total number of subjects in the dataset

time character string indicating which column of the data contains the time at which the visit occurred

id character string indicating which column of the data identifies subjects

data data frame containing the variables in the model

tmin the smallest time to include on the x-axis

tmax the largest time to include on the x-axis

xlab.abacus the label for the x-axis

ylab.abacus the label for the y-axis

pch.abacus the plotting character for the points on the abacus plot

col.abacus the colour of the rails on the abacus plot
addcensoredrows

Details

This function creates a plot for n randomly sampled individuals from the supplied dataset, with one row per subject and one point per visit. This can be useful for visualising the extent of irregularity in the visit process. For example, with perfect repeated measures data (i.e., no irregularity), the points will line up vertically. With greater irregularity, the points will be randomly scattered over time.

Value

produces a plot depicting observation times for each subject. No values are returned

Examples

```r
library(MEMSS)
data(Phenobarb)
Phenobarb$event <- 1-as.numeric(is.na(Phenobarb$conc))
data <- Phenobarb[Phenobarb$event==1,]
abacus.plot(n=20,time="time",id="Subject",data=data,tmin=0,tmax=16*24,
xlab.abacus="Time in hours",pch=16,col.abacus=gray(0.8))
```

addcensoredrows

Add rows corresponding to censoring times to a longitudinal dataset

Description

Add rows corresponding to censoring times to a longitudinal dataset

Usage

```r
addcensoredrows(data, maxfu, tinvarcols, id, time, event)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>The dataset to which rows are to be added. The data should have one row per</td>
</tr>
<tr>
<td></td>
<td>observation</td>
</tr>
<tr>
<td>maxfu</td>
<td>The maximum follow-up time per subject. If all subjects have the same follow-</td>
</tr>
<tr>
<td></td>
<td>up time, this can be supplied as a single number. Otherwise, maxfu should be</td>
</tr>
<tr>
<td></td>
<td>a dataframe with the first column specifying subject identifiers and the second</td>
</tr>
<tr>
<td></td>
<td>giving the follow-up time for each subject.</td>
</tr>
<tr>
<td>tinvarcols</td>
<td>A vector of column numbers corresponding to variables in data that are time-</td>
</tr>
<tr>
<td></td>
<td>invariant.</td>
</tr>
<tr>
<td>id</td>
<td>character string indicating which column of the data identifies subjects</td>
</tr>
<tr>
<td>time</td>
<td>character string indicating which column of the data contains the time at</td>
</tr>
<tr>
<td></td>
<td>which the visit occurred</td>
</tr>
<tr>
<td>event</td>
<td>character string indicating which column of the data indicates whether or not</td>
</tr>
<tr>
<td></td>
<td>a visit occurred. If every row corresponds to a visit, then this column will</td>
</tr>
<tr>
<td></td>
<td>consist entirely of ones</td>
</tr>
</tbody>
</table>
Value

The original dataset with extra rows corresponding to censoring times

Examples

```r
x <- c(1:3,1:2,1:5)
x0 <- c(rep(2,3),rep(0,2),rep(1,5))
id <- c(rep(1,3),rep(2,2),rep(3,5))
time <- c(0,4,6,2,3,1,3,5,6,7)
event <- c(1,1,1,0,1,0,1,1,1,1)
data <- as.data.frame(cbind(x,id,time,event,x0))
addcensoredrows(data,maxfu=8,id="id",time="time",tincvarcols=5,event="event")
```

```r
x <- c(1:3,1:2,1:5)
x0 <- c(rep(2,3),rep(0,2),rep(1,5))
id <- c(rep(1,3),rep(2,2),rep(3,5))
time <- c(0,4,6,2,3,1,3,5,6,7)
event <- c(1,1,1,0,1,0,1,1,1,1)
data <- as.data.frame(cbind(x,id,time,event,x0))
maxfu.id <- 1:3
maxfu.time <- c(6,5,8)
maxfu <- cbind(maxfu.id,maxfu.time)
maxfu <- as.data.frame(maxfu)
addcensoredrows(data,maxfu=maxfu,id="id",time="time",tincvarcols=5,event="event")
```

---

**iiw**

Given a proportional hazards model for visit intensities, compute inverse-intensity weights.

Description

For a longitudinal dataset subject to irregular observation, use a Cox proportional hazards model for visit intensities to compute inverse intensity weights

Usage

```r
iiw(phfit, data, id, time, first)
```

Arguments

- **phfit**: coxph object for the visit process
- **data**: The dataset featuring longitudinal data subject to irregular observation for which inverse-intensity weights are desired
- **id**: character string indicating which column of the data identifies subjects
- **time**: character string indicating which column of the data contains the time at which the visit occurred
Logical variable. If TRUE, the first observation for each individual is assigned an intensity of 1. This is appropriate if the first visit is a baseline visit at which recruitment to the study occurred; in this case the baseline visit is observed with probability 1.

Value

A vector of inverse-intensity weights for each row of the dataset. The first observation for each subject is assumed to have an intensity of 1.

See Also

Other iiw: iiw.weights(), iiwgee()

Examples

```r
library(nlme)
data(Phenobarb)
library(survival)
library(geepack)
data(Phenobarb)
Phenobarb$event <- 1-as.numeric(is.na(Phenobarb$conc))
data <- lagfn(Phenobarb, lagvars="dose", id="Subject", time="time", lagfirst = 0)
data <- lagfn(data, lagvars="dose.lag", id="Subject", time="time", lagfirst = 0)
data <- lagfn(data, lagvars="dose.lag.lag", id="Subject", time="time", lagfirst = 0)
data$dose.lag[is.na(data$dose.lag)] <- data$dose.lag.lag[is.na(data$dose.lag)]
data$dose.lag.lag[is.na(data$dose.lag.lag)] <- data$dose.lag.lag.lag[is.na(data$dose.lag.lag)]
data <- data[data$event==1,]
data$id <- as.numeric(data$Subject)
data <- data[data$time<16*24,]
data <- lagfn(data, lagvars=c("time","conc"), id="Subject", time="time", lagfirst = NA)
head(data)

mph <- coxph(Surv(time.lag,time,event)~I(conc.lag>0) + conc.lag + cluster(id),data=data)
summary(mph)
data$weight <- iiw(mph,data,"id","time",TRUE)
head(data)
```
Usage

```r
iiw.weights(
  formulaph,
  formulanull = NULL,
  data,
  id,
  time,
  event,
  lagvars,
  invariant = NULL,
  maxfu,
  lagfirst = lagfirst,
  first,
  frailty = FALSE
)
```

Arguments

- `formulaph` the formula for the proportional hazards model for the visit intensity that will be used to derive inverse-intensity weights. The formula should usually use the counting process format (i.e. `Surv(start,stop,event)`). If a frailty model is used, the `cluster(id)` term should appear before other covariates.

- `formulanull` if stabilised weights are to be used, the formula for the null model used to stabilise the weights.

- `data` data frame containing the variables in the model.

- `id` character string indicating which column of the data identifies subjects.

- `time` character string indicating which column of the data contains the time at which the visit occurred.

- `event` character string indicating which column of the data indicates whether or not a visit occurred. If every row corresponds to a visit, then this column will consist entirely of ones.

- `lagvars` a vector of variable names corresponding to variables which need to be lagged by one visit to fit the visit intensity model. Typically time will be one of these variables. The function will internally add columns to the data containing the values of the lagged variables from the previous visit. Values of lagged variables for a subject’s first visit will be set to NA. To access these variables in specifying the proportional hazards formulae, add "lag" to the variable you wish to lag. For example, if time is the variable for time, `time.lag` is the time of the previous visit.

- `invariant` a vector of variable names corresponding to variables in data that are time-invariant. It is not necessary to list every such variable, just those that are invariant and also included in the proportional hazards model.

- `maxfu` the maximum follow-up time(s). If everyone is followed for the same length of time, this can be given as a single value. If individuals have different follow-up times, `maxfu` should have the same number of elements as there are rows of data.
lagfirst  A vector giving the value of each lagged variable for the first time within each subject. This is helpful if, for example, time is the variable to be lagged and you know that all subjects entered the study at time zero.

first  logical variable. If TRUE, the first observation for each individual is assigned an intensity of 1. This is appropriate if the first visit is a baseline visit at which recruitment to the study occurred; in this case the baseline visit is observed with probability 1.

frailty  logical variable. If TRUE, a frailty model is fit to calculate the inverse intensity weights. If FALSE, a marginal semi-parametric model is fit. Frailty models are helpful when fitting semi-parametric joint models.

Details

Given longitudinal data with irregular visit times, fit a Cox proportional hazards model for the visit intensity, then use it to compute inverse-intensity weights

Value

a vector of inverse-intensity weights, ordered on id then time

References


See Also

Other iiw: iiwgee(), iiw()
Other iiw: iiwgee(), iiw()

Examples

library(nlme)
data(Phenobarb)
library(survival)
library(geepack)
Phenobarb$event <- 1-as.numeric(is.na(Phenobarb$conc))
data <- lagfn(Phenobarb, lagvars="dose", id="Subject", time="time", lagfirst = 0)
data <- lagfn(data, lagvars="dose.lag", id="Subject", time="time", lagfirst = 0)
data <- lagfn(data, lagvars="dose.lag.lag", id="Subject", time="time", lagfirst = 0)
data$dose.lag[is.na(data$dose.lag)] <- data$dose.lag.lag[is.na(data$dose.lag)]
data$dose.lag.lag[is.na(data$dose.lag.lag)] <- data$dose.lag.lag.lag[is.na(data$dose.lag.lag)]
data <- data[data$event==1,]
data$id <- as.numeric(data$Subject)
data <- data[data$time<16*24,]
i <- iiw.weights(Surv(time.lag,time,event)-Wt *dose.lag + dose.lag*(I(conc.lag>0) + conc.lag) +
cluster(Subject),id="id",time="time",event="event",data=data,
Fit an inverse-intensity weighted GEE.

Description

Implements inverse-intensity weighted GEEs as first described by Lin, Scharfstein and Rosenheck (2004). A Cox proportional hazards model is applied to the visit intensities, and the hazard multipliers are used to compute inverse-intensity weights. Using the approach described by Buzkova and Lumley (2007) avoids the need to compute the baseline hazard.

Usage

iiwgee(formulagee,
        formulaph,
        formulanull = NULL,
        data,
        id,
        time,
        event,
        family = gaussian,
        lagvars,
        invariant = NULL,
        maxfu,
        lagfirst,
        first)

Arguments

formulagee the formula for the GEE model to be fit. The syntax used is the same as in geeglm
formulaph the formula for the proportional hazards model for the visit intensity that will be used to derive inverse-intensity weights. The formula should usually use the counting process format (i.e. Surv(start,stop,event))
if stabilised weights are to be used, the formula for the null model used to stabilise the weights data frame containing the variables in the model character string indicating which column of the data identifies subjects character string indicating which column of the data contains the time at which the visit occurred character string indicating which column of the data indicates whether or not a visit occurred. If every row corresponds to a visit, then this column will consist entirely of ones family to be used in the GEE fit. See geeglm for documentation a vector of variable names corresponding to variables which need to be lagged by one visit to fit the visit intensity model. Typically time will be one of these variables. The function will internally add columns to the data containing the values of the lagged variables from the previous visit. Values of lagged variables for a subject’s first visit will be set to NA. To access these variables in specifying the proportional hazards formulae, add ".lag" to the variable you wish to lag. For example, if time is the variable for time, time.lag is the time of the previous visit a vector of variable names corresponding to variables in data that are time-invariant. It is not necessary to list every such variable, just those that are invariant and also included in the proportional hazards model the maximum follow-up time(s). If everyone is followed for the same length of time, this can be given as a single value. If individuals have different follow-up times, maxfu should have the same number of elements as there are rows of data A vector giving the value of each lagged variable for the first time within each subject. This is helpful if, for example, time is the variable to be lagged and you know that all subjects entered the study at time zero logical variable. If TRUE, the first observation for each individual is assigned an intensity of 1. This is appropriate if the first visit is a baseline visit at which recruitment to the study occurred; in this case the baseline visit is observed with probability 1.

Details

Let the outcome of interest be $Y$ and suppose that subject $i$ has $j^{th}$ observation at $T_{ij}$. Let $N_i(t)$ be a counting process for the number of observations for subject $i$ up to and including time $t$. Suppose that $N_i$ has intensity $\lambda$ given by

$$\lambda_i(t) = \lambda_0(t) \exp(Z_i(t)\gamma).$$

Then the inverse-intensity weights are

$$\exp(-Z_i(t)\gamma).$$

If $Y_i$ is the vector of observations for subject $i$, to be regressed onto $X_i$ (i.e. $E(Y_i|X_i) = \mu(X_i; \beta)$ with $g(\mu(X_i; beta)) = X_i\beta$, then the inverse-intensity weighted GEE equations are

$$\sum_i \frac{\partial\mu_i}{\partial\beta} V_i^{-1} \Delta_i(Y_i; X_i\beta) = 0$$
where \( \Delta_i \) is a diagonal matrix with \( j^{th} \) entry equal to \( \exp(-Z_i(T_{ij} \gamma)) \) and \( \mathbf{V}_i \) is the working variance matrix. Warning: Due to the way some gee functions incorporate weights, if using inverse-intensity weighting you should use working independence.

**Value**

a list, with the following elements:

- `geefit` the fitted GEE, see documentation for `geeglm` for details
- `phfit` the fitted proportional hazards model, see documentation for `coxph` for details

**References**


**See Also**

Other iiw: `iiw.weights()`, `iiw()`

**Examples**

```r
library(nlme)
data(Phenobarb)
library(survival)
library(geepack)
Phenobarb$event <- 1-as.numeric(is.na(Phenobarb$conc))
data <- lagfn(Phenobarb, lagvars="dose", id="Subject", time="time", lagfirst = 0)
data <- lagfn(data, lagvars="dose.lag", id="Subject", time="time", lagfirst = 0)
data <- lagfn(data, lagvars="dose.lag.lag", id="Subject", time="time", lagfirst = 0)
data$dose.lag[is.na(data$dose.lag)] <- data$dose.lag.lag[is.na(data$dose.lag)]
data$dose.lag.lag[is.na(data$dose.lag)] <- data$dose.lag.lag.lag[is.na(data$dose.lag)]
data <- data[!data$event==1,]
data$id <- as.numeric(data$Subject)
data <- data[!data$time<16*24,]
miwgee <- iiwgee(conc ~ I(time^3) + log(time),Surv(time.lag,time,event)~Wt *dose.lag + dose.lag*(I(conc.lag>0) + conc.lag) +cluster(Subject), formulanull=NULL, id="id", time="time", event="event", data=data, invariant="id", lagvars=c("time", "conc"), maxfu=16*24, lagfirst=0, first=TRUE)
summary(miwgee$geefit)
summary(miwgee$phfit)

# compare to results without weighting
m <- geeglm(conc ~ I(time^3) + log(time), id=Subject, data=data); print(summary(m))
time <- (1:200)
unweighted <- cbind(rep(1,200),time^3,log(time))%*%m$coefficients
weighted <- cbind(rep(1,200),time^3,log(time))%*%miwgee$geefit$coefficients
plot(data$time,data$conc,xlim=c(0,200),pch=16)
```
lagfn

Create lagged versions the variables in data

Description

Create lagged versions the variables in data

Usage

lagfn(data, lagvars, id, time, lagfirst = NA)

Arguments

data The data to be lagged
lagvars The names of the columns in the data to be lagged
id A character indicating which column of the data contains subject identifiers. ids
are assumed to be consecutive integers, with the first subject having id 1

time A character indicating which column of the data contains the times at which each of the observations in data was made

lagfirst A vector giving the value of each lagged variable for the first time within each subject. This is helpful if, for example, time is the variable to be lagged and you know that all subjects entered the study at time zero

Value

The original data frame with lagged variables added on as columns. For example, if the data frame contains a variable named x giving the value of x for each subject i at each visit j, the returned data frame will contain a column named x.lag containing the value of x for subject i at visit j-1. If j is the first visit for subject i, the lagged value is set to NA

Examples

library(nlme)
data(Phenobarb)
head(Phenobarb)

data <- lagfn(Phenobarb,"time","Subject","time")
head(data)
Fit a semi-parametric joint model

Description

Fits a semi-parametric joint model as described by Liang et al. (2009).

Usage

Liang(
  data,
  Yname,
  Xnames,
  Wnames,
  Znames = NULL,
  formulaobs = NULL,
  id,
  time,
  invariant = NULL,
  lagvars = NULL,
  lagfirst = NULL,
  maxfu,
  baseline,
  n.knots = NULL,
  kappa = NULL
)

Arguments

data data frame containing the variables in the model
Yname character string indicating the column containing the outcome variable
Xnames vector of character strings indicating the names of the columns of the fixed effects in the outcome regression model
Wnames vector of character strings indicating the names of the columns of the random effects in the outcome regression model
Znames vector of character strings indicating the names of the columns of the covariates in the visit intensity model
formulaobs formula for the observation intensity model
id character string indicating which column of the data identifies subjects
time character string indicating which column of the data contains the time at which the visit occurred
invariant a vector of variable names corresponding to variables in data that are time-invariant. It is not necessary to list every such variable, just those that are invariant and also included in the visit intensity model
lagvars

A vector of variable names corresponding to variables which need to be lagged by one visit to fit the visit intensity model. Typically time will be one of these variables. The function will internally add columns to the data containing the values of the lagged variables from the previous visit. Values of lagged variables for a subject’s first visit will be set to NA. To access these variables in specifying the proportional hazards formulae, add ".lag" to the variable you wish to lag. For example, if time is the variable for time, time.lag is the time of the previous visit.

lagfirst

A vector giving the value of each lagged variable for the first time within each subject. This is helpful if, for example, time is the variable to be lagged and you know that all subjects entered the study at time zero.

maxfu

The maximum follow-up time per subject. If all subjects have the same follow-up time, this can be supplied as a single number. Otherwise, maxfu should be a dataframe with the first column specifying subject identifiers and the second giving the follow-up time for each subject.

baseline

An indicator for whether baseline (time=0) measurements are included by design. Equal to 1 if yes, 0 if no.

n.knots

Integer giving the number of knots to use in fitting the frailty model. See documentation for frailtyPenal for more details.

kappa

Positive smoothing parameter in the penalized likelihood estimation. See documentation for frailtyPenal for more details.

Details

This function fits a semi-parametric joint model as described in Liang (2009), using a frailty model to estimate the parameters in the visit intensity model.

Value

The regression coefficients corresponding to the fixed effects in the outcome regression model. Closed form expressions for standard errors of the regression coefficients are not available, and Liang et al (2009) recommend obtaining these through bootstrapping.

References


Examples

# replicate simulation in Liang et al.
## Not run:
library(data.table)
library(survival)
datasimi <- function(id){
  X1 <- runif(1,0,1)
  X2 <- rbinom(1,1,0.5)
  Z <- rnorm(1,1,1)
  Z1 <- rnorm(1,Z-1,1)
  gamma <- c(0.5,-0.5)
\[
\beta \leftarrow c(1,-1)
\]
\[
hazard \leftarrow z \cdot \exp(X_1/2 - X_2/2)
\]
\[
C \leftarrow \text{runif}(1,0,5.8)
\]
\[
t \leftarrow 0
\]
\[
tlast \leftarrow t
\]
\[
y \leftarrow t + X_1 - X_2 + Z_1 \cdot X_2 + \text{rnorm}(1,0,1)
\]
\[
wait \leftarrow \text{rexp}(1,\text{hazard})
\]
\[
\text{while}(tlast + \text{wait} < C)\
\{ 
\text{tnew} \leftarrow tlast + \text{wait} \\
\text{y} \leftarrow \text{c(y, tnew + X_1 - X_2 + Z_1 \cdot X_2 + \text{rnorm}(1,0,1))} \\
\text{t} \leftarrow \text{c(t, tnew)} \\
\text{tlast} \leftarrow \text{tnew} \\
\text{wait} \leftarrow \text{rexp}(1,\text{hazard})
\}
\]
\[
datai \leftarrow \text{list(id=rep(id, length(t)), t=t, y=y,} \\
\text{X1=rep(X1, length(t)), X2=rep(X2, length(t)), C=rep(C, length(t)))}
\]
\[
return(datai)
\}
\]
sim1 <- function(it, nsubj) {
\[
data <- lapply(1:nsubj, datasimi)
\]
\[
data <- \text{as.data.frame(rbindlist(data))}
\]
\[
data$event <- 1
\]
\[
C \leftarrow \text{tapply(data$C, data$id, mean)}
\]
\[
tapply(data$C, data$id, sd)
\]
\[
maxfu <- \text{cbind(1:nsubj, C)}
\]
\[
maxfu <- \text{as.data.frame(maxfu)}
\]
\[
res <- Liang(data=data, id="id", time="t", Yname="y",} \\
\text{Xnames=c("X1", "X2"),} \\
\text{Wnames=c("X2"), Znames=c("X1", "X2"), formulaobs=Surv(t.lag, t, event)-X1} \\
\text{+ X2+ cluster(id), invariant=c} \\
\text{("id", "X1", "X2"), lagvars="t", lagfirst=NA, maxfu=maxfu,} \\
\text{baseline=1, n.knots=6, kappa=10000)}
\]
\[
return(res)
\} \\
\]
# change n to 500 to replicate results of Liang et al.
\]
n <- 10
\]
s <- lapply(1:n, sim1, nsubj=200)
\]
\[
smat <- matrix(unlist(s), byrow=TRUE, ncol=2)
\]
\[
apply(smat, 2, mean)
\]
## End(Not run)

---

**Multiple outputation for longitudinal data subject to irregular observation.**

**Description**

Multiple outputation is a procedure whereby excess observations are repeatedly randomly sampled and discarded. The method was originally developed to handle clustered data where cluster size is
informative, for example when studying pups in a litter. In this case, analysis that ignores cluster size results in larger litters being over-represented in a marginal analysis. Multiple outputation circumvents this problem by randomly selecting one observation per cluster. Multiple outputation has been further adapted to handle longitudinal data subject to irregular observation; here the probability of being retained on any given outputation is inversely proportional to the visit intensity. This function creates multiply outputted datasets, analyses each separately, and combines the results to produce a single estimate.

Usage

```r
mo(
  noutput,
  fn,
  data,
  weights,
  singleobs,
  id,
  time,
  keep.first,
  var = TRUE,
  ...
)
```

Arguments

- `noutput`: the number of outputations to be used
- `fn`: the function to be applied to the outputted datasets. `fn` should return a vector or scalar; if `var=TRUE` the second column of `fn` should be an estimate of standard error.
- `data`: the original dataset on which multiple outputation is to be performed
- `weights`: the weights to be used in the outputation, i.e. the inverse of the probability that a given observation will be selected in creating an outputted dataset. Ignored if `singleobs=TRUE`
- `singleobs`: logical variable indicating whether a single observation should be retained for each subject
- `id`: character string indicating which column of the data identifies subjects
- `time`: character string indicating which column of the data contains the time at which the visit occurred
- `keep.first`: logical variable indicating whether the first observation should be retained with probability 1. This is useful if the data consists of an observation at baseline followed by follow-up at stochastic time points.
- `var`: logical variable indicating whether `fn` returns variances in addition to point estimates
- `...`: other arguments to `fn`. 
Value

a list containing the multiple outputation estimate of the function fn applied to the data, its standard
error, and the relative efficiency of using noutput outputations as opposed to an infinite number

References

- Follmann D, Proschan M, Leifer E. Multiple outputation: inference for complex clustered data
  by averaging analyses from independent data. Biometrics 2003; 59:420-429
- Pullenayegum EM. Multiple outputation for the analysis of longitudinal data subject to irregular
  observation. Statistics in Medicine (in press)

See Also

Other mo: outputation()

Examples

library(nlme)
data(Phenobarb)
library(survival)
library(geepack)

Phenobarb$event <- 1-as.numeric(is.na(Phenobarb$conc))
data <- lagfn(Phenobarb, lagvars="dose", id="Subject", time="time", lagfirst = 0)
data <- lagfn(data, lagvars="dose.lag", id="Subject", time="time", lagfirst = 0)
data <- lagfn(data, lagvars="dose.lag.lag", id="Subject", time="time", lagfirst = 0)
data$ dose.lag[is.na(data$dose.lag)] <- data$dose.lag.lag[is.na(data$dose.lag)]
data$ dose.lag.lag[is.na(data$dose.lag.lag)] <- data$dose.lag.lag.lag[is.na(data$dose.lag.lag)]
data <- data[data$event==1,]
data$id <- as.numeric(data$Subject)
data <- data[data$time<16*24,]
i <- iiw.weights(Surv(time.lag,time,event)~Wt*dose.lag +
dose.lag*I(conc.lag>0) + conc.lag) +
cluster(Subject),id="id",time="time",event="event",data=data,
invariant="Subject",lagvars=c("time","conc"),maxfu=16*24,
lagfirst=0,first=TRUE)
wt <- i$iiw.weight
wt[wt>quantile(i$iiw.weight,0.95)] <- quantile(i$iiw.weight,0.95)
data$wt <- wt
reg <- function(data){
est <- summary(geeglm(conc~I(time^3) + log(time), id=id,data=data))$coefficients[,1:2]
est <- data.matrix(est)
return(est)
}

mo(20,reg,data,wt,singleobs=FALSE,id="id",time="time",keep.first=FALSE)
# On outputation, the dataset contains small numbers of observations per subject
# and hence the GEE sandwich variance estimate underestimates variance; this is why
# the outputation-based variance estimate fails. This can be remedied by using a
# sandwich variance error correction (e.g. Fay-Graubard, Mancl-DeRouen).

## Description

Multiple outputation is a procedure whereby excess observations are repeatedly randomly sampled and discarded. The method was originally developed to handle clustered data where cluster size is informative, for example when studying pups in a litter. In this case, analysis that ignores cluster size results in larger litters being over-represented in a marginal analysis. Multiple outputation circumvents this problem by randomly selecting one observation per cluster. Multiple outputation has been further adapted to handle longitudinal data subject to irregular observation; here the probability of being retained on any given outputation is inversely proportional to the visit intensity. This function creates a single outputted dataset.

## Usage

```r
outputation(data, weights, singleobs, id, time, keep.first)
```

## Arguments

- `data`: the original dataset on which multiple outputation is to be performed
- `weights`: the weights to be used in the outputation, i.e. the inverse of the probability that a given observation will be selected in creating an outputted dataset. Ignored if `singleobs=TRUE`
- `singleobs`: logical variable indicating whether a single observation should be retained for each subject
- `id`: character string indicating which column of the data identifies subjects
- `time`: character string indicating which column of the data contains the time at which the visit occurred
- `keep.first`: logical variable indicating whether the first observation should be retained with probability 1. This is useful if the data consists of an observation at baseline followed by follow-up at stochastic time points.

## Value

the outputted dataset.

## References

- Follmann D, Proschan M, Leifer E. Multiple outputation: inference for complex clustered data by averaging analyses from independent data. Biometrics 2003; 59:420-429
- Pullenayegum EM. Multiple outputation for the analysis of longitudinal data subject to irregular observation. Statistics in Medicine (in press).
See Also

Other mo: \texttt{mo()}

Examples

```r
library(nlme)
data(Phenobarb)
library(survival)
library(geepack)
Phenobarb$event <- 1-as.numeric(is.na(Phenobarb$conc))
data <- lagfn(Phenobarb, lagvars="dose", id="Subject", time="time", lagfirst = 0)
data <- lagfn(data, lagvars="dose.lag", id="Subject", time="time", lagfirst = 0)
data <- lagfn(data, lagvars="dose.lag.lag", id="Subject", time="time", lagfirst = 0)
data$event[is.na(data$event)] <- data$event[is.na(data$event)]
data$event[is.na(data$event)] <- data$event[is.na(data$event)]
data <- data[data$event==1,]
data$id <- as.numeric(data$Subject)
data <- data[data$time<16*24,]
i <- iiw.weights(Surv(time.lag,time,event)~Wt *dose.lag + dose.lag*(I(conc.lag>0) + conc.lag) + cluster(Subject),id="id",time="time",event="event", data=data, invariant="Subject",lagvars=c("time","conc"),maxfu=16*24,lagfirst=0,first=TRUE)
data$weight <- i$iiw.weight
head(data)
data.output1 <- outputation(data,data$weight,singleobs=FALSE, id="id",time="time",keep.first=FALSE)
head(data.output1)
data.output2 <- outputation(data,data$weight,singleobs=FALSE, id="id",time="time",keep.first=FALSE)
head(data.output2)
data.output3 <- outputation(data,data$weight,singleobs=FALSE, id="id",time="time",keep.first=FALSE)
head(data.output3)
# Note that the outputted dataset varies with each command run; outputation is done at random
```
Index

abacus.plot, 2
addcensoredrows, 3

iiw, 4, 7, 10
iiw.weights, 5, 5, 10
iiwgee, 5, 7, 8

lagfn, 11
Liang, 12

mo, 14, 18

outputation, 16, 17