Package ‘IrregLong’

May 23, 2020

Type Package

Title Analysis of Longitudinal Data with Irregular Observation Times

Version 0.3.1

Date 2020-05-15

Author Eleanor Pullenayegum

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


Depends R (>= 2.10)

Imports survival, geepack, frailtypack, data.table

License GPL-3

RoxygenNote 7.1.0

Suggests knitr, rmarkdown, nlme, MEMSS

VignetteBuilder knitr

LazyData true

Language en-GB

NeedsCompilation no

Repository CRAN

Date/Publication 2020-05-23 04:20:02 UTC

R topics documented:

abacus.plot ......................................................... 2
addcensoredrows .................................................. 3
extent.of.irregularity ............................................ 4
iiw ................................................................. 6
iiw.weights ........................................................ 7
Create an abacus plot

**Description**

Create an abacus plot, depicting visits per subject over time.

**Usage**

```r
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

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))

Description

Add rows corresponding to censoring times to a longitudinal dataset

Usage

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

Arguments

data The dataset to which rows are to be added. The data should have one row per observation
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.
tinvarcols A vector of column numbers corresponding to variables in data that are time-invariant.
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
extent.of.irregularity

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(1,1,0,1,0,1,1,1,1,1)
data <- as.data.frame(cbind(x,id,time,event,x0))
addcensoredrows(data,maxfu=8,id="id",time="time",tinvarcols=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(1,1,0,1,0,1,1,1,1,1)
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",tinvarcols=5,event="event")
```

extent.of.irregularity

Measures of extent of visit irregularity Provides visual and numeric measures of the extent of irregularity in observation times in a longitudinal dataset

Description

Measures of extent of visit irregularity Provides visual and numeric measures of the extent of irregularity in observation times in a longitudinal dataset

Usage

```r
extent.of.irregularity(
  data,
  time = "time",
  id = "id",
  scheduledtimes = NULL,
  cutpoints = NULL,
  ncutpts = NULL,
  maxfu = NULL,
  plot = FALSE,
)```
Arguments

data The data containing information on subject identifiers and visit times
time A character indicating which column of the data contains the times at which each of the observations in data was made
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
scheduledtimes For studies with protocol-specified visit times, a vector of these times. Defaults to NULL, in which case it is assumed that there are no protocolized visit times
cutpoints For studies with scheduled visit times, an array of dimension ncutpts by length(scheduledtimes) by 2 giving, for ncutpts sets of left and right cutpoints for each protocolized scheduled visit times. The left-hand cutpoints correspond to cutpoints[,1] and the right-hand cutpoints to cutpoints[,2]. Defaults to NULL, in which case cutpoints are computed as described below.
ncutpts The number of sets of cutpoints to consider
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.
plot logical parameter indicating whether plots should be produced.
legendx The x-coordinate for the position of the legend in the plot of mean proportion of individuals with 0, 1 and >1 visit per bin.
legendy The y-coordinate for the position of the legend in the plot of mean proportion of individuals with 0, 1 and >1 visit per bin.
formula For studies without protocolized visit times, the formula for the null counting process model for the visit times
tau The maximum time of interest

Details

This function provides plots and a numerical summary of the extent of irregularity in visit times. For any given set of cutpoints, it computes the proportion of individuals with 0, 1 and >1 observation(s) in each bin, then takes the mean over bins. The sizes of the bins are varied and these proportions are plotted against bin size. In addition, then mean proportion of individuals with >1 visit per bin is plotted vs. the mean proportion of individuals with 0 visits per bin, and the area under the curve is calculated (AUC). An AUC of 0 represents perfect repeated measures while a Poisson Process has an AUC of 0. If cutpoints are not supplied, they are computed as follows: (a) for studies with protocolized visit times, the left- and right-hand cutpoints are positioned at the protocolized time minus (or plus, for right-hand cutpoints) (1,...,ncutpts)/ncutpts times the gap to the previous (or next, respectively) protocolized visit time; (b) for studies with no protocolized visit times, cutpoints
are calculated by finding, for each \( j \) in 1,...,\( n_{\text{cutpts}} \) the largest times for which the cumulative hazard is less than \( j \) divided by the cumulative hazard evaluated at the maximum time of interest. This corresponds to choosing cutpoints such that the expected number of visits per bin is roughly equal within each set.

Value

a list with counts equal to a 3-dimensional by \( n_{\text{cutpts}} \) matrix giving, for each set of cutpoints, the mean proportion of individuals with zero, 1 and \( >1 \) visits per bin, and AUC, the area under the curve of the plot of the proportion of individuals with \( >1 \) visit per bin vs. the proportion of individuals with 0 visits per bin.

Examples

```r
library(nlme)
library(survival)
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$Subject)
counts <- extent.of.irregularity(data,time="time",id="id",scheduledtimes=NULL, cutpoints=NULL,ncutpts=10, maxfu=16*24,plot=TRUE,legendx=NULL,legendy=NULL,formula=Surv(time.lag,time,event)~1,tau=16*24)
counts$counts
counts$auc
```

---

**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

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.

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)
```

---

**iiw.weights**  Compute inverse-intensity weights.

**Description**
Since the vector of weights is ordered on id and time, if you intend to merge these weights onto
your original dataset it is highly recommended that you sort the data before running iiw.weights.
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.
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
A 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
A 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
```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.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,)
```
iiwgee

**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**

```r
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)`)
- Other arguments passed to `geeglm` and `coxph` should be used in the `formulaph`.
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 $V_i$ 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.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,]
miiwgee <- 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(miiwgee$geefit)
summary(miiwgee$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$coefficients
weighted <- cbind(rep(1,200),time^3,log(time))%%miiwgee$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**

```r
library(nlme)
data(Phenobarb)
head(Phenobarb)

data <- lagfn(Phenobarb,"time","Subject","time")
head(data)
```
Liang

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, Xfn = NULL, Wfn = 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.

Xfn  A function that takes as its first argument the subject identifier and has time as its second argument, and returns the value of X for the specified subject at the specified time.

Wfn  A function that takes as its first argument the subject identifier and has time as its second argument, and returns the value of W for the specified subject at the specified time.

...  other arguments to Xfn and Yfn

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.

The Liang method requires a value of X and W for every time over the observation period. If Xfn is left as NULL, then the Liang function will use, for each subject and for each time t, the values of X and W at the observation time closest to t.

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)
datasiimi <- function(id){
X1 <- runif(1,0,1)
X2 <- rbinom(1,1,0.5)
Z <- rgamma(1,1,1)
Z1 <- rnorm(1,Z-1,1)
gamma <- c(0.5,-0.5)
beta <- c(1,-1)
hazard <- Z*exp(X1/2 - X2/2)
C <- runif(1,0,5.8)
t <- 0
tlast <- t
y <- t + X1-X2 + Z1*X2 + rnorm(1,0,1)
wait <- rexp(1,hazard)
while(tlast+wait<C){
  tnew <- tlast+wait
  y <- c(y,tnew + X1-X2 + Z1*X2 + rnorm(1,0,1))
t <- c(t,tnew)
tlast <- tnew
  wait <- rexp(1,hazard)
}
datai <- list(id=rep(id,length(t)),t=t,y=y,
  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 <- as.data.frame(rbindlist(data))
data$event <- 1
C <- tapply(data$C,data$id,mean)
tapply(data$C,data$id,sd)
maxfu <- cbind(1:nsubj,C)
maxfu <- as.data.frame(maxfu)
res <- Liang(data=data, id="id",time="t",Yname="y",
  Xnames=c("X1","X2"),
  Wnames=c("X2"),Znames=c("X1","X2"), formulaobs=Surv(t.lag,t,event)~X1 + X2+ cluster(id),invariant=c
("id","X1","X2"),lagvars="t",lagfirst=NA,maxfu=maxfu,
  baseline=1,n.knots=6, kappa=10000)
return(res)
}
# change n to 500 to replicate results of Liang et al.
n <- 10
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

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$dose.lag[is.na(data$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,]
i <- iwi.weights(Surv(time.lag,time,event)-Wt *dose.lag + dose.lag*(I(conc.lag>0) + conc.lag) +
outputation

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).

---

outputation  Create an outputted dataset for use with multiple outputation.

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

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: mo()

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)
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

extent.of.irregularity, 4

iiw, 6, 9, 12
iiw.weights, 7, 7, 12
iiwgee, 7, 9, 10

lagfn, 13
Liang, 14

mo, 17, 20

outputation, 18, 19