Package 'IrregLong'

February 11, 2019

Type Package

Title Analysis of Longitudinal Data with Irregular Observation Times

Version 0.1.0

Date 2019-01-28

Author Eleanor Pullenayegum

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

Description Analysis of longitudinal data for which the times of observation are random variables that are potentially associated with the outcome process. The package includes inverse-intensity weighting methods (Lin H, Scharfstein DO, Rosenheck RA (2004) <doi:10.1111/j.1467-9868.2004.b5543.x>) and multiple outputation (Pullenayegum EM (2016) <doi:10.1002/sim.6829>).

Depends R (>= 2.10)

Imports survival, geepack, frailtypack

License GPL-3

RoxygenNote 6.1.1

Suggests knitr, rmarkdown, nlme, MEMSS

VignetteBuilder knitr

LazyData true

Language en-GB

NeedsCompilation no

Repository CRAN

Date/Publication 2019-02-11 14:03:20 UTC

R topics documented:

`addcensoredrows` *Add rows corresponding to censoring times to a longitudinal dataset*

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

Value

The original dataset with extra rows corresponding to censoring times

Examples

```
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",tinvarcols=5,event="event")
```

```
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",tinvarcols=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

```
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

```
library(nlme)
data(Phenobarb)
library(survival)
library(geepack)
Phenobarb$id <- as.numeric(Phenobarb$Subject)
Phenobarb$event <- as.numeric(is.finite(Phenobarb$conc))
Phenobarb.conc <- Phenobarb[is.finite(Phenobarb$conc),]
Phenobarb.conc <- lagfn(Phenobarb.conc,c("time","conc"),"id","time")

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

`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

```
iiw.weights(formulaph, formulanull = NULL, data, id, time, event,
  lagvars, invariant, 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	The value of the 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

- Lin H, Scharfstein DO, Rosenheck RA. Analysis of Longitudinal data with Irregular, Informative Follow-up. Journal of the Royal Statistical Society, Series B (2004), 66:791-813
- Buzkova P, Lumley T. Longitudinal data analysis for generalized linear models with follow-up dependent on outcome-related variables. The Canadian Journal of Statistics 2007; 35:485-500.

See Also

Other iiw: iiwgee, iiw

Other iiw: iiwgee, iiw

Examples

```
library(nlme)
data(Phenobarb)
library(survival)
library(geepack)
Phenobarb$id <- as.numeric(Phenobarb$Subject)
Phenobarb$event <- as.numeric(is.finite(Phenobarb$conc))
Phenobarb.conc <- Phenobarb[is.finite(Phenobarb$conc),]
i <- iiw.weights(Surv(time.lag,time,event)~I(conc.lag>0) + conc.lag + cluster(Subject),
id="id",time="time",event="event",data=Phenobarb.conc,invariant="Subject",
lagvars=c("time","conc"),maxfu=NULL,lagfirst=0,first=TRUE)
Phenobarb.conc$weight <- i$iiw.weight
summary(i$m)
# can use to fit a weighted GEE
mw <- geeglm(conc ~ time*log(time) , id=Subject, data=Phenobarb.conc, weights=weight)
summary(mw)
# agrees with results through the single command iiwgee
miiwgee <- iiwgee(conc ~ time*log(time),
Surv(time.lag,time,event)~I(conc.lag>0) + conc.lag + cluster(id),
formulanull=NULL,id="id",time="time",event="event",data=Phenobarb.conc,invariant="id",
lagvars=c("time","conc"),maxfu=NULL,lagfirst=0,first=TRUE)
summary(miiwgee$geefit)
```

`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

```
iiwgee(formulagee, formulaph, formulanull = NULL, data, id, time, event,
  family = gaussian, lagvars, invariant, maxfu, lagfirst = NA, 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))
`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
family	family to be used in the GEE fit. See geeglm for documentation
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	The value of the 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.

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 λ 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 Δ_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

- Lin H, Scharfstein DO, Rosenheck RA. Analysis of Longitudinal data with Irregular, Informative Follow-up. Journal of the Royal Statistical Society, Series B (2004), 66:791-813
- Buzkova P, Lumley T. Longitudinal data analysis for generalized linear models with follow-up dependent on outcome-related variables. The Canadian Journal of Statistics 2007; 35:485-500.

See Also

Other iiw: iiw.weights, iiw

Examples

```
library(nlme)
data(Phenobarb)
library(survival)
library(geepack)
Phenobarb$id <- as.numeric(Phenobarb$Subject)
Phenobarb$event <- as.numeric(is.finite(Phenobarb$conc))
Phenobarb.conc <- Phenobarb[is.finite(Phenobarb$conc),]
miiwgee <- iiwgee(conc ~ time*log(time),
Surv(time.lag,time,event)~I(conc.lag>0) + conc.lag + cluster(id),
formulanull=NULL,id="id",time="time",event="event",data=Phenobarb.conc,invariant="id",
lagvars=c("time","conc"),maxfu=NULL,lagfirst=0,first=TRUE)
summary(miiwgee$geefit)
summary(miiwgee$phfit)

# compare to results without weighting
m <- geeglm(conc ~ time*log(time) , id=Subject, data=Phenobarb); print(summary(m))
time <- (1:200)
unweighted <- cbind(rep(1,200),time,log(time),time*log(time))%*%m$coefficients
weighted <- cbind(rep(1,200),time,log(time),time*log(time))%*%miiwgee$geefit$coefficients
plot(Phenobarb$time,Phenobarb$conc,xlim=c(0,200),pch=16)
lines(time,unweighted,type="l")
lines(time,weighted,col=2)
legend (0,60,legend=c("Unweighted","Inverse-intensity weighted"),col=1:2,bty="n",lty=1)
```

`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`	The value of the 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)
```

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, id, time, maxfu, baseline)
```

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
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
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.

Details

This function is designed to be used in conjunction with multiple outputation and hence assumes no fixed effects in the visit process model. The visit process model thus contains a baseline hazard and a random effect only.

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

Liang Y, Lu W, Ying Z. Joint modelling and analysis of longitudinal data with informative observation times. Biometrics 2009; 65:377-384.

mo *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

- Hoffman E, Sen P, Weinberg C. Within-cluster resampling. Biometrika 2001; 88:1121-1134
- 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$id <- as.numeric(Phenobarb$Subject)
Phenobarb$event <- as.numeric(is.finite(Phenobarb$conc))
Phenobarb.conc <- Phenobarb[is.finite(Phenobarb$conc),]
i <- iiw.weights(Surv(time.lag,time,event)~I(conc.lag>0) + conc.lag + cluster(Subject),
id="id",time="time",event="event",data=Phenobarb.conc,invariant="Subject",
lagvars=c("time","conc"),maxfu=NULL,lagfirst=0,first=TRUE)
Phenobarb.conc$weight <- i$iiw.weight
reg <- function(data){
  return(data.matrix(summary(geeglm(conc ~ time*log(time) ,
  id=Subject, data=data))$coefficients[,1:2]))
 }

mo(20,reg,Phenobarb.conc,Phenobarb.conc$weight,singleobs=FALSE,id="id",time="time",keep.first=FALSE)
# does not yield valid variance estimates
# once thinned the dataset contains fewer than 30 subjects for most outputations,
# so the sandwich variance estimate from the GEE is too small
```

`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

- Hoffman E, Sen P, Weinberg C. Within-cluster resampling. Biometrika 2001; 88:1121-1134
- 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$id <- as.numeric(Phenobarb$Subject)
Phenobarb$event <- as.numeric(is.finite(Phenobarb$conc))
Phenobarb.conc <- Phenobarb[is.finite(Phenobarb$conc),]
i <- iiw.weights(Surv(time.lag,time,event)~I(conc.lag>0) + conc.lag + cluster(Subject),
id="Subject",time="time",event="event",data=Phenobarb.conc,invariant="Subject",
lagvars=c("time","conc"),maxfu=NULL,lagfirst=0,first=TRUE)
Phenobarb.conc$weight <- i$iiw.weight
head(Phenobarb.conc)
data.output1 <-   outputation(Phenobarb.conc,Phenobarb.conc$weight,singleobs=FALSE,
id="id",time="time",keep.first=FALSE)
head(data.output1)
data.output2 <-   outputation(Phenobarb.conc,Phenobarb.conc$weight,singleobs=FALSE,
id="id",time="time",keep.first=FALSE)
head(data.output2)
data.output3 <-   outputation(Phenobarb.conc,Phenobarb.conc$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