Package ‘eventglm’

March 3, 2025

Type Package

Title Regression Models for Event History Outcomes

Version 1.4.5

Date 2025-02-13

Description A user friendly, easy to understand way of doing event history regression for marginal estimands of interest, including the cumulative incidence and the restricted mean survival, using the pseudo observation framework for estimation. For a review of the methodology, see Andersen and Pohar Perme (2010) <doi:10.1177/0962280209105020> or Sachs and Gabriel (2022) <doi:10.18637/jss.v102.i09>. The interface uses the well known formulation of a generalized linear model and allows for features including plotting of residuals, the use of sampling weights, and corrected variance estimation.

License GPL-3

URL `https://sachsmc.github.io/eventglm/`

BugReports `https://github.com/sachsmc/eventglm/issues/`

Encoding UTF-8

LazyData true

Suggests testthat, prodlim, knitr, rmarkdown, rio, data.table

RoxygenNote 7.3.2

Imports survival, sandwich, stats, geepack

Depends R (>= 2.10)

VignetteBuilder knitr

NeedsCompilation yes

Author Michael C Sachs [aut, cre],
Erin E Gabriel [aut],
Morten Overgaard [ctb] (Corrected variance calculation),
Thomas A Gerds [ctb] (Fast computation of leave one out cumulative incidence),
Terry Therneau [ctb] (Restricted mean computation)

Maintainer Michael C Sachs <sachsmc@gmail.com>

Repository CRAN

Date/Publication 2025-03-03 10:20:02 UTC

Contents

`calc_ipcw_pos` *Compute inverse probability of censoring weights pseudo observations*

Description

Compute inverse probability of censoring weights pseudo observations

Usage

```
calc_ipcw_pos(mr, time, causen, type, ipcw.method, Gi)
```

Arguments

`mr`	Model response object returned by Surv
`time`	Max time
`causen`	Cause of interest (numeric)
`type`	Outcome type, "cuminc", "survival", or "rmean"
`ipcw.method`	"binder" or "hajek"
`Gi`	vector of estimated censoring probabilities

`check_mod_cens` *Error check censoring model*

Description

Censoring model must take the same named arguments as the predefined modules (though they do not all have to be used), and return a vector of pseudo observations.

Usage

```
check_mod_cens(model.censoring)
```

Arguments

`model.censoring`	censoring model specification as character or function

`colon` *Chemotherapy for Stage B/C colon cancer*

Description

These are data from one of the first successful trials of adjuvant chemotherapy for colon cancer. Levamisole is a low-toxicity compound previously used to treat worm infestations in animals; 5-FU is a moderately toxic (as these things go) chemotherapy agent. There are only one record per patient for the death outcome (or censoring). This is redistributed from the survival package, with a small modification to include only the death outcome.

Usage

```
colon
```

Format

A data frame with 929 rows and 17 variables:

id id

study 1 for all patients

rx Treatment - Obs(ervation), Lev(amisole), Lev(amisole)+5-FU

sex 1=male

age in years

obstruct obstruction of colon by tumour

perfor perforation of colon

adhere adherence to nearby organs

nodes number of lymph nodes with detectable cancer

time days until death or censoring

status censoring status

differ differentiation of tumour (1=well, 2=moderate, 3=poor)

extent Extent of local spread (1=submucosa, 2=muscle, 3=serosa, 4=contiguous structures)

surg time from surgery to registration (0=short, 1=long)

node4 more than 4 positive lymph nodes

etype event type: 1=recurrence,2=death

event event indicator: censored, death

See Also

colon

`confint.pseudoglm` *Confidence Intervals for pseudoglm Model Parameters*

Description

Computes Wald confidence intervals for one or more parameters in a fitted model. Users can specify the type of variance estimate used, with the default being the robust sandwich variance estimator.

Usage

```
## S3 method for class 'pseudoglm'
confint(object, parm, level = 0.95, type = "robust", ...)
```

Arguments

`object`	A fitted model object from cumincglm or rmeanglm
`parm`	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
`level`	the confidence level required.
`type`	The type of variance estimate to use, see vcov.pseudoglm
`...`	Not used

Value

A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1-level)/2 and 1 - (1-level)/2 in

Examples

```
cumincipcw <- cumincglm(survival::Surv(etime, event) ~ age + sex,
         time = 200, cause = "pcm", link = "identity",
         model.censoring = "independent", data = mgus2)
confint(cumincipcw)
```

`cumincglm` *Generalized linear models for cumulative incidence*

Description

Using pseudo observations for the cumulative incidence, this function then runs a generalized linear model and estimates the parameters representing contrasts in the cumulative incidence at a particular set of times (specified by the `time` argument) across covariate values. The link function can be "identity" for estimating differences in the cumulative incidence, "log" for estimating ratios, and any of the other link functions supported by quasi.

Usage

```
cumincglm(
  formula,
  time,
  cause = 1,
  link = "identity",
  model.censoring = "independent",
  formula.censoring = NULL,
  ipcw.method = "binder",
  data,
  survival = FALSE,
  weights,
```

```
    subset,
    na.action,
    offset,
    control = list(...),
    model = FALSE,
    x = TRUE,
    y = TRUE,
    singular.ok = TRUE,
    contrasts = NULL,
    id,
    ...
)
```

Arguments

`formula`	A formula specifying the model. The left hand side must be a Surv object specifying a right censored survival or competing risks outcome. The status indicator, normally 0=alive, 1=dead. Other choices are TRUE/FALSE (TRUE = death) or 1/2 (2=death). For competing risks, the event variable will be a factor, whose first level is treated as censoring. The right hand side is the usual linear combination of covariates. If there are multiple time points, the special term "tve(.)" can be used to specify that the effect of the variable inside the parentheses will be time varying. In the output this will be represented as the interaction between the time points and the variable.
`time`	Numeric vector specifying the times at which the cumulative incidence or survival probability effect estimates are desired.
`cause`	Numeric or character constant specifying the cause indicator of interest.
`link`	Link function for the cumulative incidence regression model.
`model.censoring`	Type of model for the censoring distribution. Options are "stratified", which computes the pseudo-observations stratified on a set of categorical covariates, "aareg" for Aalen's additive hazards model, and "coxph" for Cox's proportional hazards model. With those options, we assume that the time to event and event indicator are conditionally independent of the censoring time, and that the censoring model is correctly specified. If "independent", we assume completely independent censoring, i.e., that the time to event and covariates are independent of the censoring time. the censoring time is independent of the covariates in the model. Can also be a custom function, see Details and the "Extending eventglm" vignette.
`formula.censoring`	A one sided formula (e.g., `~ x1 + x2`) specifying the model for the censoring distribution. If NULL, uses the same mean model as for the outcome. Missing values in any covariates for the censoring model will cause an error.
`ipcw.method`	Which method to use for calculation of inverse probability of censoring weighted pseudo observations. "binder" the default, uses the number of observations as the denominator, while the "hajek" method uses the sum of the weights as the denominator.

`data`	Data frame in which all variables of formula can be interpreted.
`survival`	Set to TRUE to use survival (one minus the cumulative incidence) as the outcome. Not available for competing risks models.
`weights`	an optional vector of 'prior weights' to be used in the fitting process. Should be NULL or a numeric vector found in data.
`subset`	an optional vector specifying a subset of observations to be used in the fitting process.
`na.action`	a function which indicates what should happen when the data contain `NA`s. The default is set by the `na.action` setting of options, and is na.fail if that is unset. The 'factory-fresh' default is na.omit. Another possible value is NULL, no action. Value na.exclude can be useful.
`offset`	this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases. One or more offset terms can be included in the formula instead or as well, and if more than one is specified their sum is used. See model.offset. If length(time) > 1, then any offset terms must appear in the formula.
`control`	a list of parameters for controlling the fitting process. This is passed to glm.control.
`model`	a logical value indicating whether model frame should be included as a component of the returned value.
`x`	logical value indicating whether the model matrix used in the fitting process should be returned as components of the returned value.
`y`	logical value indicating whether the response vector (pseudo-observations) used in the fitting process should be returned as components of the returned value.
`singular.ok`	logical; if FALSE a singular fit is an error.
`contrasts`	an optional list. See the contrasts.arg of model.matrix.default.
`id`	An optional integer vector of subject identifiers, found in data. If this is present, then generalized estimating equations will be used to fit the model. This can be used, for example, if there are multiple observations per individual represented as multiple rows in data.
`...`	Other arguments passed to glm.fit

Details

The argument "model.censoring" determines how the pseudo observations are calculated. This can be the name of a function or the function itself, which must have arguments "formula", "time", "cause", "data", "type", "formula.censoring", and "ipcw.method". If it is the name of a function, this code will look for a function with the prefix "pseudo_" first, to avoid clashes with related methods such as coxph. The function then must return a vector of pseudo observations, one for each subject in data which are used in subsequent calculations. For examples of the implementation, see the "pseudo-modules.R" file, or the vignette "Extending eventglm".

Value

A pseudoglm object, with its own methods for print, summary, and vcov. It inherits from glm, so predict and other glm methods are supported.

Examples

```
    cumincipcw <- cumincglm(Surv(etime, event) ~ age + sex,
         time = 200, cause = "pcm", link = "identity",
         model.censoring = "independent", data = mgus2)
# stratified on only the categorical covariate
     cumincipcw2 <- cumincglm(Surv(etime, event) ~ age + sex,
                         time = 200, cause = "pcm", link = "identity",
                         model.censoring = "stratified",
                         formula.censoring = ~ sex, data = mgus2)
# multiple time points
cuminct2 <- cumincglm(Surv(etime, event) ~ age + sex,
         time = c(50, 100, 200), cause = "pcm", link = "identity",
         model.censoring = "independent", data = mgus2)
 cuminct3 <- cumincglm(Surv(etime, event) ~ age + tve(sex),
         time = c(50, 100, 200), cause = "pcm", link = "identity",
         model.censoring = "independent", data = mgus2)
```

`eventglm` *Regression Models for Event History Outcomes*

Description

A user friendly, easy to understand way of doing event history regression for marginal estimands of interest, including the cumulative incidence and the restricted mean survival, using the pseudo observation framework for estimation. The interface uses the well known formulation of a generalized linear model and allows for features including plotting of residuals, the use of sampling weights, and corrected variance estimation.

Author(s)

Maintainer: Michael C Sachs <`sachsmc@gmail.com`>

Authors:

- Erin E Gabriel

Other contributors:

- Morten Overgaard (Corrected variance calculation) [contributor]
- Thomas A Gerds (Fast computation of leave one out cumulative incidence) [contributor]
- Terry Therneau (Restricted mean computation) [contributor]

References

Sachs MC, Gabriel EE (2022). "Event History Regression with Pseudo-Observations: Computational Approaches and an Implementation in R." Journal of Statistical Software, 102(9), 1-34. <doi:10.18637/jss.v102.i09>

See Also

Useful links:

- `https://sachsmc.github.io/eventglm/`
- Report bugs at `https://github.com/sachsmc/eventglm/issues/`

`get_pseudo_cuminc` *Utility to get jackknife pseudo observations of cumulative incidence*

Description

Utility to get jackknife pseudo observations of cumulative incidence

Usage

```
get_pseudo_cuminc(marginal.estimate, time, cause, mr)
```

Arguments

`marginal.estimate`	A survfit object with no covariates
`time`	Time at which to calculate the obs
`cause`	which cause
`mr`	Model response of the survival object

Value

A vector of pseudo-observations

`get_pseudo_rmean` *Utility to get jackknife pseudo observations of restricted mean*

Description

Utility to get jackknife pseudo observations of restricted mean

Usage

```
get_pseudo_rmean(marginal.estimate, time, cause, mr)
```

Arguments

`marginal.estimate`	A survfit object with no covariates
`time`	Time at which to calculate the obs
`cause`	which cause
`mr`	Model response of the survival object

Value

A vector of pseudo-observations

---

jackknife.competing.risks2

Compute jackknife pseudo-observations of the cause-specific cumulative incidence for competing risks

Description

Compute jackknife pseudo-observations of the cause-specific cumulative incidence for competing risks

Usage

```
jackknife.competing.risks2(object, times, cause, mr)
```

Arguments

`object`	A survfit object, with competing events
`times`	Times at which the cumulative incidence is computed, must be length 1
`cause`	Value indicating for which cause the cumulative incidence is to be computed, it must match one of the values available in object (see example)
`mr`	Model response, the result of a call to Surv, or a matrix with two columns: "time" (observed follow up time) and "status" (0 = censored, 1, ..., k = event types)

Value

A vector of jackknifed pseudo-observations of the cause-specific cumulative incidence at time times

Examples

```
sfit.cuminc <- survival::survfit(survival::Surv(etime, event) ~ 1, data = mgus2)
mrs <- with(mgus2, Surv(etime, event))
pseudo.obs <- jackknife.competing.risks2(sfit.cuminc, times = 200, cause = "pcm", mrs)
mean(pseudo.obs)
# agrees with
summary(sfit.cuminc, times = 200)
```

`jackknife.survival2` *Compute jackknife pseudo-observations of the survival function*

Description

Compute jackknife pseudo-observations of the survival function

Usage

```
jackknife.survival2(object, times, mr)
```

Arguments

`object`	A survfit object, with a single event (no competing risks)
`times`	Times at which the survival is computed, must be length 1
`mr`	Model response, the result of a call to Surv, or a matrix with two columns: "time" (observed follow up time) and "status" (0 = censored, 1 = event)

Value

A vector of jackknifed estimates of survival at time times

Examples

```
sfit.surv <- survival::survfit(survival::Surv(time, status) ~ 1, data = colon)
mrs <- with(colon, Surv(time, status))
pseudo.obs <- jackknife.survival2(sfit.surv, times = 1000, mrs)
mean(pseudo.obs)
# agrees with
summary(sfit.surv, times = 1000)
```

`leaveOneOut.competing.risks` *Compute jackknife pseudo-observations of the cause-specific cumulative incidence for competing risks*

Description

This version computes them for all times up to `times`, for the restricted mean lifetime lost

Usage

```
leaveOneOut.competing.risks(object, times, cause, mr)
```

Arguments

`object`	A survfit object, with competing events
`times`	Times at which the cumulative incidence is computed, must be length 1
`cause`	Value indicating for which cause the cumulative incidence is to be computed, it must match one of the values available in object (see example)
`mr`	Model response, the result of a call to Surv, or a matrix with two columns: "time" (observed follow up time) and "status" (0 = censored, 1, ..., k = event types)

Value

A vector of jackknifed values of the cause-specific cumulative incidence at time times

Examples

```
sfit.cuminc <- survival::survfit(survival::Surv(etime, event) ~ 1, data = mgus2)
mrs <- with(mgus2, Surv(etime, event))
jackvals <- leaveOneOut.competing.risks(sfit.cuminc, times = 200, cause = "pcm", mrs)
```

`leaveOneOut.competing.risks2` *Compute jackknife pseudo-observations of the cause-specific cumulative incidence for competing risks*

Description

Compute jackknife pseudo-observations of the cause-specific cumulative incidence for competing risks

Usage

```
leaveOneOut.competing.risks2(object, times, cause, mr)
```

Arguments

`object`	A survfit object, with competing events
`times`	Times at which the cumulative incidence is computed, must be length 1
`cause`	Value indicating for which cause the cumulative incidence is to be computed, it must match one of the values available in object (see example)
`mr`	Model response, the result of a call to Surv, or a matrix with two columns: "time" (observed follow up time) and "status" (0 = censored, 1, ..., k = event types)

Value

A vector of jackknifed values of the cause-specific cumulative incidence at time times

Examples

```
sfit.cuminc <- survival::survfit(survival::Surv(etime, event) ~ 1, data = mgus2)
mrs <- with(mgus2, Surv(etime, event))
jackvals <- leaveOneOut.competing.risks2(sfit.cuminc, times = 200, cause = "pcm", mrs)
```

`leaveOneOut.survival` *Compute leave one out jackknife contributions of the survival function*

Description

For each subject, the survival function is recomputed leaving that subject out. This one does the calculation for all observed times, for calculation of the restricted mean

Usage

```
leaveOneOut.survival(object, times, mr)
```

Arguments

`object`	A survfit object, with a single event (no competing risks)
`times`	Compute values at observed times up to and including this time
`mr`	Model response, the result of a call to Surv, or a matrix with two columns: "time" (observed follow up time) and "status" (0 = censored, 1 = event)

Value

A vector of jackknifed values of survival at time times

Examples

```
sfit.surv <- survival::survfit(survival::Surv(time, status) ~ 1, data = colon)
mrs <- with(colon, Surv(time, status))
jackvals <- leaveOneOut.survival(sfit.surv, 1000, mrs)
```

`leaveOneOut.survival2` *Compute leave one out jackknife contributions of the survival function*

Description

For each subject, the survival function is recomputed leaving that subject out. This is the workhorse for jackknife.survival2 and will generally not be called by the user.

Usage

```
leaveOneOut.survival2(object, times, mr)
```

Arguments

`object`	A survfit object, with a single event (no competing risks)
`times`	Times at which the survival is computed, must be length 1
`mr`	Model response, the result of a call to Surv, or a matrix with two columns: "time" (observed follow up time) and "status" (0 = censored, 1 = event)

Value

A vector of jackknifed values of survival at time times

Examples

```
sfit.surv <- survival::survfit(survival::Surv(time, status) ~ 1, data = colon)
mrs <- with(colon, Surv(time, status))
jackvals <- leaveOneOut.survival2(sfit.surv, times = 1000, mrs)
```

`match_cause` *Match cause specification against model response*

Description

Match cause specification against model response

Usage

```
match_cause(mr, cause)
```

Arguments

`mr`	model.response as returned by Surv
`cause`	Numeric or string indicating the cause of interest

`mgus2` *Monoclonal gammopathy data*

Description

Natural history of 1341 sequential patients with monoclonal gammopathy of undetermined significance (MGUS). This is a superset of the mgus data, at a later point in the accrual process. This dataset is redistributed from the survival package with an added competing risks event indicator.

Usage

```
mgus2
```

Format

A data frame with 1384 observations on the following 10 variables.

`id`	subject identifier
`age`	age at diagnosis, in years
`sex`	a factor with levels `F` `M`
`dxyr`	year of diagnosis
`hgb`	hemoglobin
`creat`	creatinine
`mspike`	size of the monoclonal serum spike
`ptime`	time until progression to a plasma cell malignancy (PCM) or last contact, in months
`pstat`	occurrence of PCM: 0=no, 1=yes
`futime`	time until death or last contact, in months
`death`	occurrence of death: 0=no, 1=yes
`etime`	time until either death, pcm, or last contact
`event`	factor indicating which event occurred first

See Also

mgus2

`print.pseudoglm` *Print method for pseudoglm*

Description

Print method for pseudoglm

Usage

```
## S3 method for class 'pseudoglm'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

`x`	A pseudoglm object, as returned by cumincglm or rmeanglm
`digits`	Number of significant digits
`...`	Not used

Value

x, invisibly

`pseudo_aareg` *Compute censoring weighted pseudo observations*

Description

Assuming that the censoring depends on covariates, the pseudo observations are calculated with the inverse probability of censoring weighted approach, where the censoring probabilities are estimated using Aalen's additive hazards model.

Usage

```
pseudo_aareg(
  formula,
  time,
  cause = 1,
  data,
  type = c("cuminc", "survival", "rmean"),
  formula.censoring = NULL,
  ipcw.method = NULL
)
```

Arguments

`formula`	A formula specifying the outcome model. The left hand side must be a Surv object specifying a right censored survival or competing risks outcome. The status indicator, normally 0=alive, 1=dead. Other choices are TRUE/FALSE (TRUE = death) or 1/2 (2=death). For competing risks, the event variable will be a factor, whose first level is treated as censoring. The right hand side is the usual linear combination of covariates.
`time`	Numeric constant specifying the time at which the cumulative incidence or survival probability effect estimates are desired.
`cause`	Numeric or character constant specifying the cause indicator of interest.
`data`	Data frame in which all variables of formula can be interpreted.
`type`	One of "survival", "cuminc", or "rmean"
`formula.censoring`	A right-sided formula specifying which variables to use in the model for the censoring distribution.
`ipcw.method`	Which method to use for calculation of inverse probability of censoring weighted pseudo observations. "binder" the default, uses the number of observations as the denominator, while the "hajek" method uses the sum of the weights as the denominator.

Value

A vector of pseudo observations

See Also

aareg

Examples

```
POi <- pseudo_aareg(Surv(time, status) ~ 1, 1500, cause = 1,
  data = colon, type = "rmean", formula.censoring = ~ sex + age,
  ipcw.method = "binder")

mean(POi)
```

---

pseudo_coxph *Compute censoring weighted pseudo observations*

Description

Assuming that the censoring depends on covariates, the pseudo observations are calculated with the inverse probability of censoring weighted approach, where the censoring probabilities are estimated using Cox's proportional hazards model.

Usage

```
pseudo_coxph(
  formula,
  time,
  cause = 1,
  data,
  type = c("cuminc", "survival", "rmean"),
  formula.censoring = NULL,
  ipcw.method = NULL
)
```

Arguments

`formula`	A formula specifying the outcome model. The left hand side must be a Surv object specifying a right censored survival or competing risks outcome. The status indicator, normally 0=alive, 1=dead. Other choices are TRUE/FALSE (TRUE = death) or 1/2 (2=death). For competing risks, the event variable will be a factor, whose first level is treated as censoring. The right hand side is the usual linear combination of covariates.
`time`	Numeric constant specifying the time at which the cumulative incidence or survival probability effect estimates are desired.
`cause`	Numeric or character constant specifying the cause indicator of interest.
`data`	Data frame in which all variables of formula can be interpreted.
`type`	One of "survival", "cuminc", or "rmean"

`formula.censoring`	A right-sided formula specifying which variables to use in the model for the censoring distribution.
`ipcw.method`	Which method to use for calculation of inverse probability of censoring weighted pseudo observations. "binder" the default, uses the number of observations as the denominator, while the "hajek" method uses the sum of the weights as the denominator.

Value

A vector of pseudo observations

See Also

coxph

Examples

```
POi <- pseudo_coxph(Surv(time, status) ~ 1, 1500, cause = 1,
  data = colon, type = "survival", formula.censoring = ~ sex + age,
  ipcw.method = "hajek")

mean(POi)
```

`pseudo_independent` *Compute pseudo observations under independent censoring*

Description

Assuming completely independent censoring, i.e., censoring does not depend on the survival time nor any covariates in the model, the pseudo observations are calculated with the standard jackknife approach

Usage

```
pseudo_independent(
  formula,
  time,
  cause = 1,
  data,
  type = c("cuminc", "survival", "rmean"),
  formula.censoring = NULL,
  ipcw.method = NULL
)
```

Arguments

`formula`	A formula specifying the model. The left hand side must be a Surv object specifying a right censored survival or competing risks outcome. The status indicator, normally 0=alive, 1=dead. Other choices are TRUE/FALSE (TRUE = death) or 1/2 (2=death). For competing risks, the event variable will be a factor, whose first level is treated as censoring. The right hand side is the usual linear combination of covariates.
`time`	Numeric constant specifying the time at which the cumulative incidence or survival probability effect estimates are desired.
`cause`	Numeric or character constant specifying the cause indicator of interest.
`data`	Data frame in which all variables of formula can be interpreted.
`type`	One of "survival", "cuminc", or "rmean"
`formula.censoring`	Not used with this method, see pseudo_stratified, pseudo_aareg or pseudo_coxph
`ipcw.method`	Not used with this method

Value

A vector of jackknife pseudo observations

Examples

```
POi <- pseudo_independent(Surv(time, status) ~ 1, 1500, cause = 1, data = colon, type = "survival")
mean(POi)
```

`pseudo_infjack` *Compute infinitesimal jackknife pseudo observations*

Description

Assuming that the censoring depends on covariates with a finite set of levels, the pseudo observations are calculated with the infinitesimal jackknife approach stratified on those covariates. If no covariates are specified in the censoring model, then the pseudo observations are calculated under the completely independent censoring assumption. This function allows survival objects with entry and exit times, thus multi-state models, recurrent events, and delayed entry/left truncation. With delayed entry, the pseudo observation approach theoretically works under the assumption that the entry time is independent of covariates.

Usage

```
pseudo_infjack(
  formula,
  time,
  cause = 1,
```

```
    data,
    type = c("cuminc", "survival", "rmean"),
    formula.censoring = NULL,
    ipcw.method = NULL
  )
```

Arguments

`formula`	A formula specifying the outcome model. The left hand side must be a Surv object specifying a right censored survival, competing risks, counting process, or multistate outcome. The status indicator, normally 0=alive, 1=dead. Other choices are TRUE/FALSE (TRUE = death) or 1/2 (2=death). For competing risks and multi state models, the event variable will be a factor, whose first level is treated as censoring. The right hand side is the usual linear combination of covariates.
`time`	Numeric constant specifying the time at which the cumulative incidence or survival probability effect estimates are desired.
`cause`	Numeric or character constant specifying the cause indicator of interest.
`data`	Data frame in which all variables of formula can be interpreted.
`type`	One of "survival", "cuminc", or "rmean"
`formula.censoring`	A optional right-sided formula specifying which variables to stratify on. All variables in this formula must be categorical.
`ipcw.method`	Not used with this method

Value

A vector of pseudo observations

See Also

survfit

Examples

```
POi <- pseudo_infjack(Surv(time, status) ~ 1, 1500, cause = 1,
  data = colon, type = "survival", formula.censoring = ~ sex)

mean(POi)
```

`pseudo_rmst2` *Compute pseudo-observations for the restricted mean survival*

Description

Compute pseudo-observations for the restricted mean survival

Usage

```
pseudo_rmst2(sfit, jacks, times, tmax, type = "cuminc")
```

Arguments

`sfit`	A survfit object
`jacks`	A matrix of leave-one-out jackknife values, subjects in the rows, times in the columns
`times`	Times at which the survival is calculated
`tmax`	Max time
`type`	"cuminc" or "survival"

Value

A vector of pseudo observations for the restricted mean or lifetime lost

---

`pseudo_stratified` *Compute pseudo observations using stratified jackknife*

Description

Assuming that the censoring depends on covariates with a finite set of levels, the pseudo observations are calculated with the jackknife approach stratified on those covariates.

Usage

```
pseudo_stratified(
  formula,
  time,
  cause = 1,
  data,
  type = c("cuminc", "survival", "rmean"),
  formula.censoring = NULL,
  ipcw.method = NULL
)
```

Arguments

`formula`	A formula specifying the model. The left hand side must be a Surv object specifying a right censored survival or competing risks outcome. The status indicator, normally 0=alive, 1=dead. Other choices are TRUE/FALSE (TRUE = death) or 1/2 (2=death). For competing risks, the event variable will be a factor, whose first level is treated as censoring. The right hand side is the usual linear combination of covariates.
`time`	Numeric constant specifying the time at which the cumulative incidence or survival probability effect estimates are desired.
`cause`	Numeric or character constant specifying the cause indicator of interest.
`data`	Data frame in which all variables of formula can be interpreted.
`type`	One of "survival", "cuminc", or "rmean"
`formula.censoring`	A right-sided formula specifying which variables to stratify on. All variables in this formula must be categorical.
`ipcw.method`	Not used with this method

Value

A vector of jackknife pseudo observations

Examples

```
POi <- pseudo_stratified(Surv(time, status) ~ 1, 1500, cause = 1,
  data = colon, formula.censoring = ~ sex, type = "rmean")
mean(POi)
```

`residuals.pseudoglm` *Pseudo-observation scaled residuals*

Description

Computes residuals according to the recommendations of Pohar-Perme and Andersen (2009) <doi: 10.1002/sim.3401>.

Usage

```
## S3 method for class 'pseudoglm'
residuals(object, type = NULL, ...)
```

Arguments

`object`	A pseudoglm object, as returned by cumincglm or rmeanglm
`type`	Either "scaled" (the default for cumulative incidence outcomes) or one of the types available in residuals.glm for restricted mean outcomes, with the default being "deviance".
`...`	Arguments passed on to residuals.glm.

Details

The scaled residuals are computed as

$$\hat{\epsilon}_i = \frac{\hat{E}(V_i) - \hat{Y}_i}{\sqrt{\hat{Y}_i(1 - \hat{Y}_i)}}$$

When the outcome is the cumulative incidence, the denominator corresponds to an estimate of the standard error of the conditional estimate of the outcome in the absence of censoring. For the restricted mean, no such rescaling is done and the computation is passed off to residuals.glm.

Value

A numeric vector of residuals

References

Perme MP, Andersen PK. Checking hazard regression models using pseudo-observations. Stat Med. 2008;27(25):5309-5328. <doi:10.1002/sim.3401>

`rmeanglm` *Generalized linear models for the restricted mean survival*

Description

Using pseudo observations for the restricted mean, or the restricted mean lifetime lost in the competing risks case, this function then runs a generalized linear model to estimate associations of the restricted mean/lifetime lost up to a particular time (specified by the `time` argument) with covariates. The link function can be "identity" for estimating differences in the restricted mean, "log" for estimating ratios, and any of the other link functions supported by quasi.

Usage

```
rmeanglm(
  formula,
  time,
  cause = 1,
  link = "identity",
  model.censoring = "independent",
  formula.censoring = NULL,
  ipcw.method = "binder",
  data,
  weights,
  subset,
  na.action,
  offset,
  control = list(...),
  model = FALSE,
```

```
  x = TRUE,
  y = TRUE,
  singular.ok = TRUE,
  contrasts = NULL,
  id,
  ...
)
```

Arguments

`formula`	A formula specifying the model. The left hand side must be a Surv object specifying a right censored survival or competing risks outcome. The status indicator, normally 0=alive, 1=dead. Other choices are TRUE/FALSE (TRUE = death) or 1/2 (2=death). For competing risks, the event variable will be a factor, whose first level is treated as censoring. The right hand side is the usual linear combination of covariates.
`time`	Numeric constant specifying the time up to which the restricted mean effect estimates are desired.
`cause`	Numeric or character constant specifying the cause indicator of interest.
`link`	Link function for the restricted mean regression model.
`model.censoring`	Type of model for the censoring distribution. Options are "stratified", which computes the pseudo-observations stratified on a set of categorical covariates, "aareg" for Aalen's additive hazards model, and "coxph" for Cox's proportional hazards model. With those options, we assume that the time to event and event indicator are conditionally independent of the censoring time, and that the censoring model is correctly specified. If "independent", we assume completely independent censoring, i.e., that the time to event and covariates are independent of the censoring time. the censoring time is independent of the covariates in the model. Can also be a custom function, see Details and the "Extending eventglm" vignette.
`formula.censoring`	A one sided formula (e.g., ~ `x1 + x2`) specifying the model for the censoring distribution. If NULL, uses the same mean model as for the outcome. Missing values in any covariates for the censoring model will cause an error.
`ipcw.method`	Which method to use for calculation of inverse probability of censoring weighted pseudo observations. "binder" the default, uses the number of observations as the denominator, while the "hajek" method uses the sum of the weights as the denominator.
`data`	Data frame in which all variables of formula can be interpreted.
`weights`	an optional vector of 'prior weights' to be used in the fitting process. Should be NULL or a numeric vector.
`subset`	an optional vector specifying a subset of observations to be used in the fitting process.

`na.action`	a function which indicates what should happen when the data contain `NA`s. The default is set by the `na.action` setting of options, and is na.fail if that is unset. The 'factory-fresh' default is na.omit. Another possible value is NULL, no action. Value na.exclude can be useful.
`offset`	this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases. One or more offset terms can be included in the formula instead or as well, and if more than one is specified their sum is used. See model.offset.
`control`	a list of parameters for controlling the fitting process. This is passed to glm.control.
`model`	a logical value indicating whether model frame should be included as a component of the returned value.
`x`	logical value indicating whether the model matrix used in the fitting process should be returned as components of the returned value.
`y`	logical value indicating whether the response vector (pseudo-observations) used in the fitting process should be returned as components of the returned value.
`singular.ok`	logical; if FALSE a singular fit is an error.
`contrasts`	an optional list. See the contrasts.arg of model.matrix.default.
`id`	An optional integer vector of subject identifiers, found in data. If this is present, then generalized estimating equations will be used to fit the model. This can be used, for example, if there are multiple observations per individual represented as multiple rows in data.
`...`	Other arguments passed to glm.fit

Details

The argument "model.censoring" determines how the pseudo observations are calculated. This can be the name of a function or the function itself, which must have arguments "formula", "time", "cause", "data", "type", "formula.censoring", and "ipcw.method". If it is the name of a function, this code will look for a function with the prefix "pseudo_" first, to avoid clashes with related methods such as coxph. The function then must return a vector of pseudo observations, one for each subject in data which are used in subsequent calculations. For examples of the implementation, see the "pseudo-modules.R" file, or the vignette "Extending eventglm".

Value

A pseudoglm object, with its own methods for print, summary, and vcov. It inherits from glm, so predict and other glm methods are supported.

Examples

```
    rmeanipcw <- rmeanglm(Surv(etime, event) ~ age + sex,
         time = 200, cause = "pcm", link = "identity",
         model.censoring = "independent", data = mgus2)
# stratified on only the categorical covariate
     rmeanipcw2 <- rmeanglm(Surv(etime, event) ~ age + sex,
                         time = 200, cause = "pcm", link = "identity",
```

```
                          model.censoring = "stratified",
                          formula.censoring = ~ sex, data = mgus2)
```

`summary.pseudoglm` *Summary method*

Description

Summary method

Usage

```
## S3 method for class 'pseudoglm'
summary(
  object,
  correlation = FALSE,
  symbolic.cor = FALSE,
  type = "robust",
  ...
)
```

Arguments

`object`	A pseudoglm object, as returned by cumincglm or rmeanglm
`correlation`	logical; if TRUE, the correlation matrix of the estimated parameters is returned and printed.
`symbolic.cor`	logical; If TRUE, print the correlations in a symbolic form rather than as numbers.
`type`	The method to use for variance estimation; one of "corrected", "robust", "naive", or "cluster"
`...`	Not used

Value

An object of class summary.glm

---

vcov.pseudoglm *Compute covariance matrix of regression coefficient estimates*

Description

Compute covariance matrix of regression coefficient estimates

Usage

```
## S3 method for class 'pseudoglm'
vcov(object, type = "robust", ...)
```

Arguments

`object`	A pseudoglm object, as returned by cumincglm or rmeanglm.
`type`	The method to use for variance estimation; one of "corrected", "robust", "naive", or "cluster"
`...`	Not used

Details

The "corrected" variance estimate for the cumulative incidence is as described in Overgaard et al. (2017) <doi:10.1214/16-AOS1516>, with code adapted from Overgaard's Stata program. This method does not handle ties and only has marginal benefits in reasonable sample sizes. The default is "robust" which uses the sandwich estimator vcovHC as implemented in the sandwich package. "cluster" is another option if you have clustered observations that uses the vcovCL function in sandwich. Finally "naive" uses the same method as glm to compute the variance, and is known to be anti-conservative. The bootstrap is another recommended option that can be implemented using other tools; there is an example in the vignette.

Value

A numeric matrix containing the variance-covariance estimates

References

Overgaard, Morten; Parner, Erik Thorlund; Pedersen, Jan. Asymptotic theory of generalized estimating equations based on jack-knife pseudo-observations. Ann. Statist. 45 (2017), no. 5, 1988–2015. <doi:10.1214/16-AOS1516>.

See Also

vcovHC, vcovCL

Index