Package: semicmprskcoxmsm (via r-universe)

September 14, 2024

Type Package

Title Use Inverse Probability Weighting to Estimate Treatment Effect for Semi Competing Risks Data

Version 0.2.0

Author Yiran Zhang

Maintainer Yiran Zhang <yiz038@health.ucsd.edu>

Description Use inverse probability weighting methods to estimate treatment effect under marginal structure model (MSM) for the transition hazard of semi competing risk data, i.e. illness death model. We implement two specific such models, the usual Markov illness death structural model and the general Markov illness death structural model. We also provide the predicted three risks functions from the marginal structure models. Zhang, Y. and Xu, R. (2022) <arXiv:2204.10426>.

License GPL (>= 2)

Imports ggplot2, survival, stats, twang, graphics, fastGHQuad, Rcpp

Suggests knitr, rmarkdown

NeedsCompilation no

Repository CRAN

Date/Publication 2022-04-29 23:40:02 UTC

Contents

`bayesian_boot_irrd` *Obtaining Bayesian Bootstrap Sample for Individual Risk Difference and Risk Ratio.*

Description

`bayesian_boot_irrd` provides the bootstrap sample for individual risk difference and risk ratio, it can be used for further inferences.

Usage

```
bayesian_boot_irrd(dat2,B,sigma_2_0, EM_initial, varlist, t1_star,t)
```

Arguments

`dat2`	The dataset, includes non-terminal events, terminal events as well as event indicator.
`B`	Number of bootstraps that the user want to run, typically we use B = 500.
`sigma_2_0`	Initial value for sigma_2 for the general Markov model
`EM_initial`	Initial value for the EM algorithm, the output of `OUT_em_weights`.
`varlist`	Confounder list for the propensity score model.
`t1_star`	Fixed non-terminal event time for estimating risk difference/ratio for terminal event following the non-terminal event.
`t`	Fixed time point of interest to compare the individual risk difference / ratio.

Details

For each bootstrap sample:

1. Generate n standard exponential (mean and variance 1) random variates : $u_1, u_2, ..., u_n$;

2. The weights for the Bayesian bootstrap are: $w_i^{boot} = u_i/\bar{u}$, where $\bar{u} = n^{-1}\sum_{i=1}^{n} u_i$;

3. Calculate the propensity score and IP weights w_i^{IPW} based on Bayesian bootstrap weighted data, and assigned the weights for fitting the MSM general Markov model as $w_i = w_i^{boot} * w_i^{IPW}$.

4. After obtaining $\hat{\theta}$ and $\hat{b}_i$, for each individual i, calculate the IRR and IRD by plugging $\hat{\theta}, \hat{b}_i$ and a=0, a=1 separately at time t.

The 95% prediction intervals (PI) cam be obtained by the normal approximation using bootstrap standard error.

Value

RD1_boot	A n times B matrix as the Bayesian bootstrap sample for each data point. The sample is for individual risk difference for time to non-terminal event at time t.
RD2_boot	A n times B matrix as the Bayesian bootstrap sample for each data point. The sample is for individual risk difference for time to terminal event without non-terminal event at time t.
RD3_boot	A n times B matrix as the Bayesian bootstrap sample for each data point. The sample is for individual risk difference for time to terminal event following non-terminal event by t1_start at time t.
RR1_boot	A n times B matrix as the Bayesian bootstrap sample for each data point. The sample is for individual risk ratio for time to non-terminal event at time t.
RR2_boot	A n times B matrix as the Bayesian bootstrap sample for each data point. The sample is for individual risk ratio for time to terminal event without non-terminal event at time t.
RR3_boot	A n times B matrix as the Bayesian bootstrap sample for each data point. The sample is for individual risk ratio for time to terminal event following non-terminal event by t1_start at time t.

---

cif_est_usual *Estimating Three Cumulative Incidence Functions Using the Usual Markov Model*

Description

cif_est_usual estimates the cumulative incidence function (CIF, i.e.risk) based on the MSM illness-death usual Markov model.

Usage

```
cif_est_usual(data,X1,X2,event1,event2,w,Trt,
              t1_star = t1_star)
```

Arguments

data	The dataset, includes non-terminal events, terminal events as well as event indicator.
X1	Time to non-terminal event, could be censored by terminal event or lost to follow up.
X2	Time to terminal event, could be censored by lost to follow up.
event1	Event indicator for non-terminal event.
event2	Event indicator for terminal event.
w	IP weights.
Trt	Treatment variable.
t1_star	Fixed non-terminal event time for estimating CIF function for terminal event following the non-terminal event.

Details

After estimating the parameters in the illness-death model λ_j^a using IPW, we could estimate the corresponding CIF:

$$\hat{P}(T_1^a < t, \delta_1^a = 1) = \int_0^t \hat{S}^a(u) d\hat{\Lambda}_1^a(u),$$

$$\hat{P}(T_2^a < t, \delta_1^a = 0, \delta_2^a = 1) = \int_0^t \hat{S}^a(u) d\hat{\Lambda}_2^a(u),$$

and

$$\hat{P}(T_2^a < t_2 \mid T_1^a < t_1, T_2^a > t_1) = 1 - e^{-\int_{t_1}^{t_2} d\hat{\Lambda}_{12}^a(u)},$$

where $\hat{S}^a$ is the estimated overall survial function for joint T_1^a, T_2^a, $\hat{S}^a(u) = e^{-\hat{\Lambda}_1^a(u)} - \hat{\Lambda}_2^a(u)$. We obtain three hazards by fitting the MSM illness-death model $\hat{\Lambda}_j^a(u) = \hat{\Lambda}_{0j}(u) e^{\hat{\beta}_j * a}$, $\hat{\Lambda}_{12}^a(u) = \hat{\Lambda}_{03}(u) e^{\hat{\beta}_3 * a}$, and $\hat{\Lambda}_{0j}(u)$ is a Breslow-type estimator of the baseline cumulative hazard.

Value

Returns a table containing the estimated CIF for the event of interest for control and treated group.

References

Meira-Machado, Luis and Sestelo, Marta (2019). "Estimation in the progressive illness-death model: A nonexhaustive review," *Biometrical Journal* 61(2), 245–263.

`conditional_cif_b` *Estimating Three Conditional Cumulative Incidence Functions Using the General Markov Model Conditional on Random Effect*

Description

`conditional_cif_b` estimates the cumulative incidence function based on the MSM illness-death general Markov model conditional on the fixed random effect b.

Usage

```
conditional_cif_b(res1,
                  t1_star,
                  b)
```

Arguments

res1	The output from em_illness_death_phmm_weight, the general Markov model result.
t1_star	Fixed non-terminal event time for estimating CIF function for terminal event following the non-terminal event.
b	Fixed random effect value.

Details

Similar as cif_est_usual, after estimating the parameters in the illness-death model λ_j^a using IPW, we could estimate the corresponding conditional CIF under fixed b:

$$\hat{P}(T_1^a < t, \delta_1^a = 1 \mid b) = \int_0^t \hat{S}^a(u \mid b) d\hat{\Lambda}_1^a(u \mid b),$$

$$\hat{P}(T_2^a < t, \delta_1^a = 0, \delta_2^a = 1 \mid b) = \int_0^t \hat{S}^a(u \mid b) d\hat{\Lambda}_2^a(u \mid b),$$

and

$$\hat{P}(T_2^a < t_2 \mid T_1^a < t_1, T_2^a > t_1 \mid b) = 1 - e^{-\int_{t_1}^{t_2} d\hat{\Lambda}_{12}^a(u|b)},$$

where $\hat{S}^a$ is the estimated overall survial function for joint T_1^a, T_2^a, $\hat{S}^a(u) = e^{-\hat{\Lambda}_1^a(u)} - \hat{\Lambda}_2^a(u)$. We obtain three hazards by fitting the MSM illness-death model $\hat{\Lambda}_j^a(u) = \hat{\Lambda}_{0j}(u)e^{\hat{\beta}_j * a}$, $\hat{\Lambda}_{12}^a(u) = \hat{\Lambda}_{03}(u)e^{\hat{\beta}_3 * a}$, and $\hat{\Lambda}_{0j}(u)$ is a Breslow-type estimator of the baseline cumulative hazard.

where $S(t \mid b; a) = \exp[-\int_0^t \{\lambda_{01}(u)e^{\beta_1 a+b} + \lambda_{02}(u)e^{\beta_2 a+b}\}du] = \exp\{-e^{\beta_1 a+b}\Lambda_{01}(t) - e^{\beta_2 a+b}\Lambda_{02}(t)\}$

Value

a1	The step function for estimated CIF conditional on b for time to non-terminal event for control group.
b1	The step function for estimated CIF conditional on b for time to non-terminal event for treated group.
a2	The step function for estimated CIF conditional on b for time to terminal event without non-terminal event for control group.
b2	The step function for estimated CIF conditional on b for time to terminal event without non-terminal event for treated group.
a3	The step function for estimated CIF conditional on b for time to terminal event following non-terminal event by t1_start for control group.
b3	The step function for estimated CIF conditional on b for time to terminal event without non-terminal event by t1_start for treated group.
cif.1	A data frame with time and estimated CIF conditional on b if is treated or controlled for time to non-terminal event.

cif.2	A data frame with time and estimated CIF conditional on b if is treated or controlled for time to terminal event without non-terminal event.
cif.3	A data frame with time and estimated CIF conditional on b if is treated or controlled for time to terminal event without non-terminal event by t1_start.

See Also

cif_est_usual

---

doPS *Generate the Inverse Probability Treatment Weights*

Description

doPS calculates the unstabilized and stabilized inverse probability treatment weights (IPW) for average treatment effect using propensity score. The propensity score is calculated by twang package using the boosted logistic regression.

Usage

```
doPS(data,Trt,Trt.name,VARS.,logistic = FALSE,w=NULL)
```

Arguments

data	The dataset, includes treatment assignment as well as covariates.
Trt	The name of the treatment variable in the dataset.
Trt.name	The treated group name of the treatment variable in the dataset.
VARS.	The vector of the name of potential confounding variables in the dataset.
logistic	A logical value indicating whether use logistic regression (TRUE) or non-parametric boosted tree (FALSE).
w	Optional sampling weights.

Details

The treatment variable should only contain 2 levels of treatment, and one should be viewed as treated group and another is control group.

For stabilized weights:

For the treated individuals, we assign the IPW: w = Pr(T=1)/Pr(T=1|X=x), for control individuals, the stabilized weight is: w = (1-Pr(T=1))/(1-Pr(T=1|X=x)).

Value

doPS returns an object of class "PS". An object of class "PS" is a list containing the following components:

Data	A new dataset which excludes all the missing value on the potential confounders from input data, add the propensity score and IPW into the new dataset. **ps_ate** The estimated propensity scores with estimand of interest as ATE. **ipw_ate_unstab** Unstabilized ipw calculated from ps_ate. **ipw_ate_stab** Stabilized ipw calculated from ps_ate.
ps	an object of class ps, See the help for ps for details of the ps class.

See Also

ps

Examples

```
n <- 500
set.seed(1234)
Cens = runif(n,0.7,0.9)
set.seed(1234)
OUT1 <- sim_cox_msm_semicmrsk(beta1 = 1,beta2 = 1,beta3 = 0.5,
                              sigma_2 = 1,
                              alpha0 = 0.5, alpha1 = 0.1, alpha2 = -0.1, alpha3 = -0.2,
                              n=n, Cens = Cens)
data_test <- OUT1$data0

## Get the PS weights
vars <- c("Z1","Z2","Z3")
ps1 <- doPS(data = data_test,
            Trt = "A",
            Trt.name = 1,
            VARS. = vars,
            logistic = TRUE,w=NULL)
w <- ps1$Data$ipw_ate_stab
```

---

em_illness_death_phmm_weight

Using EM Type Algorithm for MSM Illness-death General Markov Model

---

Description

Under the general Markov illness-death model, with normal frailty term which is a latent variable. We use the EM type algorithm to estimate the coefficient in the MSM illness-death general Markov model.

Usage

```
em_illness_death_phmm_weight(data,X1,X2,event1,event2,w,Trt,
                             EM_initial,sigma_2_0)
```

Arguments

data	The dataset, includes non-terminal events, terminal events as well as event indicator.
X1	Time to non-terminal event, could be censored by terminal event or lost to follow up.
X2	Time to terminal event, could be censored by lost to follow up.
event1	Event indicator for non-terminal event.
event2	Event indicator for terminal event.
w	IP weights.
Trt	Treatment variable.
EM_initial	Initial value for the EM algorithm, the output of OUT_em_weights.
sigma_2_0	Initial value for σ^2, the variance of zero-mean normal frailty, usually starts with 1.

Details

Similar as the usual Markov model. We postulate the semi-parametric Cox models with a frailty term for three transition rates in marginal structural illness-death model:

$$\lambda_1(t_1; a) = \lambda_{01}(t)e^{\beta_1 a+b}, t_1 > 0;$$

$$\lambda_2(t_2; a) = \lambda_{02}(t)e^{\beta_2 a+b}, t_2 > 0;$$

and

$$\lambda_{12}(t_2 \mid t_1; a) = \lambda_{03}(t_2)e^{\beta_3 a+b}, 0 < t_1 < t_2,$$

where $b \sim N(0, 1)$. Since b is not observed in the data, we use the IP weighted EM type algorithm to estimate all the parameters in the MSM illness-death general Markov model.

Value

beta1	The EM sequence for estimating β_1 at each iteration.
beta2	The EM sequence for estimating β_2 at each iteration.
beta3	The EM sequence for estimating β_3 at each iteration.
Lambda01	List of two dataframes for estimated Λ_{01} and λ_{01} when EM converges.
Lambda02	List of two dataframes for estimated Λ_{02} and λ_{02} when EM converges.
Lambda03	List of two dataframes for estimated Λ_{03} and λ_{03} when EM converges.

sigma_2	The EM sequence for estimating σ^2 at each iteration.
loglik	The EM sequence for log-likelihood at each iteration.
em.n	Number of EM steps to converge.
data	Data after running the EM.

Examples

```
n <- 500
set.seed(1234)
Cens = runif(n,0.7,0.9)
set.seed(1234)
OUT1 <- sim_cox_msm_semicmrsk(beta1 = 1,beta2 = 1,beta3 = 0.5,
                              sigma_2 = 1,
                              alpha0 = 0.5, alpha1 = 0.1, alpha2 = -0.1, alpha3 = -0.2,
                              n=n, Cens = Cens)
data_test <- OUT1$data0

## Get the PS weights
vars <- c("Z1","Z2","Z3")
ps1 <- doPS(data = data_test,
            Trt = "A",
            Trt.name = 1,
            VARS. = vars,
            logistic = TRUE,w=NULL)
w <- ps1$Data$ipw_ate_stab

### Fit the General Markov model
EM_initial <- OUT_em_weights(data = data_test,
                             X1 = "X1",
                             X2 = "X2",
                             event1 = "delta1",
                             event2 = "delta2",
                             w = w,
                             Trt = "A")

res1 <- em_illness_death_phmm_weight(data = data_test,
                                     X1 = "X1",
                                     X2 = "X2",
                                     event1 = "delta1",
                                     event2 = "delta2",
                                     w = w,
                                     Trt = "A",
                                     EM_initial = EM_initial,
                                     sigma_2_0 = 2)

print(paste("The estimated value for beta1 is:", round(res1$beta1[res1$em.n],5) ) )
```

get_hazard *Compute the (Cumulative) Baseline Hazard from Cox Model*

Description

Compute the Breslow type baseline hazard and cumulative baseline hazard at each event time from a Cox model.

Usage

```
get_hazard(fit)
```

Arguments

fit	The results of a coxph fit.

Details

See also basehaz, we only extract the estimated baseline hazard and baseline cumulative hazard from the results of a coxph fit.

Value

A list contains two dataframes.

Lambda	See also basehaz, returns the Breslow type cumulative baseline hazard.
lambda	Returns the Breslow type baseline hazard.

See Also

basehaz

get_hazard_offset_weights *Compute the (Cumulative) Baseline Hazard from Cox Model with Offsets*

Description

Compute the Breslow type baseline hazard and cumulative baseline hazard at each event time from a weighted Cox model with offsets.

Usage

```
get_hazard_offset_weights(fit,data,time1= NULL,time2,w)
```

Arguments

fit	The results of a weighted coxph fit.
data	The original data for fitting the weighted Cox model.
time1	The default is NULL. For left truncation data, which refers to transition rate for terminal event following non-terminal events, this argument is the time to non-terminal event.
time2	For right censored data, this is the event time or censoring time. For left truncation data, the argument is the time to terminal event or the censoring time.
w	IP weights.

Details

See also get_hazard, handles the offset term in coxph for predicting the baseline hazard.

Value

A list contains two dataframes.

Lambda	See also get_hazard, returns a step function for cumulative baseline hazard.
lambda	Returns a dataframe for baseline hazard.
cum_base_haz	Returns a dataframe for cumulative baseline hazard.

See Also

get_hazard, basehaz

---

individual_RR_RD *Estimating Three Individual Risk Difference and Risk Ratio Using the General Markov Model Conditional on Predicted Random Effect*

Description

individual_RR_RD estimates the individual risk difference and risk ratio based on the MSM illness-death general Markov model conditional on predicted random effect for each data point at a fixed time point.

Usage

```
individual_RR_RD(dat1,res1,t1_star ,t)
```

Arguments

dat1	The dataset, includes non-terminal events, terminal events as well as event indicator.
res1	The output from em_illness_death_phmm_weight, the general Markov model result, the result data includes the predicted random effect.
t1_star	Fixed non-terminal event time for estimating risk difference/ratio for terminal event following the non-terminal event.
t	Fixed time point of interest to compare the individual risk difference / ratio.

Details

Similar as cif_est_usual, after estimating the parameters in the illness-death model λ_j^a using IPW, we could estimate the corresponding conditional CIF under the predicted b:

$$\hat{P}(T_1^a < t, \delta_1^a = 1 \mid b) = \int_0^t \hat{S}^a(u \mid b) d\hat{\Lambda}_1^a(u \mid b),$$

$$\hat{P}(T_2^a < t, \delta_1^a = 0, \delta_2^a = 1 \mid b) = \int_0^t \hat{S}^a(u \mid b) d\hat{\Lambda}_2^a(u \mid b),$$

and

$$\hat{P}(T_2^a < t_2 \mid T_1^a < t_1, T_2^a > t_1 \mid b) = 1 - e^{-\int_{t_1}^{t_2} d\hat{\Lambda}_{12}^a(u|b)},$$

The frailty term, or equivalently, the random effect b represents the unobserved heterogeneity among the individuals. As such, the above conditional risk represents individual risk, and the risk contrasts the individual risk contrasts. We therefore have the individual risk difference (IRD) and the individual risk ratio (IRR).

Under the random effects model, for $i = 1, 2, ..., n$, the predicted random effect is $\hat{b}_i = E(b_i \mid O_i, \hat{\theta})$. We then obtain the predicted IRD and the predicted IRR.

Value

Returns a data frame that includes the individual risk difference / ratio for three type of events.

---

initial_fit_em_weights

Fit the MSM Cox Model with IP Weights

Description

Fit the MSM cox model with IPW as the initial value for EM algorithm to fit the illness-death general Markov model

Usage

```
initial_fit_em_weights(data,X1,X2,event1,event2,w,Trt)
```

Arguments

data	The dataset, includes non-terminal events, terminal events as well as event indicator.
X1	Time to non-terminal event, could be censored by terminal event or lost to follow up.
X2	Time to terminal event, could be censored by lost to follow up.
event1	Event indicator for non-terminal event.
event2	Event indicator for terminal event.
w	IP weights.
Trt	Treatment variable.

Details

As initial values we use for β_j, $j = 1, 2, 3$, the estimates from IP weighted Cox regression without the offsets, i.e. from the usual Markov model.

Value

A list of objects from survival package:

event1	An object of class Surv for non-terminal event.
event2	An object of class Surv for terminal event without non-terminal event.
event3	An object of class Surv for terminal event following non-terminal event.
fit1	An object of class coxph representing the fit for time to non-terminal event. See coxph.object for details.
fit2	An object of class coxph representing the fit for time to terminal event without non-terminal event.
fit3	An object of class coxph representing the fit for time to terminal event following non-terminal event.

See Also

Surv, coxph

initial_lambda_em — *Compute the Initial (Cumulative) Baseline Hazard From the MSM Illness-death Model*

Description

Compute the Breslow type baseline hazard and cumulative baseline hazard at each event time from the MSM illness-death model.

Usage

```
initial_lambda_em (OUT)
```

Arguments

OUT	The results of a initial_fit_em_weights fit.

Details

See also get_hazard

Value

A list contains six dataframes: including baseline hazard and cumulative baseline hazard for non-terminal event, terminal event without non-terminal event, and terminal event following non-terminal event.

See Also

get_hazard

OUT_em_weights — *Initial Value For Fitting the General Markov Model*

Description

Compute the initial value for fitting the MSM illness-death general Markov model using EM type algorithm

Usage

```
OUT_em_weights(data,X1,X2,event1,event2,w,Trt)
```

Arguments

data	The dataset, includes non-terminal events, terminal events as well as event indicator.
X1	Time to non-terminal event, could be censored by terminal event or lost to follow up.
X2	Time to terminal event, could be censored by lost to follow up.
event1	Event indicator for non-terminal event.
event2	Event indicator for terminal event.
w	IP weights.
Trt	Treatment variable.

Details

See usual_illness_death_weight

Value

A list of vectors and dataframes:

beta1	Initial value for β_1, the coefficient for the non-terminal event model.
beta2	Initial value for β_2, the coefficient for the terminal event without non-terminal event model.
beta3	Initial value for β_3, the coefficient for the terminal event following non-terminal event model.
lambda1	Initial value for λ_{01}, the estimated baseline hazard for the non-terminal event model.
lambda2	Initial value for λ_{02}, the estimated baseline hazard for the terminal event without non-terminal event model.
lambda3	Initial value for λ_{03}, the estimated baseline hazard for the terminal event following non-terminal event model.
Lambda1	Initial value for Λ_{01}, the estimated cumulative baseline hazard for the non-terminal event model.
Lambda2	Initial value for Λ_{02}, the estimated cumulative baseline hazard for the terminal event without non-terminal event model.
Lambda3	Initial value for Λ_{03}, the estimated cumulative baseline hazard for the terminal event following non-terminal event model.
event1	An object of class Surv for non-terminal event.
event2	An object of class Surv for terminal event without non-terminal event.
event3	An object of class Surv for terminal event following non-terminal event.

See Also

usual_illness_death_weight

Examples

```
n <- 500
set.seed(1234)
Cens = runif(n,0.7,0.9)
set.seed(1234)
OUT1 <- sim_cox_msm_semicmrsk(beta1 = 1,beta2 = 1,beta3 = 0.5,
                              sigma_2 = 1,
                              alpha0 = 0.5, alpha1 = 0.1, alpha2 = -0.1, alpha3 = -0.2,
                              n=n, Cens = Cens)
data_test <- OUT1$data0

## Get the PS weights
vars <- c("Z1","Z2","Z3")
ps1 <- doPS(data = data_test,
            Trt = "A",
            Trt.name = 1,
            VARS. = vars,
            logistic = TRUE,w= NULL)
w <- ps1$Data$ipw_ate_stab

### Get the initial value
EM_initial <- OUT_em_weights(data = data_test,
                             X1 = "X1",
                             X2 = "X2",
                             event1 = "delta1",
                             event2 = "delta2",
                             w = w,
                             Trt = "A")
```

`plot.PS` *Plotting Histogram of Propensity Score and Balancing Plot for Covariates in the Propensity Score Model*

Description

Displays a the histogram plots for the propensity score, stratified by treated and control group and a graph of standardized mean difference of potential confounders before and after weigthing.

Usage

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

Arguments

`x`	The results of `doPS` function.
`...`	the other arguments you want to put in the built-in plot function.

Details

Only available when logistic = FALSE in doPS. The standardized mean difference (SMD), defined as the (weighted) treatment group mean minus the (weighted) control group mean divided by the (weighted) pooled sample (treatment and control) standard deviation. SMD between -0.1 and 0.1 typically indicates good balance.

Value

Histogram of propensity score and balancing plot for covariates in the propensity score model corresponding to the output from `doPS`.

See Also

`bal.table`

---

sim_cox_msm_semicmrsk *Simulating Semi-competing Risks with Right-censored Survival Data under Marginal Structural Illness-death Cox Model*

Description

The function to simulate semi-competing risk with right-censored survival data under marginal structural illness-death Cox model.

Usage

```
sim_cox_msm_semicmrsk(beta1,beta2,beta3,sigma_2,
        alpha0,alpha1,alpha2,alpha3,
        n,Cens)
```

Arguments

`beta1`	True value of β_1 in the illness-death model.
`beta2`	True value of β_2 in the illness-death model.
`beta3`	True value of β_3 in the illness-death model.
`sigma_2`	True value of variance of normal frailty σ^2 in the illness-death model, if $\sigma^2 = 0$, then there is no frailty term.
`alpha0`	True value of α_0 in the propensity score model.
`alpha1`	True value of α_1 in the propensity score model.
`alpha2`	True value of α_2 in the propensity score model.
`alpha3`	True value of α_3 in the propensity score model.
`n`	Sample size.
`Cens`	Censoring distribution.

Details

We simulate data followed by Xu(2010) to generate semi-competing risk data under illness-death model, where we have baseline hazard $\lambda_{01}(t) = \lambda_{02}(t) = 2exp(-t)I(0 \leq t \leq 3)+2exp(-3)I(t \geq 3)$, and $\lambda_{03}(t) = 2\lambda_{01}(t)$.

We also have the propensity score model to generate treatment assignment $P_A = logit^{-1}(\alpha_0 + \alpha_1 Z_1 + \alpha_2 Z_2 + \alpha_3 Z3)$.

Value

Returns a data frame that contains time to non-terminal event, T1, terminal event, T2 and censoring time C with their event indicator, delta1 and delta2. Three covariates Z1, Z2, Z3, and treatment assignment A are also included.

Examples

```
n <- 500
set.seed(1234)
Cens = runif(n,0.7,0.9)
set.seed(1234)
OUT1 <- sim_cox_msm_semicmrsk(beta1 = 1,beta2 = 1,beta3 = 0.5,
                              sigma_2 = 1,
                              alpha0 = 0.5, alpha1 = 0.1, alpha2 = -0.1, alpha3 = -0.2,
                              n=n, Cens = Cens)
data_test <- OUT1$data0
```

`usual_illness_death_weight` *Fit MSM Illness-death Usual Markov Model For Semi-competing Risks Data*

Description

Fit the marginal structural three-state illness-death model with Cox representation and IP weights for semi-competing risks data. Inference under this model can be carried out using estimating equations with IP weights.

Usage

```
usual_illness_death_weight(data,X1,X2,event1,event2,w,Trt)
```

Arguments

`data`	The dataset, includes non-terminal events, terminal events as well as event indicator.
`X1`	Time to non-terminal event, could be censored by terminal event or lost to follow up.
`X2`	Time to terminal event, could be censored by lost to follow up.

event1	Event indicator for non-terminal event.
event2	Event indicator for terminal event.
w	IP weights.
Trt	Treatment variable.

Details

Let T_1, T_2 be the time to non-terminal event and terminal event, A be the treatment assignment. We postulate the semi-parametric Cox models for three transition rates in marginal structural illness-death model:

$$\lambda_1(t_1; a) = \lambda_{01}(t)e^{\beta_1 a}, t_1 > 0;$$

$$\lambda_2(t_2; a) = \lambda_{02}(t)e^{\beta_2 a}, t_2 > 0;$$

and

$$\lambda_{12}(t_2 \mid t_1; a) = \lambda_{03}(t_2)e^{\beta_3 a}, 0 < t_1 < t_2.$$

The coefficients as well as Breslow type baseline hazards can be estimated by fitting the IP weights Cox proportional hazards models. Meanwhile, if we assume the estimated weights as known, then the robust sandwich variance estimator can be used to obtain the estimated variance.

The usual Markov model is also the same as the initial value for the general Markov model.

Value

A list of values and dataframes:

beta1	Estimated β_1, the coefficient for the non-terminal event model.
beta2	Estimated β_2, the coefficient for the terminal event without non-terminal event model.
beta3	Estimated β_3, the coefficient for the terminal event following non-terminal event model.
sd_beta1	Model based standard error for β_1.
sd_beta2	Model based standard error for β_2.
sd_beta3	Model based standard error for β_3.
Lambda01	See also get_hazard. List of two dataframes for estimated Λ_{01} and λ_{01}, the estimated (cumulative) baseline hazard for the non-terminal event model.
Lambda02	List of two dataframes for estimated Λ_{02} and λ_{02}, the estimated (cumulative) baseline hazard for the terminal event without non-terminal event model.
Lambda03	List of two dataframes for estimated Λ_{03} and λ_{03}, the estimated (cumulative) baseline hazard for the terminal event following non-terminal event model.

Examples

```
n <- 500
set.seed(1234)
Cens = runif(n,0.7,0.9)
set.seed(1234)
OUT1 <- sim_cox_msm_semicmrsk(beta1 = 1,beta2 = 1,beta3 = 0.5,
                              sigma_2 = 1,
                              alpha0 = 0.5, alpha1 = 0.1, alpha2 = -0.1, alpha3 = -0.2,
                              n=n, Cens = Cens)
data_test <- OUT1$data0

## Get the PS weights
vars <- c("Z1","Z2","Z3")
ps1 <- doPS(data = data_test,
            Trt = "A",
            Trt.name = 1,
            VARS. = vars,
            logistic = TRUE,w=NULL)
w <- ps1$Data$ipw_ate_stab

### Fit the Usual Markov model
res1 <- usual_illness_death_weight(data = data_test,
                                   X1 = "X1",
                                   X2 = "X2",
                                   event1 = "delta1",
                                   event2 = "delta2",
                                   w = w,
                                   Trt = "A")
print(paste("The estimated value for beta1 is:", round(res1$beta1,5) ) )
```

`var_em_illness_death_phmm` *Variance of parameters in MSM Illness-death General Markov Model*

Description

Use bootstrap to obtain the variance estimator for parameters in MSM illness-death general markov model.

Usage

```
var_em_illness_death_phmm(data,sigma_2_0,VARS.)
```

Arguments

`data`	The output dataset from `em_illness_death_phmm_weight`.
`sigma_2_0`	Initial value for σ^2, the variance of zero-mean normal frailty, usually starts with 1.
`VARS.`	Confounder sets.

Details

See `em_illness_death_phmm_weight`. In each bootstrap, the propensity score model needs to be re-fitted, and fit the MSM illness-death general markov model with new IP weights.

Value

List of bootstrap SE for all the parameters in the general Markov model

Index