Package 'SemiCompRisks'

June 28, 2017

Type Package

Title Hierarchical Models for Parametric and Semi-Parametric Analyses of Semi-Competing Risks Data

Version 2.7

Date 2017-6-28

Author Kyu Ha Lee, Catherine Lee, and Sebastien Haneuse

Maintainer Kyu Ha Lee <klee@hsph.harvard.edu>

Description Parametric and semi-parametric analyses of semi-competing risks/univariate survival data. For semi-competing risks data, the package contains implementations of hierarchical models for independent data (Lee et al., 2015; <doi:10.1111/rssc.12078>, Lee et al. 2017) and cluster-correlated data (Lee et al., 2016; <doi:10.1111/biom.12696> <doi:10.1080/01621459.2016.1164052>).

License GPL (>= 2)

Suggests R.rsp

VignetteBuilder R.rsp

Depends MASS, survival, R (>= 3.3.1)

LazyLoad yes

NeedsCompilation yes

Repository CRAN

Date/Publication 2017-06-28 17:34:45 UTC

R topics documented:

`SemiCompRisks-package` *Algorithms for fitting parametric and semi-parametric models to semi-competing risks data / univariate survival data.*

Description

The package provides functions to perform the analysis of semi-competing risks or univariate survival data with either hazard regression (HReg) models or accelerated failure time (AFT) models. The framework is flexible in the sense that:
1) it can handle cluster-correlated or independent data;
2) the option to choose between parametric (Weibull) and semi-parametric (mixture of piecewise exponential) specification for baseline hazard function(s) is available;
3) for clustered data, the option to choose between parametric (multivariate Normal for semicompeting risks data, Normal for univariate survival data) and semi-parametric (Dirichlet process mixture) specification for random effects distribution is available;
4) for semi-competing risks data, the option to choose between Makov and semi-Makov model is available.

Details

The package includes following functions:

`BayesID_HReg`	Bayesian analysis of semi-competing risks data using HReg models
`BayesID_AFT`	Bayesian analysis of semi-competing risks data using AFT models
`BayesSurv_HReg`	Bayesian analysis of univariate survival data using HReg models
`BayesSurv_AFT`	Bayesian analysis of univariate survival data using AFT models
`FreqID_HReg`	Frequentist analysis of semi-competing risks data using HReg models
`FreqSurv_HReg`	Frequentist analysis of univariate survival data using HReg models
`initiate.startValues_HReg`	Initiating starting values for Bayesian estimations with HReg models
`initiate.startValues_AFT`	Initiating starting values for Bayesian estimations with AFT models
`simID`	Simulating semi-competing risks data under Weibull/Weibull-MVN model
`simSurv`	Simulating survival data under Weibull/Weibull-Normal model

Package: SemiCompRisks

Type:	Package
Version:	2.7
Date:	2017-6-28
License:	GPL (>= 2)
LazyLoad:	yes

Author(s)

Kyu Ha Lee, Catherine Lee, and Sebastien Haneuse
Maintainer: Kyu Ha Lee <klee@hsph.harvard.edu>

References

Lee, K. H., Haneuse, S., Schrag, D., and Dominici, F. (2015), Bayesian semiparametric analysis of semicompeting risks data: investigating hospital readmission after a pancreatic cancer diagnosis, *Journal of the Royal Statistical Society: Series C*, 64, 2, 253-273.

Lee, K. H., Dominici, F., Schrag, D., and Haneuse, S. (2016), Hierarchical models for semicompeting risks data with application to quality of end-of-life care for pancreatic cancer, *Journal of the American Statistical Association*, 111, 515, 1075-1095.

Lee, K. H., Rondeau, V., and Haneuse, S. (2017), Accelerated failure time models for semicompeting risks data in the presence of complex censoring, *Biometrics*, in press.

`BayesID_AFT` *The function to implement Bayesian parametric and semi-parametric analyses for semi-competing risks data in the context of accelerated failure time (AFT) models.*

Description

Independent semi-competing risks data can be analyzed using AFT models that have a hierarchical structure. The proposed models can accomodate left-truncated and/or interval-censored data. An efficient computational algorithm that gives users the flexibility to adopt either a fully parametric (log-Normal) or a semi-parametric (Dirichlet process mixture) model specification is developed.

Usage

```
BayesID_AFT(Y, lin.pred, data, model = "LN", hyperParams, startValues,
mcmcParams, path=NULL)
```

Arguments

Y	a data.frame containing (interval-censored and/or left-truncated) semi-competing risks outcomes from n subjects. It is of dimension $n \times 5$: the columns correspond to c_j, c_{j+1}, c_k, c_{k+1}, L. See Details and Examples below.
lin.pred	a list containing three formula objects whose right-hand side specifies the covariate terms for the transition g=1,2,3.
data	a data.frame in which to interpret the variables named in the formulas in lin.pred.
model	The specification of baseline survival distribution: "LN" or "DPM".
hyperParams	a list containing lists or vectors for hyperparameter values in hierarchical models. Components include, theta (a numeric vector for hyperparameter in the prior of subject-specific frailty variance component), LN (a list containing numeric vectors for log-Normal hyperparameters: LN.ab1, LN.ab2, LN.ab3), DPM (a list containing numeric vectors for DPM hyperparameters: DPM.mu1, DPM.mu2, DPM.mu3, DPM.sigSq1, DPM.sigSq2, DPM.sigSq3, DPM.ab1, DPM.ab2, DPM.ab3, Tau.ab1, Tau.ab2, Tau.ab3). See Details and Examples below.
startValues	a list containing vectors of starting values for model parameters. It can be specified as the object returned by the function initiate.startValues_AFT.
mcmcParams	a list containing variables required for MCMC sampling. Components include, run (a list containing numeric values for setting for the overall run: numReps, total number of scans; thin, extent of thinning; burninPerc, the proportion of burn-in). storage (a list containing numeric values for storing posterior samples for subject- and cluster-specific random effects: nGam_save, the number of γ to be stored; nY1_save, the number of $y1$ to be stored; nY2_save, the number of $y2$ to be stored; nY1.NA_save, the number of $y1.NA$ to be stored). tuning (a list containing numeric values relevant to tuning parameters for specific updates in Metropolis-Hastings (MH) algorithm: mhProp_theta_var, the variance of proposal density for θ; betag.prop.var, the variance of proposal density for β_g; mug.prop.var, the variance of proposal density for μ_g; zetag.prop.var, the variance of proposal density for $1/\sigma_g^2$; gamma.prop.var, the variance of proposal density for γ). See Details and Examples below.
path	the name of directory where the results are saved.

Details

We view the semi-competing risks data as arising from an underlying illness-death model system in which individuals may undergo one or more of three transitions: 1) from some initial condition to non-terminal event, 2) from some initial condition to terminal event, 3) from non-terminal event to terminal event. Let T_{i1}, T_{i2} denote time to non-terminal and terminal event from subject $i = 1, ..., n$. We propose to directly model the times of the events via the following AFT model specification:

$$\log(T_{i1}) = x_{i1}^\top \beta_1 + \gamma_i + \epsilon_{i1}, T_{i1} > 0,$$

$$\log(T_{i2}) = x_{i2}^\top \beta_2 + \gamma_i + \epsilon_{i2}, T_{i2} > 0,$$

$$\log(T_{i2} - T_{i1}) = x_{i3}^\top \beta_3 + \gamma_i + \epsilon_{i3}, T_{i2} > T_{i1},$$

where x_{ig} is a vector of transition-specific covariates, β_g is a corresponding vector of transition-specific regression parameters and ϵ_{ig} is a transition-specific random variable whose distribution

determines that of the corresponding transition time, $g \in \{1, 2, 3\}$. γ_i is a study participant-specific random effect that induces positive dependence between the two event times, thereby performing a role analogous to that performed by frailties in models for the hazard function. Let L_i denote the time at study entry (i.e. the left-truncation time). Furthermore, suppose that study participant i was observed at follow-up times $\{c_{i1}, \ldots, c_{im_i}\}$ and let c_i^* denote the time to the end of study or to administrative right-censoring. Considering interval-censoring for both events, the times to non-terminal and terminal event for the i^{th} study participant satisfy $c_{ij} \leq T_{i1} < c_{ij+1}$ for some j and $c_{ik} \leq T_{i2} < c_{ik+1}$ for some k, respectively. Then the observed outcomes for the i^{th} study participant can be succinctly denoted by $\{c_{ij}, c_{ij+1}, c_{ik}, c_{ik+1}, L_i\}$.

For the Bayesian semi-parametric analysis, we proceed by adopting independent DPM of normal distributions for each ϵ_{ig}. More precisely, ϵ_{ig} is taken to be an independent draw from a mixture of M_g normal distributions with means and variances (μ_{gr}, σ^2_{gr}), for $r \in \{1, \ldots, M_g\}$. Since the class-specific $(\mu_{gr}, \sigma^2_{gr})$ are not known, they are taken to be draws from some common distribution, G_{g0}, often referred to as the centering distribution. Furthermore, since the 'true' class membership for any given study participant is not known, we let p_{gr} denote the probability of belonging to the r^{th} class for transition g and $p_g = (p_{g1}, \ldots, p_{gM_g})$ the collection of such probabilities. Note, p_g is defined at the level of the population (i.e. is not study participant-specific) and its components add up to 1.0. In the absence of prior knowledge regarding the distribution of class memberships for the n individuals across the M_g classes, p_g is assumed to follow a conjugate symmetric Dirichlet$(\tau_g/M_g, \ldots, \tau_g/M_g)$ distribution, where τ_g is referred to as the precision parameter. The finite mixture distribution can then be succinctly represented as:

$$\epsilon_{ig}|r_i \sim Normal(\mu_{r_i}, \sigma^2_{r_i}),$$

$$(\mu_{gr}, \sigma^2_{gr}) \sim G_{g0}, \;\; for\, r = 1, \ldots, M_g,$$

$$r_i|p_g \sim Discrete(r_i|p_{g1}, \ldots, p_{gM_g}),$$

$$p_g \sim Dirichlet(\tau_g/M_g, \ldots, \tau_g/M_g).$$

Letting M_g approach infinity, this specification is referred to as a DPM of normal distributions. In our proposed framework, we specify a Gamma(a_{τ_g}, b_{τ_g}) hyperprior for τ_g. For regression parameters, we adopt non-informative flat priors on the real line. For γ=$\{\gamma_1, \ldots, \gamma_n\}$, we assume that each γ_i is an independent random draw from a Normal(0, θ) distribution. In the absence of prior knowledge on the variance component θ, we adopt a conjugate inverse-Gamma hyperprior, IG(a_θ, b_θ). Finally, We take the G_{g0} as a normal distribution centered at μ_{g0} with a variance σ^2_{g0} for μ_{gr} and an IG(a_{σ_g}, b_{σ_g}) for σ^2_{gr}.

For the Bayesian parametric analysis, we build on the log-Normal formulation and take the ϵ_{ig} to follow independent Normal(μ_g, σ^2_g) distributions, g=1,2,3. For location parameters $\{\mu_1, \mu_2, \mu_3\}$, we adopt non-informative flat priors on the real line. For $\{\sigma^2_1, \sigma^2_2, \sigma^2_3\}$, we adopt independent inverse Gamma distributions, denoted IG($a_{\sigma g}$, $b_{\sigma g}$). For β_g, γ, and θ, we adopt the same priors as those adopted for the DPM model.

Value

`BayesID_AFT` returns an object of class `Bayes_AFT`.

Note

The posterior samples of γ are saved separately in `working directory/path`. For a dataset with large n, `nGam_save` should be carefully specified considering the system memory and the storage capacity.

Author(s)

Kyu Ha Lee and Sebastien Haneuse
Maintainer: Kyu Ha Lee <klee@hsph.harvard.edu>

References

Lee, K. H., Rondeau, V., and Haneuse, S. (2017), Accelerated failure time models for semicompeting risks data in the presence of complex censoring, *Biometrics*, in press.

See Also

`initiate.startValues_AFT`, `print.Bayes_AFT`, `summary.Bayes_AFT`, `plot.Bayes_AFT`

Examples

```
## Not run:
# loading a data set
data(scrData)
Y <- matrix(NA, dim(scrData)[1], 5)
Y[,1] <- Y[,2] <- scrData[,1]
Y[,3] <- Y[,4] <- scrData[,3]
Y[which(scrData[,2] == 0),2] <- Inf
Y[which(scrData[,4] == 0),4] <- Inf
Y[,5] <- rep(0, dim(scrData)[1])

form1 <- as.formula( ~ x1 + x2 + x3)
form2 <- as.formula( ~ x1 + x2)
form3 <- as.formula( ~ x1 + x2)
lin.pred <- list(form1, form2, form3)

######################
## Hyperparameters ##
######################

## Subject-specific random effects variance component
##
theta.ab <- c(0.5, 0.05)

## log-Normal model
##
LN.ab1 <- c(0.3, 0.3)
LN.ab2 <- c(0.3, 0.3)
LN.ab3 <- c(0.3, 0.3)
```

```
## DPM model
##
DPM.mu1 <- log(12)
DPM.mu2 <- log(12)
DPM.mu3 <- log(12)

DPM.sigSq1 <- 100
DPM.sigSq2 <- 100
DPM.sigSq3 <- 100

DPM.ab1 <-  c(2, 1)
DPM.ab2 <-  c(2, 1)
DPM.ab3 <-  c(2, 1)

Tau.ab1 <- c(1.5, 0.0125)
Tau.ab2 <- c(1.5, 0.0125)
Tau.ab3 <- c(1.5, 0.0125)

##
hyperParams <- list(theta=theta.ab,
LN=list(LN.ab1=LN.ab1, LN.ab2=LN.ab2, LN.ab3=LN.ab3),
DPM=list(DPM.mu1=DPM.mu1, DPM.mu2=DPM.mu2, DPM.mu3=DPM.mu3, DPM.sigSq1=DPM.sigSq1,
DPM.sigSq2=DPM.sigSq2, DPM.sigSq3=DPM.sigSq3, DPM.ab1=DPM.ab1, DPM.ab2=DPM.ab2,
DPM.ab3=DPM.ab3, Tau.ab1=Tau.ab1, Tau.ab2=Tau.ab2, Tau.ab3=Tau.ab3))

###################
## MCMC SETTINGS ##
###################

## Setting for the overall run
##
numReps    <- 300
thin       <- 3
burninPerc <- 0.5

## Setting for storage
##
nGam_save <- 10
nY1_save <- 10
nY2_save <- 10
nY1.NA_save <- 10

## Tuning parameters for specific updates
##
##  - those common to all models
betag.prop.var <- c(0.01,0.01,0.01)
mug.prop.var <- c(0.1,0.1,0.1)
zetag.prop.var <- c(0.1,0.1,0.1)
gamma.prop.var <- 0.01

##
mcmcParams <- list(run=list(numReps=numReps, thin=thin, burninPerc=burninPerc),
```

```
storage=list(nGam_save=nGam_save, nY1_save=nY1_save, nY2_save=nY2_save, nY1.NA_save=nY1.NA_save),
tuning=list(betag.prop.var=betag.prop.var, mug.prop.var=mug.prop.var,
zetag.prop.var=zetag.prop.var, gamma.prop.var=gamma.prop.var))

#################################################################
## Analysis of Independent Semi-competing risks data ############
#################################################################

###############
## logNormal ##
###############

##
myModel <- "LN"
myPath  <- "Output/01-Results-LN/"

startValues      <- vector("list", 2)
startValues[[1]] <- initiate.startValues_AFT(Y, lin.pred, scrData, model=myModel)
startValues[[2]] <- initiate.startValues_AFT(Y, lin.pred, scrData, model=myModel, theta = 0.20)

##
fit_LN <- BayesID_AFT(Y, lin.pred, scrData, model=myModel, hyperParams,
startValues, mcmcParams, path=myPath)

fit_LN
summ.fit_LN <- summary(fit_LN); names(summ.fit_LN)
summ.fit_LN
plot(fit_LN, time = seq(0, 35, 1), tseq=seq(from=0, to=30, by=5))
plot(fit_LN, time = seq(0, 35, 1), tseq=seq(from=0, to=30, by=5), plot.est = "BH")
names(fit_LN.plot <- plot(fit_LN, time = seq(0, 35, 1),  tseq=seq(0, 30, 5), plot=FALSE))


#########
## DPM ##
#########

##
myModel <- "DPM"
myPath  <- "Output/02-Results-DPM/"

startValues      <- vector("list", 2)
startValues[[1]] <- initiate.startValues_AFT(Y, lin.pred, scrData, model=myModel)
startValues[[2]] <- initiate.startValues_AFT(Y, lin.pred, scrData, model=myModel, theta = 0.23)

##
fit_DPM <- BayesID_AFT(Y, lin.pred, scrData, model=myModel, hyperParams,
startValues, mcmcParams, path=myPath)

fit_DPM
summ.fit_DPM <- summary(fit_DPM); names(summ.fit_DPM)
summ.fit_DPM
plot(fit_DPM, time = seq(0, 35, 1),  tseq=seq(from=0, to=30, by=5))
plot(fit_DPM, time = seq(0, 35, 1),  tseq=seq(from=0, to=30, by=5), plot.est = "BH")
```

```
names(fit_DPM.plot <- plot(fit_DPM, time = seq(0, 35, 1), tseq=seq(0, 30, 5), plot=FALSE))

## End(Not run)
```

BayesID_HReg *The function to implement Bayesian parametric and semi-parametric analyses for semi-competing risks data in the context of hazard regression (HReg) models.*

Description

Independent/cluster-correlated semi-competing risks data can be analyzed using HReg models that have a hierarchical structure. The priors for baseline hazard functions can be specified by either parametric (Weibull) model or non-parametric mixture of piecewise exponential models (PEM). The option to choose between parametric multivariate normal (MVN) and non-parametric Dirichlet process mixture of multivariate normals (DPM) is available for the prior of cluster-specific random effects distribution. The conditional hazard function for time to the terminal event given time to non-terminal event can be modeled based on either Markov (it does not depend on the timing of the non-terminal event) or semi-Markov assumption (it does depend on the timing).

Usage

```
BayesID_HReg(Y, lin.pred, data, cluster=NULL, model=c("semi-Markov", "Weibull"),
        hyperParams, startValues, mcmc, path=NULL)
```

Arguments

Y	a data.frame containing semi-competing risks outcomes from n subjects. It is of dimension $n \times 4$: the columns correspond to y_1, δ_1, y_2, δ_2.
lin.pred	a list containing three formula objects that correspond to h_g, g=1,2,3.
data	a data.frame in which to interpret the variables named in the formulas in lin.pred.
cluster	a vector of cluster information for n subjects. The cluster membership must be consecutive positive integers, $1 : J$.
model	a character vector that specifies the type of components in a model. The first element is for the assumption on h_3: "semi-Markov" or "Markov". The second element is for the specification of baseline hazard functions: "Weibull" or "PEM". The third element needs to be set only for clustered semi-competing risks data and is for the specification of cluster-specific random effects distribution: "MVN" or "DPM".
hyperParams	a list containing lists or vectors for hyperparameter values in hierarchical models. Components include, theta (a numeric vector for hyperparameter in the prior of subject-specific frailty variance component), WB (a list containing numeric vectors for Weibull hyperparameters: WB.ab1, WB.ab2, WB.ab3), PEM (a list containing numeric vectors for PEM hyperparameters: PEM.ab1, PEM.ab2, PEM.ab3, PEM.alpha1, PEM.alpha2, PEM.alpha3). Models for clustered data

require additional components, `MVN` (a list containing numeric vectors for MVN hyperparameters: `Psi_v`, `rho_v`), `DPM` (a list containing numeric vectors for DPM hyperparameters: `Psi0`, `rho0`, `aTau`, `bTau`). See Details and Examples below.

`startValues` a list containing vectors of starting values for model parameters. It can be specified as the object returned by the function `initiate.startValues_HReg`.

`mcmc` a list containing variables required for MCMC sampling. Components include, `run` (a list containing numeric values for setting for the overall run: `numReps`, total number of scans; `thin`, extent of thinning; `burninPerc`, the proportion of burn-in). `storage` (a list containing numeric values for storing posterior samples for subject- and cluster-specific random effects: `nGam_save`, the number of γ to be stored; `storeV`, a vector of three logical values to determine whether all the posterior samples of V_1, V_2, V_3 are to be stored). `tuning` (a list containing numeric values relevant to tuning parameters for specific updates in Metropolis-Hastings-Green (MHG) algorithm: `mhProp_theta_var`, the variance of proposal density for θ; `mhProp_Vg_var`, the variance of proposal density for V_g in DPM models; `mhProp_alphag_var`, the variance of proposal density for α_g in Weibull models; `Cg`, a vector of three proportions that determine the sum of probabilities of choosing the birth and the death moves in PEM models. The sum of the three elements should not exceed 0.6; `delPertg`, the perturbation parameters in the birth update in PEM models. The values must be between 0 and 0.5; If `rj.scheme`=1, the birth update will draw the proposal time split from $1 : s_{max}$. If `rj.scheme`=2, the birth update will draw the proposal time split from uniquely ordered failure times in the data. Only required for PEM models; `Kg_max`, the maximum number of splits allowed at each iteration in MHG algorithm for PEM models; `time_lambda1`, `time_lambda2`, `time_lambda3` - time points at which the log-hazard functions are calculated for `plot.Bayes_HReg`, Only required for PEM models). See Details and Examples below.

`path` the name of directory where the results are saved.

Details

We view the semi-competing risks data as arising from an underlying illness-death model system in which individuals may undergo one or more of three transitions: 1) from some initial condition to non-terminal event, 2) from some initial condition to terminal event, 3) from non-terminal event to terminal event. Let t_{ji1}, t_{ji2} denote time to non-terminal and terminal event from subject $i = 1, ..., n_j$ in cluster $j = 1, ..., J$. The system of transitions is modeled via the specification of three hazard functions:

$$h_1(t_{ji1}|\gamma_{ji}, x_{ji1}, V_{j1}) = \gamma_{ji} h_{01}(t_{ji1}) \exp(x_{ji1}^\top \beta_1 + V_{j1}), t_{ji1} > 0,$$

$$h_2(t_{ji2}|\gamma_{ji}, x_{ji2}, V_{j2}) = \gamma_{ji} h_{02}(t_{ji2}) \exp(x_{ji2}^\top \beta_2 + V_{j2}), t_{ji2} > 0,$$

$$h_3(t_{ji2}|t_{ji1}, \gamma_{ji}, x_{ji3}, V_{j3}) = \gamma_{ji} h_{03}(t_{ji2}) \exp(x_{ji3}^\top \beta_3 + V_{j3}), 0 < t_{ji1} < t_{ji2},$$

where γ_{ji} is a subject-specific frailty and V_j=(V_{j1}, V_{j2}, V_{j3}) is a vector of cluster-specific random effects (each specific to one of the three possible transitions), taken to be independent of x_{ji1}, x_{ji2}, and x_{ji3}. For $g \in \{1, 2, 3\}$, h_{0g} is an unspecified baseline hazard function and β_g is a vector of p_g log-hazard ratio regression parameters. The h_{03} is assumed to be Markov with respect to t_1.

We refer to the model specified by three conditional hazard functions as the Markov model. An alternative specification is to model the risk of terminal event following non-terminal event as a function of the sojourn time. Specifically, retaining h_1 and h_2 as above, we consider modeling h_3 as follows:

$$h_3(t_{ji2}|t_{ji1}, \gamma_{ji}, x_{ji3}, V_{j3}) = \gamma_{ji} h_{03}(t_{ji2} - t_{ji1}) \exp(x_{ji3}^{\top}\beta_3 + V_{j3}), 0 < t_{ji1} < t_{ji2}.$$

We refer to this alternative model as the semi-Markov model.
For parametric MVN prior specification for a vector of cluster-specific random effects, we assume V_j arise as i.i.d. draws from a mean 0 MVN distribution with variance-covariance matrix Σ_V. The diagonal elements of the 3×3 matrix Σ_V characterize variation across clusters in risk for non-terminal, terminal and terminal following non-terminal event, respectively, which is not explained by covariates included in the linear predictors. Specifically, the priors can be written as follows:

$$V_j \sim MVN(0, \Sigma_V),$$

$$\Sigma_V \sim inverse - Wishart(\Psi_v, \rho_v).$$

For DPM prior specification for V_j, we consider non-parametric Dirichlet process mixture of MVN distributions: the V_j's are draws from a finite mixture of M multivariate Normal distributions, each with their own mean vector and variance-covariance matrix, (μ_m, Σ_m) for $m = 1, ..., M$. Let $m_j \in \{1, ..., M\}$ denote the specific component to which the jth cluster belongs. Since the class-specific (μ_m, Σ_m) are unknown they are taken to be draws from some distribution, G_0, often referred to as the centering distribution. Furthermore, since the true class memberships are not known, we denote the probability that the jth cluster belongs to any given class by the vector $p = (p_1, ..., p_M)$ whose components add up to 1.0. In the absence of prior knowledge regarding the distribution of class memberships for the J clusters across the M classes, a natural prior for p is the conjugate symmetric $Dirichlet(\tau/M, ..., \tau/M)$ distribution; the hyperparameter, τ, is often referred to as a the precision parameter. The prior can be represented as follows (M goes to infinity):

$$V_j|m_j \sim MVN(\mu_{m_j}, \Sigma_{m_j}),$$

$$(\mu_m, \Sigma_m) \sim G_0, \quad for\ m = 1, ..., M,$$

$$m_j|p \sim Discrete(m_j|p_1, ..., p_M),$$

$$p \sim Dirichlet(\tau/M, ..., \tau/M),$$

where G_0 is taken to be a multivariate Normal/inverse-Wishart (NIW) distribution for which the probability density function is the following product:

$$f_{NIW}(\mu, \Sigma|\Psi_0, \rho_0) = f_{MVN}(\mu|0, \Sigma) \times f_{inv-Wishart}(\Sigma|\Psi_0, \rho_0).$$

We consider $Gamma(a_\tau, b_\tau)$ as the prior for concentration parameter τ.

For non-parametric PEM prior specification for baseline hazard functions, let $s_{g,\max}$ denote the largest observed event time for each transition $g \in \{1, 2, 3\}$. Then, consider the finite partition of the relevant time axis into $K_g + 1$ disjoint intervals: $0 < s_{g,1} < s_{g,2} < ... < s_{g,K_g+1} = s_{g,\max}$. For notational convenience, let $I_{g,k} = (s_{g,k-1}, s_{g,k}]$ denote the k^{th} partition. For given a partition, $s_g = (s_{g,1}, \ldots, s_{g,K_g+1})$, we assume the log-baseline hazard functions is piecewise constant:

$$\lambda_{0g}(t) = \log h_{0g}(t) = \sum_{k=1}^{K_g+1} \lambda_{g,k} I(t \in I_{g,k}),$$

where $I(\cdot)$ is the indicator function and $s_{g,0} \equiv 0$. Note, this specification is general in that the partitions of the time axes differ across the three hazard functions. our prior choices are, for $g \in \{1, 2, 3\}$:

$$\lambda_g | K_g, \mu_{\lambda_g}, \sigma^2_{\lambda_g} \sim MVN_{K_g+1}(\mu_{\lambda_g} 1, \sigma^2_{\lambda_g} \Sigma_{\lambda_g}),$$

$$K_g \sim Poisson(\alpha_g),$$

$$\pi(s_g | K_g) \propto \frac{(2K_g + 1)! \prod_{k=1}^{K_g+1} (s_{g,k} - s_{g,k-1})}{(s_{g,K_g+1})^{(2K_g+1)}},$$

$$\pi(\mu_{\lambda_g}) \propto 1,$$

$$\sigma^{-2}_{\lambda_g} \sim Gamma(a_g, b_g).$$

Note that K_g and s_g are treated as random and the priors for K_g and s_g jointly form a time-homogeneous Poisson process prior for the partition. The number of time splits and their positions are therefore updated within our computational scheme using reversible jump MCMC.

For parametric Weibull prior specification for baseline hazard functions, $h_{0g}(t) = \alpha_g \kappa_g t^{\alpha_g - 1}$. In our Bayesian framework, our prior choices are, for $g \in \{1, 2, 3\}$:

$$\pi(\alpha_g) \sim Gamma(a_g, b_g),$$

$$\pi(\kappa_g) \sim Gamma(c_g, d_g).$$

Our prior choice for remaining model parameters in all of four models (Weibull-MVN, Weibull-DPM, PEM-MVN, PEM-DPM) is given as follows:

$$\pi(\beta_g) \propto 1,$$

$$\gamma_{ji} | \theta \sim Gamma(\theta^{-1}, \theta^{-1}),$$

$$\theta^{-1} \sim Gamma(\psi, \omega).$$

We provide a detailed description of the hierarchical models for cluster-correlated semi-competing risks data. The models for independent semi-competing risks data can be obtained by removing cluster-specific random effects, V_j, and its corresponding prior specification from the description given above.

Value

`BayesID_HReg` returns an object of class `Bayes_HReg`.

Note

The posterior samples of γ and V_g are saved separately in `working directory/path`. For a dataset with large n, `nGam_save` should be carefully specified considering the system memory and the storage capacity.

Author(s)

Kyu Ha Lee and Sebastien Haneuse
Maintainer: Kyu Ha Lee <klee@hsph.harvard.edu>

References

Lee, K. H., Haneuse, S., Schrag, D., and Dominici, F. (2015), Bayesian semiparametric analysis of semicompeting risks data: investigating hospital readmission after a pancreatic cancer diagnosis, *Journal of the Royal Statistical Society: Series C*, 64, 2, 253-273.

Lee, K. H., Dominici, F., Schrag, D., and Haneuse, S. (2016), Hierarchical models for semicompeting risks data with application to quality of end-of-life care for pancreatic cancer, *Journal of the American Statistical Association*, 111, 515, 1075-1095.

See Also

initiate.startValues_HReg, print.Bayes_HReg, summary.Bayes_HReg, plot.Bayes_HReg

Examples

```
## Not run:
# loading a data set
data(scrData)
Y <- scrData[,c(1,2,3,4)]
cluster <- scrData[,5]
form1 <- as.formula( ~ x1 + x2 + x3)
form2 <- as.formula( ~ x1 + x2)
form3 <- as.formula( ~ x1 + x2)
lin.pred <- list(form1, form2, form3)

######################
## Hyperparameters ##
######################

## Subject-specific frailty variance component
##  - prior parameters for 1/theta
##
theta.ab <- c(0.7, 0.7)

## Weibull baseline hazard function: alphas, kappas
##
WB.ab1 <- c(0.5, 0.01) # prior parameters for alpha1
WB.ab2 <- c(0.5, 0.01) # prior parameters for alpha2
WB.ab3 <- c(0.5, 0.01) # prior parameters for alpha3
##
WB.cd1 <- c(0.5, 0.05) # prior parameters for kappa1
WB.cd2 <- c(0.5, 0.05) # prior parameters for kappa2
WB.cd3 <- c(0.5, 0.05) # prior parameters for kappa3

## PEM baseline hazard function
##
PEM.ab1 <- c(0.7, 0.7) # prior parameters for 1/sigma_1^2
PEM.ab2 <- c(0.7, 0.7) # prior parameters for 1/sigma_2^2
PEM.ab3 <- c(0.7, 0.7) # prior parameters for 1/sigma_3^2
```

```
##
PEM.alpha1 <- 10 # prior parameters for K1
PEM.alpha2 <- 10 # prior parameters for K2
PEM.alpha3 <- 10 # prior parameters for K3

## MVN cluster-specific random effects
##
Psi_v <- diag(1, 3)
rho_v <- 100

## DPM cluster-specific random effects
##
Psi0  <- diag(1, 3)
rho0  <- 10
aTau  <- 1.5
bTau  <- 0.0125

##
hyperParams <- list(theta=theta.ab,
                WB=list(WB.ab1=WB.ab1, WB.ab2=WB.ab2, WB.ab3=WB.ab3,
                       WB.cd1=WB.cd1, WB.cd2=WB.cd2, WB.cd3=WB.cd3),
                   PEM=list(PEM.ab1=PEM.ab1, PEM.ab2=PEM.ab2, PEM.ab3=PEM.ab3,
                   PEM.alpha1=PEM.alpha1, PEM.alpha2=PEM.alpha2, PEM.alpha3=PEM.alpha3),
                   MVN=list(Psi_v=Psi_v, rho_v=rho_v),
                   DPM=list(Psi0=Psi0, rho0=rho0, aTau=aTau, bTau=bTau))

###################
## MCMC SETTINGS ##
###################

## Setting for the overall run
##
numReps    <- 2000
thin       <- 10
burninPerc <- 0.5

## Settings for storage
##
nGam_save <- 0
storeV    <- rep(TRUE, 3)

## Tuning parameters for specific updates
##
##  - those common to all models
mhProp_theta_var  <- 0.05
mhProp_Vg_var     <- c(0.05, 0.05, 0.05)
##
## - those specific to the Weibull specification of the baseline hazard functions
mhProp_alphag_var <- c(0.01, 0.01, 0.01)
##
## - those specific to the PEM specification of the baseline hazard functions
Cg        <- c(0.2, 0.2, 0.2)
delPertg  <- c(0.5, 0.5, 0.5)
```

```
rj.scheme <- 1
Kg_max    <- c(50, 50, 50)
sg_max    <- c(max(Y$time1[Y$event1 == 1]),
               max(Y$time2[Y$event1 == 0 & Y$event2 == 1]),
               max(Y$time2[Y$event1 == 1 & Y$event2 == 1]))

time_lambda1 <- seq(1, sg_max[1], 1)
time_lambda2 <- seq(1, sg_max[2], 1)
time_lambda3 <- seq(1, sg_max[3], 1)

##
mcmc.WB  <- list(run=list(numReps=numReps, thin=thin, burninPerc=burninPerc),
                 storage=list(nGam_save=nGam_save, storeV=storeV),
                 tuning=list(mhProp_theta_var=mhProp_theta_var,
                 mhProp_Vg_var=mhProp_Vg_var, mhProp_alphag_var=mhProp_alphag_var))

##
mcmc.PEM <- list(run=list(numReps=numReps, thin=thin, burninPerc=burninPerc),
                 storage=list(nGam_save=nGam_save, storeV=storeV),
                 tuning=list(mhProp_theta_var=mhProp_theta_var,
                 mhProp_Vg_var=mhProp_Vg_var, Cg=Cg, delPertg=delPertg,
                 rj.scheme=rj.scheme, Kg_max=Kg_max,
                 time_lambda1=time_lambda1, time_lambda2=time_lambda2,
                 time_lambda3=time_lambda3))

#####################
## Starting Values ##
#####################

##
Sigma_V <- diag(0.1, 3)
Sigma_V[1,2] <- Sigma_V[2,1] <- -0.05
Sigma_V[1,3] <- Sigma_V[3,1] <- -0.06
Sigma_V[2,3] <- Sigma_V[3,2] <- 0.07

#####################################################################
## Analysis of Independent Semi-competing risks data ############
#####################################################################

#############
## WEIBULL ##
#############

##
myModel <- c("semi-Markov", "Weibull")
myPath  <- "Output/01-Results-WB/"

startValues      <- vector("list", 2)
startValues[[1]] <- initiate.startValues_HReg(Y, lin.pred, scrData, model=myModel)
startValues[[2]] <- initiate.startValues_HReg(Y, lin.pred, scrData, model=myModel, theta = 0.23)

##
fit_WB <- BayesID_HReg(Y, lin.pred, scrData, cluster=NULL, model=myModel,
```

```
                hyperParams, startValues, mcmc.WB, path=myPath)

fit_WB
summ.fit_WB <- summary(fit_WB); names(summ.fit_WB)
summ.fit_WB
plot(fit_WB, tseq=seq(from=0, to=30, by=5))
plot(fit_WB, tseq=seq(from=0, to=30, by=5), plot.est = "BH")
names(fit_WB.plot <- plot(fit_WB, tseq=seq(0, 30, 5), plot=FALSE))

#########
## PEM ##
#########

##
myModel <- c("semi-Markov", "PEM")
myPath  <- "Output/02-Results-PEM/"

startValues      <- vector("list", 2)
startValues[[1]] <- initiate.startValues_HReg(Y, lin.pred, scrData, model=myModel)
startValues[[2]] <- initiate.startValues_HReg(Y, lin.pred, scrData, model=myModel, theta = 0.23)

##
fit_PEM <- BayesID_HReg(Y, lin.pred, scrData, cluster=NULL, model=myModel,
                hyperParams, startValues, mcmc.PEM, path=myPath)

fit_PEM
summ.fit_PEM <- summary(fit_PEM); names(summ.fit_PEM)
summ.fit_PEM
plot(fit_PEM)
plot(fit_PEM, plot.est = "BH")
names(fit_PEM.plot <- plot(fit_PEM, plot=FALSE))

#####################################################################
## Analysis of Correlated Semi-competing risks data #############
#####################################################################

#################
## WEIBULL-MVN ##
#################

##
myModel <- c("semi-Markov", "Weibull", "MVN")
myPath  <- "Output/03-Results-WB_MVN/"

startValues      <- vector("list", 2)
startValues[[1]] <- initiate.startValues_HReg(Y, lin.pred, scrData, model=myModel, cluster)
startValues[[2]] <- initiate.startValues_HReg(Y, lin.pred, scrData, model=myModel, cluster,
MVN.SigmaV=Sigma_V)

##
fit_WB_MVN <- BayesID_HReg(Y, lin.pred, scrData, cluster, model=myModel,
                    hyperParams, startValues, mcmc.WB, path=myPath)
```

```
fit_WB_MVN
summ.fit_WB_MVN <- summary(fit_WB_MVN); names(summ.fit_WB_MVN)
summ.fit_WB_MVN
plot(fit_WB_MVN, tseq=seq(from=0, to=30, by=5))
plot(fit_WB_MVN, tseq=seq(from=0, to=30, by=5), plot.est = "BH")
names(fit_WB_MVN.plot <- plot(fit_WB_MVN, tseq=seq(0, 30, 5), plot=FALSE))

#################
## WEIBULL-DPM ##
#################

##
myModel <- c("semi-Markov", "Weibull", "DPM")
myPath  <- "Output/04-Results-WB_DPM/"

startValues      <- vector("list", 2)
startValues[[1]] <- initiate.startValues_HReg(Y, lin.pred, scrData, model=myModel, cluster)
startValues[[2]] <- initiate.startValues_HReg(Y, lin.pred, scrData, model=myModel, cluster,
                    MVN.SigmaV=Sigma_V)

##
fit_WB_DPM <- BayesID_HReg(Y, lin.pred, scrData, cluster, model=myModel,
                    hyperParams, startValues, mcmc.WB, path=myPath)

fit_WB_DPM
summ.fit_WB_DPM <- summary(fit_WB_DPM); names(summ.fit_WB_DPM)
summ.fit_WB_DPM
plot(fit_WB_DPM, tseq=seq(from=0, to=30, by=5))
plot(fit_WB_DPM, tseq=seq(from=0, to=30, by=5), plot.est = "BH")
names(fit_WB_DPM.plot <- plot(fit_WB_DPM, tseq=seq(0, 30, 5), plot=FALSE))

##############
## PEM-MVN ##
##############

##
myModel <- c("semi-Markov", "PEM", "MVN")
myPath  <- "Output/05-Results-PEM_MVN/"

startValues      <- vector("list", 2)
startValues[[1]] <- initiate.startValues_HReg(Y, lin.pred, scrData, model=myModel, cluster)
startValues[[2]] <- initiate.startValues_HReg(Y, lin.pred, scrData, model=myModel, cluster,
                    MVN.SigmaV=Sigma_V)

##
fit_PEM_MVN <- BayesID_HReg(Y, lin.pred, scrData, cluster, model=myModel,
                    hyperParams, startValues, mcmc.PEM, path=myPath)

fit_PEM_MVN
summ.fit_PEM_MVN <- summary(fit_PEM_MVN); names(summ.fit_PEM_MVN)
summ.fit_PEM_MVN
plot(fit_PEM_MVN)
plot(fit_PEM_MVN, plot.est = "BH")
```

```
names(fit_PEM_MVN.plot <- plot(fit_PEM_MVN, plot=FALSE))

#############
## PEM-DPM ##
#############

##
myModel <- c("semi-Markov", "PEM", "DPM")
myPath  <- "Output/06-Results-PEM_DPM/"

startValues      <- vector("list", 2)
startValues[[1]] <- initiate.startValues_HReg(Y, lin.pred, scrData, model=myModel, cluster)
startValues[[2]] <- initiate.startValues_HReg(Y, lin.pred, scrData, model=myModel, cluster,
                    MVN.SigmaV=Sigma_V)

##
fit_PEM_DPM <- BayesID_HReg(Y, lin.pred, scrData, cluster, model=myModel,
                    hyperParams, startValues, mcmc.PEM, path=myPath)

fit_PEM_DPM
summ.fit_PEM_DPM <- summary(fit_PEM_DPM); names(summ.fit_PEM_DPM)
summ.fit_PEM_DPM
plot(fit_PEM_DPM)
plot(fit_PEM_DPM, plot.est = "BH")
names(fit_PEM_DPM.plot <- plot(fit_PEM_DPM, plot=FALSE))


## End(Not run)
```

BayesSurv_AFT *The function to implement Bayesian parametric and semi-parametric analyses for univariate survival data in the context of accelerated failure time (AFT) models.*

Description

Independent univariate survival data can be analyzed using AFT models that have a hierarchical structure. The proposed models can accomodate left-truncated and/or interval-censored data. An efficient computational algorithm that gives users the flexibility to adopt either a fully parametric (log-Normal) or a semi-parametric (Dirichlet process mixture) model specification is developed.

Usage

```
BayesSurv_AFT(Y, lin.pred, data, model = "LN", hyperParams, startValues,
                mcmcParams, path = NULL)
```

Arguments

Y	a data.frame containing (interval-censored and/or left-truncated) univariate time-to-event outcome from n subjects. It is of dimension $n \times 3$: the columns correspond to c_j, c_{j+1}, L. See Details and Examples below.
lin.pred	a formula object whose right-hand side specifies the covariate terms.
data	a data.frame in which to interpret the variables named in the formulas in lin.pred.
model	The specification of baseline survival distribution: "LN" or "DPM".
hyperParams	a list containing lists or vectors for hyperparameter values in hierarchical models. Components include, LN (a list containing numeric vectors for log-Normal hyperparameters: LN.ab), DPM (a list containing numeric vectors for DPM hyperparameters: DPM.mu, DPM.sigSq, DPM.ab, Tau.ab). See Details and Examples below.
startValues	a list containing vectors of starting values for model parameters. It can be specified as the object returned by the function initiate.startValues_AFT.
mcmcParams	a list containing variables required for MCMC sampling. Components include, run (a list containing numeric values for setting for the overall run: numReps, total number of scans; thin, extent of thinning; burninPerc, the proportion of burn-in). tuning (a list containing numeric values relevant to tuning parameters for specific updates in Metropolis-Hastings (MH) algorithm: beta.prop.var, the variance of proposal density for β; mu.prop.var, the variance of proposal density for μ; zeta.prop.var, the variance of proposal density for $1/\sigma^2$).
path	the name of directory where the results are saved.

Details

The function BayesSurv_AFT implements Bayesian semi-parametric (DPM) and parametric (log-Normal) models to univariate time-to-event data in the presence of left-truncation and/or interval-censoring. Consider a univariate AFT model that relates the covariate x_i to survival time T_i for the i^{th} subject:

$$\log(T_i) = x_i^\top \beta + \epsilon_i,$$

where ϵ_i is a random variable whose distribution determines that of T_i and β is a vector of regression parameters. Considering the interval censoring, the time to the event for the i^{th} subject satisfies $c_{ij} \leq T_i < c_{ij+1}$. Let L_i denote the left-truncation time. For the Bayesian parametric analysis, we take ϵ_i to follow the Normal(μ, σ^2) distribution for ϵ_i. The following prior distributions are adopted for the model parameters:

$$\pi(\beta, \mu) \propto 1,$$

$$\sigma^2 \sim \text{Inverse-Gamma}(a_\sigma, b_\sigma).$$

For the Bayesian semi-parametric analysis, we assume that ϵ_i is taken as draws from the DPM of normal distributions:

$$\epsilon \sim DPM(G_0, \tau).$$

We refer readers to print.Bayes_AFT for a detailed illustration of DPM specification. We adopt a non-informative flat prior on the real line for the regression parameters β and a Gamma(a_τ, b_τ) hyperprior for the precision parameter τ.

Value

BayesSurv_AFT returns an object of class Bayes_AFT.

Author(s)

Kyu Ha Lee and Sebastien Haneuse
Maintainer: Kyu Ha Lee <klee@hsph.harvard.edu>

References

Lee, K. H., Rondeau, V., and Haneuse, S. (2017), Accelerated failure time models for semicompeting risks data in the presence of complex censoring, *Biometrics*, in press.

See Also

initiate.startValues_AFT, print.Bayes_AFT, summary.Bayes_AFT, plot.Bayes_AFT

Examples

```
## Not run:
# loading a data set
data(survData)
Y <- matrix(NA, dim(survData)[1], 3)
Y[,1] <- Y[,2] <- survData[,1]
Y[which(scrData[,2] == 0),2] <- Inf
Y[,3] <- rep(0, dim(survData)[1])

lin.pred <- as.formula( ~ cov1 + cov2)

#####################
## Hyperparameters ##
#####################

## log-Normal model
##
LN.ab <- c(0.3, 0.3)

## DPM model
##
DPM.mu <- log(12)
DPM.sigSq <- 100
DPM.ab <-  c(2, 1)
Tau.ab <- c(1.5, 0.0125)

##
hyperParams <- list(LN=list(LN.ab=LN.ab),
DPM=list(DPM.mu=DPM.mu, DPM.sigSq=DPM.sigSq, DPM.ab=DPM.ab, Tau.ab=Tau.ab))
```

```
###################
## MCMC SETTINGS ##
###################

## Setting for the overall run
##
numReps    <- 100
thin       <- 1
burninPerc <- 0.5

## Tuning parameters for specific updates
##
##  - those common to all models
beta.prop.var <- 0.01
mu.prop.var <- 0.1
zeta.prop.var <- 0.1

##
mcmcParams <- list(run=list(numReps=numReps, thin=thin, burninPerc=burninPerc),
tuning=list(beta.prop.var=beta.prop.var, mu.prop.var=mu.prop.var,
zeta.prop.var=zeta.prop.var))

#####################################################################
## Analysis of Independent univariate survival data #############
#####################################################################

###############
## logNormal ##
###############

##
myModel <- "LN"
myPath  <- "Output/01-Results-LN/"

startValues      <- vector("list", 2)
startValues[[1]] <- initiate.startValues_AFT(Y, lin.pred, survData, model=myModel)
startValues[[2]] <- initiate.startValues_AFT(Y, lin.pred, survData, model=myModel,
beta=c(0.05, -0.05))

##
fit_LN <- BayesSurv_AFT(Y, lin.pred, survData, model=myModel, hyperParams,
startValues, mcmcParams, path=myPath)

fit_LN
summ.fit_LN <- summary(fit_LN); names(summ.fit_LN)
summ.fit_LN
plot(fit_LN, time = seq(0, 35, 1), tseq=seq(from=0, to=30, by=5))
plot(fit_LN, time = seq(0, 35, 1), tseq=seq(from=0, to=30, by=5), plot.est = "BH")
names(fit_LN.plot <- plot(fit_LN, time = seq(0, 35, 1),  tseq=seq(0, 30, 5), plot=FALSE))


#########
## DPM ##
```

```
#########

##
myModel <- "DPM"
myPath  <- "Output/02-Results-DPM/"

startValues      <- vector("list", 2)
startValues[[1]] <- initiate.startValues_AFT(Y, lin.pred, survData, model=myModel)
startValues[[2]] <- initiate.startValues_AFT(Y, lin.pred, survData, model=myModel,
beta=c(0.05, -0.05))

##
fit_DPM <- BayesSurv_AFT(Y, lin.pred, survData, model=myModel, hyperParams,
startValues, mcmcParams, path=myPath)

fit_DPM
summ.fit_DPM <- summary(fit_DPM); names(summ.fit_DPM)
summ.fit_DPM
plot(fit_DPM, time = seq(0, 35, 1),  tseq=seq(from=0, to=30, by=5))
plot(fit_DPM, time = seq(0, 35, 1),  tseq=seq(from=0, to=30, by=5), plot.est = "BH")
names(fit_DPM.plot <- plot(fit_DPM, time = seq(0, 35, 1), tseq=seq(0, 30, 5), plot=FALSE))

## End(Not run)
```

`BayesSurv_HReg` *The function to implement Bayesian parametric and semi-parametric regression analyses for univariate time-to-event data in the context of hazard regression (HReg) models.*

Description

Independent/cluster-correlated univariate right-censored survival data can be analyzed using hierarchical models. The prior for the baseline hazard function can be specified by either parametric (Weibull) model or non-parametric mixture of piecewise exponential models (PEM).

Usage

```
BayesSurv_HReg(Y, lin.pred, data, cluster=NULL, model="Weibull",
          hyperParams, startValues, mcmc, path=NULL)
```

Arguments

`Y`	a data.frame containing univariate time-to-event outcomes from `n` subjects. It is of dimension $n \times 2$: the columns correspond to y, δ.
`lin.pred`	a formula object that corresponds to h.
`data`	a data.frame in which to interpret the variables named in the formula in `lin.pred`.
`cluster`	a vector of cluster information for `n` subjects. The cluster membership must be consecutive positive integers, $1 : J$.

`model` a character vector that specifies the type of components in a model. The first element is for the specification of baseline hazard functions: "Weibull" or "PEM". The second element needs to be set only for clustered data and is for the specification of cluster-specific random effects distribution: "Normal" or "DPM".

`hyperParams` a list containing lists or vectors for hyperparameter values in hierarchical models. Components include, `WB` (a list containing a numeric vector for Weibull hyperparameters: `WB.ab`), `PEM` (a list containing numeric vectors for PEM hyperparameters: `PEM.ab`, `PEM.alpha`). Models for clustered data require additional components, `Normal` (a list containing a numeric vector for hyperparameters in a Normal prior: `Normal.ab`), `DPM` (a list containing numeric vectors for DPM hyperparameters: `DPM.ab`, `aTau`, `bTau`). See Details and Examples below.

`startValues` a list containing vectors of starting values for model parameters. It can be specified as the object returned by the function `initiate.startValues_HReg`.

`mcmc` a list containing variables required for MCMC sampling. Components include, `run` (a list containing numeric values for setting for the overall run: `numReps`, total number of scans; `thin`, extent of thinning; `burninPerc`, the proportion of burn-in). `storage` (a list containing numeric values for storing posterior samples for cluster-specific random effects: `storeV`, a logical value to determine whether all the posterior samples of V are to be stored.) `tuning` (a list containing numeric values relevant to tuning parameters for specific updates in Metropolis-Hastings-Green (MHG) algorithm: `mhProp_V_var`, the variance of proposal density for V in DPM models; `mhProp_alpha_var`, the variance of proposal density for α in Weibull models; `C`, a numeric value for the proportion that determines the sum of probabilities of choosing the birth and the death moves in PEM models. The value should not exceed 0.8; `delPert`, the perturbation parameter in the birth update in PEM models. The values must be between 0 and 0.5; If `rj.scheme`=1, the birth update will draw the proposal time split from $1 : s_{max}$. If `rj.scheme`=2, the birth update will draw the proposal time split from uniquely ordered failure times in the data. Only required for PEM models; `K_max`, the maximum number of splits allowed at each iteration in MHG algorithm for PEM models; `time_lambda` - time points at which the log-hazard function is calculated for `plot.Bayes_HReg`, Only required for PEM models). See Details and Examples below.

`path` the name of directory where the results are saved.

Details

The function `BayesSurv_HReg` implements Bayesian semi-parametric (piecewise exponential mixture) and parametric (Weibull) models to univariate time-to-event data. Let t_{ji} denote time to event of interest from subject $i = 1, ..., n_j$ in cluster $j = 1, ..., J$. The covariates x_{ji} are incorporated via Cox proportional hazards model:

$$h(t_{ji}|x_{ji}) = h_0(t_{ji}) \exp(x_{ji}^\top \beta + V_j), t_{ji} > 0,$$

where h_0 is an unspecified baseline hazard function and β is a vector of p log-hazard ratio regression parameters. V_j's are cluster-specific random effects. For parametric Normal prior specification for a vector of cluster-specific random effects, we assume V arise as i.i.d. draws from a mean 0 Normal

distribution with variance σ^2. Specifically, the priors can be written as follows:

$$V_j \sim Normal(0, \sigma^2),$$

$$\zeta = 1/\sigma^2 \sim Gamma(a_N, b_N).$$

For DPM prior specification for V_j, we consider non-parametric Dirichlet process mixture of Normal distributions: the V_j's' are draws from a finite mixture of M Normal distributions, each with their own mean and variance, (μ_m, σ_m^2) for $m = 1, ..., M$. Let $m_j \in \{1, ..., M\}$ denote the specific component to which the jth cluster belongs. Since the class-specific (μ_m, σ_m^2) are not known they are taken to be draws from some distribution, G_0, often referred to as the centering distribution. Furthermore, since the true class memberships are unknown, we denote the probability that the jth cluster belongs to any given class by the vector $p = (p_1, ..., p_M)$ whose components add up to 1.0. In the absence of prior knowledge regarding the distribution of class memberships for the J clusters across the M classes, a natural prior for p is the conjugate symmetric $Dirichlet(\tau/M, ..., \tau/M)$ distribution; the hyperparameter, τ, is often referred to as a the precision parameter. The prior can be represented as follows (M goes to infinity):

$$V_j | m_j \sim Normal(\mu_{m_j}, \sigma^2_{m_j}),$$

$$(\mu_m, \sigma_m^2) \sim G_0, \;\; for\; m = 1, ..., M,$$

$$m_j | p \sim Discrete(m_j | p_1, ..., p_M),$$

$$p \sim Dirichlet(\tau/M, ..., \tau/M),$$

where G_0 is taken to be a multivariate Normal/inverse-Gamma (NIG) distribution for which the probability density function is the following product:

$$f_{NIG}(\mu, \sigma^2 | \mu_0, \zeta_0, a_0, b_0) = f_{Normal}(\mu | 0, 1/\zeta_0^2) \times f_{Gamma}(\zeta = 1/\sigma^2 | a_0, b_0).$$

In addition, we use $Gamma(a_\tau, b_\tau)$ as the hyperprior for τ.

For non-parametric prior specification (PEM) for baseline hazard function, let $s_{\max}$ denote the largest observed event time. Then, consider the finite partition of the relevant time axis into $K + 1$ disjoint intervals: $0 < s_1 < s_2 < ... < s_{K+1} = s_{\max}$. For notational convenience, let $I_k = (s_{k-1}, s_k]$ denote the k^{th} partition. For given a partition, $s = (s_1, \ldots, s_{K+1})$, we assume the log-baseline hazard functions is piecewise constant:

$$\lambda_0(t) = \log h_0(t) = \sum_{k=1}^{K+1} \lambda_k I(t \in I_k),$$

where $I(\cdot)$ is the indicator function and $s_0 \equiv 0$. In our proposed Bayesian framework, our prior choices are:

$$\pi(\beta) \propto 1,$$

$$\lambda | K, \mu_\lambda, \sigma_\lambda^2 \sim MVN_{K+1}(\mu_\lambda 1, \sigma_\lambda^2 \Sigma_\lambda),$$

$$K \sim Poisson(\alpha),$$

$$\pi(s|K) \propto \frac{(2K+1)! \prod_{k=1}^{K+1} (s_k - s_{k-1})}{(s_{K+1})^{(2K+1)}},$$

$$\pi(\mu_\lambda) \propto 1,$$

$$\sigma_\lambda^{-2} \sim Gamma(a, b).$$

Note that K and s are treated as random and the priors for K and s jointly form a time-homogeneous Poisson process prior for the partition. The number of time splits and their positions are therefore updated within our computational scheme using reversible jump MCMC.

For parametric Weibull prior specification for baseline hazard function, $h_0(t) = \alpha\kappa t^{\alpha-1}$. In our Bayesian framework, our prior choices are:

$$\pi(\beta) \propto 1,$$

$$\pi(\alpha) \sim Gamma(a, b),$$

$$\pi(\kappa) \sim Gamma(c, d).$$

We provide a detailed description of the hierarchical models for cluster-correlated univariate survival data. The models for independent data can be obtained by removing cluster-specific random effects, V_j, and its corresponding prior specification from the description given above.

Value

`BayesSurv_HReg` returns an object of class `Bayes_HReg`.

Note

The posterior samples of V_g are saved separately in `working directory/path`.

Author(s)

Kyu Ha Lee and Sebastien Haneuse
Maintainer: Kyu Ha Lee <klee@hsph.harvard.edu>

References

Lee, K. H., Haneuse, S., Schrag, D., and Dominici, F. (2015), Bayesian semiparametric analysis of semicompeting risks data: investigating hospital readmission after a pancreatic cancer diagnosis, *Journal of the Royal Statistical Society: Series C*, 64, 2, 253-273.

Lee, K. H., Dominici, F., Schrag, D., and Haneuse, S. (2016), Hierarchical models for semicompeting risks data with application to quality of end-of-life care for pancreatic cancer, *Journal of the American Statistical Association*, 111, 515, 1075-1095.

See Also

`initiate.startValues_HReg`, `print.Bayes_HReg`, `summary.Bayes_HReg`, `plot.Bayes_HReg`

Examples

```
## Not run:
# loading a data set
data(survData)
Y <- survData[,c(1,2)]
cluster <- survData[,3]
lin.pred <- as.formula( ~ cov1 + cov2)

#####################
## Hyperparameters ##
#####################

## Weibull baseline hazard function: alpha1, kappa1
##
WB.ab <- c(0.5, 0.01) # prior parameters for alpha
##
WB.cd <- c(0.5, 0.05) # prior parameters for kappa

## PEM baseline hazard function:
##
PEM.ab <- c(0.7, 0.7) # prior parameters for 1/sigma^2
##
PEM.alpha <- 10 # prior parameters for K

## Normal cluster-specific random effects
##
Normal.ab  <- c(0.5, 0.01)  # prior for zeta

## DPM cluster-specific random effects
##
DPM.ab <- c(0.5, 0.01)
aTau  <- 1.5
bTau  <- 0.0125

##
hyperParams <- list(WB=list(WB.ab=WB.ab, WB.cd=WB.cd),
                    PEM=list(PEM.ab=PEM.ab, PEM.alpha=PEM.alpha),
                    Normal=list(Normal.ab=Normal.ab),
                    DPM=list(DPM.ab=DPM.ab, aTau=aTau, bTau=bTau))

###################
## MCMC SETTINGS ##
###################

## Setting for the overall run
##
numReps    <- 2000
thin       <- 10
burninPerc <- 0.5

## Settings for storage
```

```
##
storeV    <- TRUE

## Tuning parameters for specific updates
##
##  - those common to all models
mhProp_V_var     <- 0.05
##
## - those specific to the Weibull specification of the baseline hazard functions
mhProp_alpha_var <- 0.01
##
## - those specific to the PEM specification of the baseline hazard functions
C        <- 0.2
delPert  <- 0.5
rj.scheme <- 1
K_max    <- 50
s_max    <- max(Y$time[Y$event == 1])
time_lambda <- seq(1, s_max, 1)

##
mcmc.WB  <- list(run=list(numReps=numReps, thin=thin, burninPerc=burninPerc),
                    storage=list(storeV=storeV),
                 tuning=list(mhProp_alpha_var=mhProp_alpha_var, mhProp_V_var=mhProp_V_var))
##
mcmc.PEM <- list(run=list(numReps=numReps, thin=thin, burninPerc=burninPerc),
                    storage=list(storeV=storeV),
                    tuning=list(mhProp_V_var=mhProp_V_var, C=C, delPert=delPert,
                    rj.scheme=rj.scheme, K_max=K_max, time_lambda=time_lambda))

#################################################################
## Analysis of Independent Univariate Survival Data ############
#################################################################

####################
## WEIBULL ##
####################

##
myModel <- "Weibull"
myPath  <- "Output/01-Results-WB/"

startValues      <- vector("list", 2)
startValues[[1]] <- initiate.startValues_HReg(Y, lin.pred, survData, model=myModel)
startValues[[2]] <- initiate.startValues_HReg(Y, lin.pred, survData, model=myModel,
WB.alpha=1.12)

##
fit_WB <- BayesSurv_HReg(Y, lin.pred, survData, cluster=NULL, model=myModel,
                  hyperParams, startValues, mcmc.WB, path=myPath)

fit_WB
summ.fit_WB <- summary(fit_WB); names(summ.fit_WB)
summ.fit_WB
```

```
plot(fit_WB, tseq=seq(from=0, to=30, by=5))
plot(fit_WB, tseq=seq(from=0, to=30, by=5), plot.est = "BH")
names(fit_WB.plot <- plot(fit_WB, tseq=seq(0, 30, 5), plot=FALSE))

#########
## PEM ##
#########

##
myModel <- "PEM"
myPath  <- "Output/02-Results-PEM/"

startValues      <- vector("list", 2)
startValues[[1]] <- initiate.startValues_HReg(Y, lin.pred, survData, model=myModel)
startValues[[2]] <- initiate.startValues_HReg(Y, lin.pred, survData, model=myModel,
beta=rep(0.1,2))

##
fit_PEM <- BayesSurv_HReg(Y, lin.pred, survData, cluster=NULL, model=myModel,
                  hyperParams, startValues, mcmc.PEM, path=myPath)

fit_PEM
summ.fit_PEM <- summary(fit_PEM); names(summ.fit_PEM)
summ.fit_PEM
plot(fit_PEM)
plot(fit_PEM, plot.est = "BH")
names(fit_PEM.plot <- plot(fit_PEM, plot=FALSE))

#################################################################
## Analysis of Correlated Univariate Survival Data ############
#################################################################

####################
## WEIBULL-NORMAL ##
####################

##
myModel <- c("Weibull", "Normal")
myPath  <- "Output/03-Results-WB_Normal/"

startValues      <- vector("list", 2)

startValues[[1]] <- initiate.startValues_HReg(Y, lin.pred, survData, model=myModel, cluster)
startValues[[2]] <- initiate.startValues_HReg(Y, lin.pred, survData, model=myModel, cluster,
Normal.zeta=0.95)

##
fit_WB_N <- BayesSurv_HReg(Y, lin.pred, survData, cluster, model=myModel,
                        hyperParams, startValues, mcmc.WB, path=myPath)

fit_WB_N
summ.fit_WB_N <- summary(fit_WB_N); names(summ.fit_WB_N)
summ.fit_WB_N
```

```
plot(fit_WB_N, tseq=seq(from=0, to=30, by=5))
plot(fit_WB_N, tseq=seq(from=0, to=30, by=5), plot.est = "BH")
names(fit_WB_N.plot <- plot(fit_WB_N, tseq=seq(0, 30, 5), plot=FALSE))

#################
## WEIBULL-DPM ##
#################

##
myModel <- c("Weibull", "DPM")
myPath  <- "Output/04-Results-WB_DPM/"

startValues      <- vector("list", 2)
startValues[[1]] <- initiate.startValues_HReg(Y, lin.pred, survData, model=myModel, cluster)
startValues[[2]] <- initiate.startValues_HReg(Y, lin.pred, survData, model=myModel, cluster,
Normal.zeta=0.95)

##
fit_WB_DPM <- BayesSurv_HReg(Y, lin.pred, survData, cluster, model=myModel,
                        hyperParams, startValues, mcmc.WB, path=myPath)

fit_WB_DPM
summ.fit_WB_DPM <- summary(fit_WB_DPM); names(summ.fit_WB_DPM)
summ.fit_WB_DPM
plot(fit_WB_DPM, tseq=seq(from=0, to=30, by=5))
plot(fit_WB_DPM, tseq=seq(from=0, to=30, by=5), plot.est = "BH")
names(fit_WB_DPM.plot <- plot(fit_WB_DPM, tseq=seq(0, 30, 5), plot=FALSE))

################
## PEM-NORMAL ##
################

##
myModel <- c("PEM", "Normal")
myPath  <- "Output/05-Results-PEM_Normal/"

startValues      <- vector("list", 2)
startValues[[1]] <- initiate.startValues_HReg(Y, lin.pred, survData, model=myModel, cluster)
startValues[[2]] <- initiate.startValues_HReg(Y, lin.pred, survData, model=myModel, cluster,
Normal.zeta=0.95)

##
fit_PEM_N <- BayesSurv_HReg(Y, lin.pred, survData, cluster, model=myModel,
                            hyperParams, startValues, mcmc.PEM, path=myPath)

fit_PEM_N
summ.fit_PEM_N <- summary(fit_PEM_N); names(summ.fit_PEM_N)
summ.fit_PEM_N
plot(fit_PEM_N)
plot(fit_PEM_N, plot.est = "BH")
names(fit_PEM_N.plot <- plot(fit_PEM_N, plot=FALSE))

#############
```

```
## PEM-DPM ##
#############

##
myModel <- c("PEM", "DPM")
myPath  <- "Output/06-Results-PEM_DPM/"

startValues      <- vector("list", 2)
startValues[[1]] <- initiate.startValues_HReg(Y, lin.pred, survData, model=myModel, cluster)
startValues[[2]] <- initiate.startValues_HReg(Y, lin.pred, survData, model=myModel, cluster,
Normal.zeta=0.95)

##
fit_PEM_DPM <- BayesSurv_HReg(Y, lin.pred, survData, cluster, model=myModel,
                        hyperParams, startValues, mcmc.PEM, path=myPath)

fit_PEM_DPM
summ.fit_PEM_DPM <- summary(fit_PEM_DPM); names(summ.fit_PEM_DPM)
summ.fit_PEM_DPM
plot(fit_PEM_DPM)
plot(fit_PEM_DPM, plot.est = "BH")
names(fit_PEM_DPM.plot <- plot(fit_PEM_DPM, plot=FALSE))

## End(Not run)
```

BMT *Data on 137 Bone Marrow Transplant Patients*

Description

Data on 137 Bone Marrow Transplant Patients

Usage

```
data(BMT)
```

Format

a data frame with 137 observations on the following 22 variables.

`g` disease group; 1-ALL, 2-AML low-risk, 3-high-risk

`T1` time (in days) to death or on study time

`T2` disease-Free survival time (time to relapse, death or end of study)

`delta1` death indicator; 1-Dead, 0-Alive

`delta2` relapse indicator; 1-Relapsed, 0-Disease-Free

`delta3` disease-Free survival indicator; 1-Dead or relapsed, 0-Alive disease-free

`TA` time (in days) to acute graft-versus-host disease

deltaA acute graft-versus-host disease indicator; 1-Developed acute graft-versus-host disease, 0-Never developed acute graft-versus-host disease

TC time (in days) to chronic graft-versus-host disease

deltaC chronic graft-versus-host disease indicator; 1-Developed chronic graft-versus-host disease, 0-Never developed chronic graft-versus-host disease

TP time (in days) to return of platelets to normal levels

deltaP platelet recovery indicator; 1-Platelets returned to normal levels, 0-Platelets never returned to normal levels

Z1 patient age in years

Z2 donor age in years

Z3 patient sex; 1-Male, 2-Female

Z4 donor sex; 1-Male, 2-Female

Z5 patient CMV status; 1-CMV positive, 0-CMV negative

Z6 donor CMV status; 1-CMV positive, 0-CMV negative

Z7 waiting time to transplant in days

Z8 FAB; 1-FAB Grade 4 or 5 and AML, 0-Otherwise

Z9 hospital; 1-The Ohio State University, 2-Alfred, 3-St. Vincent, 4-Hahnemann

Z10 MTX used as a graft-versus-host-prophylactic; 1-Yes, 0-No

Source

1. R package "KMsurv".
2. Klein, J. P. and Moeschberger M. L. (2005). Survival Analysis: Techniques for Censored and Truncated Data.

References

Klein, J. P. and Moeschberger M. L. (2005). Survival Analysis: Techniques for Censored and Truncated Data.

Examples

```
data(BMT)
```

FreqID_HReg *The function to fit parametric Weibull models for the frequentist anlaysis of semi-competing risks data.*

Description

Independent semi-competing risks data can be analyzed using hierarchical models. Markov or semi-Markov assumption can be adopted for the conditional hazard function for time to the terminal event given time to non-terminal event.

Usage

```
FreqID_HReg(Y, lin.pred, data, model="semi-Markov", frailty = TRUE)
```

Arguments

Y	a data.frame containing semi-competing risks outcomes from n subjects. It is of dimension $n \times 4$: the columns correspond to y_1, δ_1, y_2, δ_2.
lin.pred	a list containing three formula objects that correspond to h_g, g=1,2,3.
data	a data.frame in which to interpret the variables named in the formulas in lin.pred.
model	a character value that specifies the type of a model based on the assumption on h_3: "semi-Markov" or "Markov".
frailty	a logical value to determine whether to include the subject-specific shared frailty term, γ, into the model.

Details

See BayesID_HReg for a detailed description of the models.

Value

FreqID_HReg returns an object of class Freq_HReg.

Author(s)

Sebastien Haneuse and Kyu Ha Lee
Maintainer: Kyu Ha Lee <klee@hsph.harvard.edu>

References

Lee, K. H., Haneuse, S., Schrag, D., and Dominici, F. (2015), Bayesian semiparametric analysis of semicompeting risks data: investigating hospital readmission after a pancreatic cancer diagnosis, *Journal of the Royal Statistical Society: Series C*, 64, 2, 253-273.

See Also

print.Freq_HReg, summary.Freq_HReg, plot.Freq_HReg, BayesID_HReg.

Examples

```
## Not run:
# loading a data set
data(scrData)
Y <- scrData[,c(1,2,3,4)]
form1 <- as.formula( ~ x1 + x2 + x3)
form2 <- as.formula( ~ x1 + x2)
form3 <- as.formula( ~ x1 + x2)
```

```
lin.pred <- list(form1, form2, form3)

fit_WB <- FreqID_HReg(Y, lin.pred, data=scrData, model="semi-Markov")

fit_WB
summ.fit_WB <- summary(fit_WB); names(summ.fit_WB)
summ.fit_WB
plot(fit_WB, tseq=seq(from=0, to=30, by=5))
plot(fit_WB, tseq=seq(from=0, to=30, by=5), plot.est = "BH")
names(fit_WB.plot <- plot(fit_WB, tseq=seq(0, 30, 5), plot=FALSE))

## End(Not run)
```

`FreqSurv_HReg` *The function to fit parametric Weibull models for the frequentist analysis of univariate survival data.*

Description

Independent univariate right-censored survival data can be analyzed using hierarchical models.

Usage

```
FreqSurv_HReg(Y, lin.pred, data)
```

Arguments

`Y`	a data.frame containing univariate time-to-event outcomes from `n` subjects. It is of dimension $n \times 2$: the columns correspond to y, δ.
`lin.pred`	a formula object that corresponds to h.
`data`	a data.frame in which to interpret the variables named in the formula in `lin.pred`.

Details

See `BayesSurv_HReg` for a detailed description of the models.

Value

`FreqSurv_HReg` returns an object of class `Freq_HReg`.

Author(s)

Sebastien Haneuse and Kyu Ha Lee
Maintainer: Kyu Ha Lee <klee@hsph.harvard.edu>

References

Lee, K. H., Haneuse, S., Schrag, D., and Dominici, F. (2015), Bayesian semiparametric analysis of semicompeting risks data: investigating hospital readmission after a pancreatic cancer diagnosis, *Journal of the Royal Statistical Society: Series C*, 64, 2, 253-273.

See Also

print.Freq_HReg, summary.Freq_HReg, plot.Freq_HReg, BayesSurv_HReg.

Examples

```
## Not run:
# loading a data set
data(survData)
Y <- survData[,c(1,2)]
lin.pred <- as.formula( ~ cov1 + cov2)

fit_WB <- FreqSurv_HReg(Y, lin.pred, data=survData)
fit_WB
summ.fit_WB <- summary(fit_WB); names(summ.fit_WB)
summ.fit_WB
plot(fit_WB, tseq=seq(from=0, to=30, by=5))
plot(fit_WB, tseq=seq(from=0, to=30, by=5), plot.est = "BH")
names(fit_WB.plot <- plot(fit_WB, tseq=seq(0, 30, 5), plot=FALSE))

## End(Not run)
```

---

initiate.startValues_AFT

The function that initiates starting values for a single chain.

Description

The function initiates starting values for a single chain for accelrated failture time (AFT) models. Users are allowed to set some non-null values to starting values for a set of parameters. The function will automatically generate starting values for any parameters whose values are not specified.

Usage

```
initiate.startValues_AFT(Y, lin.pred, data, model,
                         beta1=NULL, beta2=NULL, beta3=NULL, beta=NULL,
                         gamma=NULL, theta=NULL,
                         y1=NULL, y2=NULL, y=NULL,
                         LN.mu=NULL, LN.sigSq=NULL,
                         DPM.class1=NULL, DPM.class2=NULL, DPM.class3=NULL,
                         DPM.class=NULL, DPM.mu1=NULL, DPM.mu2=NULL,
```

```
DPM.mu3=NULL, DPM.mu=NULL, DPM.zeta1=NULL,
DPM.zeta2=NULL, DPM.zeta3=NULL, DPM.zeta=NULL,
DPM.tau=NULL)
```

Arguments

Y	For BayesID_AFT, it is a data.frame containing semi-competing risks outcomes from n subjects. See BayesID_AFT. For BayesSurv_AFT, it is a data.frame containing univariate time-to-event outcomes from n subjects. See BayesSurv_AFT.
lin.pred	For BayesID_AFT, it is a list containing three formula objects that correspond to the transition g=1,2,3. For BayesSurv_AFT, it is a formula object that corresponds to $log(t)$.
data	a data.frame in which to interpret the variables named in the formula(s) in lin.pred.
model	a character vector that specifies the type of components in a model. Check BayesID_AFT and BayesSurv_AFT.
beta1	starting values of β_1 for BayesID_AFT.
beta2	starting values of β_2 for BayesID_AFT.
beta3	starting values of β_3 for BayesID_AFT.
beta	starting values of β for BayesSurv_AFT.
gamma	starting values of γ for BayesID_AFT.
theta	starting values of θ for BayesID_AFT.
y1	starting values of $log(t_1)$ for BayesID_AFT.
y2	starting values of $log(t_2)$ for BayesID_AFT.
y	starting values of $log(t)$ for BayesSurv_AFT.
LN.mu	starting values of β_0 in logNormal models for BayesID_AFT and BayesSurv_AFT.
LN.sigSq	starting values of σ^2 in logNormal models for BayesID_AFT and BayesSurv_AFT.
DPM.class1	starting values of the class membership for transition 1 in DPM models for BayesID_AFT.
DPM.class2	starting values of the class membership for transition 2 in DPM models for BayesID_AFT.
DPM.class3	starting values of the class membership for transition 3 in DPM models for BayesID_AFT.
DPM.class	starting values of the class membership in DPM models for BayesSurv_AFT.
DPM.mu1	starting values of μ_1 in DPM models for BayesID_AFT.
DPM.mu2	starting values of μ_2 in DPM models for BayesID_AFT.
DPM.mu3	starting values of μ_3 in DPM models for BayesID_AFT.
DPM.mu	starting values of μ in DPM models for BayesSurv_AFT.
DPM.zeta1	starting values of ζ_1 in DPM models for BayesID_AFT.
DPM.zeta2	starting values of ζ_2 in DPM models for BayesID_AFT.
DPM.zeta3	starting values of ζ_3 in DPM models for BayesID_AFT.
DPM.zeta	starting values of ζ in DPM models for BayesSurv_AFT.
DPM.tau	starting values of τ in DPM models for BayesID_AFT and BayesSurv_AFT.

Value

`initiate.startValues_AFT` returns a list containing starting values for a sigle chain that can be used for `BayesID_AFT` and `BayesSurv_AFT`.

Author(s)

Sebastien Haneuse and Kyu Ha Lee
Maintainer: Kyu Ha Lee <klee@hsph.harvard.edu>

References

Lee, K. H., Rondeau, V., and Haneuse, S. (2017), Accelerated failure time models for semicompeting risks data in the presence of complex censoring, *Biometrics*, in press.

See Also

`BayesID_AFT`, `BayesSurv_AFT`

Examples

```
## See Examples in \code{\link{BayesID_AFT}} and \code{\link{BayesSurv_AFT}}.
```

---

initiate.startValues_HReg

The function that initiates starting values for a single chain.

Description

The function initiates starting values for a single chain for hazard regression (HReg) models. Users are allowed to set some non-null values to starting values for a set of parameters. The function will automatically generate starting values for any parameters whose values are not specified.

Usage

```
initiate.startValues_HReg(Y, lin.pred, data, model, cluster = NULL,
                    beta1 = NULL, beta2 = NULL, beta3 = NULL, beta = NULL,
                    gamma.ji = NULL, theta = NULL,
                    V.j1 = NULL, V.j2 = NULL, V.j3 = NULL, V.j = NULL,
                    WB.alpha = NULL, WB.kappa = NULL,
                PEM.lambda1=NULL, PEM.lambda2=NULL, PEM.lambda3=NULL, PEM.lambda=NULL,
                    PEM.s1=NULL, PEM.s2=NULL, PEM.s3=NULL, PEM.s=NULL,
                    PEM.mu_lam=NULL, PEM.sigSq_lam=NULL,
                    MVN.SigmaV = NULL, Normal.zeta = NULL,
                    DPM.class = NULL, DPM.tau = NULL)
```

Arguments

Y	For BayesID_HReg, it is a data.frame containing semi-competing risks outcomes from n subjects. For BayesSurv_HReg, it is a data.frame containing univariate time-to-event outcomes from n subjects.
lin.pred	For BayesID_HReg, it is a list containing three formula objects that correspond to $h_g()$, g=1,2,3. For BayesSurv_HReg, it is a formula object that corresponds to $h()$.
data	a data.frame in which to interpret the variables named in the formula(s) in lin.pred.
model	a character vector that specifies the type of components in a model. Check BayesID_HReg and BayesSurv_HReg.
cluster	a vector of cluster information for n subjects. The cluster membership must be set to consecutive positive integers, $1:J$.
beta1	starting values of β_1 for BayesID_HReg.
beta2	starting values of β_2 for BayesID_HReg.
beta3	starting values of β_3 for BayesID_HReg.
beta	starting values of β for BayesSurv_HReg.
gamma.ji	starting values of γ for BayesID_HReg.
theta	starting values of θ for BayesID_HReg.
V.j1	starting values of V_{j1} for BayesID_HReg.
V.j2	starting values of V_{j2} for BayesID_HReg.
V.j3	starting values of V_{j3} for BayesID_HReg.
V.j	starting values of V_j for BayesSurv_HReg.
WB.alpha	starting values of the Weibull parameters, α_g for BayesID_HReg. starting values of the Weibull parameter, α for BayesSurv_HReg.
WB.kappa	starting values of the Weibull parameters, κ_g for BayesID_HReg. starting values of the Weibull parameter, κ for BayesSurv_HReg.
PEM.lambda1	starting values of the PEM parameters, λ_1 for BayesID_HReg.
PEM.lambda2	starting values of the PEM parameters, λ_2 for BayesID_HReg.
PEM.lambda3	starting values of the PEM parameters, λ_3 for BayesID_HReg.
PEM.lambda	starting values of λ for BayesSurv_HReg.
PEM.s1	starting values of the PEM parameters, s_1 for BayesID_HReg.
PEM.s2	starting values of the PEM parameters, s_2 for BayesID_HReg.
PEM.s3	starting values of the PEM parameters, s_3 for BayesID_HReg.
PEM.s	starting values of s for BayesSurv_HReg.
PEM.mu_lam	starting values of the PEM parameters, $\mu_{\lambda,g}$ for BayesID_HReg. starting values of the PEM parameter, μ_λ for BayesSurv_HReg.
PEM.sigSq_lam	starting values of the PEM parameters, $\sigma^2_{\lambda,g}$ for BayesID_HReg. starting values of the PEM parameter, σ^2_λ for BayesSurv_HReg.

MVN.SigmaV	starting values of Σ_V in DPM models for BayesID_HReg.
Normal.zeta	starting values of ζ in DPM models for BayesSurv_HReg.
DPM.class	starting values of the class membership in DPM models for BayesID_HReg and BayesSurv_HReg.
DPM.tau	starting values of τ in DPM models for BayesID_HReg and BayesSurv_HReg.

Value

initiate.startValues_HReg returns a list containing starting values for a sigle chain that can be used for BayesID_HReg and BayesSurv_HReg.

Author(s)

Sebastien Haneuse and Kyu Ha Lee
Maintainer: Kyu Ha Lee <klee@hsph.harvard.edu>

References

Lee, K. H., Haneuse, S., Schrag, D., and Dominici, F. (2015), Bayesian semiparametric analysis of semicompeting risks data: investigating hospital readmission after a pancreatic cancer diagnosis, *Journal of the Royal Statistical Society: Series C*, 64, 2, 253-273.

Lee, K. H., Dominici, F., Schrag, D., and Haneuse, S. (2016), Hierarchical models for semicompeting risks data with application to quality of end-of-life care for pancreatic cancer, *Journal of the American Statistical Association*, 111, 515, 1075-1095.

See Also

BayesID_HReg, BayesSurv_HReg

Examples

```
## See Examples in \code{\link{BayesID_HReg}} and \code{\link{BayesSurv_HReg}}.
```

---

methods *Methods for objects of classes,* Bayes_HReg/Bayes_AFT/Freq_HReg.

Description

The Bayes_HReg class represents results from Bayesian analysis of semi-competing risks or univariate time-to-event data in the context of hazard regression models.
The Bayes_AFT class represents results from Bayesian analysis of semi-competing risks or univariate time-to-event data in the context of AFT models.
The Freq_HReg class represents results from Frequentist analysis of semi-competing risks or univariate time-to-event data in the context of hazard regression models.

Usage

```
## S3 method for class 'Bayes_HReg'
print(x, digits=3, ...)
## S3 method for class 'Bayes_AFT'
print(x, digits=3, ...)
## S3 method for class 'summ.Bayes_HReg'
print(x, digits=3, ...)
## S3 method for class 'summ.Bayes_AFT'
print(x, digits=3, ...)
## S3 method for class 'Freq_HReg'
print(x, digits=3, ...)
## S3 method for class 'summ.Freq_HReg'
print(x, digits=3, ...)
## S3 method for class 'Bayes_HReg'
summary(object, digits=3, ...)
## S3 method for class 'Bayes_AFT'
summary(object, digits=3, ...)
## S3 method for class 'Freq_HReg'
summary(object, digits=3, ...)
## S3 method for class 'Freq_HReg'
plot(x, tseq=c(0, 5, 10), plot=TRUE, plot.est="BS", xlab=NULL, ylab=NULL, ...)
## S3 method for class 'Bayes_HReg'
plot(x, tseq=c(0, 5, 10), plot=TRUE, plot.est="BS", xlab=NULL, ylab=NULL, ...)
## S3 method for class 'Bayes_AFT'
plot(x, time, tseq=c(0, 5, 10), plot=TRUE, plot.est="BS", xlab=NULL, ylab=NULL, ...)
```

Arguments

x	an object of class Bayes_HReg or Bayes_AFT or Freq_HReg.
digits	a numeric value indicating the number of digits to display.
object	an object of class Bayes_HReg or Bayes_AFT orFreq_HReg.
time	the points at which the baseline survival/hazard functions are evaluated.
tseq	the points at which tick-marks are to be drawn. Required only if the object x is returned by parametric Weibull-HReg/log-Normal-AFT/DPM-AFT models.
plot	If TRUE (the default) then either estimated baseline hazard functions or estimated baseline survival functions are produced depending on the value of plot.est. If FALSE, the summaries which the plots are based on are returned.
plot.est	used only if plot is TRUE. If BS (the default) then estimated baseline survival functions are produced. If BH then estimated baseline hazard functions are produced.
xlab	a title for the x axis.
ylab	a title for the y axis.
...	additional arguments.

See Also

BayesID_HReg, BayesID_AFT, BayesSurv_HReg, BayesSurv_AFT, FreqID_HReg, FreqSurv_HReg.

`old.functions` *Old functions*

Description

Since version 2.5, the functions BayesID(), BayesSurv(), FreqID(), FreqSurv(), initiate.startValues() have been renamed as BayesID_HReg(), BayesSurv_HReg(), FreqID_HReg(), FreqSurv_HReg(), initiate.startValues_HReg(), respectively. If one of the old functions is called, a warning message will be displayed with the corresponding new function name.

Usage

```
BayesID(...)
BayesSurv(...)
FreqID(...)
FreqSurv(...)
initiate.startValues(...)
```

Arguments

`...`	arguments used for the old functions.

`scrData` *A simulated clustered semi-competing risks data set*

Description

Simulated semi-competing risks data

Usage

```
data(scrData)
```

Format

a data frame with 2000 observations on the following 14 variables.

`time1` the time to non-terminal event

`event1` the censoring indicators for the non-terminal event time; 1=event observed, 0=censored/truncated

`time2` the time to terminal event

`event2` the censoring indicators for the terminal event time; 1=event observed, 0=censored

`cluster` cluster numbers

`x1` a vector of continuous covarate

`x2` a vector of continuous covarate

`x3` a vector of continuous covarate

Examples

```
data(scrData)
```

`simID` *The function that simulates independent/cluster-correlated semi-competing risks data under semi-Markov Weibull/Weibull-MVN models.*

Description

The function to simulate independent/cluster-correlated semi-competing risks data under semi-Markov Weibull/Weibull-MVN models.

Usage

```
simID(cluster=NULL, x1, x2, x3, beta1.true, beta2.true, beta3.true,
alpha1.true, alpha2.true, alpha3.true,
kappa1.true, kappa2.true, kappa3.true,
theta.true, SigmaV.true=NULL, cens)
```

Arguments

`cluster`	a vector of cluster information for `n` subjects. The cluster membership must be set to consecutive positive integers, $1 : J$. Required only when generating clustered data.
`x1`	covariate matrix, `n` observations by `p1` variables.
`x2`	covariate matrix, `n` observations by `p2` variables.
`x3`	covariate matrix, `n` observations by `p3` variables.
`beta1.true`	true value for β_1.
`beta2.true`	true value for β_2.
`beta3.true`	true value for β_3.
`alpha1.true`	true value for α_1.
`alpha2.true`	true value for α_2.
`alpha3.true`	true value for α_3.
`kappa1.true`	true value for κ_1.
`kappa2.true`	true value for κ_2.
`kappa3.true`	true value for κ_3.

theta.true	true value for θ.
SigmaV.true	true value for Σ_V. Required only when generating clustered data.
cens	a vector with two numeric elements. The right censoring times are generated from Uniform($cens[1]$, $cens[2]$).

Value

simIDcor returns a data.frame containing semi-competing risks outcomes from n subjects. It is of dimension $n \times 4$: the columns correspond to y_1, δ_1, y_2, δ_2.

y1	a vector of n times to the non-terminal event
y2	a vector of n times to the terminal event
delta1	a vector of n censoring indicators for the non-terminal event time (1=event occurred, 0=censored)
delta2	a vector of n censoring indicators for the terminal event time (1=event occurred, 0=censored)

Author(s)

Kyu Ha Lee and Sebastien Haneuse
Maintainer: Kyu Ha Lee <klee@hsph.harvard.edu>

Examples

```
library(MASS)
set.seed(123456)

J = 110
nj = 50
n = J * nj

cluster <- rep(1:J, each = nj)

kappa1.true <- 0.05
kappa2.true <- 0.01
kappa3.true <- 0.01
alpha1.true <- 0.8
alpha2.true <- 1.1
alpha3.true <- 0.9
beta1.true <- c(0.5, 0.8, -0.5)
beta2.true <- c(0.5, 0.8, -0.5)
beta3.true <- c(1, 1, -1)
SigmaV.true <- matrix(0.25,3,3)
theta.true <- 0.5
cens <- c(90, 90)

cov1 <- matrix(rnorm((length(beta1.true)-1)*n, 0, 1), n, length(beta1.true)-1)
cov2 <- sample(c(0, 1), n, replace = TRUE)
x1 <- as.data.frame(cbind(cov1, cov2))
```

```
x2 <- as.data.frame(cbind(cov1, cov2))
x3 <- as.data.frame(cbind(cov1, cov2))

simData <- simID(cluster, x1, x2, x3, beta1.true, beta2.true, beta3.true,
alpha1.true, alpha2.true, alpha3.true,
kappa1.true, kappa2.true, kappa3.true,
theta.true, SigmaV.true, cens)
```

`simSurv` *The function that simulates independent/cluster-correlated right-censored survival data under Weibull/Weibull-Normal model.*

Description

The function to simulate independent/cluster-correlated right-censored survival data under Weibull/Weibull-Normal model.

Usage

```
simSurv(cluster=NULL, x, beta.true, alpha.true, kappa.true, sigmaV.true=NULL, cens)
```

Arguments

`cluster`	a vector of cluster information for `n` subjects. The cluster membership must be set to consecutive positive integers, $1 : J$. Required only when generating clustered data.
`x`	covariate matrix, `n` observations by `p` variables.
`beta.true`	true value for β.
`alpha.true`	true value for α.
`kappa.true`	true value for κ.
`sigmaV.true`	true value for σ_V. Required only when generating clustered data.
`cens`	a vector with two numeric elements. The right censoring times are generated from Uniform($cens[1]$, $cens[2]$).

Value

`simSurv` returns a data.frame containing univariate time-to-event outcomes from `n` subjects. It is of dimension $n \times 2$: the columns correspond to y, δ.

`y`	a vector of `n` times to the event
`delta`	a vector of `n` censoring indicators for the event time (1=event occurred, 0=censored)

Author(s)

Kyu Ha Lee and Sebastien Haneuse
Maintainer: Kyu Ha Lee <klee@hsph.harvard.edu>

Examples

```
set.seed(123456)

J = 110
nj = 50
n = J * nj

cluster <- rep(1:J, each = nj)

x = matrix(0, n, 2)
x[,1] = rnorm(n, 0, 2)
x[,2] = sample(c(0, 1), n, replace = TRUE)

beta.true = c(0.5, 0.5)

alpha.true = 1.5
kappa.true = 0.02
sigmaV.true = 0.1

cens <- c(30, 40)

simData <- simSurv(cluster, x, beta.true, alpha.true, kappa.true,
sigmaV.true, cens)
```

---

survData *A simulated clustered univariate survival data.*

Description

Simulated univariate survival data.

Usage

```
data(survData)
```

Format

a data frame with 2000 observations on the following 4 variables.

`time` the time to event

`event` the censoring indicators for the event time; 1=event observed, 0=censored

`cluster` cluster numbers

`cov1` the first column of covariate matrix x

`cov2` the second column of covariate matrix x

Examples

```
data(scrData)
```

Index