1. `BayesSurv_HReg` : independent, univariate time-to-event data fit to a Cox PH model with Weibull baseline hazard
2. `BayesSurv_HReg` : independent, univariate time-to-event data fit to a Cox PH model with PEM baseline hazard
3. `BayesSurv_AFT` : independent, univariate time-to-event data fit to an AFT model with LN baseline survival distribution
4. `BayesSurv_AFT` : independent, univariate time-to-event data fit to an AFT model with DPM baseline survival distribution
5. `BayesSurv_HReg` : cluster-correlated, univariate time-to-event data fit to a Cox PH model with Weibull baseline hazard
6. `BayesSurv_HReg` : cluster-correlated, univariate time-to-event data fit to a Cox PH model with PEM baseline hazard
7. `BayesID_HReg` : independent semi-competing risks data using an illness-death model with Weibull baseline hazards
8. `BayesID_HReg` : independent semi-competing risks data using an illness-death model with PEM baseline hazards
9. `BayesID_AFT` : independent semi-competing risks data using an AFT illness-death model with LN baseline survival distribution
10. `BayesID_AFT` : independent semi-competing risks data using an AFT illness-death model with DPM baseline survival distribution
11. `BayesID_HReg` : cluster-correlated semi-competing risks data using an illness-death model with Weibull baseline hazards
12. `BayesID_HReg` : cluster-correlated semi-competing risks data using an illness-death model with PEM baseline hazards

1. `BayesSurv_HReg` applied to independent, univariate time-to-event data fit to a Cox PH model with Weibull baseline hazard

Definition of the survival model

Let t_i denote the time-to-event of interest for individuals $i = 1, \ldots, n$, subject to right censoring at time c_i. Let (y_i, δ_i, x_i) denote independent observations, where $y_i = \min(t_i, c_i)$, $\delta_i = \mathbb{1}(t_i \leq c_i)$, and x_i is a vector of covariates for individual i. The following Cox proportional hazards model is assumed

$$h(t_i|x_i) = h_0(t_i)\exp\left(x_i^\top \beta\right), \ \ t_i > 0,$$

where the baseline hazard h_0 is defined parametrically by a Weibull hazard, $h_0(t) = \alpha\kappa t^{\alpha-1}$.

In the Bayesian framework, priors must be specified for the regression parameter, β, and the shape and scale parameters of baseline hazard function, α and κ, respectively. The following specifications are made

$$\begin{aligned}\pi(\beta) &\propto 1,\\ \pi(\alpha) &\sim Gamma(a, b),\\ \pi(\kappa) &\sim Gamma(c, d).\end{aligned}$$

Hyperparameters

The hyperparameters a and b must be specified for the prior distribution of α which is a Gamma distribution with mean ab and variance ab^2. Similarly, the hyperparameters c and d must be specified for the Gamma prior of κ.

Arguments to specify

Model-related	
`Formula`	a `Formula` object that corresponds to the hazard $h(t_i|x_i)$: $y + \delta \sim x$.
`data`	an $(n{\times}q)$-dimensional data.frame; the q-columns correspond to q covariate vectors named in the formula in `Formula`.
Hyperparameters	
`WB.ab`	a 2-vector of positive hyperparameters a and b of the prior distribution for the shape parameter α of the Weibull baseline hazard. Example: `WB.ab <- c(0.5, 0.01)`.
`WB.cd`	a 2-vector of positive hyperparameters c and d of the prior distribution for the scale parameter κ of the Weibull baseline hazard. Example: `WB.cd <- c(0.5, 0.05)`.
MCMC Settings	
`numReps`	total number of scans
`thin`	extent of thinning, e.g. if `thin=10` retain every 10^{th} sample.
`burninPerc`	the proportion of burn-in (samples to be discarded before analyzing the data).
`mhProp_alpha_var`	the shape parameter α is updated using a Metropolis-Hastings random walk step generating proposals from a Gamma distribution with variance `mhProp_alpha_var`.
Starting Values	
`startValues`	use `initiate.startValues_HReg(Formula, data, model, nChain, beta = NULL, WB.alpha = NULL, WB.kappa = NULL)` which initiates starting values for β, α and κ in the Metropolis-Hastings algorithm if left unspecified. Users may set non-null starting values for any of these parameters.
Storage	
`path`	name of the directory where results are stored. Can leave unspecified.

Implementation

```
data(survData)
form <- Formula(time + event ~ cov1 + cov2)
##
WB.ab <- c(0.5, 0.01) # prior parameters for alpha
WB.cd <- c(0.5, 0.05) # prior parameters for kappa
hyperParams <- list(WB=list(WB.ab=WB.ab, WB.cd=WB.cd))
##
numReps <- 2000
burninPerc <- 0.5
thin <- 10
mhProp_alpha_var <- 0.01
mcmc <- list(run=list(numReps=numReps, thin=thin, burninPerc=burninPerc),
             tuning=list(mhProp_alpha_var=mhProp_alpha_var))
##
myModel <- "Weibull"
myPath  <- "Output/01-Results-WB/"
startValues <- initiate.startValues_HReg(form, survData, model=myModel, nChain=2)
##
fit_WB <- BayesSurv_HReg(form, survData, id=NULL, model=myModel, hyperParams, startValues, mcmc, path=myPath)
summary(fit_WB)
pred_WB <- predict(fit_WB, tseq=seq(from=0, to=30, by=5))
plot(pred_WB, plot.est="Haz")
plot(pred_WB, plot.est="Surv")
```

2. `BayesSurv_HReg` applied to independent, univariate time-to-event data fit to a Cox PH model with PEM baseline hazard

Definition of the survival model

Let t_i denote the time-to-event of interest for individuals $i = 1, \dots, n$, subject to right censoring at time c_i. Let (y_i, δ_i, x_i) denote independent observations, where $y_i = \min(t_i, c_i)$, $\delta_i = \mathbb{1}(t_i \le c_i)$, and x_i is a vector of covariates for individual i. The following Cox proportional hazards model is assumed

$$h(t_i|x_i) = h_0(t_i)\exp\left(x_i^\top \beta\right), \ \ t_i > 0.$$

The baseline hazard h_0 is defined non-parametrically by a mixture of piecewise exponential functions as follows

$$\lambda_0(t) = \log h_0(t) = \sum_{k=1}^{K+1} \lambda_k \mathbb{1}\left\{t \in (s_{k-1}, s_k]\right\},$$

where λ_k is constant and the time interval between 0 and the largest observed failure time, denoted s_k, is partitioned into $K + 1$ disjoint intervals: $0 < s_1 < \cdots < s_{K+1}$.

In the Bayesian framework, priors must be specified for the regression parameter, β, the number of intervals, K, and the partition points $(s_1, \dots, s_{K+1})$, respectively. The following specifications are made

$$\begin{aligned}
\pi(\beta) &\propto 1, \\
\lambda|K, \mu_\lambda, \sigma_\lambda^2 &\sim MVN_{K+1}(\mu_\lambda \mathbb{1}, \sigma_\lambda^2 \Sigma_\lambda) \\
K &\sim Poisson(\alpha), \\
\pi(s|K) &\propto \frac{(2K+1)! \prod_{k=1}^{K+1}(s_k - s_{k-1})}{(s_{K+1})^{(2K+1)}}, \\
\pi(\mu_\lambda) &\propto 1, \\
\sigma_\lambda^{-2} &\sim Gamma(a, b).
\end{aligned}$$

The prior specification for λ follows a MVN-ICAR (see Supplemental Material to Lee, Haneuse, Schrag and Dominici, 2015). Note that K and s jointly form a time-homogeneous Poisson process prior for the partition.

Hyperparameters

The hyperparameter α must be specified for the prior distribution of K, as well as a and b, the rate and shape of the Gamma distributed hyperprior for σ_λ^{-2}.

Arguments to specify

Model-related	
`Formula`	a `Formula` object that corresponds to the hazard $h(t_i|x_i)$: $y + \delta \sim x$.
`data`	an $(n{\times}q)$-dimensional data.frame; the q-columns correspond to q covariate vectors named in the formula in `Formula`.
Hyperparameters	
`PEM.ab`	a 2-vector of positive hyperparameters a and b of the prior distribution for σ_λ^{-2}. Example: `PEM.ab <- c(0.7,0.7)`.
`PEM.alpha`	hyperparameter α of the prior distribution for K, which is one less than the number of partition points. Example: `PEM.alpha <- 10`.
MCMC Settings	
`numReps`	total number of scans
`thin`	extent of thinning, e.g. if `thin=10` retain every 10^{th} sample.
`burninPerc`	the proportion of burn-in (samples to be discarded before analyzing the data).
`C`	a numeric value for the proportion that determines the sum of probabilities choosing the birth and death moves.[1]
`delPert`	the perturbation parameter in the birth updates; values must be between 0 and 0.5.[1]
`rj.scheme`	`rj.scheme=1`: the birth update will draw the proposal time split from $1 : s_{max}$; `rj.scheme=2`: the birth update will draw the proposal time split from uniquely ordered failure times in the data.
`K_max`	the number of splits allowed in each iteration of the Metropolis-Hastings-Green algorithm.
`s_max`	the largest observed failure time, given by `s_max <- max(data$time[data$event==1])`
`time_lambda`	time points at which the λ is monitored for convergence. Example: `time_lambda <- seq(1, s_max, 1)`. The chains for these monitoring points can be found in `lambda.fin` in the chains of the `BayesSurv_HReg` object.
Starting Values	
`startValues`	use `initiate.startValues_HReg(Formula, data, model, nChain, beta = NULL)` which initiates all necessary starting values in the Metropolis-Hastings-Green algorithm. Users may set non-null starting values for `beta`.
Storage	
`path`	name of the directory where results are stored. Can leave unspecified.

---

[1] See Section A in Supplemental Material to Lee et al. (2015)

Implementation

```
data(survData)
form <- Formula(time + event ~ cov1 + cov2)
##
PEM.ab <- c(0.7, 0.7) # prior parameters for 1/sigma^2
PEM.alpha <- 10 # prior parameters for K
hyperParams <- list(PEM=list(PEM.ab=PEM.ab, PEM.alpha=PEM.alpha))
##
numReps <- 2000
burninPerc <- 0.5
thin <- 10
C <- 0.2
delPert  <- 0.5
rj.scheme <- 2
K_max    <- 50
s_max    <- max(survData$time[survData$event == 1])
time_lambda <- seq(1, s_max, 0.5)
mcmc <- list(run=list(numReps=numReps, thin=thin, burninPerc=burninPerc),
             tuning=list(C=C, delPert=delPert, rj.scheme=rj.scheme,
                         K_max=K_max, s_max=s_max, time_lambda=time_lambda) )
##
myModel <- "PEM"
myPath  <- "Output/02-Results-PEM/"
startValues      <- initiate.startValues_HReg(form, survData, model=myModel, nChain=2)
##
fit_PEM <- BayesSurv_HReg(form, survData, id=NULL, model=myModel,
                  hyperParams, startValues, mcmc, path=myPath)
summary(fit_PEM)
pred_PEM <- predict(fit_PEM)
plot(pred_PEM, plot.est="Haz")
plot(pred_PEM, plot.est="Surv")
```

3. `BayesSurv_AFT` applied to independent, univariate time-to-event data fit to an AFT model with log-Normal baseline survival distribution

Definition of the survival model

Let t_i denote the time-to-event of interest for individuals $i = 1, \ldots, n$. In the presence of interval censoring, the time-to-event for the i^{th} subject satisfies $c_{ij} \leq t_i < c_{ij+1}$. Let $(c_{ij}, c_{ij+1}, L_i, x_i)$ denote independent observations, where L_i is the left-truncation time and x_i is a vector of covariates for individual i. The following AFT model is assumed

$$\log(t_i) = x_i^\top \beta + \epsilon_i, \;\; t_i > 0.$$

We take ϵ_i to follow the Normal(μ, σ^2) distribution for ϵ_i for the parametric AFT model. In the Bayesian framework, priors must be specified for β, μ, and σ^2. The following specifications are made

$$\pi(\beta, \mu) \propto 1,$$
$$\sigma^2 \sim Inverse - Gamma(a_\sigma, b_\sigma).$$

Hyperparameters

The hyperparameters, a_σ and b_σ, must be specified for the prior distribution of σ^2.

Arguments to specify

Model-related	
`Formula`	a `Formula` object that corresponds to $\log(t_i)$: $L \vert y_L + y_U \sim x$.
`data`	an $(n{\times}q)$-dimensional data.frame; the q-columns correspond to q covariate vectors named in the formula in `Formula`.
Hyperparameters	
`LN.ab`	a 2-vector of positive hyperparameters a and b of the prior distribution for σ^2. Example: `LN.ab <- c(0.7,0.7)`.
MCMC Settings	
`numReps`	total number of scans
`thin`	extent of thinning, e.g. if `thin=10` retain every 10^{th} sample.
`burninPerc`	the proportion of burn-in (samples to be discarded before analyzing the data).
`beta.prop.var`	the parameter β is updated using a Metropolis-Hastings random walk step generating proposals from a Normal distribution with variance `beta.prop.var`.
`mu.prop.var`	the parameter μ is updated using a Metropolis-Hastings random walk step generating proposals from a Normal distribution with variance `mu.prop.var`.
`zeta.prop.var`	the parameter $\zeta = 1/\sigma^2$ is updated using a Metropolis-Hastings random walk step generating proposals from a log-Normal distribution with variance `zeta.prop.var`.
Starting Values	
`startValues`	use `initiate.startValues_AFT(Formula, data, model, nChain, beta = NULL, y = NULL, LN.mu = NULL, LN.sigSq = NULL)` which initiates all necessary starting values in the Metropolis-Hastings algorithm. Users may set non-null starting values for `beta`, `y`, `LN.mu`, `LN.sigSq`.
Storage	
`path`	name of the directory where results are stored. Can leave unspecified.

Implementation

```
data(survData)
survData$yL <- survData$yU <- survData[,1]
survData$yU[which(survData[,2] == 0)] <- Inf
survData$LT <- rep(0, dim(survData)[1])
form <- Formula(LT | yL + yU ~ cov1 + cov2)
##
LN.ab <- c(0.3, 0.3)
hyperParams <- list(LN=list(LN.ab=LN.ab))
##
numReps    <- 1000
thin       <- 10
burninPerc <- 0.5
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))
##
myModel <- "LN"
myPath  <- "Output/01-Results-LN/"
startValues      <- initiate.startValues_AFT(form, survData, model=myModel, nChain=2)
##
fit_LN <- BayesSurv_AFT(form, survData, model=myModel, hyperParams,
startValues, mcmcParams, path=myPath)

summary(fit_LN)
pred_LN <- predict(fit_LN, time = seq(0, 35, 1), tseq=seq(from=0, to=30, by=5))
plot(pred_LN, plot.est="Haz")
plot(pred_LN, plot.est="Surv")
```

Definition of the survival model

Let t_i denote the time-to-event of interest for individuals $i = 1, \dots, n$. Considering interval censoring, the time-to-event for the i^{th} subject satisfies $c_{ij} \le t_i < c_{ij+1}$. Let $(c_{ij}, c_{ij+1}, L_i, x_i)$ denote independent observations, where L_i is the left-truncation time and x_i is a vector of covariates for individual i. The following AFT model is assumed

$$\log(t_i) = x_i^\top \beta + \epsilon_i, \ \ t_i > 0,$$

where ϵ_i is assumed to be taken as draws from the DPM of normal distributions:

$$\begin{aligned}
\epsilon_i | r_i &\sim \text{Normal}(\mu_{r_i}, \sigma^2_{r_i}),\\
(\mu_r, \sigma^2_r) &\sim G_0, \ \text{ for } r = 1, \dots, M,\\
r_i | p &\sim Discrete(r_i | p_1, \dots, p_M),\\
p &\sim Dirichlet(\tau/M, \dots, \tau/M).
\end{aligned}$$

In the Bayesian framework, priors must be specified for the unknown parameters. We take the G_0 as a normal distribution centered at μ_0 with a variance σ_0^2 for μ_r and an inverse-Gamma(a_σ, b_σ) for σ_r^2. For β, we adopt non-informative flat priors on the real line. Finally, we specify a Gamma(a_τ, b_τ) hyperprior for the precision parameter τ.

Hyperparameters

The hyperparameter $(\mu_0, \sigma_0^2, a_\sigma, b_\sigma)$ must be specified for the centering distribution G_0, as well as a_τ and b_τ, the rate and shape of the Gamma distributed hyperprior for τ.

Arguments to specify

Model-related	
`Formula`	a `Formula` object that corresponds to $\log(t_i)$: $L \vert y_L + y_U \sim x$.
`data`	an $(n \times q)$-dimensional data.frame; the q-columns correspond to q covariate vectors named in the formula in `Formula`.
Hyperparameters	
`DPM.mu`	a hyperparameter μ_0 of the centering distribution G_0.
`DPM.sigSq`	a positive-valued hyperparameter σ_0^2 of the centering distribution G_0.
`DPM.ab`	a 2-vector of positive hyperparameters a_σ and b_σ of the centering distribution G_0.
`Tau.ab`	a 2-vector of positive hyperparameters a_τ and b_τ of the hyperprior distribution for τ. Example: `Tau.ab <- c(1.5, 0.0125)`.
MCMC Settings	
`numReps`	total number of scans
`thin`	extent of thinning, e.g. if `thin=10` retain every 10^{th} sample.
`burninPerc`	the proportion of burn-in (samples to be discarded before analyzing the data).
`beta.prop.var`	the parameter β is updated using a Metropolis-Hastings random walk step generating proposals from a Normal distribution with variance `beta.prop.var`.
`mu.prop.var`	the parameter μ_r is updated using a Metropolis-Hastings random walk step generating proposals from a Normal distribution with variance `mu.prop.var`.
`zeta.prop.var`	the parameter $\zeta_r = 1/\sigma_r^2$ is updated using a Metropolis-Hastings random walk step generating proposals from a log-Normal distribution with variance `zeta.prop.var`.
Starting Values	
`startValues`	use `initiate.startValues_AFT(Formula, data, model, nChain, beta = NULL, y = NULL, DPM.class = NULL, DPM.mu = NULL, DPM.zeta=NULL, DPM.tau=NULL)` which initiates all necessary starting values in the Metropolis-Hastings algorithm. Users may set non-null starting values for `beta`, `y`, `DPM.class`, `DPM.mu`, `DPM.zeta`, `DPM.tau`.
Storage	
`path`	name of the directory where results are stored. Can leave unspecified.

Implementation

```
data(survData)
survData$yL <- survData$yU <- survData[,1]
survData$yU[which(survData[,2] == 0)] <- Inf
survData$LT <- rep(0, dim(survData)[1])
form <- Formula(LT | yL + yU ~ cov1 + cov2)
##
DPM.mu <- log(12)
DPM.sigSq <- 100
DPM.ab <-  c(2, 1)
Tau.ab <- c(1.5, 0.0125)
hyperParams <- list(DPM=list(DPM.mu=DPM.mu, DPM.sigSq=DPM.sigSq, DPM.ab=DPM.ab, Tau.ab=Tau.ab))
##
numReps    <- 1000
thin       <- 10
burninPerc <- 0.5
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))
##
myModel <- "DPM"
myPath  <- "Output/02-Results-DPM/"
startValues      <- initiate.startValues_AFT(form, survData, model=myModel, nChain=2)
##
fit_DPM <- BayesSurv_AFT(form, survData, model=myModel, hyperParams,
startValues, mcmcParams, path=myPath)

summary(fit_DPM)
pred_DPM <- predict(fit_DPM, time = seq(0, 35, 1), tseq=seq(from=0, to=30, by=5))
plot(pred_DPM, plot.est="Haz")
plot(pred_DPM, plot.est="Surv")
```

5. `BayesSurv_HReg` applied to cluster-correlated, univariate time-to-event data fit to a Cox PH model with Weibull baseline hazard

Definition of the survival model

Let t_{ji} denote the time-to-event of interest for individuals $i = 1, \ldots, n_j$ in cluster $j = 1, \ldots J$, subject to right censoring at time c_{ji}. Let $(y_{ji}, \delta_{ji}, x_{ji})$ denote independent observations, where $y_{ji} = \min\left(t_{ji}, c_{ji}\right)$, $\delta_{ji} = \mathbb{1}(t_{ji} \leq c_{ji})$, and x_{ji} is a vector of covariates for individual i. The following Cox proportional hazards model is assumed

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

where the V_j's are cluster-specific random effects and the baseline hazard h_0 is defined parametrically by a Weibull hazard, $h_0(t) = \alpha\kappa t^{\alpha-1}$.

In the Bayesian framework, priors must be specified for the regression parameter, β, the cluster-specific random effects, V_j, and the shape and scale parameters of baseline hazard function, α and κ, respectively. The prior distributions for β, α and κ are given below.

$$\begin{aligned}
\pi(\beta) &\propto 1, \\
\pi(\alpha) &\sim Gamma(a, b), \\
\pi(\kappa) &\sim Gamma(c, d).
\end{aligned}$$

We provide two possible prior specifications for the cluster-specific random effects below.

$$\begin{array}{rcl}
V_j & \sim & Normal(0, \sigma^2), \\
\zeta = \dfrac{1}{\sigma^2} & \sim & Gamma(a_N, b_N).
\end{array}
\qquad\qquad
\begin{array}{rcl}
V_j|m_j & \sim & Normal(\mu_m, \sigma_m^2), \\
(\mu_m, \sigma_m^2) & \sim & G_0, \quad \text{for} \quad m = 1, \ldots M, \\
m_j|p & \sim & Discrete(m_j|p_1, \ldots, p_M), \\
p & \sim & Dirichlet(\tau/M, \ldots, \tau/M), \\
\tau & \sim & Gamma(a_\tau, b_\tau).
\end{array}$$

In the first column, the individual specific-random effects are assumed to be $\overset{iid}{\sim} N(0, \sigma^2)$. In the second column, the cluster-specific random effects are drawn from a mixture of M normal distributions each with mean and variance (μ_m, σ_m^2) which are distributed as a multivariate Normal/Inverse-Gamma (NIG), denoted by G_0; we refer to this as the Dirichlet process mixture (DPM) prior. The probability density of G_0 is defined by the product

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

We assume $\mu_0 = 0$ and $\zeta_0 = 1$.

Hyperparameters

a, b	:	shape and rate of Gamma prior for α
c, d	:	shape and rate of Gamma prior for κ
a_N, b_N	:	mean and variance of normal prior for V_j
a_0, b_0	:	shape and rate of Gamma component of the prior distribution, G_0, of (μ_m, σ_m^2) (DPM prior)
a_τ, b_τ	:	shape and rate of Gamma hyperprior for τ (DPM prior)

Arguments to specify

Model-related	
`Formula`	a `Formula` object that corresponds to the hazard $h(t_i|x_i)$: $y + \delta \sim x$.
`data`	an $(n{\times}q)$-dimensional data.frame; the q-columns correspond to q covariate vectors named in the formula in `Formula`.
`model`	a character vector that specifies the type of components in the model. Use `model <- c("Weibull", "Normal")` for Normal prior for V_j and use `model <- c("Weibull", "DPM")` for DPM prior.
`id`	an n-vector of cluster information where cluster membership corresponds to one of the positive integers $1, \ldots, J$.
Hyperparameters	
`WB.ab`	a 2-vector of positive hyperparameters a and b of the prior distribution for the shape parameter α of the Weibull baseline hazard. Example: `WB.ab <- c(0.5, 0.01)`.
`WB.cd`	a 2-vector of positive hyperparameters c and d of the prior distribution for the scale parameter κ of the Weibull baseline hazard. Example: `WB.cd <- c(0.5, 0.05)`.
Normal prior for V_j	
`Normal.ab`	a 2-vector of positive hyperparameters a_N and b_N of the prior for $1/\sigma^2$, the precision of the normally distributed cluster-specific random effects. Example: `Normal.ab <- c(0.5, 0.01)`.
DPM prior for V_j	
`DPM.ab`	a 2-vector of positive hyperparameters a_0 and b_0 of the prior for (μ_m, σ_m^2), the parameters of the normally distributed cluster-specific random effects. Example: `DPM.ab <- c(0.5, 0.01)`.
`aTau`	a positive-valued hyperparameter corresponding to the shape parameter, a_τ, of the Gamma prior of τ.
`bTau`	a positive-valued hyperparameter corresponding to the rate parameter, b_τ, of the Gamma prior of τ.
MCMC Settings	
`numReps`	total number of scans
`thin`	extent of thinning, e.g. if `thin=10` retain every 10^{th} sample.
`burninPerc`	the proportion of burn-in (samples to be discarded before analyzing the data).
`mhProp_alpha_var`	the shape parameter α is updated using a Metropolis-Hastings random walk algorithm which generates proposals from a Gamma distribution with variance `mhProp_alpha_var`.
`mhProp_V_var`	the cluster-specific random effects, V_{ji}, are updated using a Metropolis-Hastings random walk algorithm which generates proposals from a Normal distribution with variance `mhProp_V_var`
Starting Values	
`startValues`	use `initiate.startValues_HReg(Formula, data, model, id, nChain, beta = NULL, WB.alpha = NULL, WB.kappa = NULL, V.j = NULL, Normal.zeta = NULL, DPM.class = NULL, DPM.tau = NULL)` which initiates starting values for β, α, κ, V_j, ζ (in the DPM model for V_j) and τ in the Metropolis-Hastings-Green algorithm if left unspecified; `DPM.class` sets the starting value for class membership in the DPM model. Users may set non-null starting values for any of these parameters.

Storage	
`path`	name of the directory where results are stored. Can leave unspecified.
`storeV`	a `TRUE/FALSE` logical constant indicating storage of V_j values.

Implementation

```
data(survData)
id=survData$cluster
form <- Formula(time + event ~ cov1 + cov2)
##
WB.ab <- c(0.5, 0.01) # prior parameters for alpha
WB.cd <- c(0.5, 0.05) # prior parameters for kappa
Normal.ab  <- c(0.5, 0.01)  # for Normal random effects
DPM.ab <- c(0.5, 0.01) # For DPM
    aTau  <- 1.5
    bTau  <- 0.0125
hyperParams.WB.Normal <- list(WB=list(WB.ab=WB.ab, WB.cd=WB.cd),
                        Normal=list(Normal.ab=Normal.ab))
hyperParams.WB.DPM <- list(WB=list(WB.ab=WB.ab, WB.cd=WB.cd),
                        DPM=list(DPM.ab=DPM.ab, aTau=aTau, bTau=bTau))
##
numReps <- 2000
burninPerc <- 0.5
thin <- 10
mhProp_alpha_var <- 0.01
mhProp_V_var     <- 0.05
storeV <- TRUE
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))
##
myModel.WB.Normal <- c("Weibull","Normal")
myPath.WB.Normal  <- "Output/03-Results-WB_Normal/"
startValues.WB.Normal <- initiate.startValues_HReg(form, survData, id, model=myModel.WB.Normal, nChain=2)
##
fit_WB_N <- BayesSurv_HReg(form, survData, id, model=myModel.WB.Normal, hyperParams.WB.Normal,
  startValues.WB.Normal, mcmc.WB, path=myPath.WB.Normal)
summary(fit_WB_N)
pred_WB_N <- predict(fit_WB_N, tseq=seq(from=0, to=30, by=5))
plot(pred_WB_N, plot.est="Haz")
plot(pred_WB_N, plot.est="Surv")
##
myModel.WB.DPM <- c("Weibull","DPM")
myPath.WB.DPM  <- "Output/04-Results-WB_DPM/"
startValues.WB.DPM <- initiate.startValues_HReg(form, survData, id, model=myModel.WB.DPM, nChain=2)
##
fit_WB_DPM <- BayesSurv_HReg(form, survData, id, model=myModel.WB.DPM, hyperParams.WB.DPM,
  startValues.WB.DPM, mcmc.WB, path=myPath.WB.DPM)
summary(fit_WB_DPM)
pred_WB_DPM <- predict(fit_WB_DPM, tseq=seq(from=0, to=30, by=5))
plot(pred_WB_DPM, plot.est="Haz")
plot(pred_WB_DPM, plot.est="Surv")
```

6. `BayesSurv` applied to cluster-correlated, univariate time-to-event data fit to a Cox PH model with PEM baseline hazard

Definition of the survival model

Let t_{ji} denote the time-to-event of interest for individuals $i = 1, \ldots, n_j$ in cluster $j = 1, \ldots J$, subject to right censoring at time c_{ji}. Let $(y_{ji}, \delta_{ji}, x_{ji})$ denote independent observations, where $y_{ji} = \min\left(t_{ji}, c_{ji}\right)$, $\delta_{ji} = \mathbb{1}(t_{ji} \leq c_{ji})$, and x_{ji} is a vector of covariates for individual i. The following Cox proportional hazards model is assumed

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

The baseline hazard h_0 is defined non-parametrically by a mixture of piecewise exponential functions as follows

$$\lambda_0(t) = \log h_0(t) = \sum_{k=1}^{K+1} \lambda_k \mathbb{1}\left\{t \in (s_{k-1}, s_k]\right\},$$

where λ_k is constant and the time interval between 0 and the largest observed failure time, denoted s_k, is partitioned into $K + 1$ disjoint intervals: $0 < s_1 < \cdots < s_{K+1}$.

In the Bayesian framework, priors must be specified for the regression parameter, β, the number of intervals, K, and the partition points $(s_1, \ldots, s_{K+1})$, respectively. The following specifications are made

$$\begin{aligned}
\pi(\beta) &\propto 1, \\
\lambda|K, \mu_\lambda, \sigma_\lambda^2 &\sim MVN_{K+1}(\mu_\lambda \mathbb{1}, \sigma_\lambda^2 \Sigma_\lambda) \\
K &\sim Poisson(\alpha), \\
\pi(s|K) &\propto \frac{(2K+1)! \prod_{k=1}^{K+1}(s_k - s_{k-1})}{(s_{K+1})^{(2K+1)}}, \\
\pi(\mu_\lambda) &\propto 1, \\
\sigma_\lambda^{-2} &\sim Gamma(a, b).
\end{aligned}$$

The prior specification for λ follows a MVN-ICAR (see Supplemental Material to Lee, Haneuse, Schrag and Dominici, 2015). Note that K and s jointly form a time-homogeneous Poisson process prior for the partition.

We provide two possible prior specifications for the cluster-specific random effects below.

$$\begin{array}{rcl}
V_j & \sim & Normal(0, \sigma^2), \\
\zeta = \dfrac{1}{\sigma^2} & \sim & Gamma(a_N, b_N).
\end{array}
\qquad\qquad
\begin{array}{rcl}
V_j|m_j & \sim & Normal(\mu_m, \sigma_m^2), \\
(\mu_m, \sigma_m^2) & \sim & G_0, \;\; \text{for} \;\; m = 1, \ldots M, \\
m_j|p & \sim & Discrete(m_j|p_1, \ldots, p_M), \\
p & \sim & Dirichlet(\tau/M, \ldots, \tau/M), \\
\tau & \sim & Gamma(a_\tau, b_\tau).
\end{array}$$

In the first column, the individual specific-random effects are assumed to be $\overset{iid}{\sim} N(0, \sigma^2)$. In the second column, the cluster-specific random effects are drawn from a mixture of M normal distributions each with mean and variance (μ_m, σ_m^2) which are distributed as a multivariate Normal/Inverse-Gamma (NIG), denoted by G_0; we refer to this as the Dirichlet process mixture (DPM) prior. The probability density of G_0 is defined by the product

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

We assume $\mu_0 = 0$ and $\zeta_0 = 1$.

Hyperparameters

α	: hyperparameter of K
a, b	: shape and rate of Gamma prior for σ_λ^{-2}
a_N, b_N	: mean and variance of normal prior for V_j
a_0, b_0	: shape and rate of Gamma component of the prior distribution, G_0, of (μ_m, σ_m^2)
a_τ, b_τ	: shape and rate of Gamma hyperprior for τ

Arguments to specify

Model-related

`Formula`	a `Formula` object that corresponds to the hazard $h(t_i|x_i)$: $y + \delta \sim x$.
`data`	an $(n{\times}q)$-dimensional data.frame; the q-columns correspond to q covariate vectors named in the formula in `Formula`.
`model`	a character vector that specifies the type of components in the model. Use `model <- c("PEM", "DPM")`.
`id`	an n-vector of cluster information where cluster membership corresponds to one of the positive integers $1, \ldots, J$.

Hyperparameters

`PEM.ab`	a 2-vector of positive hyperparameters a and b of the prior distribution for σ_λ^{-2}. Example: `PEM.ab <- c(0.7,0.7)`.
`PEM.alpha`	hyperparameter α of the prior distribution for K, which is one less than the number of partition points. Example: `PEM.alpha <- 10`.

Normal prior for V_j

`Normal.ab`	a 2-vector of positive hyperparameters a_N and b_N of the prior for $1/\sigma^2$, the precision of the normally distributed cluster-specific random effects. Example: `Normal.ab <- c(0.5, 0.01)`.

DPM prior for V_j

`DPM.ab`	a 2-vector of positive hyperparameters a_0 and b_0 of the prior for (μ_m, σ_m^2), the parameters of the normally distributed cluster-specific random effects. Example: `DPM.ab <- c(0.5, 0.01)`.
`aTau`	a positive-valued hyperparameter corresponding to the shape parameter, a_τ, of the Gamma prior of τ.
`bTau`	a positive-valued hyperparameter corresponding to the rate parameter, b_τ, of the Gamma prior of τ.

MCMC Settings

`numReps`	total number of scans
`thin`	extent of thinning, e.g. if `thin=10` retain every 10^{th} sample.
`burninPerc`	the proportion of burn-in (samples to be discarded before analyzing the data).

`mhProp_V_var`	the cluster-specific random effects, V_{ji}, are updated using a Metropolis-Hastings random walk algorithm which generates proposals from a Normal distribution with variance `mhProp_V_var`
`C`	a numeric value for the proportion that determines the sum of probabilities choosing the birth and death moves.[2]
`delPert`	the perturbation parameter in the birth updates; values must be between 0 and 0.5.[2]
`rj.scheme`	`rj.scheme=1`: the birth update will draw the proposal time split from $1 : s_{max}$; `rj.scheme=2`: the birth update will draw the proposal time split from uniquely ordered failure times in the data.
`K_max`	the number of splits allowed in each iteration of the Metropolis-Hastings-Green algorithm.
`s_max`	the largest observed failure time, given by `s_max <- max(data$time[data$event==1])`
`time_lambda`	time points at which the λ is monitored for convergence. Example: `time_lambda <- seq(1, s_max, 1)`. The chains for these monitoring points can be found in `lambda.fin` in the chains of the `BayesSurv` object.
Starting Values	
`startValues`	use `initiate.startValues_HReg(Formula, data, model, nChain, beta = NULL, V.j=NULL, Normal.zeta=NULL, DPM.class=NULL, DPM.tau=NULL)` which initiates starting values for β, V_j, ζ (in the DPM model for V_j) and τ in the Metropolis-Hastings-Green algorithm if left unspecified; `DPM.class` sets the starting value for class membership in the DPM model. Users may set non-null starting values for any of these parameters.
Storage	
`path`	name of the directory where results are stored. Can leave unspecified.
`storeV`	a `TRUE/FALSE` logical constant indicating storage of V_j values.

Implementation

```
data(survData)
id=survData$cluster
form <- Formula(time + event ~ cov1 + cov2)
##
PEM.ab <- c(0.7, 0.7) # prior parameters for 1/sigma^2
PEM.alpha <- 10 # prior parameters for K
Normal.ab  <- c(0.5, 0.01)  # for Normal random effects
DPM.ab <- c(0.5, 0.01) # For DPM
    aTau  <- 1.5
    bTau  <- 0.0125
hyperParams.PEM.Normal <- list(PEM=list(PEM.ab=PEM.ab, PEM.alpha=PEM.alpha),
                        Normal=list(Normal.ab=Normal.ab))
hyperParams.PEM.DPM <- list(PEM=list(PEM.ab=PEM.ab, PEM.alpha=PEM.alpha),
                        DPM=list(DPM.ab=DPM.ab, aTau=aTau, bTau=bTau))
##
numReps <- 2000
burninPerc <- 0.5
thin <- 10
mhProp_V_var     <- 0.05
storeV <- TRUE
C <- 0.2
delPert  <- 0.5
rj.scheme <- 2
K_max    <- 50
s_max    <- max(survData$time[survData$event == 1])
time_lambda <- seq(1, s_max, 0.5)
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, s_max=s_max, time_lambda=time_lambda) )
##
myModel.PEM.Normal <- c("PEM","Normal")
myPath.PEM.Normal  <- "Output/05-Results-PEM_Normal/"
startValues.PEM.Normal <- initiate.startValues_HReg(form, survData, id, model=myModel.PEM.Normal, nChain=2)
##
fit_PEM_N <- BayesSurv_HReg(form, survData, id, model=myModel.PEM.Normal, hyperParams.PEM.Normal,
  startValues.PEM.Normal, mcmc.PEM, path=myPath.PEM.Normal)
summary(fit_PEM_N)
pred_PEM_N <- predict(fit_PEM_N)
plot(pred_PEM_N, plot.est="Haz")
plot(pred_PEM_N, plot.est="Surv")
##
myModel.PEM.DPM <- c("PEM","DPM")
myPath.PEM.DPM  <- "Output/06-Results-PEM_DPM/"
startValues.PEM.DPM <- initiate.startValues_HReg(form, survData, id, model=myModel.PEM.DPM, nChain=2)
##
fit_PEM_DPM <- BayesSurv_HReg(form, survData, id, model=myModel.PEM.DPM, hyperParams.PEM.DPM,
  startValues.PEM.DPM, mcmc.PEM, path=myPath.PEM.DPM)
pred_PEM_DPM <- predict(fit_PEM_DPM)
plot(pred_PEM_DPM, plot.est="Haz")
plot(pred_PEM_DPM, plot.est="Surv")
```

---

[2] See Section A in Supplemental Material to Lee et al. (2015)

7. `BayesID_HReg` to analyze independent semi-competing risks data using an illness-death model with Weibull baseline hazards

Definition of the survival model

Let t_{i1} and t_{i2} denote the time to nonterminal event and terminal event from subject $i = 1, \ldots, n$, subject to right censoring at time c_i. Let $(y_{i1}, y_{i2}, \delta_{i1}, \delta_{i2}, x_i)$ denote independent observations, where $y_{i1} = \min(t_{i1}, t_{i2}, c_i)$, $\delta_{i1} = \mathbb{1}\{t_{i1} \leq \min(t_{i2}, c_i)\}$, $y_{i2} = \min(t_{i2}, c_i)$, $\delta_{i2} = \mathbb{1}\{t_{i2} \leq c_i\}$, and x_i is a vector of covariates for individual i. The independent semi-competing risks data are assumed to arise from an illness-death model system with transitions that are modeled through the following three hazard functions:

$$h_1(t_{i1}|\gamma_{ji}, x_{i1}) = \gamma_{ji} h_{01}(t_{i1}) \exp\left(x_{i1}^\top \beta_1\right), \quad t_{i1} > 0, \tag{1}$$

$$h_2(t_{i2}|\gamma_{ji}, x_{i2}) = \gamma_{ji} h_{02}(t_{i2}) \exp\left(x_{i2}^\top \beta_2\right), \quad t_{i2} > 0, \tag{2}$$

$$h_3(t_{i2}|t_{i1}, \gamma_{ji}, x_{i3}) = \gamma_{ji} h_{03}(t_{i2}) \exp\left(x_{i3}^\top \beta_3\right), \quad t_{i2} > 0, \tag{3}$$

where γ_{ji} is a subject-specific frailty with vectors of covariates x_{i1}, x_{i2} and x_{i3} which are subsets of x_i. The baseline hazard functions are defined parametrically by Weibull hazards of the form $h_{0g}(t) = \alpha_g \kappa_g t^{\alpha_g - 1}$, for $g \in \{1, 2, 3\}$. The baseline hazard function h_{03} is assumed to be Markov with respect to t_{i1}; we will refer to the set of conditional hazard functions in (1)-(3) as the Markov model. Alternatively, we consider modeling h_3 as follows:

$$h_3(t_{i2}|t_{i1}, \gamma_{ji}, x_{i3}) = \gamma_{ji} h_{03}(t_{i2} - t_{i1}) \exp\left(x_{i3}^\top \beta_3\right), \quad 0 < t_{i1} < t_{i2}. \tag{4}$$

We will refer to the set of conditional hazard functions in (1), (2) and (4) as the semi-Markov model.

In the Bayesian framework, priors must be specified for the regression parameter, β_g, the shape and scale parameters of baseline hazard function, α_g and κ_g, and the frailty parameter, γ_{ji}, respectively, for $g \in \{1, 2, 3\}$. The following specifications are made

$$\begin{aligned}
\pi(\beta_g) &\propto 1, \\
\alpha_g &\sim Gamma(a_g, b_g), \\
\kappa_g &\sim Gamma(c_g, d_g), \\
\gamma_{ji}|\theta &\sim Gamma(\theta^{-1}, \theta^{-1}), \\
\theta^{-1} &\sim Gamma(\psi, \omega).
\end{aligned}$$

Hyperparameters

a_g, b_g : shape and rate of Gamma prior for α_g for $g \in \{1, 2, 3\}$
c_g, d_g : shape and rate of Gamma prior for κ_g for $g \in \{1, 2, 3\}$
ψ : the shape of Gamma prior for θ^{-1}
ω : the rate of Gamma prior for θ^{-1}

Arguments to specify

Model-related	
`Formula`	a `Formula` object that corresponds to h_g, for $g \in \{1, 2, 3\}$: $y_1 + \delta_1 | y_2 + \delta_2 \sim x_1 | x_2 | x_3$.
`data`	an $(n{\times}q)$-dimensional data.frame; the q-columns correspond to q covariate vectors named in the formula in `Formula` below.
`model`	`c("Markov","Weibull")` for Markov definition of h_3 in (3); `c("semi-Markov","Weibull")` for semi-Markov definition of h_3 in (4).
Hyperparameters	
`WB.ab1`	a 2-vector of positive hyperparameters a_1 and b_1 of the prior distribution for the shape parameter α_1 of the Weibull baseline hazard. Example: `WB.ab1 <- c(0.5, 0.01)`.
`WB.ab2`	a 2-vector of positive hyperparameters a_2 and b_2 of the prior for α_2.
`WB.ab3`	a 2-vector of positive hyperparameters a_3 and b_3 of the prior for α_3.
`WB.cd1`	a 2-vector of positive hyperparameters c_1 and d_1 of the prior distribution for the scale parameter κ_1 of the Weibull baseline hazard. Example: `WB.cd1 <- c(0.5, 0.05)`.
`WB.cd2`	a 2-vector of positive hyperparameters c_2 and d_2 of the prior for κ_2.
`WB.cd3`	a 2-vector of positive hyperparameters c_3 and d_3 of the prior for κ_3.
`theta`	a 2-vector of positive hyperparameters ψ and ω for the hyperprior θ.
MCMC Settings	
. `numReps`	total number of scans
`thin`	extent of thinning, e.g. if `thin=10` retain every 10^{th} sample.
`burninPerc`	the proportion of burn-in (samples to be discarded before analyzing the data).
`mhProp_theta_var`	the parameter θ is updated using a Metropolis-Hastings random walk step generating proposals from a Gamma distribution with variance `mhProp_theta_var`.
`mhProp_alphag_var`	a 3-vector which specifies the variances of the three random walk Metropolis-Hastings Gamma proposal distributions corresponding to α_1, α_2, α_3.
Starting Values	
`startValues`	use `initiate.startValues_HReg(Formula, data, model, nChain, beta1 = NULL, beta2 = NULL, beta3 = NULL, gamma.ji=NULL, theta = NULL, WB.alpha = NULL, WB.kappa = NULL)` which initiates starting values for β_g, γ_{ji}, θ, α_g, and κ_g in the Metropolis-Hastings algorithm if left unspecified. Users may set non-null starting values for any of these parameters.
Storage	
`path`	name of the directory where results are stored. Can leave unspecified.
`nGam_save`	the number of γ to be stored.

Implementation

```
data(scrData)
form <- Formula(time1 + event1 | time2 + event2 ~ x1 + x2 + x3 | x1 + x2 | x1 + x2)
##
WB.ab1 <- c(0.5, 0.01)
WB.ab2 <- c(0.5, 0.01)
WB.ab3 <- c(0.5, 0.01)
WB.cd1 <- c(0.5, 0.05)
WB.cd2 <- c(0.5, 0.05)
WB.cd3 <- c(0.5, 0.05)
theta <- c(0.7, 0.7) # prior params for 1/theta
hyperParams <- list(theta=theta,
                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))
##
numReps    <- 2000
thin       <- 10
burninPerc <- 0.25
mhProp_theta_var  <- 0.05
mhProp_alphag_var <- c(0.01, 0.01, 0.01)
nGam_save <- 0
mcmc.WB  <- list(run=list(numReps=numReps, thin=thin, burninPerc=burninPerc),
                    storage=list(nGam_save=nGam_save),
                    tuning=list(mhProp_theta_var=mhProp_theta_var, mhProp_alphag_var=mhProp_alphag_var))
##
myModel <- c("semi-Markov", "Weibull")
myPath  <- "Output/01-Results-WB/"
startValues <- initiate.startValues_HReg(form, scrData, model=myModel, nChain=2)
##
fit_WB <- BayesID_HReg(form, scrData, id=NULL, model=myModel,
                hyperParams, startValues, mcmc.WB, path=myPath)
fit_WB
summary(fit_WB)
pred_WB <- predict(fit_WB, tseq=seq(from=0, to=30, by=5))
plot(pred_WB, plot.est="Haz")
plot(pred_WB, plot.est="Surv")
```

8. `BayesID_HReg` to analyze independent semi-competing risks data using an illness-death model with PEM baseline hazards

Definition of the survival model

Let t_{i1} and t_{i2} denote the time to nonterminal event and terminal event from subject $i = 1, \dots, n$, subject to right censoring at time c_i. Let $(y_{i1}, y_{i2}, \delta_{i1}, \delta_{i2}, x_i)$ denote independent observations, where $y_{i1} = \min(t_{i1}, t_{i2}, c_i)$, $\delta_{i1} = \mathbb{1}\{t_{i1} \leq \min(t_{i2}, c_i)\}$, $y_{i2} = \min(t_{i2}, c_i)$, $\delta_{i2} = \mathbb{1}\{t_{i2} \leq c_i\}$, and x_i is a vector of covariates for individual i. The independent semi-competing risks data are assumed to arise from an illness-death model system with transitions that are modeled through the following three hazard functions:

$$h_1(t_{i1}|\gamma_{ji}, x_{i1}) = \gamma_{ji} h_{01}(t_{i1}) \exp\left(x_{i1}^\top \beta_1\right), \quad t_{i1} > 0, \tag{5}$$

$$h_2(t_{i2}|\gamma_{ji}, x_{i2}) = \gamma_{ji} h_{02}(t_{i2}) \exp\left(x_{i2}^\top \beta_2\right), \quad t_{i2} > 0, \tag{6}$$

$$h_3(t_{i2}|t_{i1}, \gamma_{ji}, x_{i3}) = \gamma_{ji} h_{03}(t_{i2}) \exp\left(x_{i3}^\top \beta_3\right), \quad t_{i2} > 0, \tag{7}$$

where γ_{ji} is a subject-specific frailty with vectors of covariates x_{i1}, x_{i2} and x_{i3} which are subsets of x_i. The baseline hazard h_0 is defined non-parametrically by a mixture of piecewise exponential functions as follows

$$\lambda_0(t) = \log h_0(t) = \sum_{k=1}^{K+1} \lambda_k \mathbb{1}\{t \in (s_{k-1}, s_k]\},$$

where λ_k is constant and the time interval between 0 and the largest observed failure time, denoted s_k, is partitioned into $K+1$ disjoint intervals: $0 < s_1 < \cdots < s_{K+1}$. The baseline hazard function h_{03} is assumed to be Markov with respect to t_{i1}; we will refer to the set of conditional hazard functions in (5)-(7) as the Markov model. Alternatively, we consider modeling h_3 as follows:

$$h_3(t_{i2}|t_{i1}, \gamma_{ji}, x_{i3}) = \gamma_{ji} h_{03}(t_{i2} - t_{i1}) \exp\left(x_{i3}^\top \beta_3\right), \quad 0 < t_{i1} < t_{i2}. \tag{8}$$

We will refer to the set of conditional hazard functions in (5), (6) and (8) as the semi-Markov model.

In the Bayesian framework, priors must be specified for the regression parameter, β, the number of intervals, K, the partition points $(s_1, \dots, s_{K+1})$, and the frailty, γ_{ji}, respectively. The following specifications are made

$$\begin{aligned}
\pi(\beta) &\propto 1, \\
\lambda|K, \mu_\lambda, \sigma_\lambda^2 &\sim MVN_{K+1}(\mu_\lambda \mathbb{1}, \sigma_\lambda^2 \Sigma_\lambda) \\
K &\sim Poisson(\alpha), \\
\pi(s|K) &\propto \frac{(2K+1)! \prod_{k=1}^{K+1}(s_k - s_{k-1})}{(s_{K+1})^{(2K+1)}}, \\
\pi(\mu_\lambda) &\propto 1, \\
\sigma_\lambda^{-2} &\sim Gamma(a, b), \\
\gamma_{ji}|\theta &\sim Gamma(\theta^{-1}, \theta^{-1}), \\
\theta^{-1} &\sim Gamma(\psi, \omega).
\end{aligned}$$

The prior specification for λ follows a MVN-ICAR (see Supplemental Material to Lee, Haneuse, Schrag and Dominici, 2015). Note that K and s jointly form a time-homogeneous Poisson process prior for the partition.

Hyperparameters

α	:	parameter corresponding to the Poisson prior of K
a, b	:	shape and rate of Gamma prior for σ_λ^{-2}
ψ	:	the shape of Gamma prior for θ^{-1}
ω	:	the rate of Gamma prior for θ^{-1}

Arguments to specify

Model-related

`Formula`	a `Formula` object that corresponds to h_g, for $g \in \{1, 2, 3\}$: $y_1 + \delta_1 | y_2 + \delta_2 \sim x_1 | x_2 | x_3$.
`data`	an $(n \times q)$-dimensional data.frame; the q-columns correspond to q covariate vectors named in the formula in `Formula` below.
`model`	`c("Markov","PEM")` for Markov definition of h_3 in (7); `c("semi-Markov","PEM")` for semi-Markov definition of h_3 in (8).

Hyperparameters

`PEM.ab1`	a 2-vector of positive hyperparameters a_1 and b_1 which represent the shape and rate of the Gamma prior for $\sigma_{\lambda,1}^{-2}$. Example: `PEM.ab1 <- c(0.7,0.7)`.
`PEM.ab2`	a 2-vector of positive hyperparameters a and b of the prior distribution for $\sigma_{\lambda,2}^{-2}$.
`PEM.ab3`	a 2-vector of positive hyperparameters a and b of the prior distribution for $\sigma_{\lambda,3}^{-2}$.
`PEM.alpha1`	hyperparameter α of the prior distribution for K_1, which is one less than the number of partition points.
`PEM.alpha2`	hyperparameter α of the prior distribution for K_2, which is one less than the number of partition points.
`PEM.alpha3`	hyperparameter α of the prior distribution for K_3, which is one less than the number of partition points.
`theta`	a 2-vector of positive hyperparameters ψ and ω for the hyperprior θ.

MCMC Settings

`numReps`	total number of scans
`thin`	extent of thinning, e.g. if `thin=10` retain every 10^{th} sample.
`burninPerc`	the proportion of burn-in (samples to be discarded before analyzing the data).
`mhProp_theta_var`	the parameter θ is updated using a Metropolis-Hastings random walk step generating proposals from a Gamma distribution with variance `mhProp_theta_var`.

`Cg`	a 3-vector for the proportion that determines the sum of probabilities choosing the birth and death moves for each of the baseline hazards, h_{0g}, for $g \in \{1, 2, 3\}$. [3]
`delPertg`	a 3-vector for the perturbation parameter in the birth updates for all three baseline hazard functions; values must be between 0 and 0.5.[3]
`rj.scheme`	`rj.scheme=1`: the birth update will draw the proposal time split from $1 : s_{max}$; `rj.scheme=2`: the birth update will draw the proposal time split from uniquely ordered failure times in the data.
`Kg_max`	a 3-vector for the number of splits allowed in each iteration of the Metropolis-Hastings-Green algorithm for the three baseline hazard functions.
`sg_max`	the largest observed failure time, given by `sg_max <- c( max(data$time1[data$event1==1]), max(data$time2[data$event1==0 & data$event2==1]), max(data$time2[data$event1==1] & data$event2==1]))`
`time_lambda1`	time points at which the λ_1 is monitored for convergence. Example: `time_lambda1 <- seq(1, sg_max[1], 1)`. The chains for these monitoring points can be found in `lambda.fin` in the chains of the `BayesID` object.
`time_lambda2`	time points at which the λ_2 is monitored for convergence. Example: `time_lambda2 <- seq(1, sg_max[2], 1)`.
`time_lambda3`	time points at which the λ_3 is monitored for convergence. Example: `time_lambda3 <- seq(1, sg_max[3], 1)`.
Starting Values	
`startValues`	use `initiate.startValues_HReg(form, data, model, nChain)` which initiates all necessary starting values. Users may set non-null starting values for any of the following: `beta1, beta2, beta3, gamma.ji, theta`.
Storage	
`path`	name of the directory where results are stored. Can leave unspecified.
`nGam_save`	the number of γ to be stored.

Implementation

```
data(scrData)
form <- Formula(time1 + event1 | time2 + event2 ~ x1 + x2 + x3 | x1 + x2 | x1 + x2)
##
theta <- c(0.7, 0.7)
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
hyperParams <- list(theta=theta,
            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))
##
numReps    <- 2000
thin       <- 10
burninPerc <- 0.25
mhProp_theta_var  <- 0.05
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(scrData$time1[scrData$event1 == 1]),
                max(scrData$time2[scrData$event1 == 0 & scrData$event2 == 1]),
                max(scrData$time2[scrData$event1 == 1 & scrData$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)
nGam_save <- 0
mcmc.PEM <- list(run=list(numReps=numReps, thin=thin, burninPerc=burninPerc),
                   storage=list(nGam_save=nGam_save),
                   tuning=list(mhProp_theta_var=mhProp_theta_var,
                               Cg=Cg, delPertg=delPertg,
                               rj.scheme=rj.scheme, Kg_max=Kg_max, sg_max=sg_max,
                               time_lambda1=time_lambda1, time_lambda2=time_lambda2,
                               time_lambda3=time_lambda3))
##
myModel <- c("semi-Markov", "PEM")
myPath  <- "Output/02-Results-PEM/"
startValues <- initiate.startValues_HReg(form, scrData, model=myModel, nChain=2)
##
fit_PEM <- BayesID_HReg(form, scrData, id=NULL, model=myModel,
                   hyperParams, startValues, mcmc.PEM, path=myPath)
fit_PEM
summ.fit_PEM <- summary(fit_PEM); names(summ.fit_PEM)
summ.fit_PEM
pred_PEM <- predict(fit_PEM)
plot(pred_PEM, plot.est="Haz")
plot(pred_PEM, plot.est="Surv")
```

---

[3] See Section A in Supplemental Material to Lee et al. (2015)

9. `BayesID_AFT` to analyze independent semi-competing risks data using an illness-death model with log-Normal baseline survival distribution

Definition of the survival model

Let t_{i1}, t_{i2} denote time to non-terminal and terminal event from subject $i = 1, \ldots, n$. The independent semi-competing risks data are assumed to arise from an illness-death model system with transitions that are modeled through the following three hazard functions:

$$\begin{aligned}
\log(t_{i1}) &= x_{i1}^{\top}\beta_1 + \gamma_i + \epsilon_{i1}, \quad t_{i1} > 0, \\
\log(t_{i2}) &= x_{i2}^{\top}\beta_2 + \gamma_i + \epsilon_{i2}, \quad t_{i2} > 0, \\
\log(t_{i2} - t_{i1}) &= x_{i3}^{\top}\beta_3 + \gamma_i + \epsilon_{i3}, \quad t_{i2} > t_{i1},
\end{aligned}$$

where γ_i is a study participant-specific random effect, 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, $g \in \{1, 2, 3\}$. In the presence of interval censoring, the times-to-event for the i^{th} subject satisfy $c_{ij} \leq t_{i1} < c_{ij+1}$ for some j and $c_{ik} \leq t_{i2} < c_{ik+1}$ for some k. Let $\{c_{ij}, c_{ij+1}, c_{ik}, c_{ik+1}, L_i, x_{i1}, x_{i2}, x_{i3}\}$ denote independent observations, where L_i is the left-truncation time.

For the parametric AFT illness-death model, we build on the log-Normal formulation and take the ϵ_{ig} to follow independent Normal(μ_g, σ_g^2) distributions, g=1,2,3. In the Bayesian framework, priors must be specified for the unknown parameters. The following specifications are made

$$\begin{aligned}
\pi(\beta_g, \mu_g) &\propto 1, \\
\sigma_g^2 &\sim inverse-Gamma(a_{\sigma_g}, b_{\sigma_g}), \\
\gamma_i|\theta &\sim Normal(0, \theta), \\
\theta^{-1} &\sim inverse-Gamma(a_{\theta}, b_{\theta}).
\end{aligned}$$

Hyperparameters

The hyperparameters a_{σ_g} and b_{σ_g} must be specified for the prior of σ_g^2, as well as a_θ and b_θ, the rate and shape of the inverse-Gamma distributed hyperprior for θ.

Arguments to specify

Model-related	
`Formula`	a `Formula` object that corresponds to h_g, for $g \in \{1, 2, 3\}$: $L \mid y_{1L} + y_{1U} \mid y_{2L} + y_{2U} \sim x1 \mid x2 \mid x3$.
`data`	an $(n \times q)$-dimensional data.frame; the q-columns correspond to q covariate vectors named in the formula in `Formula`.
Hyperparameters	
`theta`	a 2-vector of positive hyperparameters a_θ and b_θ for the hyperprior θ.
`LN.ab1`	a 2-vector of positive hyperparameters a_{σ_1} and b_{σ_1} which represent the shape and rate of the inverse-Gamma prior for σ_1^2. Example: `LN..ab1 <- c(0.3,0.3)`.
`LN.ab2`	a 2-vector of positive hyperparameters a_{σ_2} and b_{σ_2} which represent the shape and rate of the inverse-Gamma prior for σ_2^2. Example: `LN..ab2 <- c(0.3,0.3)`.
`LN.ab3`	a 2-vector of positive hyperparameters a_{σ_3} and b_{σ_3} which represent the shape and rate of the inverse-Gamma prior for σ_3^2. Example: `LN..ab3 <- c(0.3,0.3)`.
MCMC Settings	
`numReps`	total number of scans
`thin`	extent of thinning, e.g. if `thin=10` retain every 10^{th} sample.
`burninPerc`	the proportion of burn-in (samples to be discarded before analyzing the data).
`betag.prop.var`	the parameter β_g is updated using a Metropolis-Hastings random walk step generating proposals from a Normal distribution with variance `betag.prop.var`.
`gamma.prop.var`	the parameter γ is updated using a Metropolis-Hastings random walk step generating proposals from a Normal distribution with variance `gamma.prop.var`.
`mug.prop.var`	the parameter μ_g is updated using a Metropolis-Hastings random walk step generating proposals from a Normal distribution with variance `mug.prop.var`.
`zetag.prop.var`	the parameter $\zeta_g = 1/\sigma_g^2$ is updated using a Metropolis-Hastings random walk step generating proposals from a log-Normal distribution with variance `zetag.prop.var`.
Starting Values	
`startValues`	use `initiate.startValues_AFT(Formula, data, model, nChain)` which initiates all necessary starting values. Users may set non-null starting values for any of the following: `beta1`, `beta2`, `beta3`, `gamma`, `theta`, `y1`, `y2`, `LN.mu`, `LN.sigSq`.
Storage	
`nGam_save`	the number of γ to be stored
`nY1_save`	the number of $\log(t_1)$ to be stored
`nY2_save`	the number of $\log(t_2)$ to be stored
`nY1.NA_save`	the number of $\mathbb{1}\{t_1 > t_2\}$ to be stored
`path`	name of the directory where results are stored. Can leave unspecified.

Implementation

```
data(scrData)
scrData$y1L <- scrData$y1U <- scrData[,1]
scrData$y1U[which(scrData[,2] == 0)] <- Inf
scrData$y2L <- scrData$y2U <- scrData[,3]
scrData$y2U[which(scrData[,4] == 0)] <- Inf
scrData$LT <- rep(0, dim(scrData)[1])
form <- Formula(LT | y1L + y1U | y2L + y2U  ~ x1 + x2 + x3 | x1 + x2 | x1 + x2)
##
theta.ab <- c(0.5, 0.05)
LN.ab1 <- c(0.3, 0.3)
LN.ab2 <- c(0.3, 0.3)
LN.ab3 <- c(0.3, 0.3)
hyperParams <- list(theta=theta.ab,
LN=list(LN.ab1=LN.ab1, LN.ab2=LN.ab2, LN.ab3=LN.ab3))
##
numReps    <- 300
thin       <- 3
burninPerc <- 0.5
nGam_save <- 10
nY1_save <- 10
nY2_save <- 10
nY1.NA_save <- 10
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))
##
myModel <- "LN"
myPath  <- "Output/01-Results-LN/"
startValues      <- initiate.startValues_AFT(form, scrData, model=myModel, nChain=2)
##
fit_LN <- BayesID_AFT(form, scrData, model=myModel, hyperParams,
startValues, mcmcParams, path=myPath)
summary(fit_LN)
pred_LN <- predict(fit_LN, time = seq(0, 35, 1), tseq=seq(from=0, to=30, by=5))
plot(pred_LN, plot.est="Haz")
plot(pred_LN, plot.est="Surv")
```

10. `BayesID_AFT` to analyze independent semi-competing risks data using an illness-death model with DPM baseline survival distribution

Definition of the survival model

Let t_{i1}, t_{i2} denote time to non-terminal and terminal event from subject $i = 1, \ldots, n$. The independent semi-competing risks data are assumed to arise from an illness-death model system with transitions that are modeled through the following three hazard functions:

$$
\begin{aligned}
\log(t_{i1}) &= x_{i1}^{\top}\beta_1 + \gamma_i + \epsilon_{i1}, \quad t_{i1} > 0,\\
\log(t_{i2}) &= x_{i2}^{\top}\beta_2 + \gamma_i + \epsilon_{i2}, \quad t_{i2} > 0,\\
\log(t_{i2} - t_{i1}) &= x_{i3}^{\top}\beta_3 + \gamma_i + \epsilon_{i3}, \quad t_{i2} > t_{i1},
\end{aligned}
$$

where γ_i is a study participant-specific random effect, 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, $g \in \{1, 2, 3\}$. In the presence of interval censoring, the times-to-event for the i^{th} subject satisfy $c_{ij} \leq t_{i1} < c_{ij+1}$ for some j and $c_{ik} \leq t_{i2} < c_{ik+1}$ for some k. Let $\{c_{ij}, c_{ij+1}, c_{ik}, c_{ik+1}, L_i, x_{i1}, x_{i2}, x_{i3}\}$ denote independent observations, where L_i is the left-truncation time.

For our semi-parametric AFT illness-death model, ϵ_{ig} is assumed to be taken as draws from the independent DPM of normal distributions:

$$
\begin{aligned}
\epsilon_{ig}|r_i &\sim Normal(\mu_{r_i}, \sigma^2_{r_i}),\\
(\mu_{gr}, \sigma^2_{gr}) &\sim G_{g0}, \ \text{for } r = 1, \ldots, M_g,\\
r_i|p_g &\sim Discrete(r_i|p_{g1}, \ldots, p_{gM_g}),\\
p_g &\sim Dirichlet(\tau_g/M_g, \ldots, \tau_g/M_g).
\end{aligned}
$$

In the Bayesian framework, priors must be specified for the unknown parameters. 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$(a_{\sigma_g}, b_{\sigma_g})$ for σ^2_{gr}. For regression parameters $\{\beta_1, \beta_2, \beta_3\}$, we adopt non-informative flat priors on the real line. For γ, we assume that each γ_i is an independent random draw from a Normal$(0, \theta)$ 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 specify a Gamma(a_{τ_g}, b_{τ_g}) hyperprior for the precision parameter τ_g.

Hyperparameters

a_θ, b_θ	:	the shape and rate of inverse-Gamma prior for θ
μ_{g0}, σ^2_{g0}, a_{σ_g}, b_{σ_g}	:	hyperparameters for G_{g0}
a_{τ_g}, b_{τ_g}	:	shape and rate of Gamma hyperprior for τ_g

Arguments to specify

Model-related	
`Formula`	a `Formula` object that corresponds to h_g, for $g \in \{1, 2, 3\}$: $L|y_{1L} + y_{1U}|y_{2L} + y_{2U} \sim x1|x2|x3$.
`data`	an $(n \times q)$-dimensional data.frame; the q-columns correspond to q covariate vectors named in the formula in `Formula`.
Hyperparameters	
`theta`	a 2-vector of positive hyperparameters a_θ and b_θ for the hyperprior θ.
`DPM.mu1`	a hyperparameter μ_{10}
`DPM.mu2`	a hyperparameter μ_{20}
`DPM.mu3`	a hyperparameter μ_{30}
`DPM.sigSq1`	a hyperparameter σ^2_{10}
`DPM.sigSq2`	a hyperparameter σ^2_{20}
`DPM.sigSq3`	a hyperparameter σ^2_{30}
`DPM.ab1`	a 2-vector of positive hyperparameters a_{σ_1}, b_{σ_1}
`DPM.ab2`	a 2-vector of positive hyperparameters a_{σ_2}, b_{σ_2}
`DPM.ab3`	a 2-vector of positive hyperparameters a_{σ_3}, b_{σ_3}
`Tau.ab1`	a 2-vector of positive hyperparameters a_{τ_1}, b_{τ_1}
`Tau.ab2`	a 2-vector of positive hyperparameters a_{τ_2}, b_{τ_2}
`Tau.ab3`	a 2-vector of positive hyperparameters a_{τ_3}, b_{τ_3}
MCMC Settings	
`numReps`	total number of scans
`thin`	extent of thinning, e.g. if `thin=10` retain every 10^{th} sample.
`burninPerc`	the proportion of burn-in (samples to be discarded before analyzing the data).
`betag.prop.var`	the parameter β_g is updated using a Metropolis-Hastings random walk step generating proposals from a Normal distribution with variance `betag.prop.var`.
`gamma.prop.var`	the parameter γ is updated using a Metropolis-Hastings random walk step generating proposals from a Normal distribution with variance `gamma.prop.var`.
`mug.prop.var`	the parameter μ_{gr} is updated using a Metropolis-Hastings random walk step generating proposals from a Normal distribution with variance `mug.prop.var`.
`zetag.prop.var`	the parameter $\zeta_{gr} = 1/\sigma^2_{gr}$ is updated using a Metropolis-Hastings random walk step generating proposals from a log-Normal distribution with variance `zetag.prop.var`.
Starting Values	
`startValues`	use `initiate.startValues_AFT(Formula, data, model, nChain)` which initiates all necessary starting values. Users may set non-null starting values for any of the following: `beta1, beta2, beta3, gamma, theta, y1, y2, DPM.class1, DPM.class2, DPM.class3, DPM.mu1, DPM.mu2, DPM.mu3, DPM.zeta1, DPM.zeta2, DPM.zeta3, DPM.tau`.
Storage	
`nGam_save`	the number of γ to be stored
`nY1_save`	the number of $\log(t_1)$ to be stored
`nY2_save`	the number of $\log(t_2)$ to be stored
`nY1.NA_save`	the number of $\mathbb{1}\{t_1 > t_2\}$ to be stored
`path`	name of the directory where results are stored. Can leave unspecified.

Implementation

```
data(scrData)
scrData$y1L <- scrData$y1U <- scrData[,1]
scrData$y1U[which(scrData[,2] == 0)] <- Inf
scrData$y2L <- scrData$y2U <- scrData[,3]
scrData$y2U[which(scrData[,4] == 0)] <- Inf
scrData$LT <- rep(0, dim(scrData)[1])
form <- Formula(LT | y1L + y1U | y2L + y2U  ~ x1 + x2 + x3 | x1 + x2 | x1 + x2)
##
theta.ab <- c(0.5, 0.05)
##
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,
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))
##
numReps    <- 300
thin       <- 3
burninPerc <- 0.5
nGam_save <- 10
nY1_save <- 10
nY2_save <- 10
nY1.NA_save <- 10
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))
##
myModel <- "DPM"
myPath  <- "Output/02-Results-DPM/"
startValues      <- initiate.startValues_AFT(form, scrData, model=myModel, nChain=2)
##
fit_DPM <- BayesID_AFT(form, scrData, model=myModel, hyperParams,
startValues, mcmcParams, path=myPath)
summary(fit_DPM);
pred_DPM <- predict(fit_DPM, time = seq(0, 35, 1), tseq=seq(from=0, to=30, by=5))
plot(pred_DPM, plot.est="Haz")
plot(pred_DPM, plot.est="Surv")
```

11. `BayesID_HReg` to analyze cluster-correlated semi-competing risks data using an illness-death model with Weibull baseline hazards

Definition of the survival model

Let t_{ji1} and t_{ji2} denote the time to nonterminal event and terminal event from subject $i = 1, \ldots, n_j$ in cluster $j = 1, \ldots, J$, subject to right censoring at time c_{ji}. Let $(y_{ji1}, y_{ji2}, \delta_{ji1}, \delta_{ji2}, x_{ji})$ denote independent observations, where $y_{ji1} = \min\left(t_{ji1}, t_{ji2}, c_{ji}\right)$, $\delta_{ji1} = \mathbb{1}\left\{t_{ji1} \leq \min\left(t_{ji2}, c_{ji}\right)\right\}$, $y_{ji2} = \min\left(t_{ji2}, c_{ji}\right)$, $\delta_{ji2} = \mathbb{1}\left\{t_{ji2} \leq c_{ji}\right\}$, and x_{ji} is a vector of covariates for individual i. The independent semi-competing risks data are assumed to arise from an illness-death model system with transitions that are modeled through the following three hazard functions:

$$h_1(t_{ji1}|\gamma_{ji}, x_{ji1}, V_{j1}) \quad = \gamma_{ji} h_{01}(t_{ji1}) \exp\left(x_{ji1}^{\top}\beta_1 + V_{j1}\right), \;\; t_{ji1} > 0, \tag{9}$$

$$h_2(t_{ji2}|\gamma_{ji}, x_{ji2}, V_{j2}) \quad = \gamma_{ji} h_{02}(t_{ji2}) \exp\left(x_{ji2}^{\top}\beta_2 + V_{j2}\right), \;\; t_{ji2} > 0, \tag{10}$$

$$h_3(t_{ji2}|t_{ji1}, \gamma_{ji}, x_{ji3}, V_{j3}) \quad = \gamma_{ji} h_{03}(t_{ji2}) \exp\left(x_{ji3}^{\top}\beta_3 + V_{j3}\right), \;\; t_{ji2} > 0, \tag{11}$$

where γ_{ji} is a subject-specific frailty, $V_j = (V_{j1}, V_{j2}, V_{j3})$ is a vector of cluster-specific random effects, andx_{ji1}, x_{ji2} and x_{ji3} which are subsets of x_i are vectors of covariates. The baseline hazard functions are defined parametrically by Weibull hazards of the form $h_{0g}(t) = \alpha_g \kappa_g t^{\alpha_g - 1}$, for $g \in \{1, 2, 3\}$. The baseline hazard function h_{03} is assumed to be Markov with respect to t_{ji1}; we will refer to the set of conditional hazard functions in (9)-(11) as the Markov model. Alternatively, we consider modeling h_3 as follows:

$$h_3(t_{ji2}|t_{ji1}, \gamma_{ji}, x_{ji3}, V_{j3}) \quad = \gamma_{ji} h_{03}(t_{ji2} - t_{ji1}) \exp\left(x_{ji3}^{\top}\beta_3 + V_{j3}\right), \;\; 0 < t_{ji1} < t_{ji2}. \tag{12}$$

We will refer to the set of conditional hazard functions in (9), (10) and (12) as the semi-Markov model.

In the Bayesian framework, priors must be specified for the regression parameter, β_g, the shape and scale parameters of baseline hazard function, α_g and κ_g, and the frailty parameter, γ_{ji}, respectively, for $g \in \{1, 2, 3\}$. The following specifications are made

$$\begin{aligned} \pi(\beta_g) &\propto 1, \\ \alpha_g &\sim Gamma(a_g, b_g), \\ \kappa_g &\sim Gamma(c_g, d_g), \\ \gamma_{ji}|\theta &\sim Gamma(\theta^{-1}, \theta^{-1}), \\ \theta^{-1} &\sim Gamma(\psi, \omega). \end{aligned}$$

We provide two possible prior specifications for the cluster-specific random effects below.

$$\begin{array}{rcl} V_j & \sim & MVN(0, \Sigma_V), \\ \Sigma_V & \sim & Inverse-Wishart(\Psi_v, \rho_v). \end{array} \qquad\qquad \begin{array}{rcl} V_j|m_j & \sim & MVN(\mu_{m_j}, \Sigma_{m_j}), \\ (\mu_m, \Sigma_m) & \sim & G_0, \;\; \text{for} \;\; m = 1, \ldots M, \\ m_j|p & \sim & Discrete(m_j|p_1, \ldots, p_M), \\ p & \sim & Dirichlet(\tau/M, \ldots, \tau/M), \\ \tau & \sim & Gamma(a_\tau, b_\tau). \end{array}$$

In the first column, the individual specific-random effects are assumed to be $\overset{iid}{\sim} MVN(0, \Sigma_V)$. In the second column, the cluster-specific random effects are drawn from a mixture of M multivariate normal distributions each with mean vector and covariance matrix (μ_m, Σ_m) which are distributed as a multivariate Normal/Inverse-Wishart (NIW), denoted by G_0; we refer to this as the Dirichlet process mixture (DPM) prior. The probability density of G_0 is defined by the product

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

Hyperparameters

a_g, b_g : shape and rate of Gamma prior for α_g for $g \in \{1, 2, 3\}$
c_g, d_g : shape and rate of Gamma prior for κ_g for $g \in \{1, 2, 3\}$
ψ : the shape of Gamma prior for θ^{-1}
ω : the rate of Gamma prior for θ^{-1}
Ψ_0, ρ_0 : shape and scale of Inverse-Wishart component of the prior distribution, G_0, of (μ_m, Σ_m) (DPM prior)
a_τ, b_τ : shape and rate of Gamma hyperprior for τ (DPM prior)

Arguments to specify

Model-related	
`Formula`	a `Formula` object that corresponds to h_g, for $g \in \{1, 2, 3\}$: $y_1 + \delta_1 \| y_2 + \delta_2 \sim x_1 \| x_2 \| x_3$.
`data`	an $(n{\times}q)$-dimensional data.frame; the q-columns correspond to q covariate vectors named in the formula in `Formula`.
`model`	`c("Markov","Weibull")` for Markov definition of h_3 in (11); `c("semi-Markov","Weibull")` for semi-Markov definition of h_3 in (12).
`id`	an n-vector of cluster information where cluster membership corresponds to one of the positive integers $1, \ldots, J$.
Hyperparameters	
`WB.ab1`	a 2-vector of positive hyperparameters a_1 and b_1 of the prior distribution for the shape parameter α_1 of the Weibull baseline hazard. Example: `WB.ab1 <- c(0.5, 0.01)`.
`WB.ab2`	a 2-vector of positive hyperparameters a_2 and b_2 of the prior for α_2.
`WB.ab3`	a 2-vector of positive hyperparameters a_3 and b_3 of the prior for α_3.
`WB.cd1`	a 2-vector of positive hyperparameters c_1 and d_1 of the prior distribution for the scale parameter κ_1 of the Weibull baseline hazard. Example: `WB.cd1 <- c(0.5, 0.05)`.
`WB.cd2`	a 2-vector of positive hyperparameters c_2 and d_2 of the prior for κ_2.
`WB.cd3`	a 2-vector of positive hyperparameters c_3 and d_3 of the prior for κ_3.
`theta`	a 2-vector of positive hyperparameters ψ and ω for the hyperprior θ.
MVN prior for V_{ji}	
`Psi_v`	a positive-definite scale matrix of the Inverse-Wishart prior for the cluster random effects, V_{ji} . Example: `Psi_v <- diag(1,3)`.
`rho_v`	the degrees of freedom of the Inverse-Wishart prior for V_{ji}. Example: `rho_v <- 100`.

DPM prior for V_{ji}	
`Psi0`	a positive-definite scale matrix of the Inverse-Wishart component of G_0. Example: `Psi0 <- diag(1,3)`.
`rho0`	the degrees of freedom of the Inverse-Wishart component of G_0. Example: `rho0 <- 10`.
`aTau`	a positive-valued hyperparameter corresponding to the shape parameter, a_τ, of the Gamma prior of τ.
`bTau`	a positive-valued hyperparameter corresponding to the rate parameter, b_τ, of the Gamma prior of τ.
MCMC Settings	
`numReps`	total number of scans
`thin`	extent of thinning, e.g. if `thin=10` retain every 10^{th} sample.
`burninPerc`	the proportion of burn-in (samples to be discarded before analyzing the data).
`mhProp_theta_var`	the parameter θ is updated using a Metropolis-Hastings random walk step generating proposals from a Gamma distribution with variance `mhProp_theta_var`.
`mhProp_alphag_var`	a 3-vector which specifies the variances of the three random walk Metropolis-Hastings Gamma proposal distributions corresponding to $\alpha_1,\ \alpha_2,\ \alpha_3$.
`mhProp_Vg_var`	a 3-vector which specifies the variances of the three random walk Metropolis-Hastings proposals from normal distributions with the same variance `mhProp_Vg_var`.
Starting Values	
`startValues`	use `initiate.startValues_HReg(Formula, data, model, id, nChain)` which initiates all necessary starting values. Users may set non-null starting values for: `beta1, beta2, beta3, theta, WB.alpha, WB.kappa, gamma.ji, V.j1, V.j2, V.j3, MVN.SigmaV, DPM.tau, DPM.class`.
Storage	
`path`	name of the directory where results are stored. Can leave unspecified.
`nGam_save`	the number of γ to be stored.
`storeV`	a 3-vector of `TRUE/FALSE` logical constants indicating storage of V_{ji} values for $g = 1, 2, 3$. Example: `storeV <- rep(TRUE, 3)`.

Implementation

```
data(scrData)
id=scrData$cluster
form <- Formula(time1 + event1 | time2 + event2 ~ x1 + x2 + x3 | x1 + x2 | x1 + x2)
##
WB.ab1 <- c(0.5, 0.01)
WB.ab2 <- c(0.5, 0.01)
WB.ab3 <- c(0.5, 0.01)
WB.cd1 <- c(0.5, 0.05)
WB.cd2 <- c(0.5, 0.05)
WB.cd3 <- c(0.5, 0.05)
theta <- c(0.7, 0.7) # prior params for 1/theta
Psi_v <- diag(1, 3) # MVN cluster-specific random effects
rho_v <- 100
Psi0  <- diag(1, 3) # DPM cluster-specific random effects
rho0  <- 10
aTau  <- 1.5
bTau  <- 0.0125
hyperParams.WB.MVN <- list(theta=theta,
                    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),
                           MVN=list(Psi_v=Psi_v, rho_v=rho_v))
hyperParams.WB.DPM <- list(theta=theta,
                    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),
                           DPM=list(Psi0=Psi0, rho0=rho0, aTau=aTau, bTau=bTau))
##
numReps    <- 2000
thin       <- 10
burninPerc <- 0.25
mhProp_theta_var  <- 0.05
mhProp_alphag_var <- c(0.01, 0.01, 0.01)
mhProp_Vg_var     <- c(0.05, 0.05, 0.05)
nGam_save <- 0
storeV    <- rep(TRUE, 3)
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_alphag_var=mhProp_alphag_var,
                    mhProp_Vg_var =mhProp_Vg_var))
##
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
##
myModel <- c("semi-Markov", "Weibull", "MVN")
myPath  <- "Output/03-Results-WB_MVN/ "
startValues      <- initiate.startValues_HReg(form, scrData, model=myModel, id, nChain=2)
##
```

```
fit_WB_MVN <- BayesID_HReg(form, scrData, id, model=myModel,
                        hyperParams.WB.MVN, 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
pred_WB_MVN <- predict(fit_WB_MVN, tseq=seq(from=0, to=30, by=5))
plot(pred_WB_MVN, plot.est="Haz")
plot(pred_WB_MVN, plot.est="Surv")
##
myModel <- c("semi-Markov", "Weibull", "DPM")
myPath  <- "Output/04-Results-WB_DPM/"
startValues      <- initiate.startValues_HReg(form, scrData, model=myModel, id, nChain=2)
fit_WB_DPM <- BayesID_HReg(form, scrData, id, model=myModel,
                        hyperParams.WB.DPM, 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
pred_WB_DPM <- predict(fit_WB_MVN, tseq=seq(from=0, to=30, by=5))
plot(pred_WB_DPM, plot.est="Haz")
plot(pred_WB_DPM, plot.est="Surv")
```

Definition of the survival model

Let t_{ji1} and t_{ji2} denote the time to nonterminal event and terminal event from subject $i = 1, \ldots, n_j$ in cluster $j = 1, \ldots, J$, subject to right censoring at time c_{ji}. Let $(y_{ji1}, y_{ji2}, \delta_{ji1}, \delta_{ji2}, x_{ji})$ denote independent observations, where $y_{ji1} = \min\left(t_{ji1}, t_{ji2}, c_{ji}\right)$, $\delta_{ji1} = \mathbb{1}\left\{t_{ji1} \leq \min\left(t_{ji2}, c_{ji}\right)\right\}$, $y_{ji2} = \min\left(t_{ji2}, c_{ji}\right)$, $\delta_{ji2} = \mathbb{1}\left\{t_{ji2} \leq c_{ji}\right\}$, and x_{ji} is a vector of covariates for individual i. The independent semi-competing risks data are assumed to arise from an illness-death model system with transitions that are modeled through the following three hazard functions:

$$\begin{aligned}
h_1(t_{ji1}|\gamma_{ji}, x_{ji1}, V_{j1}) &= \gamma_{ji} h_{01}(t_{ji1}) \exp\left(x_{ji1}^{\top}\beta_1 + V_{j1}\right), \ \ t_{ji1} > 0, && (13)\\
h_2(t_{ji2}|\gamma_{ji}, x_{ji2}, V_{j2}) &= \gamma_{ji} h_{02}(t_{ji2}) \exp\left(x_{ji2}^{\top}\beta_2 + V_{j2}\right), \ \ t_{ji2} > 0, && (14)\\
h_3(t_{ji2}|t_{ji1}, \gamma_{ji}, x_{ji3}, V_{j3}) &= \gamma_{ji} h_{03}(t_{ji2}) \exp\left(x_{ji3}^{\top}\beta_3 + V_{j3}\right), \ \ t_{ji2} > 0, && (15)
\end{aligned}$$

where γ_{ji} is a subject-specific frailty, $V_j = (V_{j1}, V_{j2}, V_{j3})$ is a vector of cluster-specific random effects, andx_{ji1}, x_{ji2} and x_{ji3} which are subsets of x_i are vectors of covariates. The baseline hazard h_0 is defined non-parametrically by a mixture of piecewise exponential functions as follows

$$\lambda_0(t) = \log h_0(t) = \sum_{k=1}^{K+1} \lambda_k \mathbb{1}\left\{t \in (s_{k-1}, s_k]\right\},$$

where λ_k is constant and the time interval between 0 and the largest observed failure time, denoted s_k, is partitioned into $K+1$ disjoint intervals: $0 < s_1 < \cdots < s_{K+1}$. The baseline hazard function h_{03} is assumed to be Markov with respect to t_{i1}; we will refer to the set of conditional hazard functions in (9)-(11) as the Markov model. Alternatively, we consider modeling h_3 as follows:

$$h_3(t_{ji2}|t_{ji1}, \gamma_{ji}, x_{ji3}, V_{j3}) = \gamma_{ji} h_{03}(t_{ji2} - t_{ji1}) \exp\left(x_{ji3}^{\top}\beta_3 + V_{j3}\right), \ \ 0 < t_{ji1} < t_{ji2}. \quad (16)$$

We will refer to the set of conditional hazard functions in (9), (10) and (12) as the semi-Markov model.

In the Bayesian framework, priors must be specified for the regression parameter, β, the number of intervals, K, the partition points $(s_1, \ldots, s_{K+1})$, and the frailty, γ_{ji}, respectively. The following specifications are made

$$\begin{aligned}
\pi(\beta) &\propto 1,\\
\lambda|K, \mu_\lambda, \sigma_\lambda^2 &\sim MVN_{K+1}(\mu_\lambda \mathbb{1}, \sigma_\lambda^2 \Sigma_\lambda)\\
K &\sim Poisson(\alpha),\\
\pi(s|K) &\propto \frac{(2K+1)! \prod_{k=1}^{K+1}(s_k - s_{k-1})}{(s_{K+1})^{(2K+1)}},\\
\pi(\mu_\lambda) &\propto 1,\\
\sigma_\lambda^{-2} &\sim Gamma(a, b),\\
\gamma_{ji}|\theta &\sim Gamma(\theta^{-1}, \theta^{-1}),\\
\theta^{-1} &\sim Gamma(\psi, \omega).
\end{aligned}$$

The prior specification for λ follows a MVN-ICAR (see Supplemental Material to Lee, Haneuse, Schrag and Dominici, 2015). Note that K and s jointly form a time-homogeneous Poisson process prior for the partition.

We provide two possible prior specifications for the cluster-specific random effects below.

$$\begin{array}{rcl}
V_j & \sim & MVN(0, \Sigma_V),\\
\Sigma_V & \sim & Inverse-Wishart(\Psi_v, \rho_v).
\end{array}
\qquad\qquad
\begin{array}{rcl}
V_j|m_j & \sim & MVN(\mu_{m_j}, \Sigma_{m_j}),\\
(\mu_m, \Sigma_m) & \sim & G_0, \ \text{ for } \ m = 1, \ldots M,\\
m_j|p & \sim & Discrete(m_j|p_1, \ldots, p_M),\\
p & \sim & Dirichlet(\tau/M, \ldots, \tau/M),\\
\tau & \sim & Gamma(a_\tau, b_\tau).
\end{array}$$

In the first column, the individual specific-random effects are assumed to be $\overset{iid}{\sim} MVN(0, \Sigma_V)$. In the second column, the cluster-specific random effects are drawn from a mixture of M multivariate normal distributions each with mean vector and covariance matrix (μ_m, Σ_m) which are distributed as a multivariate Normal/Inverse-Wishart (NIW), denoted by G_0; we refer to this as the Dirichlet process mixture (DPM) prior. The probability density of G_0 is defined by the product

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

Hyperparameters

α	:	parameter corresponding to the Poisson prior of K
a, b	:	shape and rate of Gamma prior for σ_λ^{-2}
ψ	:	the shape of Gamma prior for θ^{-1}
ω	:	the rate of Gamma prior for θ^{-1}
Ψ_0, ρ_0	:	shape and scale of Inverse-Wishart component of the prior distribution, G_0, of (μ_m, Σ_m) (DPM prior)
a_τ, b_τ	:	shape and rate of Gamma hyperprior for τ (DPM prior)

Arguments to specify

Model-related

`Formula`	a `Formula` object that corresponds to h_g, for $g \in \{1, 2, 3\}$: $y_1 + \delta_1 | y_2 + \delta_2 \sim x_1 | x_2 | x_3$.
`data`	an $(n \times q)$-dimensional data.frame; the q-columns correspond to q covariate vectors named in the formula in `Formula`.
`model`	`c("Markov","PEM")` for Markov definition of h_3 in (11); `c("semi-Markov","PEM")` for semi-Markov definition of h_3 in (12).
`id`	an n-vector of cluster information where cluster membership corresponds to one of the positive integers $1, \ldots, J$.

Hyperparameters

`PEM.ab1`	a 2-vector of positive hyperparameters a_1 and b_1 which represent the shape and rate of the Gamma prior for $\sigma^{-2}_{\lambda,1}$. Example: `PEM.ab1 <- c(0.7,0.7)`.
`PEM.ab2`	a 2-vector of positive hyperparameters a and b of the prior distribution for $\sigma^{-2}_{\lambda,2}$.
`PEM.ab3`	a 2-vector of positive hyperparameters a and b of the prior distribution for $\sigma^{-2}_{\lambda,3}$.
`PEM.alpha1`	hyperparameter α of the prior distribution for K_1, which is one less than the number of partition points.
`PEM.alpha2`	hyperparameter α of the prior distribution for K_2, which is one less than the number of partition points.
`PEM.alpha3`	hyperparameter α of the prior distribution for K_3, which is one less than the number of partition points.
`theta`	a 2-vector of positive hyperparameters ψ and ω for the hyperprior θ.

MVN prior for V_{ji}

`Psi_v`	a positive-definite scale matrix of the Inverse-Wishart prior for the cluster random effects, V_{ji} . Example: `Psi_v <- diag(1,3)`.
`rho_v`	the degrees of freedom of the Inverse-Wishart prior for V_{ji}. Example: `rho_v <- 100`.

DPM prior for V_{ji}

`Psi0`	a positive-definite scale matrix of the Inverse-Wishart component of G_0. Example: `Psi0 <- diag(1,3)`.
`rho0`	the degrees of freedom of the Inverse-Wishart component of G_0. Example: `rho0 <- 10`.
`aTau`	a positive-valued hyperparameter corresponding to the shape parameter, a_τ, of the Gamma prior of τ.
`bTau`	a positive-valued hyperparameter corresponding to the rate parameter, b_τ, of the Gamma prior of τ.

MCMC Settings

`numReps`	total number of scans
`thin`	extent of thinning, e.g. if `thin=10` retain every 10^{th} sample.
`burninPerc`	the proportion of burn-in (samples to be discarded before analyzing the data).
`mhProp_theta_var`	the parameter θ is updated using a Metropolis-Hastings random walk step generating proposals from a Gamma distribution with variance `mhProp_theta_var`.
`mhProp_Vg_var`	3-vector which specifies the variances of the three random walk Metropolis-Hastings proposals from normal distributions with the same variance `mhProp_Vg_var`.
`Cg`	a 3-vector for the proportion that determines the sum of probabilities choosing the birth and death moves for each of the baseline hazards, h_{0g}, for $g \in \{1, 2, 3\}$. [4]
`delPertg`	a 3-vector for the perturbation parameter in the birth updates for all three baseline hazard functions; values must be between 0 and 0.5.[4]
`rj.scheme`	`rj.scheme=1`: the birth update will draw the proposal time split from $1 : s_{max}$; `rj.scheme=2`: the birth update will draw the proposal time split from uniquely ordered failure times in the data.
`Kg_max`	a 3-vector for the number of splits allowed in each iteration of the Metropolis-Hastings-Green algorithm for the three baseline hazard functions.
`sg_max`	the largest observed failure time, given by

```
sg_max <- c( max(data$time1[data$event1==1]),
                  max(data$time2[data$event1==0 & data$event2==1]),
                 max(data$time2[data$event1==1] & data$event2==1]))
```

`time_lambda1`	time points at which the λ_1 is monitored for convergence. Example: `time_lambda1 <- seq(1, sg_max[1], 1)`. The chains for these monitoring points can be found in `lambda.fin` in the chains of the `BayesID_HReg` object.
`time_lambda2`	time points at which the λ_2 is monitored for convergence. Example: `time_lambda2 <- seq(1, sg_max[2], 1)`.
`time_lambda3`	time points at which the λ_3 is monitored for convergence. Example: `time_lambda3 <- seq(1, sg_max[3], 1)`.

Starting Values

`startValues`	use `initiate.startValues_HReg(Formula, data, model, id, nChain)` which initiates all necessary starting values. Users may set non-null starting values for any of the following: `beta1, beta2, beta3, gamma.ji, theta, V.j1, V.j2, V.j3, MVN.SigmaV, DPM.tau, DPM.class`.

Storage

`path`	name of the directory where results are stored. Can leave unspecified.
`nGam_save`	the number of γ to be stored.
`storeV`	a 3-vector of `TRUE/FALSE` logical constants indicating storage of V_{ji} values for $g = 1, 2, 3$. Example: `storeV <- rep(TRUE, 3)`.

Implementation

```
data(scrData)
id=scrData$cluster
form <- Formula(time1 + event1 | time2 + event2 ~ x1 + x2 + x3 | x1 + x2 | x1 + x2)
##
theta <- c(0.7, 0.7)
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
Psi_v <- diag(1, 3) # MVN cluster-specific random effects
rho_v <- 100
Psi0  <- diag(1, 3) # DPM cluster-specific random effects
rho0  <- 10
aTau  <- 1.5
bTau  <- 0.0125
```

---

[4] See Section A in Supplemental Material to Lee et al. (2015)

```
hyperParams.PEM.MVN <- list(theta=theta,
           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))
hyperParams.PEM.DPM <- list(theta=theta,
           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),
           DPM=list(Psi0=Psi0, rho0=rho0, aTau=aTau, bTau=bTau))

##
numReps    <- 2000
thin       <- 10
burninPerc <- 0.25
mhProp_theta_var  <- 0.05
mhProp_Vg_var     <- c(0.05, 0.05, 0.05)
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(scrData$time1[scrData$event1 == 1]),
               max(scrData$time2[scrData$event1 == 0 & scrData$event2 == 1]),
               max(scrData$time2[scrData$event1 == 1 & scrData$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)
nGam_save <- 0
storeV <- rep(TRUE, 3)
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, sg_max=sg_max,
                             time_lambda1=time_lambda1, time_lambda2=time_lambda2,
                             time_lambda3=time_lambda3))
##
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
##
myModel <- c("semi-Markov", "PEM", "MVN")
myPath  <- "Output/05-Results-PEM_MVN/"
startValues      <- initiate.startValues_HReg(form, scrData, model=myModel, id, nChain=2)
##
fit_PEM_MVN <- BayesID_HReg(form, scrData, id, model=myModel,
                    hyperParams.PEM.MVN, 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
pred_PEM_MVN <- predict(fit_PEM_MVN)
plot(pred_PEM_MVN, plot.est="Haz")
plot(pred_PEM_MVN, plot.est="Surv")
##
myModel <- c("semi-Markov", "PEM", "DPM")
myPath  <- "Output/06-Results-PEM_DPM/"
startValues      <- initiate.startValues_HReg(form, scrData, model=myModel, id, nChain=2)
##
fit_PEM_DPM <- BayesID_HReg(form, scrData, id, model=myModel,
                      hyperParams.PEM.DPM, 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
pred_PEM_DPM <- predict(fit_PEM_DPM)
plot(pred_PEM_DPM, plot.est="Haz")
plot(pred_PEM_DPM, plot.est="Surv")
```