A basic semicompeting risks model in Stan

Leah Comment

10/25/2017

Illness-death models

Introduction

The compartmental model representation of the issue of semicompeting risks conceptualizes an observational unit as belonging to exactly one state at any given time point. These models apply to contexts outside of illness and death, but it is illustrative to think of a person who may be "healthy," "ill," or "dead." Developing illness is a non-terminal event and dying is the terminal event. When illness occurs, it *must* occur prior to death. However, some subjects may die without ever experiencing the illness of interest.

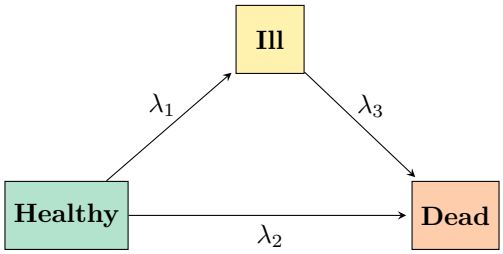


Figure 1: A compartmental model representation of an illness-death model in semicompeting risks

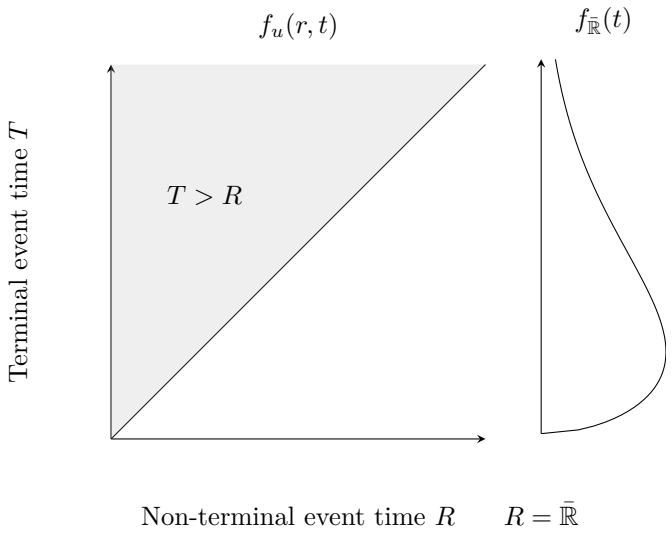
Notation

Suppose we have a non-terminal event R and a terminal event T. Both are subject to a censoring time which may vary by person. Let T_i denote the terminal event time for person i. When the non-terminal event occurs, $R_i \in \mathbb{R}_+$; when the non-terminal event is truncated by death, $R_i \equiv \bar{\mathbb{R}}$, a non-real number. The potential censoring time is C_i. The observed non-terminal event time is $Y_i^R = \min(R_i, T_i, C_i)$, with $\min(x, \bar{\mathbb{R}}) \equiv x$ for any $x \in \mathbb{R}$. The non-terminal event indicator δ_i^R is $\mathbf{1}(Y_i^R = R_i)$. Similarly, $Y_i^T = \min(T_i, C_i)$, and δ_i^R is $\mathbf{1}(Y_i^T = T_i)$.

Potential observed data patterns

Four distinct data patterns are possible:

Type	Description	(Y_i^R, Y_i^T)	(δ_i^R, δ_i^T)
1	Observe neither non-terminal nor terminal event	(c_i, c_i)	$(0, 0)$
2	Observe non-terminal event; terminal event is censored	(y_i^R, c_i)	$(1, 0)$
3	Observe terminal event without non-terminal event	(y_i^T, y_i^T)	$(0, 1)$
4	Observe non-terminal and terminal events	(y_i^R, y_i^T)	$(1, 1)$



Likelihood contributions by observed data pattern

The joint density for T and R is shown in . The portion where both event types are observed is only defined on the "upper wedge" where $T > R$; the joint density is $f_u(r,t)$. When the non-terminal event does not occur ($R = \bar{\mathrm{R}}$), the density of T is give $f_{\bar{\mathrm{R}}}(t)$.

For now, we assume a semi-Markov model for λ_3 where the hazard of the terminal event only depends on covariates and time elapsed since the non-terminal event. Ignoring measure theoretic issues and using $P(X = x)$ to refer to the pdf $f(x)$ for any continuous random variable,

$$\begin{aligned}
f_u(r,t) &= P(R = r, T = t) \text{ for } t > r \\
&= P(R = r, T > r) P(T = t | R = r, T > r) \\
&= \underbrace{\frac{P(R = r, T > r)}{P(R > r, T > r)}}_{\lambda_1(r)} \underbrace{P(R > r, T > r)}_{S_1(r)S_2(r)} \underbrace{\frac{P(T = t | R = r, T > r)}{P(T > t | R = r)}}_{\lambda_3(t-r)} \underbrace{P(T > t | R = r)}_{S_3(t-r)} \\
&= \lambda_1(r)\lambda_3(t-r) \exp\left\{-\Lambda_1(r) - \Lambda_3(t-r)\right\}
\end{aligned}$$

In a Markov model for death following the non-terminal event, $\lambda_3(t-r)$, $\Lambda_3(t-r)$, and $S_3(t-r)$ above would be replaced by $\lambda_3(t)$, $\Lambda_3(t)$, and $S_3(t)$.

The density for terminal event times among those who never experience the non-terminal event is

$$\begin{aligned}
f_{\bar{\mathbb{R}}} &= P(T = t, R = \bar{\mathbb{R}}) \\
&= \underbrace{\frac{P(R > t, T = t)}{P(R > r, T > r)}}_{\lambda_2(t)} \underbrace{P(R > r, T > r)}_{S_1(t)S_2(t)} \\
&= \lambda_2(t) \exp\left\{-\Lambda_1(t) + \Lambda_2(t)\right\}
\end{aligned}$$

Likelihood contributions by data pattern

For the semi-Markov model, the general form of the likelihood contributions is

(non-terminal contribution) × (terminal contribution)

TODO(LCOMM): add colors to contributions below

Type 1 (observe neither event type):

$$L_{1i} = P(R > c_i, T > c_i) = S_1(c_i)S_2(c_i) = S_1(y_i^R)S_2(y_i^T)$$

Type 2 (observe only non-terminal event):

$$L_{2i} = P(R = y_i^R, T > c_i) = \int_{c_i}^{\infty} f_u(y_i^R, u)du = f_1(y_i^R)S_2(y_i^R)S_3(c_i - y_i^R)$$

Type 3 (observe only terminal event):

$$L_{3i} = P(R > y_i^T, T = y_i^T) = \int_{y_i^T}^{\infty} f_u(u, y_i^R)du = f_2(y_i^T)S_1(y_i^T)$$

Type 4 (observe both event types):

$$L_{4i} = P(R = y_i^R, T = y_i^T) = f_u(y_i^R, y_i^T) = f_1(y_i^R)S_2(y_i^R)f_3(y_i^T - y_i^R)$$

TODO(LCOMM): fill in notation gaps above

Model formulation

Weibull models

The hazard of a Weibull regression model with shape α is

$$h(t|x_i) = \alpha t^{\alpha-1} \exp\{\beta_0 + x_i'\beta\}$$

With $\kappa = \exp(\beta_0)$, this is the parameterization used in Lee et al.

Alternatively, one can use the Stan hazard parameterization, where

$$h(t) = \frac{\alpha}{\sigma}\left(\frac{t}{\sigma}\right)^{\alpha-1}$$

To incorporate regression parameters into the model, we note that

$$\sigma^{-\alpha} = \exp\{x_i'\beta\}$$

and thus

$$\sigma = \exp\left\{-\frac{x_i'\beta}{\alpha}\right\}$$

Fitting semicompeting risk Weibull hazard models in Stan

```
# Packages
suppressPackageStartupMessages(library("rstan"))
rstan_options(auto_write = TRUE)
options(mc.cores = parallel::detectCores() - 1)
```

```r
# Read in semicompeting risk Weibull simulated data
dat_ID <- readRDS("dat_ID.Rdata")

# Package for Stan
x_m <- cbind(1, dat_ID$x_c)
stan_dat <-  list(N = nrow(dat_ID),
                  P_1 = ncol(x_m),
                  X_1 = x_m,
                  P_2 = ncol(x_m),
                  X_2 = x_m,
                  P_3 = ncol(x_m),
                  X_3 = x_m,
                  Yr = dat_ID$R,
                  dYr = dat_ID$delta_R,
                  Yt = dat_ID$T,
                  dYt = dat_ID$delta_T)
```

The Stan code to fit these models is shown below.

```
data {
  // number of observations
  int<lower=0> N;
  // number of columns in 1st design matrix, including intercept
  int<lower=1> P_1;
  // design matrix for non-terminal model
  matrix[N, P_1] X_1;
  // number of columns in 2nd design matrix, including intercept
  int<lower=1> P_2;
  // design matrix for terminal model w/o non-terminal history
  matrix[N, P_2] X_2;
  // number of columns in 3rd design matrix, including intercept
  int<lower=1> P_3;
  // design matrix for terminal model with non-terminal history
  matrix[N, P_3] X_3;
  // observed non-terminal time
  vector<lower=0>[N] Yr;
  // indicator of event observation for non-terminal event
  int<lower=0,upper=1> dYr[N];
  // observed terminal time
  vector<lower=0>[N] Yt;
  // indicator of event observation for terminal event
  int<lower=0,upper=1> dYt[N];
}

transformed data {
  // Duration of gap between non-terminal and terminal events
  vector[N] YtYrdiff;
  YtYrdiff = Yt - Yr;
}

parameters {
  // vectors of regression parameters
  vector[P_1] beta1;
  vector[P_2] beta2;
```

```
  vector[P_3] beta3;

  // shape parameters (the one in exponent of time)
  // alpha > 1 -> hazard increases over time, more clumping
  real<lower=0> alpha1;
  real<lower=0> alpha2;
  real<lower=0> alpha3;

  // scale parameters
  // bigger sigma -> slower event occurrence (double check this)
  //real<lower=0> sigma1;
  //real<lower=0> sigma2;
  //real<lower=0> sigma3;
}

model {
  // linear predictors
  vector[N] lp1;
  vector[N] lp2;
  vector[N] lp3;
  lp1 = X_1 * beta1;
  lp2 = X_2 * beta2;
  lp3 = X_3 * beta3;

  // no priors -> use Stan defaults

  // likelihood
  for (n in 1:N){
    if (dYr[n] == 0 && dYt[n] == 0) {

      // type 1: observe neither event
      target += weibull_lccdf(Yr[n] | alpha1, exp(-(lp1[n])/alpha1)) +
                weibull_lccdf(Yt[n] | alpha2, exp(-(lp2[n])/alpha2));

    } else if (dYr[n] == 1 && dYt[n] == 0) {

      // type 2: observe non-terminal but terminal censored
      target += weibull_lpdf(Yr[n] | alpha1, exp(-(lp1[n])/alpha1)) +
                weibull_lccdf(Yr[n] | alpha2, exp(-(lp2[n])/alpha2)) +
                weibull_lccdf(YtYrdiff[n] | alpha3, exp(-(lp3[n])/alpha3));

    } else if (dYr[n] == 0 && dYt[n] == 1) {

      // type 3: observed terminal with no prior non-terminal
      target += weibull_lccdf(Yr[n] | alpha1, exp(-(lp1[n])/alpha1)) +
                weibull_lpdf(Yt[n] | alpha2, exp(-(lp2[n])/alpha2));

    } else if (dYr[n] == 1 && dYt[n] == 1) {

      // type 4: both non-terminal and terminal observed
      target += weibull_lpdf(Yr[n] | alpha1, exp(-(lp1[n])/alpha1)) +
                weibull_lccdf(Yr[n] | alpha2, exp(-(lp2[n])/alpha2)) +
                weibull_lpdf(YtYrdiff[n] | alpha3, exp(-(lp3[n])/alpha3));
```

```
    }
  }
}
```

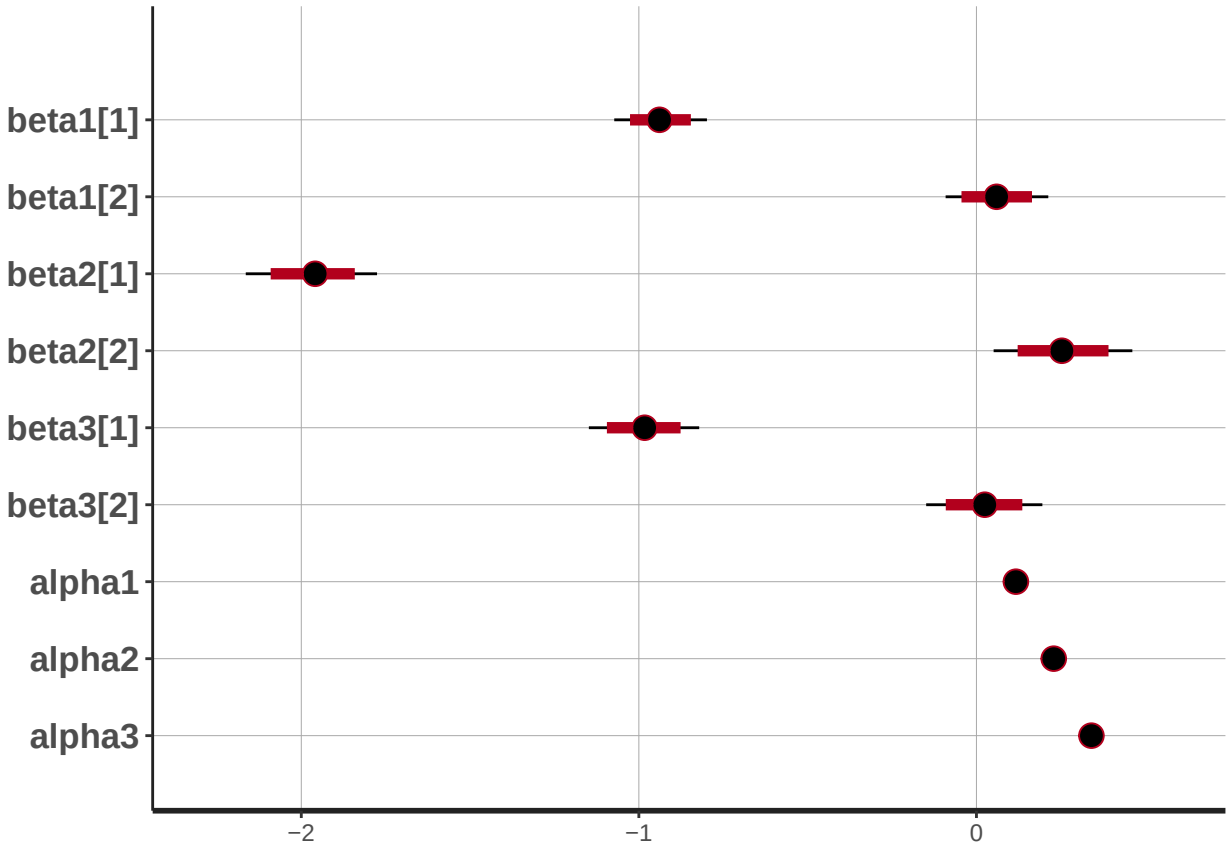
```
# Fit Weibull semicompeting model
library("rstan")
fit1 <- stan(file = "semicompeting_weibull.stan", data = stan_dat,
             iter = 1000, chains = 4)
summary(fit1)[["summary"]]
```

```
##                   mean      se_mean          sd          2.5%
## beta1[1] -9.386601e-01 1.561794e-03 0.069845551 -1.072907e+00
## beta1[2]  6.035495e-02 1.791009e-03 0.080096346 -9.133991e-02
## beta2[1] -1.962410e+00 2.227488e-03 0.098383963 -2.164367e+00
## beta2[2]  2.543048e-01 2.354872e-03 0.105313064  5.055478e-02
## beta3[1] -9.844034e-01 1.898033e-03 0.084882608 -1.147983e+00
## beta3[2]  2.302398e-02 1.987692e-03 0.088892270 -1.489141e-01
## alpha1    1.168827e-01 8.775352e-05 0.003924457  1.091760e-01
## alpha2    2.289271e-01 2.120797e-04 0.009484491  2.104455e-01
## alpha3    3.407741e-01 2.439019e-04 0.010907624  3.196419e-01
## lp__     -3.214298e+03 6.978859e-02 2.057081446 -3.219054e+03
##                   25%           50%           75%         97.5%     n_eff
## beta1[1] -9.843496e-01 -9.390387e-01 -8.918447e-01    -0.7984249 2000.0000
## beta1[2]  4.356931e-03  5.990967e-02  1.137667e-01     0.2127391 2000.0000
## beta2[1] -2.029885e+00 -1.958928e+00 -1.893099e+00    -1.7748420 1950.8231
## beta2[2]  1.817880e-01  2.529893e-01  3.270500e-01     0.4617319 2000.0000
## beta3[1] -1.042230e+00 -9.829984e-01 -9.251052e-01    -0.8212657 2000.0000
## beta3[2] -3.818477e-02  2.470272e-02  8.395738e-02     0.1957215 2000.0000
## alpha1    1.142895e-01  1.168423e-01  1.194361e-01     0.1248782 2000.0000
## alpha2    2.225156e-01  2.289380e-01  2.351512e-01     0.2476558 2000.0000
## alpha3    3.336239e-01  3.405563e-01  3.478461e-01     0.3626364 2000.0000
## lp__     -3.215470e+03 -3.213951e+03 -3.212786e+03 -3211.2691786  868.8286
##               Rhat
## beta1[1] 0.9995252
## beta1[2] 0.9994217
## beta2[1] 1.0021486
## beta2[2] 1.0008264
## beta3[1] 1.0014051
## beta3[2] 1.0001848
## alpha1   0.9981954
## alpha2   0.9991849
## alpha3   1.0001593
## lp__     1.0043064
```

```
plot(fit1)
```

```
## ci_level: 0.8 (80% intervals)
```

```
## outer_level: 0.95 (95% intervals)
```



TODO(LCOMM): Look into reparameterization to reduce numerical issues

```
# Compare to Lee et al
library("SemiCompRisks")
```

```
## Loading required package: MASS
```

```
## Loading required package: survival
```

```
# Objects
Y = data.frame(dat_ID$R, dat_ID$delta_R,
               dat_ID$T, dat_ID$delta_T)
lin.pred = list(formula(~ x_c),
                formula(~ x_c),
                formula(~ x_c))

### 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
```

```
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))

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

## Setting for the overall run
numReps    <- 5000
thin       <- 10
burninPerc <- 0.6

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

## Tuning parameters for specific updates
mhProp_theta_var  <- 0.05
mhProp_Vg_var     <- c(0.05, 0.05, 0.05)

## Specific to the Weibull specification of the baseline hazard functions
mhProp_alphag_var <- c(0.01, 0.01, 0.01)
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))
myModel <- c("semi-Markov", "Weibull")
startValues      <- vector("list", 2)
startValues[[1]] <- initiate.startValues_HReg(Y, lin.pred, dat_ID, model = myModel)
```

```
##
## [1] "Start values are initiated for semi-competing risks Weibull model..."
```

```
startValues[[2]] <- initiate.startValues_HReg(Y, lin.pred, dat_ID, model = myModel,
                                              theta = 0.23)
```

```
##
## [1] "Start values are initiated for semi-competing risks Weibull model..."
```

```
startValues[[3]] <- initiate.startValues_HReg(Y, lin.pred, dat_ID, model = myModel,
                                              theta = 0.72)
```

```
##
## [1] "Start values are initiated for semi-competing risks Weibull model..."
```

```
startValues[[4]] <- initiate.startValues_HReg(Y, lin.pred, dat_ID, model = myModel,
                                              theta = 0.55)
```

```
##
## [1] "Start values are initiated for semi-competing risks Weibull model..."
```

```
# Fit SemiCompRisks models
fit2 <- BayesID_HReg(Y, lin.pred, data = dat_ID,
                     hyperParams = hyperParams, startValues = startValues,
```

```
                    mcmc = mcmc.WB, path = "SemiCompRisks/Output/")
```

```
## chain:  1
## chain:  2
## chain:  3
## chain:  4
```

```
print(fit2)
```

```
##
## Analysis of independent semi-competing risks data
## semi-Markov assumption for h3
##
## Number of chains:     4
## Number of scans:      5000
## Thinning:             10
## Percentage of burnin: 60%
##
## ######
## Potential Scale Reduction Factor
##
## Variance of frailties, theta:
##  2.338
##
## Regression coefficients:
##     beta1 beta2 beta3
## x_c 1.094 1.059 1.037
##
## Baseline hazard function components:
##          h1    h2    h3
## kappa 1.004 1.065 1.090
## alpha 1.289 1.197 1.442
##
## ######
## Estimates
##
## Variance of frailties, theta:
##  Estimate    SD    LL  UL
##     0.233 0.076 0.118 0.4
##
## Regression coefficients:
##     Estimate    SD    LL    UL
## x_c    0.049 0.095 0.869 1.244
## x_c    0.245 0.117 1.075 1.675
## x_c    0.041 0.113 0.809 1.248
```

```
str(summary(fit2))
```

```
## List of 6
##  $ classFit: chr [1:4] "Bayes_HReg" "ID" "Ind" "WB"
##  $ psrf    :List of 3
##   ..$ theta: num [1, 1] 2.34
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr ""
##   .. .. ..$ : chr ""
```

```
##   ..$ coef : num [1, 1:3] 1.09 1.06 1.04
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr "x_c"
##   .. .. ..$ : chr [1:3] "beta1" "beta2" "beta3"
##   ..$ h0   : num [1:2, 1:3] 1 1.29 1.06 1.2 1.09 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:2] "kappa" "alpha"
##   .. .. ..$ : chr [1:3] "h1" "h2" "h3"
##  $ theta   : num [1, 1:3] 0.233 0.118 0.4
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr ""
##   .. ..$ : chr [1:3] "theta" "LL" "UL"
##  $ coef    : num [1, 1:9] 1.05 0.869 1.244 1.278 1.075 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr "x_c"
##   .. ..$ : chr [1:9] "exp(beta1)" "LL" "UL" "exp(beta2)" ...
##  $ h0      : num [1:2, 1:9] -2.079 -0.842 -2.134 -0.992 -2.026 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:2] "Weibull: log-kappa" "Weibull: log-alpha"
##   .. ..$ : chr [1:9] "h1-PM" "LL" "UL" "h2-PM" ...
##  $ setup   :List of 12
##   ..$ nCov       : int [1:3] 1 1 1
##   ..$ hyperParams: num [1:14] 0.5 0.01 0.5 0.01 0.5 0.01 0.5 0.05 0.5 0.05 ...
##   ..$ startValues:List of 4
##   .. ..$ :List of 2
##   .. .. ..$ common:List of 5
##   .. .. .. ..$ beta1   : num 0.0121
##   .. .. .. ..$ beta2   : num -0.0126
##   .. .. .. ..$ beta3   : num 0.0168
##   .. .. .. ..$ gamma.ji: num [1:1500] 0.399 0.124 0.335 0.03 1.486 ...
##   .. .. .. ..$ theta   : num 1.06
##   .. .. ..$ WB    :List of 2
##   .. .. .. ..$ WB.alpha: num [1:3] 1 1 1
##   .. .. .. ..$ WB.kappa: num [1:3] 0.01 0.01 0.01
##   .. ..$ :List of 2
##   .. .. ..$ common:List of 5
##   .. .. .. ..$ beta1   : num 0.0661
##   .. .. .. ..$ beta2   : num 0.0118
##   .. .. .. ..$ beta3   : num -0.0658
##   .. .. .. ..$ gamma.ji: num [1:1500] 0.849 0.676 0.844 0.805 0.719 ...
##   .. .. .. ..$ theta   : num 0.23
##   .. .. ..$ WB    :List of 2
##   .. .. .. ..$ WB.alpha: num [1:3] 1 1 1
##   .. .. .. ..$ WB.kappa: num [1:3] 0.01 0.01 0.01
##   .. ..$ :List of 2
##   .. .. ..$ common:List of 5
##   .. .. .. ..$ beta1   : num 0.0401
##   .. .. .. ..$ beta2   : num -0.0243
##   .. .. .. ..$ beta3   : num 0.0623
##   .. .. .. ..$ gamma.ji: num [1:1500] 1.7153 0.8886 0.0485 0.4012 0.6849 ...
##   .. .. .. ..$ theta   : num 0.72
##   .. .. ..$ WB    :List of 2
##   .. .. .. ..$ WB.alpha: num [1:3] 1 1 1
##   .. .. .. ..$ WB.kappa: num [1:3] 0.01 0.01 0.01
```

```
##   .. ..$ :List of 2
##   .. .. ..$ common:List of 5
##   .. .. .. ..$ beta1   : num 0.0288
##   .. .. .. ..$ beta2   : num 0.0623
##   .. .. .. ..$ beta3   : num -0.0032
##   .. .. .. ..$ gamma.ji: num [1:1500] 1.069 0.645 0.725 2.669 1.621 ...
##   .. .. .. ..$ theta   : num 0.55
##   .. .. ..$ WB    :List of 2
##   .. .. .. ..$ WB.alpha: num [1:3] 1 1 1
##   .. .. .. ..$ WB.kappa: num [1:3] 0.01 0.01 0.01
##   ..$ mcmcParams : num [1:4] 0.01 0.01 0.01 0.05
##   ..$ nGam_save  : num 0
##   ..$ numReps    : num 5000
##   ..$ thin       : num 10
##   ..$ path       : chr "SemiCompRisks/Output/"
##   ..$ burninPerc : num 0.6
##   ..$ hz.type    : chr "Weibull"
##   ..$ model      : chr "semi-Markov"
##   ..$ nChain     : int 4
##  - attr(*, "class")= chr "summ.Bayes_HReg"
```

```r
theta_est <- summary(fit2)[["theta"]][,1]
kappa_ests <- exp(summary(fit2)[["h0"]][1, c(1,4,7)])
beta0_ests <- summary(fit2)[["h0"]][1, c(1,4,7)]
alpha_ests <- exp(summary(fit2)[["h0"]][2, c(1,4,7)])
beta_ests <- log(summary(fit2)[["coef"]][c(1,4,7)])

# print(fit2)
# str(summary(fit2)$psrf)
# plot(fit2)
# str(fit2)
#
# # Extract comparable to Stan output
# beta01 <- log(summary(fit2)[["psrf"]][["h0"]]["kappa","h1"])
# beta02 <- log(summary(fit2)[["psrf"]][["h0"]]["kappa","h2"])
# beta03 <- log(summary(fit2)[["psrf"]][["h0"]]["kappa","h3"])
# beta_x <- summary(fit2)[["psrf"]][["coef"]]
```

TODO(LCOMM): Finish comparison table