Workbook for Introduction to TTE modeling usin brms

Parametric TTE models using brms

2023-07-13

Contents

Preliminaries for R examples **1**

Preliminaries for R examples

```
library(tidyverse)
library(stringr)
library(survival)
library(survminer)
library(texreg)
library(mgcv)
library(flexsurv)
library(muhaz)
library(Hmisc)
library(brms)
library(tidybayes)
library(bayesplot)

theme_set(theme_bw())
bayesplot::color_scheme_set("viridis")
```

We'll use the simulated data from Zecchin et al., as posted on the DDMoRe repository.

The objective of this analysis is to explore the relationship between change in tumor size through 12 weeks and overall survival landmarked at 12 weeks. We will fit two models, compare them using LOO-IC and VPCs.

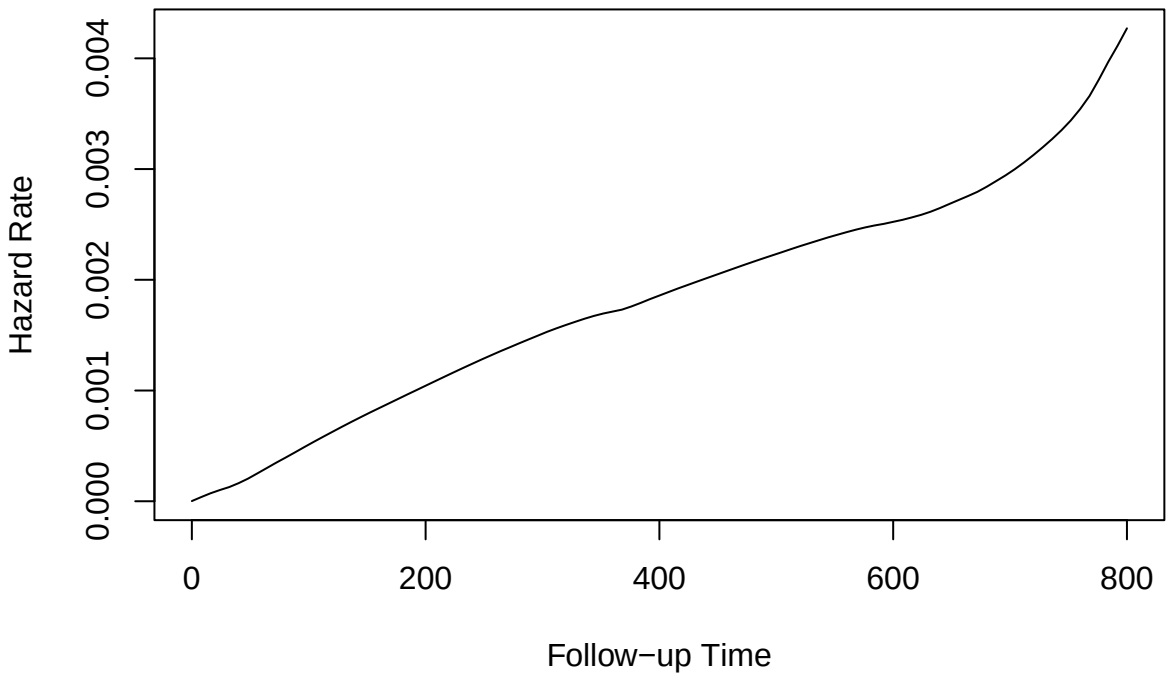
We'll start by reading the data

```
# OS data
d <- read_csv('../data/source/DDmodel0218_Simulated_OS.csv', na = c('.','-99'))

# Add week 12 (Day 84) predicted tumor size
d84 <- d %>%
  filter(TIME <= 84) %>%
  group_by(ID) %>%
  mutate(rate = KG/1000 - KD0/1000*AUC0 - KD1/100*AUC1,
         prevTIME = lag(TIME,default = 0),
         change = exp(rate * (TIME-prevTIME)),
         ipred = IBASE * 1000 * cumprod(change)
         ) %>%
  arrange(ID,TIME) %>%
  slice(n()) %>%
```

```
  mutate(ipred84 = ipred * exp(rate * (84-TIME)),
         rts84 = ipred84 / ( IBASE * 1000 ) )

dos <- d %>%
  filter(TIME>0) %>%
  group_by(ID) %>%
  mutate(meanGem = mean(AUC1),
         Group = if_else(meanGem > 0, "Cb+G", "Cb")) %>%
  ungroup() %>%
  filter(CMT==2, EVID==0) %>%
  left_join(d84 %>% select(ID, ipred84, rts84)) %>%
  mutate(rts84_f = paste0("Q", ntile(rts84, n = 4)))

# Create landmarked dataset and event times
dos84 <- dos %>%
  filter(TIME>84) %>%
  mutate(TIME = TIME-84)
```

Let's start by getting some idea about the shape of the hazard function. To do this, we can use the package `muhaz` to get a non-parametric estimate.

```
np_est = muhaz(times = dos84$TIME,
               delta = dos$DV,
               min.time=0,
               max.time = 800)

plot(np_est)
```



Using all of the data, the hazard appears to be monotonic - increasing at a roughly constant rate. Based on this, we can rule out the exponential model. Which models might be good candidates to fit this shape?

Let's try the Weibull and Gamma.

```
fit01_weibull <- brm(TIME | cens(1-DV) ~ rts84 + ECOG,
                             data = dos84,
                             family = 'weibull')
```

```
. Running /opt/R/4.1.3/lib/R/bin/R CMD SHLIB foo.c
. gcc -I"/opt/R/4.1.3/lib/R/include" -DNDEBUG   -I"/data/page/binary_tte_in_stan/rlibs/Rcpp/include/"  
. In file included from /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/Core:88:0,
.                  from /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/Dense:1,
.                  from /data/page/binary_tte_in_stan/rlibs/StanHeaders/include/stan/math/prim/mat/fun/
.                  from <command-line>:0:
. /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/src/Core/util/Macros.h:628:1: error: unkn
.  namespace Eigen {
.  ^~~~~~~~~
. /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/src/Core/util/Macros.h:628:17: error: exp
.  namespace Eigen {
.                  ^
. In file included from /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/Dense:1:0,
.                  from /data/page/binary_tte_in_stan/rlibs/StanHeaders/include/stan/math/prim/mat/fun/
.                  from <command-line>:0:
. /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/Core:96:10: fatal error: complex: No such
.  #include <complex>
.           ^~~~~~~~~
```

```
. compilation terminated.
. /opt/R/4.1.3/lib/R/etc/Makeconf:168: recipe for target 'foo.o' failed
. make: *** [foo.o] Error 1
.
. SAMPLING FOR MODEL '3c2a460e03594a5049bcdcbe70e3d91d' NOW (CHAIN 1).
. Chain 1:
. Chain 1: Gradient evaluation took 0.000233 seconds
. Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 2.33 seconds.
. Chain 1: Adjust your expectations accordingly!
. Chain 1:
. Chain 1:
. Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
. Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
. Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
. Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
. Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
. Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
. Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
. Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
. Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
. Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
. Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
. Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
. Chain 1:
. Chain 1:  Elapsed Time: 0.732225 seconds (Warm-up)
. Chain 1:                0.740348 seconds (Sampling)
. Chain 1:                1.47257 seconds (Total)
. Chain 1:
.
. SAMPLING FOR MODEL '3c2a460e03594a5049bcdcbe70e3d91d' NOW (CHAIN 2).
. Chain 2:
. Chain 2: Gradient evaluation took 0.000126 seconds
. Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 1.26 seconds.
. Chain 2: Adjust your expectations accordingly!
. Chain 2:
. Chain 2:
. Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
. Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
. Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
. Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
. Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
. Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
. Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
. Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
. Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
. Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
. Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
. Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
. Chain 2:
. Chain 2:  Elapsed Time: 0.723739 seconds (Warm-up)
. Chain 2:                0.778128 seconds (Sampling)
. Chain 2:                1.50187 seconds (Total)
. Chain 2:
.
```

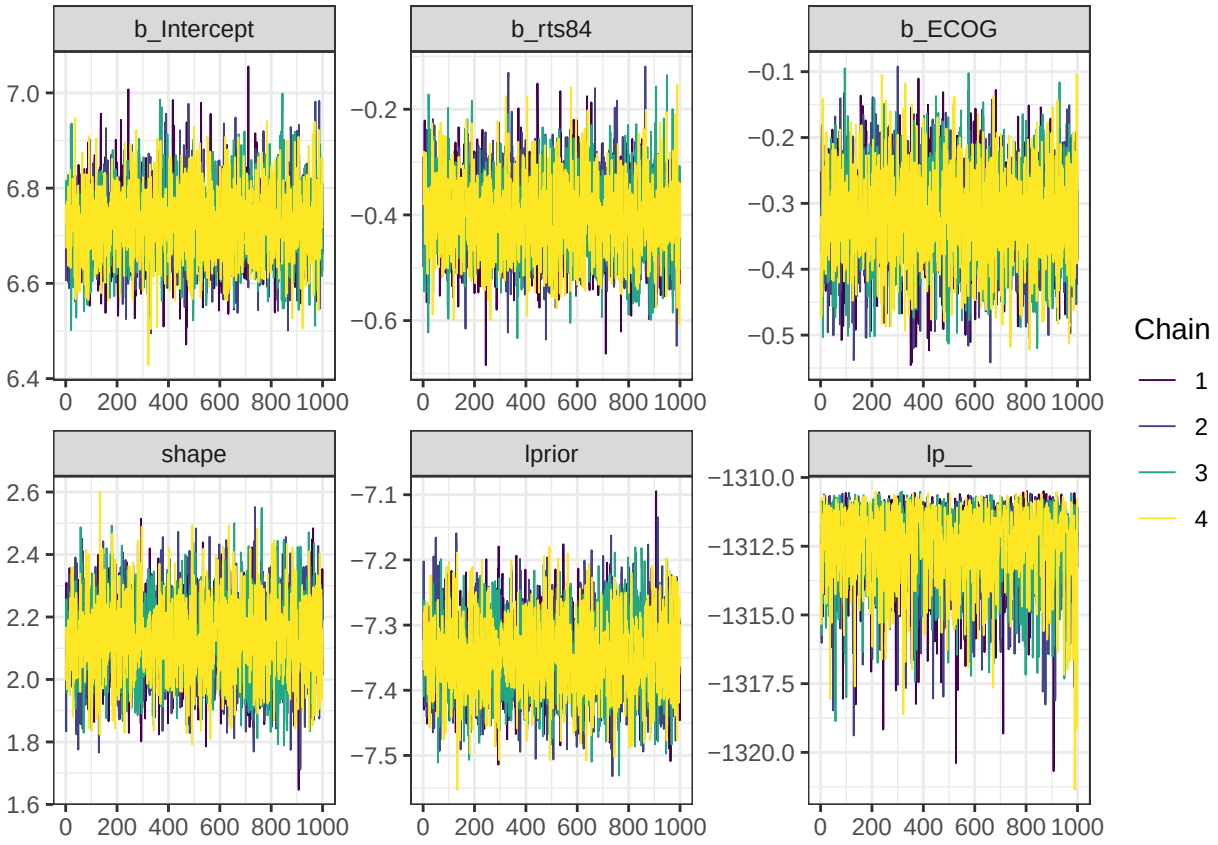
```
. SAMPLING FOR MODEL '3c2a460e03594a5049bcdcbe70e3d91d' NOW (CHAIN 3).
. Chain 3:
. Chain 3: Gradient evaluation took 0.000165 seconds
. Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 1.65 seconds.
. Chain 3: Adjust your expectations accordingly!
. Chain 3:
. Chain 3:
. Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
. Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
. Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
. Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
. Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
. Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
. Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
. Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
. Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
. Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
. Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
. Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
. Chain 3:
. Chain 3:  Elapsed Time: 0.774715 seconds (Warm-up)
. Chain 3:                0.759778 seconds (Sampling)
. Chain 3:                1.53449 seconds (Total)
. Chain 3:
.
. SAMPLING FOR MODEL '3c2a460e03594a5049bcdcbe70e3d91d' NOW (CHAIN 4).
. Chain 4:
. Chain 4: Gradient evaluation took 0.000131 seconds
. Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 1.31 seconds.
. Chain 4: Adjust your expectations accordingly!
. Chain 4:
. Chain 4:
. Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
. Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
. Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
. Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
. Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
. Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
. Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
. Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
. Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
. Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
. Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
. Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
. Chain 4:
. Chain 4:  Elapsed Time: 0.73555 seconds (Warm-up)
. Chain 4:                0.756412 seconds (Sampling)
. Chain 4:                1.49196 seconds (Total)
. Chain 4:
```

```
fit01_weibull
```

```
.  Family: weibull
.   Links: mu = log; shape = identity
. Formula: TIME | cens(1 - DV) ~ rts84 + ECOG
```

```
.    Data: dos84 (Number of observations: 336)
.   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
.          total post-warmup draws = 4000
.
. Population-Level Effects:
.           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
. Intercept     6.72      0.08     6.58     6.88 1.00     3439     2335
. rts84        -0.40      0.07    -0.54    -0.25 1.00     3734     3273
. ECOG         -0.32      0.07    -0.47    -0.18 1.00     3993     3149
.
. Family Specific Parameters:
.       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
. shape     2.13      0.12     1.88     2.38 1.00     3779     3055
.
. Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
. and Tail_ESS are effective sample size measures, and Rhat is the potential
. scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
mcmc_trace(fit01_weibull)
```



Let's use prior distributions other than default:

```
new_priors <- c(
  prior(student_t(10, 5, 3), class='Intercept'),
  prior(normal(0,3), class='b', coef='rts84'),
  prior(normal(0,1), class='b', coef='ECOG'),
  prior(lognormal(1,2),class='shape')
```

```
)
```

```
fit02_weibull <- brm(TIME | cens(1-DV) ~ rts84 + ECOG,
                     data = dos84,
                     prior = new_priors,
                     family = 'weibull')
```

```
. Running /opt/R/4.1.3/lib/R/bin/R CMD SHLIB foo.c
. gcc -I"/opt/R/4.1.3/lib/R/include" -DNDEBUG   -I"/data/page/binary_tte_in_stan/rlibs/Rcpp/include/"  -
. In file included from /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/Core:88:0,
.                  from /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/Dense:1,
.                  from /data/page/binary_tte_in_stan/rlibs/StanHeaders/include/stan/math/prim/mat/fun/
.                  from <command-line>:0:
. /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/src/Core/util/Macros.h:628:1: error: unkn
.  namespace Eigen {
.  ^~~~~~~~~
. /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/src/Core/util/Macros.h:628:17: error: exp
.  namespace Eigen {
.                  ^
. In file included from /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/Dense:1:0,
.                  from /data/page/binary_tte_in_stan/rlibs/StanHeaders/include/stan/math/prim/mat/fun/
.                  from <command-line>:0:
. /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/Core:96:10: fatal error: complex: No such
.  #include <complex>
.           ^~~~~~~~~
. compilation terminated.
. /opt/R/4.1.3/lib/R/etc/Makeconf:168: recipe for target 'foo.o' failed
. make: *** [foo.o] Error 1
.
. SAMPLING FOR MODEL '9f23344bd695b6f09ee24f43b64c189b' NOW (CHAIN 1).
. Chain 1:
. Chain 1: Gradient evaluation took 0.000208 seconds
. Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 2.08 seconds.
. Chain 1: Adjust your expectations accordingly!
. Chain 1:
. Chain 1:
. Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
. Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
. Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
. Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
. Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
. Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
. Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
. Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
. Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
. Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
. Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
. Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
. Chain 1:
. Chain 1:  Elapsed Time: 0.720094 seconds (Warm-up)
. Chain 1:                0.817605 seconds (Sampling)
. Chain 1:                1.5377 seconds (Total)
. Chain 1:
.
```

```
. SAMPLING FOR MODEL '9f23344bd695b6f09ee24f43b64c189b' NOW (CHAIN 2).
. Chain 2:
. Chain 2: Gradient evaluation took 0.000123 seconds
. Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 1.23 seconds.
. Chain 2: Adjust your expectations accordingly!
. Chain 2:
. Chain 2:
. Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
. Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
. Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
. Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
. Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
. Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
. Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
. Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
. Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
. Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
. Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
. Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
. Chain 2:
. Chain 2:  Elapsed Time: 0.736877 seconds (Warm-up)
. Chain 2:                0.725385 seconds (Sampling)
. Chain 2:                1.46226 seconds (Total)
. Chain 2:
.
. SAMPLING FOR MODEL '9f23344bd695b6f09ee24f43b64c189b' NOW (CHAIN 3).
. Chain 3:
. Chain 3: Gradient evaluation took 0.000154 seconds
. Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 1.54 seconds.
. Chain 3: Adjust your expectations accordingly!
. Chain 3:
. Chain 3:
. Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
. Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
. Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
. Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
. Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
. Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
. Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
. Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
. Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
. Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
. Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
. Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
. Chain 3:
. Chain 3:  Elapsed Time: 0.715589 seconds (Warm-up)
. Chain 3:                0.742669 seconds (Sampling)
. Chain 3:                1.45826 seconds (Total)
. Chain 3:
.
. SAMPLING FOR MODEL '9f23344bd695b6f09ee24f43b64c189b' NOW (CHAIN 4).
. Chain 4:
. Chain 4: Gradient evaluation took 0.000166 seconds
. Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 1.66 seconds.
```

```
. Chain 4: Adjust your expectations accordingly!
. Chain 4:
. Chain 4:
. Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
. Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
. Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
. Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
. Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
. Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
. Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
. Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
. Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
. Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
. Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
. Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
. Chain 4:
. Chain 4:  Elapsed Time: 0.748202 seconds (Warm-up)
. Chain 4:                0.83083 seconds (Sampling)
. Chain 4:                1.57903 seconds (Total)
. Chain 4:
```

How do the parameter estimates compare?

```
fit02_weibull
```

```
.  Family: weibull
.   Links: mu = log; shape = identity
. Formula: TIME | cens(1 - DV) ~ rts84 + ECOG
.    Data: dos84 (Number of observations: 336)
.   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
.          total post-warmup draws = 4000
.
. Population-Level Effects:
.           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
. Intercept     6.72      0.08     6.58     6.88 1.00     3588     3219
. rts84        -0.40      0.08    -0.55    -0.24 1.00     3784     3076
. ECOG         -0.32      0.07    -0.45    -0.18 1.00     3896     3115
.
. Family Specific Parameters:
.       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
. shape     2.13      0.12     1.90     2.37 1.00     3815     3132
.
. Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
. and Tail_ESS are effective sample size measures, and Rhat is the potential
. scale reduction factor on split chains (at convergence, Rhat = 1).
```

Let's look at the LOO-IC

```
loo_weibull = loo(fit01_weibull)
loo_weibull
```

```
.
. Computed from 4000 by 336 log-likelihood matrix
.
.          Estimate    SE
. elpd_loo  -1307.8  58.7
. p_loo         3.7   0.4
```

```
. looic      2615.6 117.3
. ------
. Monte Carlo SE of elpd_loo is 0.0.
.
. All Pareto k estimates are good (k < 0.5).
. See help('pareto-k-diagnostic') for details.
```

Now, let's look at on VPCs. Fortunately, it is easy to simulate from these models.

We'll simulate death times from the Weibull model. Following the slides, we'll simulate censoring times from a log-normal model.

```
weibull_sims <- add_predicted_draws(object=fit01_weibull,
                                    newdata = dos84 %>% select(ID, rts84, rts84_f, ECOG),
                                    value = 'survival_time')
```

For the censoring model, we'll use t-distribution priors with relatively low degrees of freedom.

```
censoring_priors <- prior('student_t(3,0,2.5)',class='b')

fit_censoring <- brm(TIME | cens(DV) ~ ECOG, data=dos84, prior = censoring_priors, family=lognormal())
```

```
. Running /opt/R/4.1.3/lib/R/bin/R CMD SHLIB foo.c
. gcc -I"/opt/R/4.1.3/lib/R/include" -DNDEBUG   -I"/data/page/binary_tte_in_stan/rlibs/Rcpp/include/"  -
. In file included from /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/Core:88:0,
.                  from /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/Dense:1,
.                  from /data/page/binary_tte_in_stan/rlibs/StanHeaders/include/stan/math/prim/mat/fun/
.                  from <command-line>:0:
. /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/src/Core/util/Macros.h:628:1: error: unkn
.  namespace Eigen {
.  ^~~~~~~~~
. /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/src/Core/util/Macros.h:628:17: error: exp
.  namespace Eigen {
.                 ^
. In file included from /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/Dense:1:0,
.                  from /data/page/binary_tte_in_stan/rlibs/StanHeaders/include/stan/math/prim/mat/fun/
.                  from <command-line>:0:
. /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/Core:96:10: fatal error: complex: No such
.  #include <complex>
.           ^~~~~~~~~
. compilation terminated.
. /opt/R/4.1.3/lib/R/etc/Makeconf:168: recipe for target 'foo.o' failed
. make: *** [foo.o] Error 1
.
. SAMPLING FOR MODEL '01210fb62131297c3300ce402dd3b065' NOW (CHAIN 1).
. Chain 1:
. Chain 1: Gradient evaluation took 0.00011 seconds
. Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 1.1 seconds.
. Chain 1: Adjust your expectations accordingly!
. Chain 1:
. Chain 1:
. Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
. Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
. Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
. Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
. Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
```

```
. Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
. Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
. Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
. Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
. Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
. Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
. Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
. Chain 1:
. Chain 1:  Elapsed Time: 0.325971 seconds (Warm-up)
. Chain 1:                0.331546 seconds (Sampling)
. Chain 1:                0.657517 seconds (Total)
. Chain 1:
.
. SAMPLING FOR MODEL '01210fb62131297c3300ce402dd3b065' NOW (CHAIN 2).
. Chain 2:
. Chain 2: Gradient evaluation took 7e-05 seconds
. Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.7 seconds.
. Chain 2: Adjust your expectations accordingly!
. Chain 2:
. Chain 2:
. Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
. Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
. Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
. Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
. Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
. Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
. Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
. Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
. Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
. Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
. Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
. Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
. Chain 2:
. Chain 2:  Elapsed Time: 0.321819 seconds (Warm-up)
. Chain 2:                0.348279 seconds (Sampling)
. Chain 2:                0.670098 seconds (Total)
. Chain 2:
.
. SAMPLING FOR MODEL '01210fb62131297c3300ce402dd3b065' NOW (CHAIN 3).
. Chain 3: Rejecting initial value:
. Chain 3:   Log probability evaluates to log(0), i.e. negative infinity.
. Chain 3:   Stan can't start sampling from this initial value.
. Chain 3: Rejecting initial value:
. Chain 3:   Log probability evaluates to log(0), i.e. negative infinity.
. Chain 3:   Stan can't start sampling from this initial value.
. Chain 3:
. Chain 3: Gradient evaluation took 6.7e-05 seconds
. Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.67 seconds.
. Chain 3: Adjust your expectations accordingly!
. Chain 3:
. Chain 3:
. Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
. Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
. Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
```

```
. Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
. Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
. Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
. Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
. Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
. Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
. Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
. Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
. Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
. Chain 3:
. Chain 3:  Elapsed Time: 0.325719 seconds (Warm-up)
. Chain 3:                0.322826 seconds (Sampling)
. Chain 3:                0.648545 seconds (Total)
. Chain 3:
.
. SAMPLING FOR MODEL '01210fb62131297c3300ce402dd3b065' NOW (CHAIN 4).
. Chain 4:
. Chain 4: Gradient evaluation took 6.7e-05 seconds
. Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.67 seconds.
. Chain 4: Adjust your expectations accordingly!
. Chain 4:
. Chain 4:
. Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
. Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
. Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
. Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
. Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
. Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
. Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
. Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
. Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
. Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
. Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
. Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
. Chain 4:
. Chain 4:  Elapsed Time: 0.335109 seconds (Warm-up)
. Chain 4:                0.341129 seconds (Sampling)
. Chain 4:                0.676238 seconds (Total)
. Chain 4:
```

- Check Rhat values - has the sampler converged?
- Do you know how to interpret the parameter values?

```
fit_censoring
```

```
.  Family: lognormal
.   Links: mu = identity; sigma = identity
. Formula: TIME | cens(DV) ~ ECOG
.    Data: dos84 (Number of observations: 336)
.   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
.          total post-warmup draws = 4000
.
. Population-Level Effects:
.           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
. Intercept     6.36      0.04     6.28     6.44 1.00     3095     2774
```

```
. ECOG         -0.22      0.06    -0.33    -0.10 1.00     3362     2775
.
. Family Specific Parameters:
.       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
. sigma     0.45      0.03     0.40     0.50 1.00     3084     2603
.
. Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
. and Tail_ESS are effective sample size measures, and Rhat is the potential
. scale reduction factor on split chains (at convergence, Rhat = 1).
```

Next we'll simulate censoring times

```
censoring_sims <- add_predicted_draws(object=fit_censoring,
                                      newdata = dos84 %>% select(ID, rts84, rts84_f, ECOG),
                                      value = 'censoring_time')
```

Finally, we'll censor the death times.

```
event_sims <- weibull_sims %>% left_join(censoring_sims) %>%
  mutate(event_time = pmin(survival_time, censoring_time),
         delta = survival_time < censoring_time)
```

To make the VPC, we'll write a function to accept any factor variable. That will let us make plots for RTS and ECOG.

```
vpc_stat_km <- function(.data,
                        group.var = NULL,
                        pred_times=NULL) {
  if (is.null(group.var)) stop("Must specify a grouping variable.")

  .data$group <- .data[[group.var]]

  fit <- survfit(Surv(time,event)~group, data=.data)
  if (is.null(pred_times)) {
    pred_times <- c(0,sort(unique(fit$time)))
  }
  preds = summary(fit, times=pred_times)

  data.frame(pred_times=preds$time, preds = preds$surv, group=preds$strata)
}
```

Calculate K-M estimator for observed data

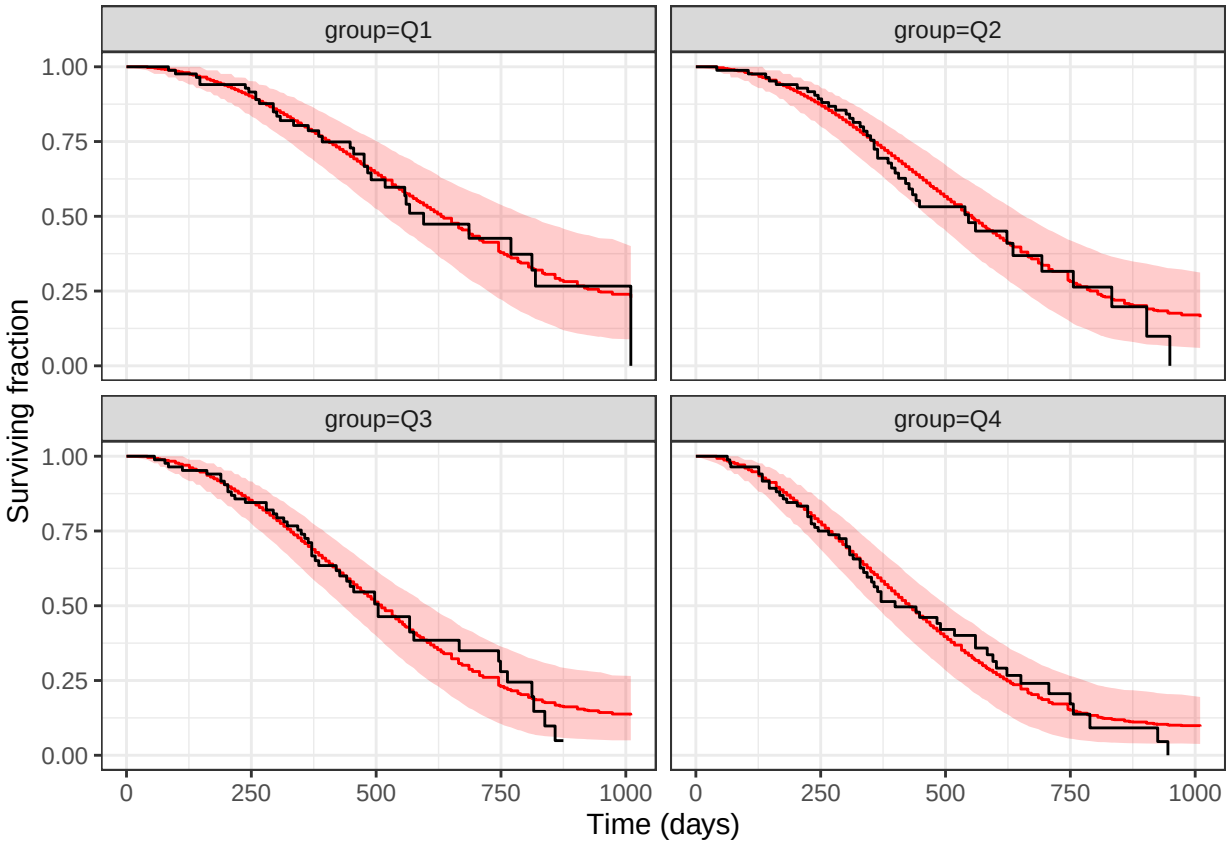
```
obs_surv = vpc_stat_km(dos84 %>% mutate(time=TIME, event=DV), group.var = 'rts84_f')

tail(obs_surv)
```

Apply the summary statistic to each simulated dataset

```
sim_surv = event_sims %>%
  mutate(time=event_time, event=as.numeric(delta)) %>%
  nest(data = -.draw) %>%
  mutate(km_est = map(data, ~vpc_stat_km(., group.var='rts84_f',pred_times=sort(unique(obs_surv$pred_ti
  select(-data) %>%
  unnest(cols=km_est)

tail(sim_surv)
```

Plot survival function VPC

```
sim_surv %>% group_by(pred_times, group) %>%
  summarise(med=mean(preds),
            lcl = quantile(preds,probs = 0.05),
            ucl = quantile(preds, probs=0.95)) %>%
  ggplot(aes(x=pred_times)) +
  geom_step(aes(y=med), color='red') +
  geom_ribbon(aes(ymin=lcl, ymax=ucl), fill='red', alpha=0.2) +
  geom_step(data=obs_surv,aes(y=preds)) +
  facet_wrap(~group) +
  ylim(0,1) + labs(x='Time (days)', y='Surviving fraction')
```



Exercise:

- Make VPCs for the censoring model by RTS quartile and ECOG status. Do you need to update the censoring model?

Answer:

For evaluating the censoring model, we'll reverse the event indicator. That is, an "event" now corresponds to a patient being censored. We'll do this for both the observed and simulated data and then plot the VPC.

```
obs_cens = vpc_stat_km(dos84 %>% mutate(time=TIME, event=1-DV), group.var = 'rts84_f')

sim_cens = event_sims %>%
  mutate(time=event_time, event=1-as.numeric(delta)) %>%
  nest(data = -.draw) %>%
  mutate(km_est = map(data, ~vpc_stat_km(., group.var='rts84_f',pred_times=sort(unique(obs_surv$pred_ti
  select(-data) %>%
  unnest(cols=km_est)
```

```
sim_cens %>% group_by(pred_times, group) %>%
  summarise(med=mean(preds),
            lcl = quantile(preds,probs = 0.05),
            ucl = quantile(preds, probs=0.95)) %>%
  ggplot(aes(x=pred_times)) +
  geom_step(aes(y=med), color='red') +
  geom_ribbon(aes(ymin=lcl, ymax=ucl), fill='red', alpha=0.2) +
  geom_step(data=obs_cens,aes(y=preds)) +
  facet_wrap(~group) +
  ylim(0,1) + labs(x='Time (days)', y='Surviving fraction')
```

This looks pretty good.

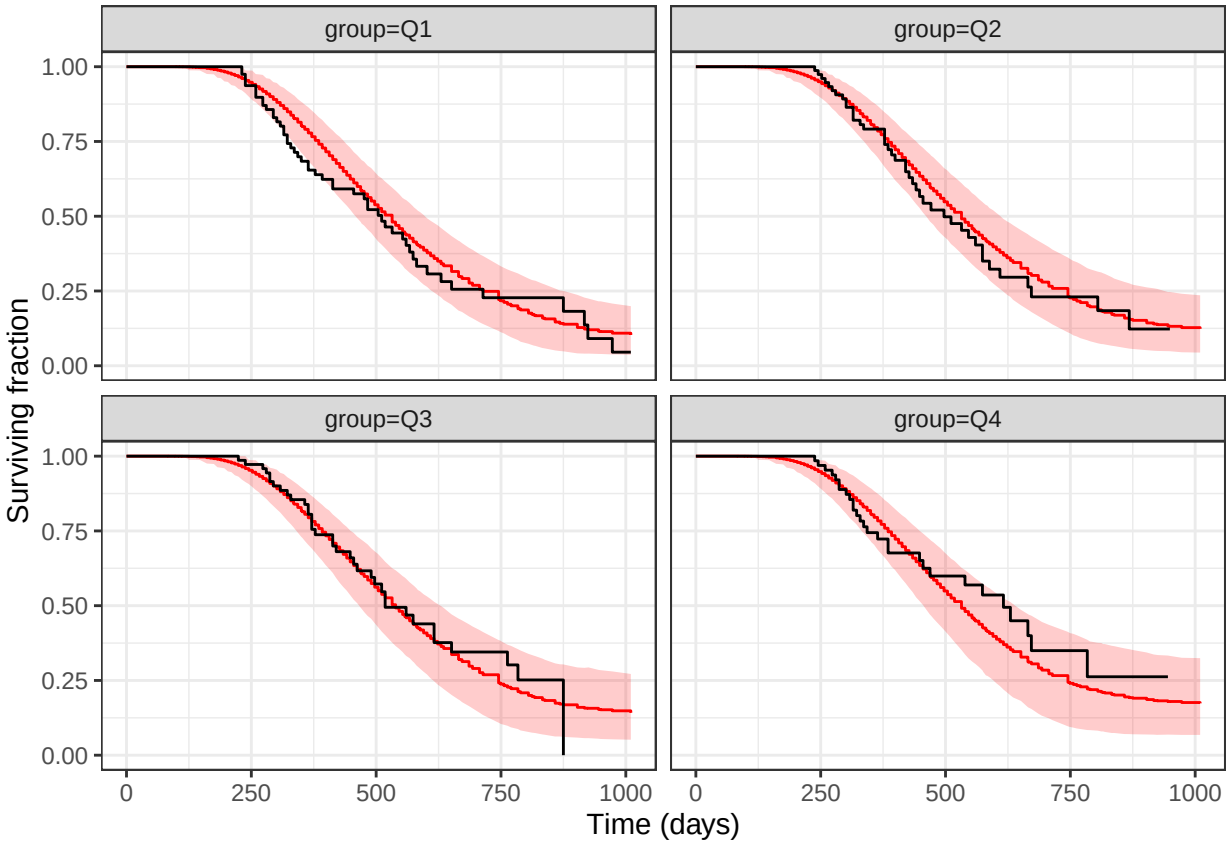


Figure 1: VPC for censoring model stratified by relative change in tumor size to Day 84.

- Fit a model or two using alternative parametric families (Gamma, lognormal, or Frechet). Include both relative tumor size at Day 84 and baseline ECOG status.

Answer: Let's look at the lognormal model as an alternative since it allows for a non-monotonic hazard function (unlike the Weibull model).

```
fit01_lognormal <-brm(TIME | cens(1 - DV) ~ rts84 + ECOG, data=dos84, family = lognormal())
```

```
. Running /opt/R/4.1.3/lib/R/bin/R CMD SHLIB foo.c
. gcc -I"/opt/R/4.1.3/lib/R/include" -DNDEBUG   -I"/data/page/binary_tte_in_stan/rlibs/Rcpp/include/"  -
. In file included from /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/Core:88:0,
.                  from /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/Dense:1,
.                  from /data/page/binary_tte_in_stan/rlibs/StanHeaders/include/stan/math/prim/mat/fun/
.                  from <command-line>:0:
. /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/src/Core/util/Macros.h:628:1: error: unkn
.  namespace Eigen {
.  ^~~~~~~~~
. /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/src/Core/util/Macros.h:628:17: error: exp
.  namespace Eigen {
.                  ^
. In file included from /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/Dense:1:0,
.                  from /data/page/binary_tte_in_stan/rlibs/StanHeaders/include/stan/math/prim/mat/fun/
.                  from <command-line>:0:
. /data/page/binary_tte_in_stan/rlibs/RcppEigen/include/Eigen/Core:96:10: fatal error: complex: No such
.  #include <complex>
.           ^~~~~~~~~
. compilation terminated.
. /opt/R/4.1.3/lib/R/etc/Makeconf:168: recipe for target 'foo.o' failed
. make: *** [foo.o] Error 1
.
. SAMPLING FOR MODEL '0fab9c793cbf5696fa0af5a6fb7ed1ba' NOW (CHAIN 1).
. Chain 1:
. Chain 1: Gradient evaluation took 0.00011 seconds
. Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 1.1 seconds.
. Chain 1: Adjust your expectations accordingly!
. Chain 1:
. Chain 1:
. Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
. Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
. Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
. Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
. Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
. Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
. Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
. Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
. Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
. Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
. Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
. Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
. Chain 1:
. Chain 1:  Elapsed Time: 0.324805 seconds (Warm-up)
. Chain 1:                0.364196 seconds (Sampling)
. Chain 1:                0.689001 seconds (Total)
. Chain 1:
```

```
.
. SAMPLING FOR MODEL '0fab9c793cbf5696fa0af5a6fb7ed1ba' NOW (CHAIN 2).
. Chain 2: Rejecting initial value:
. Chain 2:   Log probability evaluates to log(0), i.e. negative infinity.
. Chain 2:   Stan can't start sampling from this initial value.
. Chain 2: Rejecting initial value:
. Chain 2:   Log probability evaluates to log(0), i.e. negative infinity.
. Chain 2:   Stan can't start sampling from this initial value.
. Chain 2: Rejecting initial value:
. Chain 2:   Log probability evaluates to log(0), i.e. negative infinity.
. Chain 2:   Stan can't start sampling from this initial value.
. Chain 2:
. Chain 2: Gradient evaluation took 6.5e-05 seconds
. Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.65 seconds.
. Chain 2: Adjust your expectations accordingly!
. Chain 2:
. Chain 2:
. Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
. Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
. Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
. Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
. Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
. Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
. Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
. Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
. Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
. Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
. Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
. Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
. Chain 2:
. Chain 2:  Elapsed Time: 0.319496 seconds (Warm-up)
. Chain 2:                0.364446 seconds (Sampling)
. Chain 2:                0.683942 seconds (Total)
. Chain 2:
.
. SAMPLING FOR MODEL '0fab9c793cbf5696fa0af5a6fb7ed1ba' NOW (CHAIN 3).
. Chain 3:
. Chain 3: Gradient evaluation took 6.9e-05 seconds
. Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.69 seconds.
. Chain 3: Adjust your expectations accordingly!
. Chain 3:
. Chain 3:
. Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
. Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
. Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
. Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
. Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
. Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
. Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
. Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
. Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
. Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
. Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
. Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
```

```
. Chain 3:
. Chain 3:  Elapsed Time: 0.34191 seconds (Warm-up)
. Chain 3:                0.351388 seconds (Sampling)
. Chain 3:                0.693298 seconds (Total)
. Chain 3:
.
. SAMPLING FOR MODEL '0fab9c793cbf5696fa0af5a6fb7ed1ba' NOW (CHAIN 4).
. Chain 4:
. Chain 4: Gradient evaluation took 6.6e-05 seconds
. Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.66 seconds.
. Chain 4: Adjust your expectations accordingly!
. Chain 4:
. Chain 4:
. Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
. Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
. Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
. Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
. Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
. Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
. Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
. Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
. Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
. Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
. Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
. Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
. Chain 4:
. Chain 4:  Elapsed Time: 0.330681 seconds (Warm-up)
. Chain 4:                0.289004 seconds (Sampling)
. Chain 4:                0.619685 seconds (Total)
. Chain 4:
```

```
fit01_lognormal
```

```
.  Family: lognormal
.   Links: mu = identity; sigma = identity
. Formula: TIME | cens(1 - DV) ~ rts84 + ECOG
.    Data: dos84 (Number of observations: 336)
.   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
.          total post-warmup draws = 4000
.
. Population-Level Effects:
.           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
. Intercept     6.67      0.09     6.50     6.84 1.00     3834     2845
. rts84        -0.41      0.09    -0.58    -0.23 1.00     3997     3028
. ECOG         -0.38      0.08    -0.55    -0.22 1.00     3893     2771
.
. Family Specific Parameters:
.       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
. sigma     0.67      0.04     0.60     0.75 1.00     3831     2918
.
. Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
. and Tail_ESS are effective sample size measures, and Rhat is the potential
. scale reduction factor on split chains (at convergence, Rhat = 1).
```

The Rhat values show that the sampler seems to have converged and we have high ESS values.

Let's calculate the ELPD.

- Compare to the Weibull model using LOO. Which model do you think would be better at predicting OS for future patients? Does it appear that you need a non-linear effect of RTS?

We can look at the ELPD-loo values from each of our models as an approximate measure of out-of-sample predictive performance.

For the Weibull model we have:

```
loo_weibull
```

```
.
. Computed from 4000 by 336 log-likelihood matrix
.
.          Estimate    SE
. elpd_loo  -1307.8  58.7
. p_loo         3.7   0.4
. looic      2615.6 117.3
. ------
. Monte Carlo SE of elpd_loo is 0.0.
.
. All Pareto k estimates are good (k < 0.5).
. See help('pareto-k-diagnostic') for details.
```

And for the log normal model we have:

```
loo_lognormal <- loo(fit01_lognormal)

loo_lognormal
```

```
.
. Computed from 4000 by 336 log-likelihood matrix
.
.          Estimate    SE
. elpd_loo  -1317.2  59.2
. p_loo         4.3   0.7
. looic      2634.4 118.4
. ------
. Monte Carlo SE of elpd_loo is 0.0.
.
. All Pareto k estimates are good (k < 0.5).
. See help('pareto-k-diagnostic') for details.
```

Based on these the ELPD is higher for the log-normal model, indicating it might be a better fit. We can quantify this better with a pairwise comparison (which allows us to get a standard error of the difference):

```
loo_compare(loo_weibull, loo_lognormal)
```

```
.                 elpd_diff se_diff
. fit01_weibull    0.0       0.0
. fit01_lognormal -9.4       4.7
```

So, the difference in LOOIC is about twice the standard error, indicting that the log normal model is a bit better for out-of-sample predictions.
