Simulation Study

```
library(survival)
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loading required package: foreach
```

```
## Loaded glmnet 2.0-16
```

```
library(polspline)
library(knitr)
library(EnvStats)
```

```
##
## Attaching package: 'EnvStats'
```

```
## The following object is masked from 'package:Matrix':
##
##     print
```

```
## The following objects are masked from 'package:stats':
##
##     predict, predict.lm
```

```
## The following object is masked from 'package:base':
##
##     print.default
```

```
library(bda)
```

Simulating Survival Time with a Weibull Distribution

This function is based on simulate_data in https://cran.r-project.org/web/packages/rsimsum/vignettes/relhaz.html

```
#' Simulate survival times with censoring, based on a weibull baseline hazard
#'
#' This function simulates survival times with censoring, according to a weibull
#' baseline hazard that the user parameterizes. The survival/censoring times are
#' simulated for user-given covariates and coefficients.
#' @param x model matrix (including intercept) of x-values for the Cox model of survival times
#' @param fcts_select subset of fcts from a hare object containing the coefficients of interest.
#' @param params parameters shape and scale for the baseline Weibull distribution,
#' by default the exponential distribution with scale = 1
#' @param FUN random generation function for the distribution of censoring times,
#' expected to be uniform, exponential, or weibull.
#' @param ... arguments for FUN, the random generation function
#' @return dataframe appending survival time and censoring indicator to the model matrix x
#' @export
simulate_weibull <- function(x, fcts_select, params = list(shape = 1, scale = 1), FUN, ...) {

  n <- nrow(x)
```

```
  # extract unique list of covariates selected
  cov_nums <- sort(fcts_select[,1][fcts_select[,1] != 0])
  cov_names <- colnames(x)[cov_nums]
  x_select <- x[,cov_names]

  # extract the coefficient values from fcts_select
  betas <- fcts_select[,5][fcts_select[,1] != 0]

  # simulate survival times according to Bender et al. (2005)
  u <- runif(n)
  time <- (-log(u) / (params$scale * exp(x_select %*% betas)))^(1 / params$shape)

  # Winsorising tiny values for time (smaller than one day on a yearly-scale, e.g. 1 / 365.242),
  # and adding a tiny amount of white noise not to have too many concurrent values
  time <- ifelse(time < 1 / 365.242, 1 / 365.242, time)
  time[time == 1 / 365.242] <- time[time == 1 / 365.242] +
    rnorm(length(time[time == 1 / 365.242]), mean = 0, sd = 1e-4)
  # ...and make sure that the resulting value is positive
  time <- abs(time)

  # Censoring
  # to be implemented later
  cid <- sample(c(0, 1), size = n, replace = TRUE, prob = c(.75, .25))

  # return a dataframe
  data.frame(time, cid, x)
}
```

```
load("actg175.RData")

x <- model.matrix( ~ trt + age + wtkg + hemo + drugs +
                     karnof + oprior + preanti + race +
                     gender + symptom + offtrt + cd40 +
                     cd80, actg175)[,-1]

# x <- readRDS("actg175_mat.rds")

nphm_hare <- readRDS("nphm_hare.rds")

# extracting the coefficients for basis functions
# that do not correspond to knots and/or tensor products
fcts <- nphm_hare$fcts
fcts_select <- fcts[fcts[,2] == 0 & is.na(fcts[,3]),]
```

Using arbitrary Weibull parameter values to get similar survival times as original study

```
set.seed(2)

sim_mat <- simulate_weibull(x, fcts_select, params = list(shape = 500, scale = 1))

summary(sim_mat$time)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.002633 0.791474 1.255397 1.182642 1.513341 3.714213
```

Using Weibull parameter estimates from fit.Weibull and rhare

```
set.seed(1)

parm_res <- fit.Weibull(rhare(100000, cov = rep(0, nphm_hare$ncov), nphm_hare), dist="Weibull")
```

```
set.seed(2)

sim_mat2 <- simulate_weibull(x, fcts_select,
                             params = list(shape = parm_res$pars[2],
                                           scale = parm_res$pars[1]))

summary(sim_mat2$time)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
##  0.002633  0.640442  1.542383  1.538171  2.201997 12.182677
```

Coxph Simulation

```
phm_sim_mat <- coxph(Surv(time, cid) ~ ., data = sim_mat)

phm_sim_mat
```

```
## Call:
## coxph(formula = Surv(time, cid) ~ ., data = sim_mat)
##
##                 coef exp(coef)  se(coef)       z        p
## trtZDV.ddi -0.171795  0.842152  0.133199  -1.290 0.197134
## trtZDV.ZAL -0.525136  0.591475  0.131060  -4.007 6.15e-05
## trtddi     -0.077887  0.925069  0.126435  -0.616 0.537880
## age        -0.007221  0.992804  0.005819  -1.241 0.214581
## wtkg        0.004427  1.004437  0.003599   1.230 0.218567
## hemo1      -0.001595  0.998406  0.178046  -0.009 0.992853
## drugs1      0.075128  1.078022  0.137217   0.548 0.584028
## karnof     -0.011022  0.989039  0.007939  -1.388 0.165048
## oprior1     0.010359  1.010412  0.298440   0.035 0.972311
## preanti     0.212841  1.237187  0.008611  24.717  < 2e-16
## race1      -0.090127  0.913815  0.108848  -0.828 0.407666
## gender1     0.014553  1.014659  0.132836   0.110 0.912764
## symptom1   -0.483437  0.616660  0.129335  -3.738 0.000186
## offtrt1    -0.323289  0.723765  0.098733  -3.274 0.001059
## cd40       -0.265542  0.766790  0.010744 -24.716  < 2e-16
## cd80       -0.008863  0.991176  0.000376 -23.574  < 2e-16
##
## Likelihood ratio test=4874  on 16 df, p=< 2.2e-16
## n= 2139, number of events= 544
```

```
phm_sim_mat2 <- coxph(Surv(time, cid) ~ ., data = sim_mat2)

phm_sim_mat2
```

```
## Call:
## coxph(formula = Surv(time, cid) ~ ., data = sim_mat2)
##
##                  coef exp(coef)  se(coef)       z        p
## trtZDV.ddi -0.171795  0.842152  0.133199  -1.290 0.197134
## trtZDV.ZAL -0.525136  0.591475  0.131060  -4.007 6.15e-05
## trtddi     -0.077887  0.925069  0.126435  -0.616 0.537880
## age        -0.007221  0.992804  0.005819  -1.241 0.214581
## wtkg        0.004427  1.004437  0.003599   1.230 0.218567
## hemo1      -0.001595  0.998406  0.178046  -0.009 0.992853
## drugs1      0.075128  1.078022  0.137217   0.548 0.584028
## karnof     -0.011022  0.989039  0.007939  -1.388 0.165048
## oprior1     0.010359  1.010412  0.298440   0.035 0.972311
## preanti     0.212841  1.237187  0.008611  24.717  < 2e-16
## race1      -0.090127  0.913815  0.108848  -0.828 0.407666
## gender1     0.014553  1.014659  0.132836   0.110 0.912764
## symptom1   -0.483437  0.616660  0.129335  -3.738 0.000186
## offtrt1    -0.323289  0.723765  0.098733  -3.274 0.001059
## cd40       -0.265542  0.766790  0.010744 -24.716  < 2e-16
## cd80       -0.008863  0.991176  0.000376 -23.574  < 2e-16
##
## Likelihood ratio test=4874  on 16 df, p=< 2.2e-16
## n= 2139, number of events= 544
```

Whether I use the arbitrary parameters shape = 500, scale = 1, or the estimated parameters shape = 262.4285, scale = 1.008144, the Cox proportional hazards model returns the same results. As long as the shape parameter is high, the Cox proportional hazards model is robust to small perturbations in the parameter values.

Glmnet Simulation

```
cv_phmnet <- cv.glmnet(as.matrix(sim_mat2[-c(1,2)]),
                       Surv(sim_mat2$time, sim_mat2$cid),
                       family = "cox", alpha = .95)
```

```
## Warning: from glmnet Fortran code (error code -30057); Numerical error at
## 57th lambda value; solutions for larger values of lambda returned

## Warning: from glmnet Fortran code (error code -30057); Numerical error at
## 57th lambda value; solutions for larger values of lambda returned

## Warning: from glmnet Fortran code (error code -30064); Numerical error at
## 64th lambda value; solutions for larger values of lambda returned

## Warning: from glmnet Fortran code (error code -30047); Numerical error at
## 47th lambda value; solutions for larger values of lambda returned

## Warning: from glmnet Fortran code (error code -30057); Numerical error at
## 57th lambda value; solutions for larger values of lambda returned
```

```
## Warning: from glmnet Fortran code (error code -30057); Numerical error at
## 57th lambda value; solutions for larger values of lambda returned

## Warning: from glmnet Fortran code (error code -30057); Numerical error at
## 57th lambda value; solutions for larger values of lambda returned

## Warning: from glmnet Fortran code (error code -30056); Numerical error at
## 56th lambda value; solutions for larger values of lambda returned

## Warning: from glmnet Fortran code (error code -30057); Numerical error at
## 57th lambda value; solutions for larger values of lambda returned

## Warning: from glmnet Fortran code (error code -30056); Numerical error at
## 56th lambda value; solutions for larger values of lambda returned

## Warning: from glmnet Fortran code (error code -30057); Numerical error at
## 57th lambda value; solutions for larger values of lambda returned
```

```
coefficients(cv_phmnet)
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## trtZDV.ddi -0.090937080
## trtZDV.ZAL -0.126691688
## trtddi      .
## age        -0.005855786
## wtkg        0.003316663
## hemo1       0.009609051
## drugs1      0.013113569
## karnof     -0.008462179
## oprior1     0.127126752
## preanti     0.058753345
## race1      -0.044395996
## gender1     .
## symptom1   -0.193849985
## offtrt1    -0.051636873
## cd40       -0.073785538
## cd80       -0.002342212
```