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
  # cid <- sample(c(0, 1), size = n, replace = TRUE, prob = c(.75, .25))
  cid_time <- FUN(n, ...)

  cid <- ifelse(time <= cid_time, 1, 0)

  time <- pmin(time, cid_time)

  # 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),
                            FUN = rexp, 1.3)

summary(sim_mat$time)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000123 0.2066407 0.4692176 0.5644418 0.8135539 2.0275567
```

```
mean(sim_mat$cid)
```

```
## [1] 0.2431043
```

Using Weibull parameter estimates from fit.Weibull

```
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]),
                             FUN = rexp, 1.3)

summary(sim_mat2$time)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0000123 0.1901507 0.4084255 0.5705628 0.7844852 3.0695179
```

```
mean(sim_mat2$cid)
```

```
## [1] 0.2435718
```

Coxph Simulation

Using arbitrary Weibull parameter values

```
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 -0.0315080  0.9689832  0.1519655  -0.207 0.8357
```

```
## trtZDV.ZAL -0.3707338  0.6902277  0.1559796  -2.377 0.0175
## trtddi     -0.0275478  0.9728282  0.1535824  -0.179 0.8576
## age         0.0036990  1.0037058  0.0071670   0.516 0.6058
## wtkg        0.0003467  1.0003468  0.0049080   0.071 0.9437
## hemo1       0.0405321  1.0413648  0.1896735   0.214 0.8308
## drugs1      0.0377496  1.0384712  0.1773127   0.213 0.8314
## karnof     -0.0007312  0.9992691  0.0090552  -0.081 0.9356
## oprior1    -0.0030029  0.9970016  0.3480679  -0.009 0.9931
## preanti     0.1620180  1.1758814  0.0075108  21.571 <2e-16
## race1       0.0269866  1.0273540  0.1287316   0.210 0.8340
## gender1    -0.0460705  0.9549747  0.1598970  -0.288 0.7733
## symptom1   -0.3391385  0.7123838  0.1374381  -2.468 0.0136
## offtrt1    -0.2270705  0.7968646  0.1189267  -1.909 0.0562
## cd40       -0.2027929  0.8164473  0.0093967 -21.581 <2e-16
## cd80       -0.0067717  0.9932511  0.0003382 -20.025 <2e-16
##
## Likelihood ratio test=4453  on 16 df, p=< 2.2e-16
## n= 2139, number of events= 520
```

Using Weibull parameter estimates from fit.Weibull

```
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 -0.0125363  0.9875420  0.1502395  -0.083 0.93350
## trtZDV.ZAL -0.3480515  0.7060625  0.1579983  -2.203 0.02760
## trtddi     -0.0259318  0.9744016  0.1538496  -0.169 0.86615
## age         0.0069482  1.0069724  0.0068392   1.016 0.30966
## wtkg       -0.0015353  0.9984659  0.0048394  -0.317 0.75106
## hemo1       0.0087963  1.0088351  0.1808414   0.049 0.96121
## drugs1     -0.0237253  0.9765540  0.1825600  -0.130 0.89660
## karnof      0.0045485  1.0045589  0.0090543   0.502 0.61542
## oprior1     0.0555272  1.0570978  0.2967025   0.187 0.85154
## preanti     0.1623560  1.1762790  0.0074766  21.715 < 2e-16
## race1      -0.0682251  0.9340502  0.1323887  -0.515 0.60632
## gender1     0.0445937  1.0456030  0.1648220   0.271 0.78673
## symptom1   -0.3890929  0.6776713  0.1394388  -2.790 0.00526
## offtrt1    -0.2722028  0.7616998  0.1202460  -2.264 0.02359
## cd40       -0.2031480  0.8161575  0.0093562 -21.713 < 2e-16
## cd80       -0.0068301  0.9931931  0.0003365 -20.295 < 2e-16
##
## Likelihood ratio test=5013  on 16 df, p=< 2.2e-16
## n= 2139, number of events= 521
```

Weibull model (to compare with Coxph)

The Weibull model should be more powerful (with less variance in the coefficient estimates) than the Cox Proportional Hazards model, as the data is from a Weibull distribution. The coefficient estimates themselves should be similar.

```
survreg(Surv(time, cid) ~ ., data = sim_mat2)
```

```
## Call:
## survreg(formula = Surv(time, cid) ~ ., data = sim_mat2)
##
## Coefficients:
##   (Intercept)    trtZDV.ddi    trtZDV.ZAL        trtddi           age
##  2.049551e-03 -6.594153e-04  3.435753e-03 -1.092460e-04  8.621281e-06
##          wtkg         hemo1        drugs1        karnof       oprior1
## -4.074435e-05 -1.452632e-03  5.923479e-04  1.908744e-05 -2.860494e-04
##       preanti         race1       gender1      symptom1       offtrt1
## -1.652848e-03  2.134880e-04  3.011987e-04  2.991213e-03  1.793223e-03
##          cd40          cd80
##  2.059898e-03  6.978350e-05
##
## Scale= 0.034682
##
## Loglik(model)= 1033.9   Loglik(intercept only)= -908.8
##  Chisq= 3885.39 on 16 degrees of freedom, p= <2e-16
## n= 2139
```

For coefficients with very low p-values ($p < 2e\text{-}16$) according to the coxph model, the values from the Weibull model are approximately equal to those from the coxph model divided by 100.

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 -74); Convergence for 74th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned
```

```
## Warning: from glmnet Fortran code (error code -76); Convergence for 76th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned
```

```
## Warning: from glmnet Fortran code (error code -76); Convergence for 76th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned
```

```
## Warning: from glmnet Fortran code (error code -76); Convergence for 76th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned
```

```
## Warning: from glmnet Fortran code (error code -76); Convergence for 76th
```

```
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned

## Warning: from glmnet Fortran code (error code -69); Convergence for 69th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned

## Warning: from glmnet Fortran code (error code -72); Convergence for 72th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned

## Warning: from glmnet Fortran code (error code -76); Convergence for 76th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned


## Warning: from glmnet Fortran code (error code -76); Convergence for 76th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned

## Warning: from glmnet Fortran code (error code -73); Convergence for 73th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned

## Warning: from glmnet Fortran code (error code -77); Convergence for 77th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned
```

```
coefficients(cv_phmnet)
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## trtZDV.ddi  0.0051094207
## trtZDV.ZAL -0.0632111356
## trtddi      .
## age         .
## wtkg        .
## hemo1       .
## drugs1     -0.0007287251
## karnof      0.0009972092
## oprior1     0.0013966838
## preanti     0.0473722429
## race1       .
## gender1     .
## symptom1   -0.1163350309
## offtrt1    -0.0559478504
## cd40       -0.0592515887
## cd80       -0.0019513857
```