Code and documentation for the simulation setup described in “Plasmode simulation for the evaluation of pharmacoepidemiologic methods in complex healthcare databases”

Software: Jessica M Franklin

```
hdSimSetup <- function(x, idVar, outcomeVar, timeVar, treatVar,
                       form, effectRR = 1, MM = 1, nsim = 500,
                       size = nrow(x), eventRate = 0.05) {
# x = datset on which sims are based
# idVar = name of id variable
# outcomeVar = name of outcome variable
# timeVar = name of the follow-up time variable
# treatVar = name of treatment variable
# form = RHS of formula used for outcome simulation – should look like
# “~ C1 + C2 + …”.  Can include anything allowed by coxph.
# effectRR = the desired treatment effect relative risk
# MM = multiplier of confounder effects on outcome on
#  the log-scale
# nsim = number of desired outcome vectors
# size = desired size of simulated cohort studies (i.e., # of individuals)
# eventRate = desired average event rate -- default is the event
#  rate observed in the base dataset

      n <- nrow(x)

      sidx <- sapply(c(idVar, outcomeVar, timeVar, treatVar),
                       function(v) which(names(x) == v))
      names(x)[sidx] <- c("ID", "OUTCOME", "TIME", "TREAT")
      y1 <- Surv(x$TIME, x$OUTCOME)
      y2 <- Surv(x$TIME, !x$OUTCOME)
      form1 <- as.formula(paste("y1 ~", form))
      form2 <- as.formula(paste("y2 ~", form))

# estimate survival and censoring models
      smod <- coxph(form1, x = TRUE, data = x)
      fit <- survfit(smod)
      s0 <- fit$surv      # survival curve for average patient
      ts <- fit$time
      nts <- length(ts)
      cmod <- coxph(form2, data = x)
      fit <- survfit(cmod)
      c0 <- fit$surv      # censoring curve for average patient

# find event rate in base cohort (if everyone was followed to end of study)
      Xb <- as.vector(smod$x %*% coef(smod))
      mx <- colMeans(smod$x)
      xb0 <- mx %*% coef(smod)
      s0end <- min(s0)
      if(is.null(eventRate)) eventRate <- 1-mean(s0end^exp(Xb - xb0))

# find delta value needed to get approximate desired event rate under new
# parameters
      bnew <- replace(MM*coef(smod), names(coef(smod)) == "TREAT", log(effectRR))
      Xbnew <- as.vector(smod$x %*% bnew)
      sXend <- s0end^(exp(Xb - xb0))
      fn <- function(d) mean(sXend^d) - (1 - eventRate)
      delta <- uniroot(fn, lower = 0, upper = 20)$root

  # setup n X nts matrix of individual survival and censoring curves under new
  # parameters
      Sx <- matrix(unlist(lapply(s0, function(s) s^(delta*exp(Xbnew - xb0)))),
                   nrow = n)
      Xbnew <- as.vector(smod$x %*% coef(cmod))
      xb0 <- mx %*% coef(cmod)
      Cx <- matrix(unlist(lapply(c0, function(s) s^(delta*exp(Xbnew - xb0)))),
                   nrow = n)

  #### sample and simulate
      ids <- tnew <- ynew <- data.frame(matrix(nrow = size, ncol = nsim))
      for(sim in 1:nsim) {
            idxs <- sample(n, size, replace = TRUE)
            ids[,sim] <- x$ID[idxs]

      # event time
            u <- runif(size, 0, 1)
      # the first time survival drops below u
            w <- apply(Sx[idxs,] < u, 1, function(x) which(x)[1])
            stime <- ts[w]
      # for any individuals with survival that never drops below u,
      # replace with arbitrary time beyond last observed event/censoring time
            w <- Sx[idxs,nts] > u
            stime[w] <- max(ts) + 1

      # censoring time
            u <- runif(size, 0, 1)
      # the first time censor-free survival drops below u
            w <- apply(Cx[idxs,] < u, 1, function(x) which(x)[1])
            ctime <- ts[w]
      # for any individuals with censor-free survival that never drops below u,
      # replace with hard censor time at last observed event/censoring time
            w <- Cx[idxs,nts] > u
            ctime[w] <- max(ts)

      # put it together
            tnew[,sim] <- pmin(stime, ctime)
            names(tnew) <- paste("TIME", 1:nsim, sep = "")
            ynew[,sim] <- stime == tnew[,sim]
            names(ynew) <- paste("EVENT", 1:nsim, sep = "")
      }

      names(ids) <- paste("ID", 1:nsim, sep = "")
      names(tnew) <- paste("TIME", 1:nsim, sep = "")
      names(ynew) <- paste("EVENT", 1:nsim, sep = "")

      data.frame(ids, ynew, tnew)
}
```