Assessing non-inferiority in treatment trials regarding severe infectious diseases: An extension to the entire follow-up period using a cure-death multistate model Supplementary Material

Harriet Sommer* (corresponding author)
Institute for Medical Biometry and Statistics,
Faculty of Medicine and Medical Center – University of Freiburg, Germany
Stefan-Meier-Str. 26
79104 Freiburg, Germany
PHONE: +49 (0)761 203 7703
E-MAIL: sommer@imbi.uni-freiburg.de

Tobias Bluhmki* (alternate author)
Institute of Statistics, Ulm University, Germany
Helmholtzstr. 20
89081 Ulm, Germany
PHONE: +49 (0)731 50 33104
E-MAIL: tobias.bluhmki@uni-ulm.de

Jan Beyersmann
Institute of Statistics, Ulm University, Germany
Helmholtzstr. 20
89081 Ulm, Germany
E-MAIL: jan.beyersmann@uni-ulm.de

Martin Schumacher
Institute for Medical Biometry and Statistics,
Faculty of Medicine and Medical Center – University of Freiburg, Germany
Stefan-Meier-Str. 26
79104 Freiburg, Germany
E-MAIL: ms@imbi.uni-freiburg.de

*These authors contributed equally to this manuscript.

Notation

The framework illustrated in Figure 1 is a cure-death multistate model with the initial state 0, the "cure + alive" state 1, and the absorbing "death" state 2. We assume that all individuals enter the model into state 0. The estimated quantity of interest is

$$\hat{P}_{01}(0,t) = \sum_{s \leq t} \hat{P}_{00}(0,s) \cdot \frac{\#\text{ observed } 0 \to 1 \text{ transitions at time } s}{\#\text{ observed patients in state 0 just before time } s} \cdot \hat{P}_{11}(s,t), \quad \text{(A.1)}$$

the Aalen-Johansen estimator [1, 2] of the probability to be cured and alive at time t, where $t \leq \tau$ and τ is a suitably chosen timepoint of interest, for instance, the largest observed cure or death time. Equation (A.1) is a finite sum over all observed transition times s lesser or equal to t including

$$\hat{P}_{00}(0,s) = \prod_{v<s} \left(1 - \frac{\#\text{ observed transitions out of state 0 at time } v}{\#\text{ observed patients in state 0 just before time } v}\right)$$

and

$$\hat{P}_{11}(s,t) := \prod_{s<u\leq t} \left(1 - \frac{\#\text{ observed transitions out of state 1 at time } u}{\#\text{ observed patients in state 1 just before time } u}\right),$$

which are the Kaplan-Meier-type estimators of being uncured and alive prior to time s and, respectively, being cured and alive at time t given to be cured at time s. Note that $\hat{P}_{00}(0,s)$ is only a product over all v less than s. The interpretation of (A.1) is that we obtain the probability of being cured and alive at time t as the sum over all $0 \to 1$ transition times s up to time t with respect to the estimated probability of staying in the initial state 0 just prior to s times the instantaneous "risk" of being cured exactly at time s times the probability of subsequently staying cured and alive up to time t.

Determination of q: Wild bootstrap

The aim is to construct a time-simultaneous one-sided 95% confidence band with lower boundary introduced in relation (1), which covers the true difference in the treatment-specific probabilities

of being cured and alive, i.e., $P_{01}(t|A) - P_{01}(t|B)$, with a probability of 95% over the entire time interval of interest $[0, \tau]$. Since time-simultaneous statistical inference is mathematically involved here [1, 2], we extend a simulation procedure known as "wild bootstrap resampling" – originally proposed for cumulative incidence functions in a competing risk setting [3, 4] – to our multistate model at hand.

Employing the general representation given in equation (3.83) in [1] to the cure-death model at hand, we consider, given the data, the quantity

$$\begin{aligned}\zeta(t) = \sum_{s \leq t} \hat{P}_{00}(0,s) G_{01}(s) \left(\hat{P}_{11}(s,t) - \hat{P}_{01}(s,t) \right) \\ - \hat{P}_{00}(0,s) G_{02}(s) \hat{P}_{01}(s,t) - \hat{P}_{01}(0,s) G_{12}(s) \hat{P}_{11}(s,t), \end{aligned} \tag{A.2}$$

where $G_{01}(s),\ G_{02}(s),\ G_{12}(s)$ are independently computer-generated quantities following a normal distribution with expectation zero and variance

$$\sigma^2_{lm}(s) = \frac{\#\text{ observed } l \to m \text{ transitions at time } s}{(\#\text{ observed patients in state } l \text{ just before time } s)^2},$$

$l \in \{0,1\},\ m \in \{0,1\},\ l < m$. When no $l \to m$ transition is observed at time s, $G_{lm}(s)$ equals zero. The intuition behind the wild bootstrap is that relation (A.2) is a computer experiment mimicking the uncertainty of how the Aalen-Johansen estimator varies around the true quantity. This is used to construct the confidence band.

Adapting arguments applied in [3, 4] to our non-inferiority testing problem, the procedure can be realised in `R` [5] as follows:

1. Generate $G_{lm}(s)$ for each observed transition time s within the data using the function `rnorm()`.

2. Separately compute $\zeta(t|A)$ and $\zeta(t|B)$ in (A.2) for the treatment groups A and B applying the function `etm` [6].

3. Compute

$$\hat{D} = \max_{t\in[0,\tau]} \left(\zeta(t|A) - \zeta(t|B)\right). \tag{A.3}$$

4. Repeat steps 1-3 many times, say 1000 times. The critical value q is the empirical 95% quantile of the 1000 $\hat{D}$'s.

The quantile q is used to construct the desired confidence band as detailed below. An `R`-script exemplary applying the procedure to a freely accessible dataset in the context of infection control is provided in an additional supplementary file. A data overview can be found in [2].

Further details

The procedure determines q via resampling such that

$$\begin{aligned}
0.95 &= P\left(\max_{t\in[0,\tau]} \left(\zeta(t|A) - \zeta(t|B)\right) \leq q\right) \\
&\approx P\left(\left(\hat{P}_{01}(t|A) - P_{01}(t|A)\right) - \left(\hat{P}_{01}(t|B) - P_{01}(t|B)\right) \leq q \quad \text{for all } t \in [0,\tau]\right) \\
&= P\left(\hat{P}_{01}(t|A) - \hat{P}_{01}(t|B) - q \leq P_{01}(t|A) - P_{01}(t|B) \quad \text{for all } t \in [0,\tau]\right) \\
&= P\left(PCAD(t) - q \leq P_{01}(t|A) - P_{01}(t|B) \quad \text{for all } t \in [0,\tau]\right)
\end{aligned}$$

guaranteeing that the true difference in the treatment-specific probabilities of being cured and alive lies above the lower margin $PCAD(t) - q$ of the derived one-sided time-simultaneous confidence band for all $t \in [0,\tau]$ with a probability of approximately 95%.

References

[1] Aalen O, Borgan Ø, Gjessing H. 2008. Survival and Event history analysis: A process point of view. New York: Springer Science & Business Media.

[2] Beyersmann J, Allignol A, Schumacher M. 2011. Competing risks and multistate models with R. New York: Springer Science & Business Media.

[3] Lin D. 1997. Non-parametric inference for cumulative incidence functions in competing risks studies. Stat Med 16:901–910.

[4] Beyersmann J, Di Termini S, Pauly M. 2013. Weak convergence of the wild bootstrap for the Aalen-Johansen estimator of the cumulative incidence function of a competing risk. Scand J Stat 40:387–402.

[5] R Core Team. 2016. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing Vienna, Austria.

[6] Allignol A, Schumacher M, Beyersmann J. 2011. Empirical transition matrix of multi-state models: The etm package. J Stat Soft 38:1–15.

```
##############################################
##############################################
##############################################
##############################################
## One-sided linear confidence bands
## using wild bootstrap resampling
##############################################
##############################################
##############################################
##############################################

rm(list=ls())

## set random seed
set.seed(168238469)

## Load packages
library(etm)
library(kmi)


## Load functions
source("function_linearCB_diff_onesided.R")
source("function_det_xi.R")

##############################################
##############################################
## Data icu.pneu out or R-package kmi
##############################################
##############################################

## The data set is a random sample drawn from the
## SIR-3 study that aimed at analysing the effect of
## nosocomial infections on the length of ICU stay.
## Patients were included in the study if they had
## stayed at least 1 day in the unit. The sample includes
## information to assess the effect of nosocomial
## pneumonia on the length of stay. The endpoint is
## either discharge alive from the ICU or dead in the
## unit. These data are censoring complete as the
## censoring time is known for all patients.

## Load the data.
data(icu.pneu)

## A data frame with 1421 observations on the following 8
variables:
## id
## Individual patient id.
## start
## Start of the observation time.
## stop
## Failure time.
```

```
## status
## Censoring status. 0 if the observation is censored, 1
otherwise.
## event
## Event type. 2 is death in ICU, 3 is discharge alive
## pneu
## Nosocomial pneumonia indicator.
## adm.cens.exit
## Exit times for patients discharged alive are replaced by their
## administrative censoring times.
## age
## Age at inclusion
## sex
## Sex. F for female and M for male.

## Then, we have to transform the icu.pneu data
## into a format which allows us to use etm package:
data <- icu.pneu

data <- data[order(data$id, data$start), ]
masque <- diff(data$id)

data$from <- 0
data$from[c(1, masque) == 0] <- 1

data$to2 <- data$event
data$to2[data$status == 0] <- "cens"
data$to2[c(masque, 1) == 0] <- 1

## Event indicator variable to for a first event analysis:
data$to <- ifelse(data$to2 %in% c(2, 3), 2,data$to2)

## Keep the variables of interest
data <- data[, c("id",
                 "start","stop",
                 "from","to","to2",
                 "age","sex")]
names(data)[c(2, 3)] <- c("entry","exit")

## Overview of transitions
table(data$from,data$to)

###########################################
###########################################
## Set values of interest
###########################################
###########################################

## Matrix of possible transitions
tra <- matrix(FALSE, ncol=3, nrow=3)
dimnames(tra) <- list(c("0", "1", "2"),
                      c("0", "1", "2"))
tra[1, 2:3] <- TRUE
```

```
tra[2, 3] <- TRUE

## Initialize state names
state.names <- c("0","1","2")

## Set no of bootstrap iterations
noit <- 1000

## Specify groups
data.F <- subset(data,sex=="F" )
data.M <- subset(data,sex=="M")

## Set nominal level
alpha <- 0.05


## Set quantities of interest in relation (A.3)
l <- "0"
m <- "1"


## Set relevant timeframe [t.1,tau]
t.1 <- 0
tau <- 30

## Set number of cores for bootstrap computation
ncores <- 3

##############################################
##############################################
## Derive zeta in (A.6)
##############################################
##############################################

zeta.F.01 <- det.xi(data=data.F,noit,l,m,ncores=ncores)

zeta.M.01 <- det.xi(data=data.M,noit,l,m,ncores=ncores)

##############################################
##############################################
## Linear confidence band for difference
##############################################
##############################################

CB.lin.diff.01 <- linearCB.diff.onesided(data1=data.F,
                                         data2=data.M,
                                         zeta.1=zeta.F.01,
                                         zeta.2=zeta.M.01,
                                         alpha=alpha,
                                         from=l,
                                         to=m,
                                         noit=noit,
                                         tau=tau,
```

```
                                        state.names=state.names,
                                        tra=tra)

##############################################
##############################################
## Plot
##############################################
##############################################

par(mfrow=c(1,2))


etm.idm.F <- etm(data.F, c("0", "1", "2"), tra, "cens", s=0)
etm.idm.M <- etm(data.M, c("0", "1", "2"), tra, "cens", s=0)

plot(etm.idm.F, tr.choice = c("0 1"),
     lty=1, lwd=2,
     xlim=c(0,30), ylim=c(0,0.1),
     ylab="Probability", xlab="Time (days)",
     col="red", legend=FALSE)
lines(etm.idm.M, tr.choice = c("0 1"),
      lty=1, lwd=2, col="blue")
legend('topleft',
       c("Group F", "Group M"),
       col=c("red", "blue"),
       lty=c(1,1), lwd=c(2,2), bty='n')

plot(CB.lin.diff.01$times,CB.lin.diff.01$est,
     type="s", lty=1, lwd=2,
     xlim=c(0,30), ylim=c(-0.05,0.05),
     ylab="Difference", xlab="Time (days)")
abline(h = 0, lty=1)
lines(CB.lin.diff.01$times,CB.lin.diff.01$CI.lower,
      type="s", lty=2)
legend('topleft',
       c("Difference (F-M)", "95% (linear) CB"),
       col=c("black", "black"),
       lty=c(1,2), lwd=c(2,1), bty='n')
```

```
det.xi<-function(data,
                 noit,
                 from,
                 to,
                 ncores=1){

  ## Load required packages
  library(parallel)
  library(etm)

  ## Determine number of individuals
  n<-length(unique(data$id))

  ## Compute P(0,t) for all event times in (A.6)
  P.s.all<-etm(data, state.names, tra, "cens",
s=0,covariance=F,delta.na=F)
  eval.times<-P.s.all$time[P.s.all$time>0]

  ## Compute P(s,*) for all u's and all event times in (a.6)
  P.u.v.all<-mclapply(eval.times,FUN=function(v){
    ## Determine all times u<=v for summation/integration
    int.times<-eval.times[eval.times<=v]
    ## Determine for all u P(u,v)
    P.u.v.it<-lapply(int.times,FUN=function(u){
      ##  If u==v then identity matrix
      if (u==v){
        P.u.v<-diag(1,length(state.names))
        rownames(P.u.v)<-state.names
        colnames(P.u.v)<-state.names
        ## Else etm application
      }else{
        P.u.v.etm<- etm(data, state.names, tra, "cens",
s=u,t=v,covariance=F)$est
        ## Extract last estimated value
        P.u.v<-P.u.v.etm[,,dim(P.u.v.etm)[3]]
      }
      return(P.u.v)
    })
    names(P.u.v.it)<-as.character(int.times)
    return(P.u.v.it)
  },mc.cores=ncores)
  names(P.u.v.all)<-as.character(eval.times)

  ## Compute zeta for each bootstrap iteration in (A.2)
  zeta<-mclapply(1:noit,FUN=function(i){

    dpsi.it<-lapply(1:dim(P.s.all$trans)[1],FUN=function(j){

return((ifelse(P.s.all$n.risk[,as.character(P.s.all$trans[j,1])]==
0,0,
        sapply(P.s.all$n.event[as.character(P.s.all$trans[j,1]),
                               as.character(P.s.all$trans[j,2]),],
               FUN=function(l){
```

```
return(sum(rnorm(1,0,1)))})/P.s.all$n.risk[,as.character(P.s.all$t
rans[j,1])]))*sqrt(n))
        })
    names(dpsi.it)<-
paste(P.s.all$trans$from,P.s.all$trans$to,sep=" ")

    dpsi.u.all<-lapply(eval.times,FUN=function(u){
      dpsi.u<-
matrix(0,nrow=length(state.names),length(state.names))
      colnames(dpsi.u)<-state.names
      rownames(dpsi.u)<-state.names

      for(row in 1:dim(P.s.all$trans)[1]){

dpsi.u[as.character(P.s.all$trans$from[row]),as.character(P.s.all$
trans$to[row])]<-
dpsi.it[[paste(as.character(P.s.all$trans$from[row]),as.character(
P.s.all$trans$to[row]),sep=" ")]][[which(P.s.all$time==u)]]
      }
      ## Diagonal elements such that sum of each row equals zero
      diag(dpsi.u)<-(-rowSums(dpsi.u))
      return(dpsi.u)
    })
    names(dpsi.u.all)<-as.character(eval.times)

    ## For each event time v...
    zeta.it<-sapply(eval.times,FUN=function(v){
      int.times<-eval.times[eval.times<=v]
      ## Summation over all times <=v
      zeta.v<-sum(sapply(int.times,FUN=function(u){
        ## Matrix product
        summand<-P.s.all$est[,,as.character(u)]%*
%dpsi.u.all[[as.character(u)]]%*%P.u.v.all[[as.character(v)]]
[[as.character(u)]]
        return(summand[from,to])
      }))
      return(zeta.v)
    })
    return(zeta.it)
    },mc.cores=ncores)

  out<-matrix(0,nrow=noit,ncol=length(eval.times))
  rownames(out)<-as.character(1:noit)
  colnames(out)<-as.character(eval.times)
  for (i in 1:noit){
    out[i,]<-zeta[[i]]
  }
  return(out)
}
```

```
linearCB.diff.onesided<-function(data1,
                                 data2,
                                 zeta.1,
                                 zeta.2,
                                 alpha,
                                 from,
                                 to,
                                 noit,
                                 tau,
                                 state.names,
                                 tra){

  ## Determine number of individuals
  n1<-length(unique(data1$id))
  n2<-length(unique(data2$id))

  ## Compute P(s,u) for all event times
  P.s.all.1<-etm(data1, state.names, tra, "cens", s=0)
  est1<-P.s.all.1$est[from,to,]

  ## Compute P(s,u) for all event times
  P.s.all.2<-etm(data2, state.names, tra, "cens", s=0)
  est2<-P.s.all.2$est[from,to,]

  ## Determine possible transitions
  poss.trans<-data.frame(P.s.all.1$trans)
  poss.trans$transition<-paste(poss.trans$from,poss.trans$to,sep="
")

  ## Determine times within the time window
  eval.times.1<-P.s.all.1$time[P.s.all.1$time>0]
  eval.times.2<-P.s.all.2$time[P.s.all.2$time>0]

  ## Determine (sorted) unique timepoints of both groups
  combine.times<-sort(unique(c(eval.times.1,eval.times.2)))

  ## Derive times between 0 and tau
  ana.times<-combine.times[combine.times<=tau]
  ## Include previous event time if not chosen
  ## (it might happen s.th. before)!
  if(ana.times[1]>0){
    ana.times<-c(0,ana.times)
  }

  if(tail(ana.times,1)<tau){
    ana.times<-c(ana.times,tau)
  }

  ## Select bootstrap processes with lm transition
  B.1.predict<-lapply(1:noit,function(j)
{sapply(findInterval(ana.times,eval.times.1),
                                                FUN=function(k){
    if(k==0){return(0)}
```

```
    else(return(zeta.1[j,k]))
  })/sqrt(n1)})
  B.2.predict<-lapply(1:noit,function(j)
{sapply(findInterval(ana.times,eval.times.2),
                                                FUN=function(k){
    if(k==0){return(0)}
    else(return(zeta.2[j,k]))
  })/sqrt(n2)})


  ## Determine D.hat
D.hat<-lapply(1:noit,
              FUN=function(i,B1,B2){B1[[i]]-B2[[i]]},
              B1=B.1.predict,
              B2=B.2.predict)


## Derive supremum of each process
sup.D<-sapply(1:noit,function(it,index){max(D.hat[[it]])})
## Determine quantile
q.alpha<-abs(as.vector(quantile((sup.D),1-alpha)))

## Estimated difference
est<-trprob(P.s.all.1,tr.choice=paste(from,to,sep=" "),ana.times)-
trprob(P.s.all.2,tr.choice=paste(from,to,sep=" "),ana.times)

## Lower CI
CI.lower<-est-q.alpha

output<-list(times=ana.times,est=est,CI.lower=CI.lower)

return(output)
}
```