Pseudo-value regression of clustered data when cluster size is potentially informative

Samuel Anyaso-Samuel, Somnath Datta

March 2022

1 Introduction

First, we describe a suite of functions written in the R programming language for estimating the state occupation probabilities (SOPs) for clustered data under a general multistate model. Secondly, we describe the implementation of the pseudo-value regression of clustered data.

2 Marginal estimators of the state occupation probabilities

For a general multistate model, we describe functions for estimating the state occupation probabilities (SOPs). The functions can handle cases where the data may be subject to right-censoring and/or left-truncation. Currently, only the case for the independent censoring is being implemented in the functions. The functions presented here are a modification of the functions from the `msSurv` [Ferguson et al., 2012] package created for estimating the transition probability matrix and SOPs for independent data.

2.1 Dependencies

- `graph`
- `Rgraphviz`

2.2 Main functions

`msSurvClust`: Returns the estimates of the transition matrix and SOPs for the clustered data for a general multistate model.

Arguments

`Data`: Data frame with one or more records for each cluster unit. Columns should be named: "cID" (unique cluster id), "id" (unique id for the cluster unit), "start" (starting time t_0 for the transition interval. Needed if `LT=TRUE`, optional otherwise), "stop" (stopping time t_1 for the transition interval), "start.stage" (state at time t_0), and "end.stage" (state at time t_1).
`tree`: A `graphNEL` graph with the nodes and edges of the multistate model.
`cens.type`: A character string specifying whether censoring is independent ("ind") or state dependent ("dep"). Default is "ind".
`LT`: Logical argument specifying whether data are subject to left truncation. Default is `FALSE`.
`weight`: Logical argument specifying whether the estimators should be reweighted by the inverse of the cluster sizes. Default is `TRUE`.

`getProbs`: Returns the estimates of the SOPs for specific time points.

Arguments

`fit`: The list returned from the `msSurvClust` function.

`cutoffs`: A numeric vector of time/cutoff points at which the SOPs are desired.

2.3 Example

```
# Consider an example where the data shown below contain the transition outcomes
# for patients whose disease history is described by a progressive illness-death model

head(Data)
  cID id csize         stop start.stage end.stage
1   1  1    37 2.062355048069         1         2
2   1  1    37 2.550987234215         2         3
3   1  2    37 2.633097758183         1         3
4   1  3    37 2.437935798208         1         2
5   1  3    37 2.515316778606         2         3
6   1  4    37 0.865459008855         1         0

## Inputting nodes & edges of the tree structure for a progressive illness-death model
Nodes <- c("1","2","3") # states in MSM
Edges <- list("1"=list(edges=c("2","3")),"2"=list(edges=c("3")),"3"=list(edges=NULL))

## Specifying tree
treeobj <- new("graphNEL", nodes=Nodes, edgeL=Edges, edgemode="directed")

## the SOP based on inverse cluster size weighting
mod <- msSurvClust(Data=Data, tree=treeobj, cens.type="ind", LT=FALSE, weight=TRUE)

## returns the SOPs for specific time points
res <- getProbs(fit=mod, cutoffs=c(0.75, 2.0))
```

3 Pseudo-value regression

For the pseudo-value regression of clustered data, we carry out two steps, (i) obtain marginal estimators of the SOPs and (ii) fit the marginal model using estimating equations that are based on pseudo-value responses. We implement the first step using the `msSurvClust` function described above, and we fit the marginal model using the `geem` function from the `geeM` [McDaniel et al., 2013] package.

3.1 Example

```
## Pseudo-value regression based on pseudo-values obtained from
## the unweighted marginal estimators of the SOPs.
## Suppose inference is desired at time t=2.0.

head(Data)
  cID id         stop start.stage end.stage Z1         Z2 csize
1   1  1 1.44489430444         1         2  0 0.801464768009   8
2   1  1 1.56797828923         2         3  0 0.801464768009   8
3   1  2 1.65001429131         1         3  0 1.130876863097   8
4   1  3 1.80370818717         1         2  0 1.036811043772   8
5   1  3 2.04009832192         2         3  0 1.036811043772   8
6   1  4 1.60629528264         1         3  0 0.828629136865   8
```

```r
# get covariate information
covs <- Data[, !(names(Data) %in% c("csize","start.stage","end.stage","stop"))]
covs <- dplyr::distinct(covs,id,.keep_all=T)

# remove the covariate information
Data <- Data[, c("cID","id","csize","stop","start.stage","end.stage")]
ids <- sort(unique(Data$id)) # unique ids
n <- length(ids) # total number of observations

########################################
## Step-1 - obtain the marginal SOP
########################################
mod <- msSurvClust(Data=Data, tree=treeobj, cens.type="ind", LT=FALSE, weight=FALSE)
res <- getProbs(fit=mod, cutoffs=c(2.0))

########################################
## computation of the pseudo-values
########################################
ps_vals <- matrix(0,nrow=n,ncol=ncol(covs)+length(res))
colnames(ps_vals) <- c(colnames(covs), names(res))
for(ijs in 1:n) {
  # specifies the ij-th observation to be omitted
  ij = ids[ijs]

  # removes the ij-th observation from the data
  temp_dat <- Data[which(Data[,"id"] != ij), ]

  # computes the leave-one-out statistic
  tmp0 <- msSurvClust(Data=temp_dat, tree=treeobj, cens.type="ind", LT=FALSE, weight=FALSE)
  msfit_temp <- getProbs(fit=tmp0,cutoffs=cutoffs)

  pseudo <- n*res - (n-1)*msfit_temp # computes the jackknife pseudo-values
  names(pseudo) <- c("p1", "p2", "p3")
  ps_vals[ijs,] <- c(as.numeric(covs[which(covs$id == ij), ]), pseudo)
}

########################################
## Step-2 - Fit the marginal models
########################################
ps_vals <- as.data.frame(ps_vals)
ps_vals$weights <- 1/ps_vals$csize

CWGEE <- geem(formula=`p 1` ~ Z1 + Z2, data=ps_vals, useP=T, weights = weights,
          id="cID", family=gaussian(link="identity"), corstr="independence")

GEE <- geem(formula=`p 1` ~ Z1 + Z2, data=ps_vals, useP=T,
            id="cID", family=gaussian(link="identity"), corstr="independence")
```

References

Nicole Ferguson, Somnath Datta, and Guy Brock. msSurv: An R package for nonparametric estimation of multistate models. *Journal of Statistical Software*, 50:1–24, 2012.

Lee S McDaniel, Nicholas C Henderson, and Paul J Rathouz. Fast pure R implementation of GEE: application of the Matrix package. *The R journal*, 5(1):181, 2013.