intcensROC

Fast Spline Based Sieve AUC Estimator for Interval Censored Data

Jiaxing Lin   Yuan Wu   Xiaofei Wang   Kouros Owzar

2018-05-01

Outline

Introduction

This document provides an comprehensive example for using the `intcensROC` package to estimate the receiver operating characteristic (ROC) curve and time-dependent area under the curve (AUC) for interval censored survival data, that is not not applicable for existing methods.

The estimator applies a generalized gradient projection method on Spline based likelihood function to obtain the joint distribution function between survival time and biomarker and compute the ROC curve and time-dependent AUC with the estimated joint distribution function. Features of this package include:

1. The algorithm is implemented in `C++`, and ported to `R` by `Rcpp`, to facilitate fast computation.
2. The estimator uses a constrained minimization method and is designed for the interval survival data.

Function Signature and Return of `intcensROC`

Function to compute ROC curve

```
res <- intcensROC(U, V, Marker, Delta, PredictTime, gridNumber = 500)
```

Function arguments:

- `U`: An array contains left times of the censored intervals for the sample.
- `V`: An array contains right times of the censored intervals for the sample.
- `Marker`: An array contains marker levels for the samples.
- `Delta`: An array of indicator for the censoring type, use 1, 2, 3 for left, interval and right censoring types, correspondingly.
- `PredictTime`: A scalar for predict time for the ROC.
- `gridNumber`: A integer for the number of grid of the ROC curve, the default value is 500.

Function return:
A `dataframe` contains two columns

- `tp`: A array for true positive rate.
- `fp`: A array for false positive rate.

Function Signature and Return of `intcensAUC`

Function to compute AUC

```
auc <- intcensAUC(ROCdata)
```

Function argument:

`ROCdata`: A dataframe from the function `intcensROC`.

Function return:

`auc`: A scalar for AUC.

A Simple Start Off Example

A start off example to use function `intcensROC` and `intcensAUC`

```
library(intcensROC)
## example interval censored data
U <- runif(100, min = 0.1, max = 5)
V <- runif(100, min = 0.1, max = 5) + U
Marker <- runif(100, min = 5, max = 10)
Delta  <- sample.int(3, size = 100, replace = TRUE)
pTime  <- 4
## compute the ROC curve
res <- intcensROC(U, V, Marker, Delta, pTime, gridNumber = 500)
head(res)

##             fp           tp
## 1 2.938704e-07 6.709990e-07
## 2 6.429442e-07 1.401916e-06
## 3 1.102661e-06 2.253220e-06
## 4 1.728461e-06 3.285375e-06
## 5 2.575783e-06 4.558851e-06
## 6 3.700067e-06 6.134114e-06

##compute the AUC
auc <- intcensAUC(res)
print(auc)

## [1] 0.5378973
```

Example-Background

Here, we present a comprehensive example as a tutorial on how to use `intcensROC` package.

- We assume the survival time `T follows an exponential distribution with hazard rate` $\lambda = \frac{\log(2)}{24}$.
- The marker `M is assumed to follow a beta distribution with parameter` $\alpha = 2.35$ `and` $\beta = 1.87$.
- The joint distribution of (T, M) is assumed to be generated by `Clayton Coupla` with parameter $\alpha > 1$

  $$F_{T,M}(t, m) = Pr(T < t, M < m) = \{F_T(t)^{\alpha-1} + F_M(m)^{\alpha-1} - 1\}^{\frac{1}{\alpha-1}}$$

  Here $F_T(.)$ and $F_M(.)$ denote the distribution functions of T and M respectively. The dependence between T and M is denote by Kendall $\tau = \frac{\alpha-1}{\alpha+1}$
- The random assessment interval $[U, V]$ are sampled from uniform distribution, V's are sampled within $[L_0, L_c]$, and U is generated from uniform distribution on $[0, V - L_0]$. Here L_c is determined by the censoring rate $\rho = 0.3$ and $L_0 = 0.1$ is the minimum time difference between U and V.

Data Simulation

```
library(copula)
f<-function(x,L0,rate,censor){
  1/((x-L0)*rate)*exp(-L0*rate)-1/((x-L0)*rate)*exp(-x*rate)-censor}
dataSim <- function(kendall_tau = 0.3, n = 100, rho = 0.3, lambda = log(2)/6)
{
  b_alpha     <- 2.35
  b_beta      <- 1.87
  scale       <- 10
  kendall_tau <- iTau( claytonCopula(), kendall_tau)
  Int_cop     <- claytonCopula(param = kendall_tau, dim = 2)
  Int_mvdc    <- mvdc(Int_cop, c("exp","beta"),  paramMargins =
                 list(list(rate = lambda),
                 list(shape1=b_alpha,shape2=b_beta)))
  Int_obs_data <- rMvdc(n, Int_mvdc)
  colnames(Int_obs_data) <- c("event_time", "marker")
```

Data Simulation-Continued

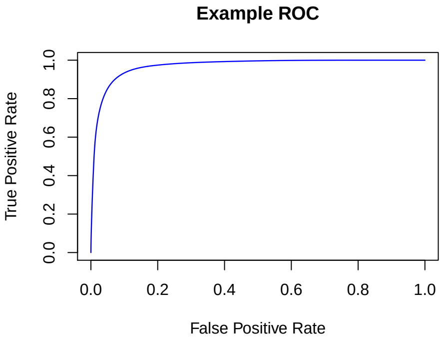
```r
  Int_obs_data[,"marker"] <- Int_obs_data[,"marker"]*scale
  L0       <-0.1; size     <-n; U        <-rep(0,size)
  L        <-uniroot(f, lower = 10^(-6), upper = 500, tol=0.000001,
             L0=L0, rate=lambda, censor=rho)
  V        <-runif(size,L0,L$root)
  for (i in 1:size)
      U[i] <-runif(1,0,(V[i]-L0))
  delta_1      <- Int_obs_data[ ,"event_time"] < U
  delta_2      <- Int_obs_data[ ,"event_time"] >= U&
                  Int_obs_data[ ,"event_time"] <= V
  delta_3      <- Int_obs_data[ ,"event_time"] > V
  data         <- data.frame(U = U, V = V, delta =
                  delta_1+2*delta_2+3*delta_3,
                  marker=Int_obs_data[,"marker"])
}
```

Compute ROC and AUC

```r
mydata <- dataSim(kendall_tau = 0.7, n = 300, rho = 0.3, lambda = log(2)/24)
roc    <- intcensROC(U=mydata[,"U"],V=mydata[,"V"], Marker=mydata[,"marker"],
                 Delta=mydata[,"delta"], PredictTime=12)
print(intcensAUC(roc))

## [1] 0.9700358

plot(roc$fp, roc$tp, type = "l", lwd = 1.2, col="blue", main = "Example ROC",
     xlab = "False Positive Rate", ylab = "True Positive Rate" )
```



Session Information

- R version 3.4.4 (2018-03-15), `x86_64-pc-linux-gnu`
- Running under: `Debian GNU/Linux 9 (stretch)`
- Matrix products: default
- BLAS: `/usr/lib/openblas-base/libblas.so.3`
- LAPACK: `/usr/lib/libopenblasp-r0.2.19.so`
- Base packages: base, datasets, grDevices, graphics, methods, stats, utils
- Other packages: copula 0.999-18, intcensROC 0.1.1, knitr 1.20
- Loaded via a namespace (and not attached): ADGofTest 0.3, Matrix 1.2-12, Rcpp 0.12.16, compiler 3.4.4, evaluate 0.10.1, grid 3.4.4, gsl 1.9-10.3, highr 0.6, lattice 0.20-35, magrittr 1.5, mvtnorm 1.0-7, numDeriv 2016.8-1, pcaPP 1.9-73, pracma 2.0.7, pspline 1.0-18, quadprog 1.5-5, stabledist 0.7-1, stats4 3.4.4, stringi 1.1.7, stringr 1.3.0, tools 3.4.4

```
## [1] "Start Time Wed May  2 15:17:54 2018"
## [1] "End Time   Wed May  2 15:17:58 2018"
```