R code for implementing the methodology proposed in Patra and Sen (2013)

March 29, 2014

In this article, we discuss the implementation of the techniques developed in Patra and Sen (2013). To run the codes, the user requires an installation of R (see R Core Team (2013)) including the Iso (see Turner (2013)) and the fdrtool (see Klaus and Strimmer (2013)) package.

The following function takes as input the data and outputs $\gamma d_n(\hat{F}_s, n, \tilde{F}_s, n)$ (see Patra and Sen (2013, Equation 7)) at equally spaced points. The gridsize determines the spacing between two consecutive points at which $\gamma d_n(\hat{F}_s, n, \tilde{F}_s, n)$ is evaluated. In the sample code, we assume that $F_b$ is the Uniform distribution.

```r
EstMixMdl <- function(data, gridsize=200) {
  n <- length(data) ## Length of the data set
  data <- sort(data) ## Sorts the data set
  data.1 <- unique(data) ## Finds the unique data points
  Fn <- ecdf(data) ## Computes the empirical DF of the data
  Fn.1 <- Fn(data.1) ## Empirical DF of the data at the data points
  ## Calculate the known F_b at the data points
  ## Note: for Uniform(0,1) F_b(x) = x
  ## Usually would need to CHANGE this
  Fb <- data.1
  ## Compute the weights (= frequency/n) of the unique data values, i.e., dF_n
  Freq <- diff(c(0,Fn.1))
  distance <- rep(0,gridsize)
  distance[0]<- sqrt(t((Fn.1-Fb)^2)%*%Freq)
  for(i in 1:gridsize) {
    a <- i/gridsize ## Assumes a value of the mixing proportion
    F.hat <- (Fn.1-(1-a)*Fb)/a ## Computes the naive estimator of F_s
    F.is=pava(F.hat,Freq,decreasing=FALSE) ## Computes the Isotonic Estimator of F_s
    F.is[which(F.is<=0)]=0
    F.is[which(F.is>=1)]=1
    distance[i] <- a*sqrt(t(((F.hat-F.is)^2)%*%Freq));
  }
  return(distance)
}
```

1
The following set of commands will give a plot of $\gamma d_n(\hat{F}_s, \hat{F}_s')$ for $\gamma \in [0,1]$.

dist.alpha <- EstMixMdl(data,gridsize)
frame()
plot((1:gridsize)/gridsize,dist.alpha,type='l',xlab="x",ylab="Distance",col="blue")

We can compute the a lower confidence bound for $\alpha_0$ using asymptotic quantiles of the Cramér-von Mises statistic, which are readily available (e.g., see Anderson and Darling (1952)). The 90%, 95%, and 99% quantiles are 0.5893, 0.6792, and 0.8622 respectively. The following computes the 95% lower confidence bound for $\alpha_0$.

q <- 0.6792
Lower.Cfd.Bound <- sum(dist.alpha>q/sqrt(n))/gridsize

To find the estimator of $\alpha_0$ discussed in Patra and Sen (2013, Section 3) with a particular choice of $c_n$, use the following lines of code.

#Here we have taken the choice of c_n to be log(log(n)).
c.n<-log(log(n))
Est<- sum(dist.alpha>c.n/sqrt(n))/gridsize

To find an heuristic estimator of $\alpha_0$ as discussed in Patra and Sen (2013, Section 4.3), use the following lines of code.

## Numerically find the 2nd derivative of ‘dist.alpha’
dder <- Comp_2ndDer(dist.alpha, gridsize)
Est <- which.max(dder)/gridsize
## Overlaid plot of the normalized 2nd derivative
lines((1:gridsize)/gridsize ,dder*(max(dist.alpha)/max(dder)),col='red')
legend("topright",c("Distance","Scaled 2nd derivative"),
 lty=c(1,1), col = c("blue","red") )

We can now estimate the distribution function $F_s$ using the estimate of $\alpha_0$ (see Patra and Sen (2013, Section 5.1)). The following function estimates the CDF. It takes as input an estimator of $\alpha_0$ together with the ECDF (empirical cumulative distribution function) and $F_b$ evaluated at data points. It outputs a matrix with naive and isotonised estimate of $F_s$ evaluated at data points.

CDFEst <- function(Fn.1,Fb,Est)
{
## Compute the weights (= frequency/n) of the unique data values, i.e., dF_n
Freq <- diff(c(0,Fn.1))
## Computes the naive estimator of F_s
Est.CDF.naive <- (Fn.1-(1-Est)*Fb)/Est
## Computes the Isotonic Estimator of F_s
Est.CDF=pava(Est.CDF.naive,Freq,decreasing=FALSE)
}
Est.CDF[which(Est.CDF<=0)]=0
Est.CDF[which(Est.CDF>=1)]=1
return(cbind(Est.CDF.naive,Est.CDF))
}

Suppose now that $F_s$ has density $f_s$. If we assume that $f_s$ is non-increasing, then we can estimate it using techniques discussed in Patra and Sen (2013, Section 5.2). The following function estimates the density. It takes as input an estimator of $\alpha_0$ together with the ECDF (empirical cumulative distribution function) and $F_b$ evaluated at data points. The output is a matrix with the data points in the first column and the corresponding values of $f_s$ in the second column.

DensEst <- function(Fn.1,Fb,Est)
{
  F.hat <- (Fn.1-(1-Est)*Fb)/Est
  Freq <- diff(c(0,Fn.1))
  F.is <- pava(F.hat,Freq,decreasing=FALSE)
  F.is[which(F.is<=0)] <- 0
  F.is[which(F.is>=1)] <- 1
  F.check <- F.is
  x <- data.1
  y <- F.check
  ll <- gcmlcm(x,y, type="lcm")
  xtemp=rep(ll$x.knots,each=2) #data points for density
  ytemp=c(0,rep(ll$slope.knots,each=2),0) #value of density
  ans<-rbind(t(xtemp),t(ytemp))
  return(ans)
}

References


