\*Tutorial para correr el Delta de Cox para Datos pareados (aunque sirve para datos no pareados si quisiéramos)

###### EN OTRO FICHERO DENOMINADO 'Script\_Sociologia\_Optimizado' ESTARAN ALGUNAS FUNCIONES LAS CUALES ESTAN, A MI PARECER, OPTIMIZADAS ######

library(readxl) #Cargamos la libreria para poder leer ficheros de excel.

#Cargamos datos de excel.

setwd("~/Desktop/three generations")

Excel\_example <- read\_excel("ab\_pa\_hi\_prueba.xlsx")

Excel\_example$Pais <- as.factor(Excel\_example$Pais)

Excel\_example <- Excel\_example[Excel\_example$Pais == "1",] #Elegir Regiones

Excel\_example <- Excel\_example[complete.cases(Excel\_example),] ### Nos quedamos con los datos completos. Toda la terna Abuelo, Padre, e Hijo

m <- data.matrix(Excel\_example) #Convertimos el fichero leido en una matriz de datos. (Esto sirve para cuando los datos estan apareados)

identidad <- c(1:dim(m)[1])

m<- cbind(identidad, m)

#---------------------------------------------------------------------------------------------------------------------------------------------#

### Todas las funciones iniciales trabajan con una matriz de transicion que tiene que venir dada ###

#---------------------------------------------------------------------------------------------------------------------------------------------#

### Definicion de la matriz colapsada en cuatro cuadrantes.

colap\_ij <- function(m, i,j)

{

 if(i>=dim(m)[1] | j>=dim(m)[2] | i<=0 | j<=0){

 print(paste("Los parametros introducidos no son correctos, introduce un valor i entre 1 y",dim(m)[1]-1,"y un valor j entre 1 y",dim(m)[2]-1))

 }else{

 matriz<-matrix(c(sum(m[1:i, 1:j]), sum(m[1:i, (j+1):dim(m)[2]]), sum(m[(i+1):dim(m)[1], 1:j]), sum(m[(i+1):dim(m)[1], (j+1):dim(m)[2]] )), ncol=2, byrow=TRUE)

 rownames(matriz) <- c("0","1")

 colnames(matriz) <- c("0","1")

 list(Comment=paste("La matriz colapsada en (",i,",",j,") es:"),Matriz=matriz)

 }

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

### Definicion de la delta asociada a una matriz colapsada

delta\_ij <- function(m, i, j)

{

 delta<-log((colap\_ij(m,i,j)$Matriz[1,1]\*colap\_ij(m,i,j)$Matriz[2,2])/(colap\_ij(m,i,j)$Matriz[1,2]\*colap\_ij(m,i,j)$Matriz[2,1]))

 list(Comment=paste("La delta asociada a la matriz colapsada en (",i,",",j,") es:"),

 Delta=delta)

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

### Varianza de la delta asociada a una matriz colapasada (En principio no sirve para gran cosa)

# Forma 1.Facil usando las peculiaridades de R

#vardelta\_ij <- function(m,i,j )

#{

# cat("\n")

# print(paste("La varianza de la delta asociada a la matriz colapsada en (",i,",",j,") es:"))

# cat("\n")

# sum(1/colap\_ij(m,i,j)$Matriz)

#}

# Forma 2. Forma directa calculando las sumas

#vardelta\_ij <- function(m,i,j )

#{

# cat("\n")

# print(paste("La varianza de la delta asociada a la matriz colapsada en (",i,",",j,") es:"))

# cat("\n")

# (1/colap\_ij(m,i,j)$Matriz[1,1])+(1/colap\_ij(m,i,j)$Matriz[2,2])+(1/colap\_ij(m,i,j)$Matriz[1,2])+(1/colap\_ij(m,i,j)$Matriz[2,1])

#}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Calculo de la delta de Cox

deltacox <- function(m)

{

 h <- matrix( c(1:((dim(m)[1]-1)\*(dim(m)[2]-1))), ncol =(dim(m)[2]-1), byrow=TRUE)

 rownames(h) <- c(1:(dim(m)[1]-1))

 colnames(h) <- c(1:(dim(m)[2]-1))

 for(i in 1:(dim(m)[1]-1))

 {

 for(j in 1:(dim(m)[2]-1))

 {h[i,j] <- delta\_ij(m,i,j)$Delta

 }

 }

 list(

 Comment1=paste("La matriz de los deltas asociaciados es la siguiente: "),

 Matriz = h,

 Comment2=paste("La delta de Cox global es: "),

 DGlobal = mean(h),

 Comment3=paste("La delta de Cox central es: "),

 DCentral=mean( h[c(-1,-(dim(m)[1]-1)), c(-1,-(dim(m)[2]-1))]))

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Calculo de las coincidencias entre dos matrices colapsadas para el caclulo de la covarianza con el signo de la misma

coincd\_abcd\_signo <- function(m, a,b,c,d)

{

 matriz <- matrix( 0, nrow =4 , ncol = 4)

 rownames(matriz) <- c("00","10","01", "11")

 colnames(matriz) <- c("00","10","01", "11")

 matriz[1,1] <- sum(m[1:min(a,c),1:min(b,d)])

 matriz[1,2] <- if(c<a){(-1)\*sum(m[(c+1):a,1:min(b,d)])} else {0}

 matriz[1,3] <- if(d<b){(-1)\*sum(m[1:min(a,c),(d+1):b])} else {0}

 matriz[1,4] <- if((c<a)&(d<b)){sum(m[(c+1):a,(d+1):b])} else {0}

 matriz[2,1] <- if(a<c){(-1)\*sum(m[(a+1):c,1:min(b,d)])} else {0}

 matriz[2,2] <- sum(m[(max(a,c)+1):(dim(m)[1]), 1:min(b,d)])

 matriz[2,3] <- if((a<c)&(d<b)){sum(m[(a+1):c,(d+1):b])} else {0}

 matriz[2,4] <- if(d<b){(-1)\*sum(m[(max(a,c)+1):(dim(m)[1]), (d+1):b])} else {0}

 matriz[3,1] <- if(b<d){(-1)\*sum(m[1:min(a,c),(b+1):d])} else {0}

 matriz[3,2] <- if((c<a)&(b<d)){sum(m[(c+1):a,(b+1):d])} else {0}

 matriz[3,3] <- sum(m[1:min(a,c), (max(b,d)+1):(dim(m)[2])])

 matriz[3,4] <- if(c<a){(-1)\*sum(m[(c+1):a, (max(b,d)+1):(dim(m)[2]) ])} else {0}

 matriz[4,1] <- if((a<c)&(b<d)){sum(m[(a+1):c,(b+1):d])} else {0}

 matriz[4,2] <- if(b<d){(-1)\*sum(m[(max(a,c)+1):(dim(m)[1]), (b+1):d])} else {0}

 matriz[4,3] <- if(a<c){(-1)\*sum(m[(a+1):c, (max(b,d)+1):(dim(m)[2]) ])} else {0}

 matriz[4,4] <- sum(m[(max(a,c)+1):(dim(m)[1]), (max(b,d)+1):(dim(m)[2])] )

 list(Comment=paste(" La matriz de coincidencias entre las matrices colapsadas en (",a,",",b,") y en (",c,",",d,") es la siguiente: "),

 Matriz=matriz)

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Calculo de los productos entre dos matrices colapsadas como denominadores para el calculo de la matriz de covarianzas.

coincd\_abcd\_prod <- function(m, a, b, c, d)

{

 n<-colap\_ij(m,a,b)$Matriz

 k<-colap\_ij(m,c,d)$Matriz

 pr <- matrix( c( n[1,1]\*k, n[2,1]\*k, n[1,2]\*k, n[2,2]\*k ), ncol=4, byrow=TRUE )

 rownames(pr) <- c("00","10","01", "11")

 colnames(pr) <- c("00","10","01", "11")

 list(Comment=paste("La matriz resultante del producto entre las matrices colapsadas en (",a,",",b,") y en (",c,",",d,") es la siguiente: "),

 Matriz=pr)

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Covarianza asociada a dos matrices colapsadas.

covarianza\_abcd <- function(m, a, b, c, d)

{

 cov<-sum(coincd\_abcd\_signo(m, a, b, c, d)$Matriz/coincd\_abcd\_prod(m,a,b,c,d)$Matriz)

 list(Comment=paste("La covarianza asociada a las matrices colapsadas en (",a,",",b,") y en (",c,",",d,") es: "),Covarianza=cov)

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Matriz de varianzas y covarianzas asociada a todas las matrices colapsadas.

# Esta matriz puede ser util si queremos ponderar los deltas acumulados con distintos pesos

# Bastaria con sumar en vez de construir la matriz si solo nos interesa el promedio

varcovarmatrix <- function(m) #ESTO HACE LA FUNCION ANTERIOR PERO A TODAS LAS POSIBLES 'COLAPSACIONES' 2 A 2

{

 varcovar <- 0

 for(a in 1:(dim(m)[1]-1))

 {

 for(b in 1:(dim(m)[2]-1))

 {

 for(c in 1:(dim(m)[1]-1))

 {

 for(d in 1:(dim(m)[2]-1))

 {

 varcovar <- c(varcovar, covarianza\_abcd(m, a,b,c,d)$Covarianza)

 }

 }

 }

 }

 varcovar <- varcovar[-1]

 varcovar <- matrix(varcovar, ncol=((dim(m)[1]-1)\*(dim(m)[2]-1)), byrow=TRUE)

 list(Comment=paste("La matriz de varianza-covarianza asociada a la matriz original es: "),VarCovar=varcovar)

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Varianza de la Delta de Cox

vardelta <- function(m)

{

 l <- matrix( rep(1, (dim(m)[1]-1)\*(dim(m)[2]-1)), ncol=dim(m)[2]-1) #Crea una matriz cuya dimension son todas las posibles colapsaciones (quitando ultima fila y ultima columna)

 for(i in 1:(dim(m)[1]-1))

 {

 for(j in 1:(dim(m)[2]-1))

 {

 if((i==1) | (j==1) | (j==(dim(m)[2]-1)) | (i==(dim(m)[1]-1))) {l[i,j] = 0}

 }

 }

 l <- matrix(t(l), nrow=1, byrow=TRUE) #HACEMOS TRASPUESTA Y LO CONVERTIMOS EN UN VECTOR

 n <- varcovarmatrix(m)$VarCovar

 d <- as.data.frame(n)

 d[is.na(d)] <- 0

 d <- data.matrix(d)

 list(Comment1=paste("La varianza de la Delta de Cox global es:"),

 VarDGlobal = sum(n)/((dim(m)[1]-1)\*(dim(m)[2]-1))^2 ,

 Comment2=paste("La varianza de la Delta de Cox central es:"),

 VarDCentral = sum(( l%\*%d%\*%t(l) )/(sum(l))^2)

 )

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Desviacion de la Delta de Cox

desvdelta <- function(m) {

 n <- vardelta(m)

 list(Comment1=paste("La desviacion de la Delta de Cox global es:"),

 DesvDGlobal = sqrt(n$VarDGlobal) ,

 Comment2=paste("La desviacion de la Delta de Cox central es:"),

 DesvDCentral = sqrt(n$VarDCentral)

 )

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Intervalo confianza de la Delta de Cox

interv <- function(m,alpha) #Se puede prescindir del 'alpha'.

{

 if(alpha>1 | alpha<0){

 print(paste("Error, el valor alpha introducido debe estar comprendido entre 0 y 1."))

 }else{

 n <- deltacox(m)

 d <- desvdelta(m)

 inf1<-n$DGlobal-qnorm(1-alpha/2)\*d$DesvDGlobal

 sup1<-n$DGlobal+qnorm(1-alpha/2)\*d$DesvDGlobal

 print(paste("Se tiene que el Delta de Cox global es",n$DGlobal,"y la desviacion de la Delta de Cox global es",d$DesvDGlobal,"."))

 cat("\n")

 print(paste("El intervalo de confianza es (",inf1,",",sup1,"), con una confianza del",(1-alpha)\*100,"%"))

 inf2<-n$DCentral-qnorm(1-alpha/2)\*d$DesvDCentral

 sup2<-n$DCentral+qnorm(1-alpha/2)\*d$DesvDCentral

 cat("\n")

 cat("\n")

 print(paste("Se tiene que el Delta de Cox central es",n$DCentral,"y la desviacion de la Delta de Cox central es",d$DesvDCentral,"."))

 cat("\n")

 print(paste("El intervalo de confianza es (",inf2,",",sup2,"), con una confianza del",(1-alpha)\*100,"%"))

 }

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

#interv <- function(m,alpha) #Esta funcion es igual que la anterior pero con alpha=0.05 (una confianza del 95%)

#{

# alpha<-0.05

# n <- deltacox(m)

# d <- desvdelta(m)

#

# inf1<-n$DGlobal-qnorm(1-alpha/2)\*d$DesvDGlobal

# sup1<-n$DGlobal+qnorm(1-alpha/2)\*d$DesvDGlobal

#

# print(paste("Se tiene que el Delta de Cox global es",n$DGlobal,"y la desviacion de la Delta de Cox global es",d$DesvDGlobal,"."))

# cat("\n")

# print(paste("El intervalo de confianza es (",inf1,",",sup1,"), con una confianza del",(1-alpha)\*100,"%"))

#

# inf2<-n$DCentral-qnorm(1-alpha/2)\*d$DesvDCentral

# sup2<-n$DCentral+qnorm(1-alpha/2)\*d$DesvDCentral

#

# cat("\n")

# cat("\n")

# print(paste("Se tiene que el Delta de Cox central es",n$DCentral,"y la desviacion de la Delta de Cox central es",d$DesvDCentral,"."))

# cat("\n")

# print(paste("El intervalo de confianza es (",inf2,",",sup2,"), con una confianza del",(1-alpha)\*100,"%"))

#

#}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Funciones para evaluar la varianza de la diferencia entre dos indicadores asociados a dos tablas:

# Muestras sin aparear, independientes

vardifdelta <- function(m1, m2)

{

 n1 <- vardelta(m1)

 n2 <- vardelta(m2)

 list(

 Comment1=paste("La varianza (global) de la diferencia es: "),

 VardifDGlobal=n1$VarDGlobal+n2$VarDGlobal,

 Comment2=paste("La varianza (central) de la diferencia es: "),

 VardifDCentral = n1$VarDCentral+n2$VarDCentral)

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

#Desviacion de la diferencia entre dos indicadores asociados a dos tablas:

desvdifdelta <- function(m1, m2)

{

 n <- vardifdelta(m1,m2)

 list(

 Comment1=paste("La desviacion (global) de la diferencia es: "),

 DesvdifDGlobal= sqrt(n$VardifDGlobal),

 Comment2=paste("La desviacion (central) de la diferencia es: "),

 DesvdifDCentral = sqrt(n$VardifDCentral)

 )

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

zexp <- function(m1,m2)

{

 d1 <- deltacox(m1)

 d2 <- deltacox(m2)

 n <- desvdifdelta(m1,m2)

 list(

 Comment1=paste("El valor del z-experimental (global) es:"),

 zdifGlobal= (d1$DGlobal-d2$DGlobal)/n$DesvdifDGlobal,

 Comment2=paste("El valor del z-experimental (central) es:"),

 zdifCentral=(d1$DCentral-d2$DCentral)/n$DesvdifDCentral

 )

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

ztest <- function(m1,m2)

{

 n <- zexp(m1,m2)

 print(paste("El valor del z-experimental (global) es",n$zdifGlobal))

 #print(n$zdifGlobal)

 cat("\n")

 print(paste("El p-valor(global) bilateral es",2\*min( pnorm( n$zdifGlobal),1-pnorm( n$zdifGlobal) )))

 cat("\n")

 cat("\n")

 #print(paste("El p-valor(global) Menor Vs Mayor es: ", 1-pnorm( n$zdifGlobal) ))

 #cat("\n")

 #print(paste("El p-valor(global) Mayor Vs Menor es: ", pnorm( n$zdifGlobal) ))

 #cat("\n")

 print(paste("El valor del z-experimental (central) es",n$zdifCentral))

 #print(n$zdifCentral)

 cat("\n")

 print(paste("El p-valor(central) es",2\*min(pnorm( n$zdifCentral) ,1-pnorm( n$zdifCentral )) ))

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

ztest\_menor\_vs\_mayor<-function(m1,m2)

{

 n <- zexp(m1,m2)

 print(paste("El valor del z-experimental (global) es",n$zdifGlobal))

 #print(n$zdifGlobal)

 cat("\n")

 print(paste("El p-valor(global) Menor Vs Mayor es",1-pnorm( n$zdifGlobal )))

 cat("\n")

 cat("\n")

 print(paste("El valor del z-experimental (central) es",n$zdifCentral))

 #print(n$zdifCentral)

 cat("\n")

 print(paste("El p-valor(central) Menor Vs Mayor es",1-pnorm( n$zdifCentral )))

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

ztest\_mayor\_vs\_menor<-function(m1,m2)

{

 n <- zexp(m1,m2)

 print(paste("El valor del z-experimental (global) es",n$zdifGlobal))

 #print(n$zdifGlobal)

 cat("\n")

 print(paste("El p-valor(global) Mayor Vs Menor es",pnorm( n$zdifGlobal )))

 cat("\n")

 cat("\n")

 print(paste("El valor del z-experimental (central) es",n$zdifCentral))

 #print(n$zdifCentral)

 cat("\n")

 print(paste("El p-valor(central) Mayor Vs Menor es",pnorm( n$zdifCentral )))

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

###HASTA AQUI UNA PRIMERA PARTE EN LA QUE LAS MUESTRAS NO ESTAN APAREADAS. ###

###AHORA SE CONSIDERARA QUE LAS MUESTRAS PUEDEN ESTAR APAREADAS. ###

# Funcion historial. Evalua la historia de un individuo en un vector de tiempos denominados vtiempo

historial <- function(identidad, dat, vtime, vranks) # Introducir rango cero para variables cualitativas

{

 recorrido = rep(0, length(vtime))

 for(i in 1:length(vtime))

 {

 if(vranks[i]==0) {recorrido[i]=dat[identidad, vtime[i]] } # Para variables cualitativas.

 else

 {

 if(dat[identidad, vtime[i]] <= quantile(dat[, vtime[i]],probs=seq(0,1,1/vranks[i]))[2])

 {

 recorrido[i]=1

 }

 for(j in 1:(vranks[i]-1))

 {

 if(dat[identidad, vtime[i] ] >

 quantile(dat[,vtime[i] ], probs=seq(0,1,1/vranks[i]))[j+1] &

 dat[identidad, vtime[i] ] <= quantile(dat[, vtime[i] ],

 probs=seq(0,1,1/vranks[i]))[j+2])

 {

 recorrido[i]=j+1

 }

 }

 }

 }

 return(recorrido)

}

# Variable identidad-> Se introduce la identidad del individuo "Fila" al que queremos conocer su historial

# Variable dat -> Matriz cuya primera columna son las identidades y el resto son los tiempos

# Variable vtiempo -> Vector con los numeros de las columnas que identifican los tiempos con los que trabajamos

# Variable vranks -> vector particiones para hacer los cuantiles asociado a cada tiempo.

## Por ejemplo, num\_ranks = 5, significa 5 quintiles.

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Crea la matriz de transicion entre dos tiempos t1 y t2 dado un numero especifico de cuantiles.

# En esta matriz se pueden calcular directamente la delta de COX y su Varianza con las funciones anteriores

transition\_t1t2 <- function(dat, vtime12, vranks12)

{

 vpos <- matrix( c(1:(length(dat[,1])\*2)), nrow =length(dat[,1]) , byrow=TRUE)

 for(i in 1:length(dat[,1]))

 {

 vpos[i,]=historial(i, dat, vtime12, vranks12)

 }

 return(table(vpos[,1], vpos[,2]))

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# A partir de ahora todas las funciones seran utiles para comparar tablas apareadas.

# Funcion colap\_ij\_t1t2. Matriz de transition-Mobility colapasada en fila i columna j.

# Tendriamos un resultado equivalente:

# usando como arguemento la matriz transtion\_t1t2 en la funcion colap\_ij definidad ya...

colap\_ij\_t1t2 <- function(dat,i,j, vtime12, vranks12){

 matriz = cbind(c(0,0),c(0,0))

 for (k in 1:length(dat[,1])){

 h= historial(k,dat,vtime12, vranks12)

 if( h[1]<=i & h[2]<=j){matriz[1,1]=matriz[1,1]+1}

 if( h[1]>i & h[2]<=j){matriz[2,1]=matriz[2,1]+1}

 if( h[1]<=i & h[2]>j){matriz[1,2]=matriz[1,2]+1}

 if( h[1]> i & h[2]>j){matriz[2,2]=matriz[2,2]+1}

 }

 return(matriz)

}

# dat -> Matriz con los datos, cuya primera columna esta formada por los individuos

# i,j -> Donde colapsamos

# t1,t2 -> tiempos inicial y final

# num\_ranks -> Numero de particiones para hacer los cuantiles

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Funcion coinc\_abcd\_signo\_i1i2i3i4. Matriz 4x4

# Esta funcion es fundamental con datos temporales pues no puede ser sustituida.

# Obviamente si (t1, t2)=(t3, t4) la salida debe coincidir con:

# usando como argumento la matriz transtion\_t1t2 en la funcion coinc\_abcd\_signo definida ya...

coinc\_abcd\_signo\_t1t2t3t4 <- function(dat,a,b,c,d, vtime1234, vranks1234){

 matriz=cbind(c(0,0,0,0),c(0,0,0,0),c(0,0,0,0),c(0,0,0,0))

 rownames(matriz) <- c("00","10","01", "11")

 colnames(matriz) <- c("00","10","01", "11")

 for(k in 1:length(dat[,1])){

 h= historial(k,dat,vtime1234, vranks1234)

 if(h[1]<=a & h[2]<=b & h[3]<=c & h[4]<=d){matriz[1,1]=matriz[1,1]+1}

 if(h[1]<=a & h[2]<=b & h[3]>c & h[4]<=d){matriz[1,2]=matriz[1,2]-1}

 if(h[1]<=a & h[2]<=b & h[3]<=c & h[4]>d){matriz[1,3]=matriz[1,3]-1}

 if(h[1]<=a & h[2]<=b & h[3]> c & h[4]>d){matriz[1,4]=matriz[1,4]+1}

 if(h[1]>a & h[2]<=b & h[3]<=c & h[4]<=d){matriz[2,1]=matriz[2,1]-1}

 if(h[1]>a & h[2]<=b & h[3]>c & h[4]<=d){matriz[2,2]=matriz[2,2]+1}

 if(h[1]>a & h[2]<=b & h[3]<=c & h[4]>d){matriz[2,3]=matriz[2,3]-1}

 if(h[1]>a & h[2]<=b & h[3]> c & h[4]>d){matriz[2,4]=matriz[2,4]+1}

 if(h[1]<=a & h[2]>b & h[3]<=c & h[4]<=d){matriz[3,1]=matriz[3,1]-1}

 if(h[1]<=a & h[2]>b & h[3]>c & h[4]<=d){matriz[3,2]=matriz[3,2]+1}

 if(h[1]<=a & h[2]>b & h[3]<=c & h[4]>d){matriz[3,3]=matriz[3,3]+1}

 if(h[1]<=a & h[2]>b & h[3]> c & h[4]>d){matriz[3,4]=matriz[3,4]-1}

 if(h[1]>a & h[2]>b & h[3]<=c & h[4]<=d){matriz[4,1]=matriz[4,1]+1}

 if(h[1]>a & h[2]>b & h[3]>c & h[4]<=d){matriz[4,2]=matriz[4,2]-1}

 if(h[1]>a & h[2]>b & h[3]<=c & h[4]>d){matriz[4,3]=matriz[4,3]-1}

 if(h[1]>a & h[2]>b & h[3]> c & h[4]>d){matriz[4,4]=matriz[4,4]+1}

 }

 return(matriz)

}

# dat -> Matriz con los datos, cuya primera columna esta formada por los individuos

# a,b,c,d -> Cortes que realizamos

# i1,i2, i3, i4 -> tiempos en parejas para dos matrices

# num\_ranks -> Numero de particiones para hacer los cuantiles

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Calculo de los productos entre dos matrices colapsadas como denominadores para el calculo de la matriz de covarianzas.

coincd\_abcd\_prod\_t1t2t3t4 <- function(dat, a, b, c, d, vtime1234, vranks1234)

{

 pr <- matrix(

 c( colap\_ij\_t1t2(dat, a, b, vtime1234[1:2],vranks1234[1:2])[1,1]\*colap\_ij\_t1t2(dat, c, d, vtime1234[3:4],vranks1234[3:4]),

 colap\_ij\_t1t2(dat, a, b, vtime1234[1:2],vranks1234[1:2])[2,1]\*colap\_ij\_t1t2(dat, c, d, vtime1234[3:4],vranks1234[3:4]),

 colap\_ij\_t1t2(dat, a, b, vtime1234[1:2],vranks1234[1:2])[1,2]\*colap\_ij\_t1t2(dat, c, d, vtime1234[3:4],vranks1234[3:4]),

 colap\_ij\_t1t2(dat, a, b, vtime1234[1:2],vranks1234[1:2])[2,2]\*colap\_ij\_t1t2(dat, c, d, vtime1234[3:4],vranks1234[3:4])),

 ncol=4, byrow=TRUE )

 rownames(pr) <- c("00","10","01", "11")

 colnames(pr) <- c("00","10","01", "11")

 return(pr)

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Covarianza entre dos cortes.

covarianza\_abcd\_t1t2t3t4 <- function(dat, a, b, c, d, vtime1234, vranks1234)

{

 sum(coinc\_abcd\_signo\_t1t2t3t4(dat,a,b,c,d, vtime1234, vranks1234)/coincd\_abcd\_prod\_t1t2t3t4(dat,a,b,c,d, vtime1234, vranks1234))

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Matriz para calcular la covarianza entre dos indicadores delta de COX para tiempos con datos apareados.

# Puede ser util tener la matriz si queremos ponderar los deltas acumulados con distintos pesos

##### OJO AQU? HAY QUE CORREGIR PARECE ARREGLADO ############################

covarmatrix\_t1t2t3t4 <- function(dat, vtime1234, vranks1234)

{

 dimension <- vranks1234

 for(i in 1:length(vtime1234))

 {

 if(vranks1234[i]==0) {dimension[i] = max(dat[,vtime1234[i]]) } # Para variables cualitativas.

 }

 varcovar <- 0

 for(a in 1:(dimension[1]-1))

 {

 for(b in 1:(dimension[2]-1))

 {

 for(c in 1:(dimension[3]-1))

 {

 for(d in 1:(dimension[4]-1))

 {

 varcovar <- c(varcovar, covarianza\_abcd\_t1t2t3t4(dat, a,b,c,d, vtime1234, vranks1234))

 }

 }

 }

 }

 varcovar <- varcovar[-1]

 varcovar <- matrix(varcovar, ncol=(dimension[3]-1)\*(dimension[4]-1), byrow=TRUE)

 list(Comment=paste("La matriz de covarianza asociada a la matriz original es: "),CovarMatrix=varcovar)

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

#Funcion para calcular las matrices 'l1' y 'l2' las cuales seran utiles para calcular la varianza de la diferencia.

matrices\_l1\_l2<-function(dat, vtime1234, vranks1234){

 dimension <- vranks1234

 for(i in 1:length(vtime1234))

 {

 if(vranks1234[i]==0) {dimension[i] = max(dat[,vtime1234[i]]) } # Para variables cualitativas.

 }

 l1 <- matrix( rep(1, (dimension[1]-1)\*(dimension[2]-1)), ncol=dimension[2]-1)

 for(i in 1:(dimension[1]-1))

 {

 for(j in 1:(dimension[2]-1))

 {

 if((i==1) | (j==1) | (j==(dimension[2]-1)) | (i==(dimension[1]-1))) {l1[i,j] = 0}

 }

 }

 l1 <- matrix(t(l1), nrow=1, byrow=TRUE)

 #REPETIMOS PARA l2

 l2 <- matrix( rep(1, (dimension[3]-1)\*(dimension[4]-1)), ncol=dimension[4]-1)

 for(i in 1:(dimension[3]-1))

 {

 for(j in 1:(dimension[4]-1))

 {

 if((i==1) | (j==1) | (j==(dimension[4]-1)) | (i==(dimension[3]-1))) {l2[i,j] = 0}

 }

 }

 l2 <- matrix(t(l2), nrow=1, byrow=TRUE)

 list(l1=l1,l2=l2)

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Muestras apareadas. Datos forzosamente en una tabla con secuencias de tiempos.

# Ojo con los redondeos... las matrices de covar tienen muchos numeros redondeados....

##### VER ERRORES vardifdelta\_paired(m,c(2,3,5,6), c(3,3,3,0))

vardifdelta\_paired <- function(dat, vtime1234, vranks1234)

{

 d=covarmatrix\_t1t2t3t4(dat, vtime1234, vranks1234)$CovarMatrix

 l1=matrices\_l1\_l2(dat, vtime1234, vranks1234)$l1

 l2=matrices\_l1\_l2(dat, vtime1234, vranks1234)$l2

 dimension <- vranks1234

 for(i in 1:length(vtime1234))

 {

 if(vranks1234[i]==0) {dimension[i] = max(dat[,vtime1234[i]]) } # Para variables cualitativas.

 }

 VarDifGlobal<- vardelta(transition\_t1t2(dat, vtime1234[1:2], vranks1234[1:2]))$VarDGlobal+vardelta(transition\_t1t2(dat, vtime1234[3:4], vranks1234[3:4]))$VarDGlobal

 -2\*sum(d)/((dimension[1]-1)\*(dimension[2]-1)\*(dimension[3]-1)\*(dimension[4]-1))

 VarDifCentral<-vardelta(transition\_t1t2(dat, vtime1234[1:2], vranks1234[1:2]))$VarDCentral+vardelta(transition\_t1t2(dat, vtime1234[3:4], vranks1234[3:4]))$VarDCentral

 -2\*( sum((l1%\*%d%\*%t(l2))/(sum(l1)\*sum(l2))) )

 list(Comment1=paste("La varianza (global) de la diferencia es: "),

 VarDifGlobal=VarDifGlobal,

 Comment2=paste("La varianza (central) de la diferencia es: "),

 VarDifCentral=VarDifCentral)

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

desdifdelta\_paired <- function(dat, vtime1234, vranks1234)

{

 DesvDifGlobal<-sqrt(vardifdelta\_paired(dat, vtime1234, vranks1234)$VarDifGlobal)

 DesvDifCentral<-sqrt(vardifdelta\_paired(dat, vtime1234, vranks1234)$VarDifCentral)

 list(Comment1=paste("La desviación (global) de la diferencia es: "),DesvDifGlobal=DesvDifGlobal,

 Comment2=paste("La desviación (central) de la diferencia es: "),DesvDifCentral=DesvDifCentral)

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

zexp\_paired<-function(dat,vtime1234,vranks1234){

 m1=transition\_t1t2(dat,vtime1234[1:2],vranks1234[1:2])

 m2=transition\_t1t2(dat,vtime1234[3:4],vranks1234[3:4])

 d<-desdifdelta\_paired(dat,vtime1234,vranks1234)

 E1<-deltacox(m1)

 E2<-deltacox(m2)

 list(

 Comment1=paste("El valor del z-experimental (global) es:"),

 zdifGlobal= (E1$DGlobal-E2$DGlobal)/d$DesvDifGlobal,

 Comment2=paste("El valor del z-experimental (central) es:"),

 zdifCentral=(E1$DCentral-E2$DCentral)/d$DesvDifCentral

 )

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

ztest\_paired <- function(dat,vtime1234,vranks1234)

{

 n <- zexp\_paired(dat,vtime1234,vranks1234)

 print(paste("El valor del z-experimental (global) es",n$zdifGlobal))

 cat("\n")

 print(paste("El p-valor(global) bilateral es",2\*min( pnorm( n$zdifGlobal),1-pnorm( n$zdifGlobal) )))

 cat("\n")

 cat("\n")

 print(paste("El valor del z-experimental (central) es",n$zdifCentral))

 cat("\n")

 print(paste("El p-valor(central) es",2\*min(pnorm( n$zdifCentral) ,1-pnorm( n$zdifCentral )) ))

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

ztest\_paired\_menor\_vs\_mayor<-function(dat,vtime1234,vranks1234)

{

 n <- zexp\_paired(dat,vtime1234,vranks1234)

 print(paste("El valor del z-experimental (global) es",n$zdifGlobal))

 cat("\n")

 print(paste("El p-valor(global) Menor Vs Mayor es",1-pnorm( n$zdifGlobal )))

 cat("\n")

 cat("\n")

 print(paste("El valor del z-experimental (central) es",n$zdifCentral))

 cat("\n")

 print(paste("El p-valor(central) Menor Vs Mayor es",1-pnorm( n$zdifCentral )))

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

ztest\_paired\_mayor\_vs\_menor<-function(dat,vtime1234,vranks1234)

{

 n <- zexp\_paired(dat,vtime1234,vranks1234)

 print(paste("El valor del z-experimental (global) es",n$zdifGlobal))

 cat("\n")

 tr

 print(paste("El p-valor(global) Menor Vs Mayor es",pnorm( n$zdifGlobal )))

 cat("\n")

 cat("\n")

 print(paste("El valor del z-experimental (central) es",n$zdifCentral))

 cat("\n")

 print(paste("El p-valor(central) Menor Vs Mayor es",pnorm( n$zdifCentral )))

}

\*Atención: Los datos deben estar incorporados en la misma carpeta junto al paquete de r.

Para trabajar con todos los datos

Si cambiamos la ruta tenemos que cambiar los siguiente:

Para trabajar con datos del primer layer solo/ se pone una almohdilla si no se quiere

setwd("~/Desktop/three generations")

Excel\_example <- read\_excel("ab\_pa\_hi\_prueba.xlsx")

Excel\_example$Pais <- as.factor(Excel\_example$Pais)

Excel\_example <- Excel\_example[Excel\_example$Pais == "1",] #Elegir Regiones

Excel\_example <- Excel\_example[complete.cases(Excel\_example),] ### Nos quedamos con los datos completos. Toda la terna Abuelo, Padre, e Hijo

m <- data.matrix(Excel\_example) #Convertimos el fichero leido en una matriz de datos. (Esto sirve para cuando los datos estan apareados)

identidad <- c(1:dim(m)[1])

m<- cbind(identidad, m)

#---------------------------------------------------------------------------------------------------------------------------------------------#

### Todas las funciones iniciales trabajan con una matriz de transicion que tiene que venir dada ###

\*\*\*Formar una MATRIZ

transition\_t1t2(m,c(2,3), c(0,0))

\*m=matriz y c(2,3) es la columna(abuelo (2) padre (3)) y c(0,0) es que está calculada para variables categóricas y no para percentiles, deciles y similares.

\*para grabar la matriz

mAPgerma<-transition\_t1t2(m,c(2,3), c(0,0))

mPHgerma<-transition\_t1t2(m,c(3,4), c(0,0))

mAHlgerma<-transition\_t1t2(m,c(2,4), c(0,0))

\*abuelo-padre

\*cálculo de la delta de cox

deltacox(mAPfrance)

\*\*Test para cada parejas

ztest\_paired(m, c(2,3,3,4), c(0,0,0,0))

ztest\_paired(m, c(2,3,2,4), c(0,0,0,0))

ztest\_paired(m, c(2,4,3,4), c(0,0,0,0))

### Cargar datos de Excel: Ejemplo

### library(readxl)

### Excel\_example <- read\_excel("F:/dir1/dir2/.../namefile.xlsx")

### ?Ojo! A veces el directorio raiz no se llama F.

### Convertimos el fichero leido en una matriz de datos.

### m <- data.matrix(Excel\_example)

#---------------------------------------------------------------------------------------------------------------------------------------------#

### Se puede meter la matriz directamente

### m <- matrix(c(11,...,1n,21,...,2n,...,m1,...mn), ncol=n, byrow=TRUE)

#---------------------------------------------------------------------------------------------------------------------------------------------#

### Ejemplo matriz 3x3

### [,1] [,2] [,3]

### [1,] 22 23 21

### [2,] 22 15 17

### [3,] 18 23 12

#---------------------------------------------------------------------------------------------------------------------------------------------#

### Todas las funciones iniciales trabajan con una matriz de transicion que tiene que venir dada

#---------------------------------------------------------------------------------------------------------------------------------------------#

### Definicion de la matriz colapsada en cuatro cuadrantes prescindiendo de las funciones parciales.

colap\_ij <- function(m, i,j)

{

 if(i>=dim(m)[1] | j>=dim(m)[2] | i<=0 | j<=0){

 print(paste("Los parametros introducidos no son correctos, introduce un valor i entre 1 y",dim(m)[1]-1,"y un valor j entre 1 y",dim(m)[2]-1))

 }else{

 matriz<-matrix(c(sum(m[1:i, 1:j]), sum(m[1:i, (j+1):dim(m)[2]]), sum(m[(i+1):dim(m)[1], 1:j]), sum(m[(i+1):dim(m)[1], (j+1):dim(m)[2]] )), ncol=2, byrow=TRUE)

 rownames(matriz) <- c("0","1")

 colnames(matriz) <- c("0","1")

 list(Comment=paste("La matriz colapsada en (",i,",",j,") es:"),Matriz=matriz)

 }

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

### Definicion de la delta asociada a una matriz colapsada

delta\_ij <- function(m, i, j)

{

delta<-log((colap\_ij(m,i,j)$Matriz[1,1]\*colap\_ij(m,i,j)$Matriz[2,2])/(colap\_ij(m,i,j)$Matriz[1,2]\*colap\_ij(m,i,j)$Matriz[2,1]))

list(Comment=paste("La delta asociada a la matriz colapsada en (",i,",",j,") es:"),

Delta=delta)

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

### Varianza de la delta asociada a una matriz colapasada (En principio no sirve para gran cosa)

# Forma 1.Facil usando las peculiaridades de R

#vardelta\_ij <- function(m,i,j )

#{

# cat("\n")

# print(paste("La varianza de la delta asociada a la matriz colapsada en (",i,",",j,") es:"))

# cat("\n")

# sum(1/colap\_ij(m,i,j)$Matriz)

#}

# Forma 2. Forma directa calculando las sumas

#vardelta\_ij <- function(m,i,j )

#{

# cat("\n")

# print(paste("La varianza de la delta asociada a la matriz colapsada en (",i,",",j,") es:"))

# cat("\n")

# (1/colap\_ij(m,i,j)$Matriz[1,1])+(1/colap\_ij(m,i,j)$Matriz[2,2])+(1/colap\_ij(m,i,j)$Matriz[1,2])+(1/colap\_ij(m,i,j)$Matriz[2,1])

#}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Calculo de la delta de Cox como sumatorio ponderado de todas las deltas...

deltacox <- function(m)

{

 h <- matrix( c(1:((dim(m)[1]-1)\*(dim(m)[2]-1))), ncol =(dim(m)[2]-1), byrow=TRUE)

 rownames(h) <- c(1:(dim(m)[1]-1))

 colnames(h) <- c(1:(dim(m)[2]-1))

 for(i in 1:(dim(m)[1]-1))

 {

 for(j in 1:(dim(m)[2]-1))

 {h[i,j] <- delta\_ij(m,i,j)$Delta

 }

 }

 list(

 Comment1=paste("La matriz de los deltas asociaciados es la siguiente: "),

 Matriz = h,

 Comment2=paste("La delta de Cox global es: "),

 DGlobal = mean(h),

 Comment3=paste("La delta de Cox central es: "),

 DCentral=mean( h[c(-1,-(dim(m)[1]-1)), c(-1,-(dim(m)[2]-1))]))

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Calculo de las coincidencias entre dos matrices colapsadas para el caclulo de la covarianza con el signo de la misma

coincd\_abcd\_signo <- function(m, a,b,c,d)

{

 matriz <- matrix( 0, nrow =4 , ncol = 4)

 rownames(matriz) <- c("00","10","01", "11")

 colnames(matriz) <- c("00","10","01", "11")

 matriz[1,1] <- sum(m[1:min(a,c),1:min(b,d)])

 matriz[1,2] <- if(c<a){(-1)\*sum(m[(c+1):a,1:min(b,d)])} else {0}

 matriz[1,3] <- if(d<b){(-1)\*sum(m[1:min(a,c),(d+1):b])} else {0}

 matriz[1,4] <- if((c<a)&(d<b)){sum(m[(c+1):a,(d+1):b])} else {0}

 matriz[2,1] <- if(a<c){(-1)\*sum(m[(a+1):c,1:min(b,d)])} else {0}

 matriz[2,2] <- sum(m[(max(a,c)+1):(dim(m)[1]), 1:min(b,d)])

 matriz[2,3] <- if((a<c)&(d<b)){sum(m[(a+1):c,(d+1):b])} else {0}

 matriz[2,4] <- if(d<b){(-1)\*sum(m[(max(a,c)+1):(dim(m)[1]), (d+1):b])} else {0}

 matriz[3,1] <- if(b<d){(-1)\*sum(m[1:min(a,c),(b+1):d])} else {0}

 matriz[3,2] <- if((c<a)&(b<d)){sum(m[(c+1):a,(b+1):d])} else {0}

 matriz[3,3] <- sum(m[1:min(a,c), (max(b,d)+1):(dim(m)[2])])

 matriz[3,4] <- if(c<a){(-1)\*sum(m[(c+1):a, (max(b,d)+1):(dim(m)[2]) ])} else {0}

 matriz[4,1] <- if((a<c)&(b<d)){sum(m[(a+1):c,(b+1):d])} else {0}

 matriz[4,2] <- if(b<d){(-1)\*sum(m[(max(a,c)+1):(dim(m)[1]), (b+1):d])} else {0}

 matriz[4,3] <- if(a<c){(-1)\*sum(m[(a+1):c, (max(b,d)+1):(dim(m)[2]) ])} else {0}

 matriz[4,4] <- sum(m[(max(a,c)+1):(dim(m)[1]), (max(b,d)+1):(dim(m)[2])] )

 list(Comment=paste(" La matriz de coincidencias entre las matrices colapsadas en (",a,",",b,") y en (",c,",",d,") es la siguiente: "),

 Matriz=matriz)

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Calculo de los productos entre dos matrices colapsadas como denominadores para el calculo de la matriz de covarianzas.

coincd\_abcd\_prod <- function(m, a, b, c, d)

{

 n<-colap\_ij(m,a,b)$Matriz

 k<-colap\_ij(m,c,d)$Matriz

 pr <- matrix( c( n[1,1]\*k, n[2,1]\*k, n[1,2]\*k, n[2,2]\*k ), ncol=4, byrow=TRUE )

 rownames(pr) <- c("00","10","01", "11")

 colnames(pr) <- c("00","10","01", "11")

 list(Comment=paste("La matriz resultante del producto entre las matrices colapsadas en (",a,",",b,") y en (",c,",",d,") es la siguiente: "),

 Matriz=pr)

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Covarianza asociada a dos matrices colapsadas.

covarianza\_abcd <- function(m, a, b, c, d)

{

 cov<-sum(coincd\_abcd\_signo(m, a, b, c, d)$Matriz/coincd\_abcd\_prod(m,a,b,c,d)$Matriz)

 list(Comment=paste("La covarianza asociada a las matrices colapsadas en (",a,",",b,") y en (",c,",",d,") es: "),Covarianza=cov)

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Matriz de varianzas y covarianzas asociada a todas las matrices colapsadas.

# Esta matriz puede ser util si queremos ponderar los deltas acumulados con distintos pesos

# Bastaria con sumar en vez de construir la matriz si solo nos interesa el promedio

varcovarmatrix <- function(m) #ESTO HACE LA FUNCION ANTERIOR PERO A TODAS LAS POSIBLES 'COLAPSACIONES' 2 A 2

{

 varcovar <- 0

 for(a in 1:(dim(m)[1]-1))

 {

 for(b in 1:(dim(m)[2]-1))

 {

 for(c in 1:(dim(m)[1]-1))

 {

 for(d in 1:(dim(m)[2]-1))

 {

 varcovar <- c(varcovar, covarianza\_abcd(m, a,b,c,d)$Covarianza)

 }

 }

 }

 }

 varcovar <- varcovar[-1]

 varcovar <- matrix(varcovar, ncol=((dim(m)[1]-1)\*(dim(m)[2]-1)), byrow=TRUE)

 list(Comment=paste("La matriz de varianza-covarianza asociada a la matriz original es: "),VarCovar=varcovar)

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Varianza de la Delta de Cox

vardelta <- function(m)

{

 l <- matrix( rep(1, (dim(m)[1]-1)\*(dim(m)[2]-1)), ncol=dim(m)[2]-1) #Crea una matriz cuya dimension son todas las posibles colapsaciones (quitando ultima fila y ultima columna)

 for(i in 1:(dim(m)[1]-1)) #¿No podemos poner i in 1:dim(l)[1]?

 {

 for(j in 1:(dim(m)[2]-1)) #¿No podemos poner j in 1:dim(l)[2]?

 {

 if((i==1) | (j==1) | (j==(dim(m)[2]-1)) | (i==(dim(m)[1]-1))) {l[i,j] = 0}

 }

 }

 l <- matrix(t(l), nrow=1, byrow=TRUE) #HACEMOS TRASPUESTA Y LO CONVERTIMOS EN UN VECTOR

 n <- varcovarmatrix(m)$VarCovar

 d <- as.data.frame(n)

 d[is.na(d)] <- 0

 d <- data.matrix(d)

 list(Comment1=paste("La varianza de la Delta de Cox global es:"),

 VarDGlobal = sum(n)/((dim(m)[1]-1)\*(dim(m)[2]-1))^2 ,

 Comment2=paste("La varianza de la Delta de Cox central es:"),

 VarDCentral = sum(( l%\*%d%\*%t(l) )/(sum(l))^2)

 )

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Desviacion de la Delta de Cox

desvdelta <- function(m) {

 n <- vardelta(m)

 list(Comment1=paste("La desviacion de la Delta de Cox global es:"),

 DesvDGlobal = sqrt(n$VarDGlobal) ,

 Comment2=paste("La desviacion de la Delta de Cox central es:"),

 DesvDCentral = sqrt(n$VarDCentral)

 )

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Intervalo confianza de la Delta de Cox

interv <- function(m,alpha)

{

 if(alpha>1 | alpha<0){

 print(paste("Error, el valor alpha introducido debe estar comprendido entre 0 y 1."))

 }else{

 n <- deltacox(m)

 d <- desvdelta(m)

 inf1<-n$DGlobal-qnorm(1-alpha/2)\*d$DesvDGlobal

 sup1<-n$DGlobal+qnorm(1-alpha/2)\*d$DesvDGlobal

 print(paste("Se tiene que el Delta de Cox global es",n$DGlobal,"y la desviacion de la Delta de Cox global es",d$DesvDGlobal,"."))

 cat("\n")

 print(paste("El intervalo de confianza es (",inf1,",",sup1,"), con una confianza del",(1-alpha)\*100,"%"))

 inf2<-n$DCentral-qnorm(1-alpha/2)\*d$DesvDCentral

 sup2<-n$DCentral+qnorm(1-alpha/2)\*d$DesvDCentral

 cat("\n")

 cat("\n")

 print(paste("Se tiene que el Delta de Cox central es",n$DCentral,"y la desviacion de la Delta de Cox central es",d$DesvDCentral,"."))

 cat("\n")

 print(paste("El intervalo de confianza es (",inf2,",",sup2,"), con una confianza del",(1-alpha)\*100,"%"))

 }

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

# Funciones para evaluar la varianza de la diferencia entre dos indicadores asociados a dos tablas:

# Muestras sin aparear, independientes

vardifdelta <- function(m1, m2)

{

 n1 <- vardelta(m1)

 n2 <- vardelta(m2)

 list(

 Comment1=paste("La varianza (global) de la diferencia es: "),

 VardifDGlobal=n1$VarDGlobal+n2$VarDGlobal,

 Comment2=paste("La varianza (central) de la diferencia es: "),

 VardifDCentral = n1$VarDCentral+n2$VarDCentral)

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

#Desviacion de la diferencia entre dos indicadores asociados a dos tablas:

desvdifdelta <- function(m1, m2)

{

 n <- vardifdelta(m1,m2)

 list(

 Comment1=paste("La desviacion (global) de la diferencia es: "),

 DesvdifDGlobal= sqrt(n$VardifDGlobal),

 Comment2=paste("La desviacion (central) de la diferencia es: "),

 DesvdifDCentral = sqrt(n$VardifDCentral)

 )

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

zexp <- function(m1,m2)

{

 d1 <- deltacox(m1)

 d2 <- deltacox(m2)

 n <- desvdifdelta(m1,m2)

 list(

 Comment1=paste("El valor del z-experimental (global) es:"),

 zdifGlobal= (d1$DGlobal-d2$DGlobal)/n$DesvdifDGlobal,

 Comment2=paste("El valor del z-experimental (central) es:"),

 zdifCentral=(d1$DCentral-d2$DCentral)/n$DesvdifDCentral

 )

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

ztest <- function(m1,m2)

{

 n <- zexp(m1,m2)

 print(paste("El valor del z-experimental (global) es",n$zdifGlobal))

 #print(n$zdifGlobal)

 cat("\n")

 print(paste("El p-valor(global) bilateral es",2\*min( pnorm( n$zdifGlobal),1-pnorm( n$zdifGlobal) )))

 cat("\n")

 cat("\n")

 #print(paste("El p-valor(global) Menor Vs Mayor es: ", 1-pnorm( n$zdifGlobal) ))

 #cat("\n")

 #print(paste("El p-valor(global) Mayor Vs Menor es: ", pnorm( n$zdifGlobal) ))

 #cat("\n")

 print(paste("El valor del z-experimental (central) es",n$zdifCentral))

 #print(n$zdifCentral)

 cat("\n")

 print(paste("El p-valor(central) es",2\*min(pnorm( n$zdifCentral) ,1-pnorm( n$zdifCentral )) ))

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

ztest\_menor\_vs\_mayor<-function(m1,m2)

{

 n <- zexp(m1,m2)

 print(paste("El valor del z-experimental (global) es",n$zdifGlobal))

 #print(n$zdifGlobal)

 cat("\n")

 print(paste("El p-valor(global) Menor Vs Mayor es",1-pnorm( n$zdifGlobal )))

 cat("\n")

 cat("\n")

 print(paste("El valor del z-experimental (central) es",n$zdifCentral))

 #print(n$zdifCentral)

 cat("\n")

 print(paste("El p-valor(central) Menor Vs Mayor es",1-pnorm( n$zdifCentral )))

}

#---------------------------------------------------------------------------------------------------------------------------------------------#

ztest\_mayor\_vs\_menor<-function(m1,m2)

{

 n <- zexp(m1,m2)

 print(paste("El valor del z-experimental (global) es",n$zdifGlobal))

 #print(n$zdifGlobal)

 cat("\n")

 print(paste("El p-valor(global) Mayor Vs Menor es",pnorm( n$zdifGlobal )))

 cat("\n")

 cat("\n")

 print(paste("El valor del z-experimental (central) es",n$zdifCentral))

 #print(n$zdifCentral)

 cat("\n")

 print(paste("El p-valor(central) Mayor Vs Menor es",pnorm( n$zdifCentral )))

}

deltacox()

ztest(,)