```
##################################################################################
# R code to find confidence intervals in the unrestricted case for a 2xc table #
# : Modeling and inference for an ordinal effect size measure                #
# : Authors: Euijung Ryu and Alan Agresti                                  #
# : Statistics in Medicine (2007)                                         #
##################################################################################


##########################################
#### Data: Shoulder-tip pain scores #####

y1<-c(19,2,1,0,0)
y2<-c( 7,3,4,3,2)


# number of columns
c<-length(y1)

# matrix A (will be used to find confidence intervals)
zero<-c(rep(0,c))
one<-c(rep(1,c))
J<-one%*%t(one)
J

DD<-diag(c)
A<-J-(0.5)*DD

for(i in 1:c) {
  for(j in 1:c) {
    if(j>i) A[i,j]<-0
  }
}
A


# for 95% confidence intervals
alpha<-0.05
```

```
#######################################################
#### Confidence Intervals for the measure theta #####



########### Halperin et al. confidence interval #####

Halperin.CI<-function(y1, y2){
        critic.Norm<-qnorm(1-alpha/2)
            x1<-y1
            x2<-y2
        n1<-sum(x1)
        n2<-sum(x2)

        pi.hat<-x1/n1
        lambda.hat<-x2/n2
        theta.hat<-c(lambda.hat)%*%A%*%pi.hat

            A11<-t(lambda.hat)%*%A%*%diag(pi.hat)%*%t(A)%*%lambda.hat
            B11<-t(pi.hat)%*%t(A)%*%diag(lambda.hat)%*%A%*%pi.hat
            A12<-A13<-c(rep(0,c-1))

            for(i in 1:(c-1)){
                    A12[i]<-pi.hat[i]*(   (1-lambda.hat[i]) * sum(lambda.hat[(i+1):c])  - (sum(lambda.hat[(i+1):c]) )^2 )
                    A13[i]<-pi.hat[i]*lambda.hat[i]*(1-lambda.hat[i])
            }
        A1<-A11-(1/(n2-1))*sum(A12)-(1/(4*(n2-1)))*(sum(A13)+pi.hat[c]*lambda.hat[c]*(1-lambda.hat[c])  )

        B12<-B13<-c(rep(0,c-1))

            for(j in 2:c){
                    B12[j-1]<-lambda.hat[j]*(  (1-pi.hat[j])*sum(pi.hat[1:(j-1)]) - (sum(pi.hat[1:(j-1)]))^2 )
                    B13[j-1]<-lambda.hat[j]*pi.hat[j]*(1-pi.hat[j])
            }

            B1<-B11-(1/(n1-1))*sum(B12)-(1/(4*(n1-1)))*(sum(B13)+ lambda.hat[1]*pi.hat[1]*(1-pi.hat[1])  )
```

```
           theta.fn<-( (n1*n2-n1-n2+2)*theta.hat-n1*n2*theta.hat^2  )/((n1-1)*(n2-1)  )+A1/(n1-1)+B1/(n2-1)

           epsilon.hat<-0
       if( (abs(A1)>1e-10 ) | (abs(B1)>1e-10 ) | abs(theta.hat)>1e-10){
                 if((abs(A1-1)>1e-10) |(abs(B1-1)>1e-10 ) | abs(theta.hat-1)>1e-10 ){
                   epsilon.hat<- ( (n1+n2-2)*theta.hat-(n2-1)*A1-(n1-1)*B1  )/((n1+n2-2)*theta.fn)
                   a.hat<-epsilon.hat
                   if(epsilon.hat<0) a.hat<-0
               if(epsilon.hat>1) a.hat<-1
               epsilon.hat<-a.hat
                 }
            }

           gam<-(n1+n2-1)-(n1+n2-2)*epsilon.hat
           CCC<-gam*critic.Norm^2/(n1*n2)

           L.H<-(CCC+2*theta.hat-sqrt(CCC^2+4*CCC*theta.hat*(1-theta.hat)))/(2*(CCC+1))
           U.H<-(CCC+2*theta.hat+sqrt(CCC^2+4*CCC*theta.hat*(1-theta.hat)))/(2*(CCC+1))
           H.CI<-c(1,1)*c(L.H, U.H)
           list(theta.hat=theta.hat, Halperin.CI=H.CI)
}

Halperin.CI(y1,y2)
```

```
########## Newcombe's pseudo-score confidence interval ################

Newcombe.CI<-function(y1, y2){
        critic.Norm<-qnorm(1-alpha/2)
            x1<-y1
            x2<-y2
        n1<-sum(x1)
        n2<-sum(x2)

        pi.hat<-x1/n1
        lambda.hat<-x2/n2
        theta.hat<-c(lambda.hat)%*%A%*%pi.hat

            aa<-(n1+n2)/2-1
            bb<-(critic.Norm^2)/(n1*n2)
            cc<-theta.hat

            c1<- -(1+bb)-2*aa*bb
            c2<-(1+2*bb+2*cc)+4*aa*bb
            c3<-2+2*bb-(bb+2*cc)-cc^2-aa*bb
            c4<-cc^2-2*bb-4*cc-aa*bb
            c5<-2*cc^2

            iter<-0
        diff<-1
            reltol=1e-6
            maxiter=200
            theta0<-0.01

        while((diff>reltol) & (iter<-iter+1)<=maxiter) {
                  theta1<-theta0
                  f<-c1*theta1^4+c2*theta1^3+c3*theta1^2+c4*theta1+c5
                  der.f<-4*c1*theta1^3+3*c2*theta1^2+2*c3*theta1+c4
                  theta2<-theta1-(1/der.f)*f
                  diff<-(theta2-theta1)^2
                  theta0<-theta2
                  theta0
                  diff
            }
            left<-theta0

             iter<-0

        diff<-1
             reltol<-1e-6
             maxiter<-200
             theta0<-.99

        while((diff>reltol) & (iter<=maxiter)) {
                    theta1<-theta0
                    f<-c1*theta1^4+c2*theta1^3+c3*theta1^2+c4*theta1+c5
                    der.f<-4*c1*theta1^3+3*c2*theta1^2+2*c3*theta1+c4
                    theta2<-theta1-(1/der.f)*f
                    diff<-(theta2-theta1)^2
                    theta0<-theta2
                    theta0
                    diff
                    iter<-iter+1
           }
           right<-theta0

           CI<-c(1,1)*c(left, right)
           list(Newcombe.CI=CI)
}
Newcombe.CI(y1,y2)
```

```
############ Unrestricted Wald-type confidence intervals ##########

Wald.CIs<-function(y1, y2){
        critic.Norm<-qnorm(1-alpha/2)
            x1<-y1
            x2<-y2
        n1<-sum(x1)
        n2<-sum(x2)

        pi.hat<-x1/n1
        lambda.hat<-x2/n2
        theta.hat<-c(lambda.hat)%*%A%*%pi.hat

            CC<-t(lambda.hat)%*%A%*%diag(pi.hat)%*%t(A)%*%lambda.hat
            DD<-t(pi.hat)%*%t(A)%*%diag(lambda.hat)%*%A%*%pi.hat

            V.1<-((n2-1)*(CC-theta.hat^2)+(n1-1)*(DD-theta.hat^2)+theta.hat*(1-theta.hat)-.25*t(pi.hat)%*%lambda.hat)/(n1*n2)
            V.1<-abs(V.1)
            W.1<-theta.hat+c(-1,1)*critic.Norm*sqrt(V.1)

            Logit.1<-Logit.2<-c(0,1)
            if( abs(theta.hat)>1e-8 & abs(theta.hat-1)>1e-8  ){
          logit<-log(theta.hat/(1-theta.hat))

          L.1<-logit-critic.Norm*sqrt(V.1)/(theta.hat*(1-theta.hat))
          U.1<-logit+critic.Norm*sqrt(V.1)/(theta.hat*(1-theta.hat))

          L.theta.W.Logit.1<-exp(L.1)/(1+exp(L.1))
          U.theta.W.Logit.1<-exp(U.1)/(1+exp(U.1))

                  Logit.1<-c(L.theta.W.Logit.1, U.theta.W.Logit.1)
            }

        list(MLE.theta=theta.hat, Wald.CI=W.1, Logit.Wald.CI=Logit.1)
}

Wald.CIs(y1,y2)
```

```
########## Unrestricted LRT, Score, and Pseudo-score confidence intervals ##################

## To use the following function, we need to get "mph.fit" function
## from Dr. Joseph Lang (joseph-lang@uiowa.edu)
## It takes some time to find these confidence intervals
## We can change "starting.boundaries" inside the following function if you want


LRT.Score.Pseudo.Score.CIs<-function(y1, y2){
        starting.boundaries<-Newcombe.CI(y1, y2)$Newcombe.CI
        theta.set<-seq(starting.boundaries[1]-0.02, starting.boundaries[2]+0.02, length=1000)

        x1<-y1
        x2<-y2
    n1<-sum(x1)
    n2<-sum(x2)

    pi.hat<-x1/n1
    lambda.hat<-x2/n2
    theta.hat<-c(lambda.hat)%*%A%*%pi.hat

    L.cal1<-c(rep(0,c))
    L.cal2<-c(rep(0,c))
    for(j in 1:c){
         if(x1[j]>0) L.cal1[j]<-x1[j]*log(pi.hat[j])
         if(x2[j]>0) L.cal2[j]<-x2[j]*log(lambda.hat[j])
    }

    L.HA<-sum(L.cal1)+sum(L.cal2)

        yy<-matrix(c(y1,y2), ncol=1)
        strata<-c(rep(0,c), rep(1,c))


        critic.Chisq<-qchisq(1-alpha,1)
        store.LRT<-store.Score<-store.Pseudo.Score<-c(rep(0,length=length(theta.set)))
        LRT.interval<-Score.interval<-Pseudo.Score.interval<-c(rep(1e+5, length=length(theta.set)))
```

```
for(k in 1:length(theta.set)){
        theta.null<-theta.set[k]
        cat("index=", k, "theta.value=", theta.null, "\n")

        h.fct<-function(m){
                p<-diag(c(1/(Z%*%t(Z)%*%m)))%*%m
                t(p[(c+1):(2*c)])%*%A%*%p[1:c]-theta.null
        }

        if(  (1-theta.hat)> .1 & ( theta.hat > .1)) {
                a<-mph.fit(yy,strata=strata,h.fct=h.fct, norm.diff.conv=10,  maxiter=200, norm.score.conv=1e-5)
        }

        if(  (1-theta.hat)> .1 & ( theta.hat <= .1)) {
                a<-mph.fit(yy,strata=strata,,h.fct=h.fct, norm.diff.conv=10,  maxiter=200, norm.score.conv=1e-5, y.eps=.1)
        }

        if(  (1-theta.hat)<= .1 & ( theta.hat > .1)) {
                a<-mph.fit(yy,strata=strata,,h.fct=h.fct, norm.diff.conv=10,  maxiter=200, norm.score.conv=1e-5, y.eps=.1)
        }

        if(  (1-theta.hat)<= .1 & ( theta.hat <= .1) ){
                a<-mph.fit(yy,strata=strata,,h.fct=h.fct, norm.diff.conv=10,  maxiter=200, norm.score.conv=1e-5, y.eps=.1)
        }

        lang<-a$p
        pi.final<-lang[1:c]
        lambda.final<-lang[(c+1):(2*c)]
        theta.final<-t(lambda.final)%*%A%*%pi.final

        lrt<-score<-pseudo.score<-1e10

        if(abs(theta.final - theta.null) <1e-5){
        L.cal10<-c(rep(0,c))
        L.cal20<-c(rep(0,c))
        for(j in 1:c){
                if(x1[j]>0) L.cal10[j]<-x1[j]*log(pi.final[j])
                if(x2[j]>0) L.cal20[j]<-x2[j]*log(lambda.final[j])
                }

                L.H0<-sum(L.cal10)+sum(L.cal20)
                        lrt<- -2*(L.H0-L.HA)
                        score<-a$Xsq

                CC<-t(lambda.final)%*%A%*%diag(pi.final)%*%t(A)%*%lambda.final
                DD<-t(pi.final)%*%t(A)%*%diag(lambda.final)%*%A%*%pi.final

                var.theta.final<-((n2-1)*(CC-theta.null^2)+(n1-1)*(DD-theta.null^2)+theta.null*(1-theta.null)-.25*t(pi.final)%*%lambda.final)/(n1*n2)
                pseudo.score<-(theta.hat-theta.null)^2/var.theta.final
                }

                cat("LRT.stat=", lrt, "Score.stat=", score, "Pseudo.Score.stat=", pseudo.score, "\n")

        store.LRT[k]<-lrt
                store.Score[k]<-score
                store.Pseudo.Score[k]<-pseudo.score

            if(lrt<critic.Chisq) {LRT.interval[k]<-theta.null}
            if(score<critic.Chisq) {Score.interval[k]<-theta.null}
            if(pseudo.score<critic.Chisq) {Pseudo.Score.interval[k]<-theta.null}
        }

        LRT.set<-LRT.interval[LRT.interval<1e+5]
        LRT.CI<-cbind(min(LRT.set), max(LRT.set))

        Score.set<-Score.interval[Score.interval<1e+5]
        Score.CI<-cbind(min(Score.set), max(Score.set))

        Pseudo.Score.set<-Pseudo.Score.interval[Pseudo.Score.interval<1e+5]
        Pseudo.Score.CI<-cbind(min(Pseudo.Score.set), max(Pseudo.Score.set))

        list(LRT.CI=LRT.CI, Score.CI=Score.CI, Pseudo.Score.CI=Pseudo.Score.CI)
}


LRT.Score.Pseudo.Score.CIs(y1, y2)
```