

```
#####
# R code to find simultaneous confidence intervals for binomial proportions #####
# #####
# The method was discussed in the following article #####
# #####
# Article: Simultaneous Confidence Intervals for Comparing Binomial Parameters #####
# Journal: Biometrics (2008) #####
# Authors: Alan Agresti, Matilde Bini, Bruno Bertaccini, and Euijung Ryu #####
#####
```

Simultaneous Score Intervals for Difference of Proportions

```
Diff.Score.CI<-function(y1, N1, y2, N2, alpha, t){
  Q.T.alpha<-qtukey(p=alpha, nmeans=t, lower.tail=FALSE, log.p=FALSE, df=Inf)
  card<-2000
  store.Score.Diff<-c(rep(2,card))
  dif.set<-seq(-.999, .999, length=card )
  k<-1

  for(k in 1:card){
    dif<-dif.set[k]
    score.min<-Score.diff(y1,N1,y2, N2, dif)
    if(score.min < Q.T.alpha^2/2) {store.Score.Diff[k]<-dif}
  }

  left.Score<-min(store.Score.Diff[abs(store.Score.Diff)<2])
  right.Score<-max(store.Score.Diff[abs(store.Score.Diff)<2])
  Score.CI<-c(1,1)*c(left.Score, right.Score)
  return(Score.CI)
}
```

```
Score.diff<- function (y1,N1,y2,N2,dif){
  p1x<-y1/N1
  p1y<-y2/N2
  nx<-N1
  ny<-N2
  diff = p1x-p1y-dif

  if ( abs(diff) == 0 ) {
    fmdiff = 0
  }
}
```

```

else{
  t = ny/nx
  a = 1+t
  b = -(1+ t + p1x + t*p1y + dif*(t+2))
  c = dif*dif + dif*(2*p1x + t +1) + p1x + t*p1y
  d = -p1x*dif*(1+dif)
  v = (b/a/3)^3 - b*c/(6*a*a) + d/a/2
  s = sqrt( (b/a/3)^2 - c/a/3)

  if(v>0){
    u=s
  }
  else{
    u=-s
  }

  arg.acos=v/u^3
  if(arg.acos>1){
    arg.acos=1
  }

  w = (3.141592654+acos(arg.acos))/3
  p1d = 2*u*cos(w) - b/a/3
  p2d = p1d - dif
  var = p1d*(1-p1d)/nx + p2d*(1-p2d)/ny
  fmdiff = diff^2/var
}
return(fmdiff)
}

```

Simultaneous Score Intervals for Odds Ratio

```

OR.Score.CI <- function(x1,n1,x2,n2,conflev,t){
  px = x1/n1
  py = x2/n2

  if(((x1==0) && (x2==0)) || ((x1==n1) && (x2==n2))){
    ul = 1/0
    ll = 0
  }
}

```

```

else if((x1==0) || (x2==n2)){
  ll = 0
  theta = 0.01/n2
  ul = OR.limit(x1,n1,x2,n2,conflev,theta,t,1)
}

else if((x1==n1) || (x2==0)){
  ul = 1/0
  theta = 100*n1
  ll = OR.limit(x1,n1,x2,n2,conflev,theta,t,0)
}

else{
  theta = px/(1-px)/(py/(1-py))/1.1
  ll = OR.limit(x1,n1,x2,n2,conflev,theta,t,0)
  theta=px/(1-px)/(py/(1-py))*1.1
  ul = OR.limit(x1,n1,x2,n2,conflev,theta,t,1)
}
c(ll,ul)
}

OR.limit <- function(x,nx,y,ny,conflev,lim,t,w){
  z = (((qtukey(conflev, df=Inf, nmeans=t))^2)/2)
  px = x/nx
  score= 0
  while ( score < z){
    a = ny*(lim-1)
    b = nx*lim+ny-(x+y)*(lim-1)
    c = -(x+y)
    p2d = (-b+sqrt(b^2-4*a*c))/(2*a)
    p1d = p2d*lim/(1+p2d*(lim-1))
    score = ((nx*(px-p1d))^2)*(1/(nx*p1d*(1-p1d))+1/(ny*p2d*(1-p2d)))
    ci = lim
    if(w==0) { lim = ci/1.001 }
    else{ lim = ci*1.001 }
  }
  return(ci)
}

```

```
##### Data Analysis (Table 1) #####
```

```
success<-c(25, 22, 12, 6)
```

```
size<-c(30, 25, 20, 25)
```

```
no.groups<-length(size)
```

```
alpha<-.05
```

```
no.comp<-choose(no.groups,2)
```

```
Diff.Score.CI.Set<-OR.Score.CI.Set<-index<-matrix(c(rep(0,no.comp*4)), ncol=4)
```

```
i<-0
```

```
j<-0
```

```
r<-0
```

```
for(i in 1:(no.groups-1)){
```

```
  for(j in (i+1):no.groups){
```

```
    r<-r+1
```

```
    y1<-success[i]
```

```
    y2<-success[j]
```

```
    N1<-size[i]
```

```
    N2<-size[j]
```

```
    conflev<-1-alpha
```

```
    diff.int<-Diff.Score.CI(y1, N1, y2, N2, alpha,no.groups)
```

```
    or.int<-OR.Score.CI(y1,N1,y2,N2,conflev,no.groups)
```

```
    Diff.Score.CI.Set[r,1]<-i
```

```
    Diff.Score.CI.Set[r,2]<-j
```

```
    Diff.Score.CI.Set[r,3]<-diff.int[1]
```

```
    Diff.Score.CI.Set[r,4]<-diff.int[2]
```

```
    OR.Score.CI.Set[r,1]<-i
```

```
    OR.Score.CI.Set[r,2]<-j
```

```
    OR.Score.CI.Set[r,3]<-or.int[1]
```

```
    OR.Score.CI.Set[r,4]<-or.int[2]
```

```
  }
```

```
}
```

```
## simultaneous score confidence intervals for differences ##
```

```
dimnames(Diff.Score.CI.Set)<-list(NULL, c("First group", "Second group", "Lower bound", "Upper bound"))
```

```
Diff.Score.CI.Set
```

```
## simultaneous score confidence intervals for odds ratio ##
```

```
dimnames(OR.Score.CI.Set)<-list(NULL, c("First group", "Second group", "Lower bound", "Upper bound"))
```

```
OR.Score.CI.Set
```