##############################################

# 4 functions for McNemar's Test

##############################################

##############################################

#1. McNemar power function

#INSTRUCTIONS FOR USE:

#copy the entire function at once into R

#You then call the function simply by entering: mcnemar.power(alpha,p01,p10,n,sided)

#The function has five inputs: alpha, p01, p10, n, sided

 #alpha: statistical significance level (usually .05)

 #p01: estimated proportion of affirmative response for treatment group

 #p10: estimated proportion of affirmative response for control group

 #n: sample size

 #sided: enter 1 for 1-sided and 2 for 2-sided

#Example of how to get the very first cell entry in manuscript, Table 5: mcnemar.power(.05,.05,.10,300,1)

mcnemar.power=function(alpha,p01,p10,n,sided) {

if (sided==1) {

power=pnorm((1/(2\*sqrt((p10/(p10+p01))\*(1-(p10/(p10+p01))))))\*(qnorm(alpha)+2\*abs(p10/(p10+p01)-.5)\*sqrt(n\*(p10+p01))))

return(power)

}

if (sided==2) {

power=pnorm((1/(2\*sqrt((p10/(p10+p01))\*(1-(p10/(p10+p01))))))\*(qnorm(alpha/2)+2\*abs(p10/(p10+p01)-.5)\*sqrt(n\*(p10+p01))))

return(power)

}

}

##############################################

#2. McNemar sample size function

#INSTRUCTIONS FOR USE:

#copy the entire function at once into R

#You then call the function simply by entering: mcnemar.n(alpha,power,p01,p10,sided)

#The function has five inputs: alpha, power, p01, p10, sided

 #alpha: statistical significance level (usually .05)

 #power: desired power (usually .8 or .9)

 #p01: estimated proportion of affirmative response for treatment group

 #p10: estimated proportion of affirmative response for control group

 #sided: enter 1 for 1-sided and 2 for 2-sided

#Example of how to get the very first cell entry in manuscript, Table 4: mcnemar.n(.05,.8,.05,.10,1)

mcnemar.n=function(alpha,power,p01,p10,sided) {

if (sided==1) {

n=ceiling((qnorm(1-alpha)+2\*qnorm(power)\*sqrt((p10/(p10+p01))\*(1-(p10/(p10+p01)))))^2/(4\*((p10/(p10+p01))-.5)^2\*(p10+p01)))

return(n)

}

if (sided==2) {

n=ceiling((qnorm(1-alpha/2)+2\*qnorm(power)\*sqrt((p10/(p10+p01))\*(1-(p10/(p10+p01)))))^2/(4\*((p10/(p10+p01))-.5)^2\*(p10+p01)))

return(n)

}

}

##############################################

#3. McNemar power table for a given sample size

#INSTRUCTIONS FOR USE:

#copy the entire function at once into R

#You then call the function simply by entering: mcnemar.power.table(alpha,n,sided)

#The function has three inputs: alpha, n, sided

 #alpha: statistical significance level (usually .05)

 #n: sample size

 #sided: enter 1 for 1-sided and 2 for 2-sided

#Example of how to reproduce Table 5: mcnemar.power.table(.05,300,1)

mcnemar.power.table=function(alpha,n,sided) {

mcnemar.power=function(alpha,p01,p10,n,sided) {

if (sided==1) {

power=pnorm((1/(2\*sqrt((p10/(p10+p01))\*(1-(p10/(p10+p01))))))\*(qnorm(alpha)+2\*abs(p10/(p10+p01)-.5)\*sqrt(n\*(p10+p01))))

return(power)

}

if (sided==2) {

power=pnorm((1/(2\*sqrt((p10/(p10+p01))\*(1-(p10/(p10+p01))))))\*(qnorm(alpha/2)+2\*abs(p10/(p10+p01)-.5)\*sqrt(n\*(p10+p01))))

return(power)

}

}

p10values=seq(.10,.95,.05)

p01values=seq(.05,.45,.05)

Q=matrix(nrow=18,ncol=9)

 for(i in 1:18){

 for(j in 1:9){

 Q[i,j]=mcnemar.power(alpha,p01values[j],p10values[i],n,sided)

 if (p10values[i]+p01values[j]>1){

 Q[i,j]=NA

 }

 if (j>i){

 Q[i,j]=NA

 }

 }

 }

rownames(Q) <- p10values

colnames(Q) <- p01values

return(Q)

}

##############################################

#4. McNemar sample size table for a given power

#INSTRUCTIONS FOR USE:

#copy the entire function at once into R

#You then call the function simply by entering: mcnemar.n.table(alpha,power,sided)

#The function has three inputs: alpha, power, sided

 #alpha: statistical significance level (usually .05)

 #power: desired power (usually .8 or .9)

 #sided: enter 1 for 1-sided and 2 for 2-sided

#Example of how to reproduce Table 5: mcnemar.n.table(.05,.8,1)

mcnemar.n.table=function(alpha,power,sided) {

mcnemar.n=function(alpha,power,p01,p10,sided) {

if (sided==1) {

n=ceiling((qnorm(1-alpha)+2\*qnorm(power)\*sqrt((p10/(p10+p01))\*(1-(p10/(p10+p01)))))^2/(4\*((p10/(p10+p01))-.5)^2\*(p10+p01)))

return(n)

}

if (sided==2) {

n=ceiling((qnorm(1-alpha/2)+2\*qnorm(power)\*sqrt((p10/(p10+p01))\*(1-(p10/(p10+p01)))))^2/(4\*((p10/(p10+p01))-.5)^2\*(p10+p01)))

return(n)

}

}

p10values=seq(.10,.95,.05)

p01values=seq(.05,.45,.05)

Q=matrix(nrow=18,ncol=9)

 for(i in 1:18){

 for(j in 1:9){

 Q[i,j]=mcnemar.n(alpha,power,p01values[j],p10values[i],sided)

 if (p10values[i]+p01values[j]>1){

 Q[i,j]=NA

 }

 if (j>i){

 Q[i,j]=NA

 }

 }

 }

rownames(Q) <- p10values

colnames(Q) <- p01values

return(Q)

}

##############################################

# Stuart-Maxwell sample size function for more than 2 outcomes

##############################################

#INSTRUCTIONS FOR USE:

#copy the entire function at once into R

#You then call the function simply by entering: stuart.maxwell.n(alpha,beta,X)

#The function has three inputs: X, alpha, beta

 #alpha: statistical significance level (usually .05)

 #beta: 1-(desire power) (usually .2 or .1)

 #X: This is an rX2 matrix of probabilities, where the entry of each row for is p\_ij,p\_ji corresponding to the expected probability of the outcome for the control and treatment, respectively.

 #In other words, row 1 will contain the probability of outcome 1 for the control in column 1 and for the treatment in column 2;

 #row 2 will contain the probability of outcome 2 for the control in column 1 and for the treatment in column 2; etc.

 #Note that in R, you enter all the cells in column 1, and then all the cells in column 2 when creating a matrix. We highly recommend checking the matrix for accuracy before running the sample size function.

#Example of how to get the very first result mentioned in the Stuart-Maxwell section of the manuscript:

 #First create the matrix: X2=matrix(c(.13,.13,.13,.18,.18,.18),ncol=2)

 #Call the function: stuart.maxwell.n(.05,.2,X2)

stuart.maxwell.n=function(alpha,beta,X) {

r=nrow(X)

ncp=function(alpha,beta,r) optimize(function(x) abs(pchisq(qchisq(alpha, r, lower.tail = FALSE), r, x) - beta),lower=0,upper=100)$minimum

insidesum=function(X) {

z=matrix(nrow=r,ncol=1)

for(i in 1:nrow(X)){

z[i] = (X[i,1]-X[i,2])^2/(X[i,1]+X[i,2])

}

return(sum(z))

}

return(ceiling(ncp(alpha,beta,r)\*(1/insidesum(X))))

}

##############################################

# Cochran's Q sample size calculator for 3 or 4 treatments

##############################################

#INSTRUCTIONS FOR USE:

#copy the entire function at once into R

#You then call the function simply by entering: cochranQn(X,alpha,beta)

#The function has three inputs: X, alpha, beta

 #alpha: statistical significance level (usually .05)

 #beta: 1-(desire power) (usually .2 or .1)

 #X: This is a row vector of probabilities, corresponding to the expected probability of the outcome for each possible treatment.

 #The probabilities must be entered in a specific order. For 3 treatments, follow the example in Table 8 and 9 of the manuscript

 #For 4 treatments, use Appendix C in the manuscript to help you set up the order

 #For either 3 or 4 treatments, the important thing is to assign a treatment to one of the subscripts and then just be consistent with it throughout the table

 #always check that your vector sums to 1 by using: sum(X)

#Example of how to get result under "Example Proportion 1" in Table 9:

 #First create the row vector: X1=c(.05,.13,.08,.02,.18,.03,.01,.5)

 #Call the function: cochranQn(X1,.05,.2)

cochranQn=function(X,alpha,beta) {

chisq.rc.n=function(alpha,beta,X) {

ncp=function(alpha,beta,r) optimize(function(x) abs(pchisq(qchisq(alpha, r, lower.tail = FALSE), r, x) - beta),lower=0,upper=100)$minimum

tau=function(X) {

Qvector=function(X) {

m=nrow(X)

Q=matrix(nrow=m,ncol=1)

for(i in 1:m){

Q[i,1]=1/m

}

return(Q)

}

Avector=function(X) matrix(c(1/mean(X[,1]),1/mean(X[,2])),ncol=1)

Fmatrix=function(X) {

F=matrix(nrow=nrow(X),ncol=2)

F[,1]=X[,1]/sqrt(mean(X[,1]))

F[,2]=X[,2]/sqrt(mean(X[,2]))

return(F)

}

return(t(Qvector(X))%\*%(X^2)%\*%Avector(X)-t(Qvector(X))%\*%Fmatrix(X)%\*%t(Fmatrix(X))%\*%Qvector(X))

}

return(ceiling(ncp(alpha,beta,(nrow(X)-1)\*(ncol(X)-1))/tau(X)))

}

if (length(X)==16) {

IndTable=function(X){

Imatrix=matrix(nrow=4,ncol=2)

Imatrix[1,1]=sum(X[1],X[2],X[3],X[4],X[6],X[7],X[8],X[12])

Imatrix[1,2]=1-Imatrix[1,1]

Imatrix[2,1]=sum(X[1],X[2],X[3],X[5],X[6],X[9],X[10],X[13])

Imatrix[2,2]=1-Imatrix[2,1]

Imatrix[3,1]=sum(X[1],X[2],X[4],X[5],X[7],X[10],X[11],X[14])

Imatrix[3,2]=1-Imatrix[3,1]

Imatrix[4,1]=sum(X[1],X[3],X[4],X[5],X[8],X[9],X[11],X[15])

Imatrix[4,2]=1-Imatrix[4,1]

return(Imatrix)

}

Weight=function(X) {

contotal=function(X) {

return(sum(X[1]\*4,X[2]\*3,X[3]\*3,X[4]\*3,X[5]\*3,X[6]\*2,X[7]\*2,X[8]\*2,X[9]\*2,X[10]\*2,X[11]\*2,X[12],X[13],X[14],X[15])/4)

}

return(sum(X[2]\*3,X[3]\*3,X[4]\*3,X[5]\*3,X[6]\*4,X[7]\*4,X[8]\*4,X[9]\*4,X[10]\*4,X[11]\*4,X[12]\*3,X[13]\*3,X[14]\*3,X[15]\*3)/(12\*contotal(X)\*(1-contotal(X))))

}

}

if (length(X)==8) {

IndTable=function(X){

Imatrix=matrix(nrow=3,ncol=2)

Imatrix[1,1]=sum(X[1],X[2],X[3],X[5])

Imatrix[1,2]=1-Imatrix[1,1]

Imatrix[2,1]=sum(X[1],X[2],X[4],X[6])

Imatrix[2,2]=1-Imatrix[2,1]

Imatrix[3,1]=sum(X[1],X[3],X[4],X[7])

Imatrix[3,2]=1-Imatrix[3,1]

return(Imatrix)

}

Weight=function(X) {

contotal=function(X) {

return(sum(X[1]\*3,X[2]\*2,X[3]\*2,X[4]\*2,X[5],X[6],X[7])/3)

}

return(sum(X[2]\*2,X[3]\*2,X[4]\*2,X[5]\*2,X[6]\*2,X[7]\*2)/(6\*contotal(X)\*(1-contotal(X))))

}

}

return(ceiling(Weight(X)\*chisq.rc.n(alpha,beta,IndTable(X))))

}