####this program contains functions to calculate Sample Size using 'Corrected Method' - corrected Dupont's formula, input required is alpha, beta, pnot, psi, phi and M

CalcP1 = function(pnot,mycorr,oddsR)

{#function calculates the probability that a sampled case patient is exposed

#Input:

#pnot: probability that a sample control patient is exposed ,

#mycorr: correlation coefficient for exposure in matched pairs of case-control patients (called phi in paper),

#nmatch: number of controls per cases in the sample

#oddsR: odds ratio

#Output: myp1 is the probability that a sampled case patient is exposed

qnot = 1-pnot;

mydelta = sqrt(mycorr^2\*(oddsR-1)^2 + 4\*oddsR);

mypiece1= 2\*oddsR\*pnot\*(oddsR\*pnot+qnot);

mypiece2 = (oddsR-1)^2\*pnot\*qnot\*mycorr^2;

myp1 = 0.5\*(mypiece1 + mypiece2 - (oddsR-1)\*pnot\*qnot\*mycorr\*mydelta)/((oddsR\*pnot+qnot)^2 + mypiece2);

return(myp1)

}

ExpVar = function(pnot,myp1,oddsR,mycorr,nmatch)

{#function calculates the expectation and variance for chi-square test

#Input:

#pnot: probability that a sample control patient is exposed

#myp1 probability that a sampled case patient is exposed, would be derived from CalcP1

#mycorr: correlation coefficient for exposure in matched pairs of case-control patients (called phi in paper),

#nmatch: number of controls per cases in the sample

#oddsR: odds ratio

#Output:

#2 dimensional vector of expectation and variance

qnot=1-pnot;

myq1=1-myp1;

#p0plus is the probability that a control patient is exposed given that his matched case patient is exposed

#p0minus is the probability that a control patient is exposed given that his matched case patient is not exposed

myterm = mycorr\*sqrt(pnot\*qnot);

p0plus = pnot + myterm\*sqrt(myq1/myp1);

p0minus = pnot - myterm\*sqrt(myp1/myq1);

#print(p0plus);

#print(p0minus); these commands would print intermediate results if needed

#tVector is the vector of probabilities of observing i (vector index) exposed subjects among a case patient and his nmatch controls.

tVector = myp1\*dbinom(c(1:nmatch)-1,size=nmatch,prob=p0plus) + myq1\*dbinom(c(1:nmatch),size=nmatch,prob=p0minus);

rho1<-p0minus/p0plus

rho2<-(1-p0minus)/(1-p0plus)

rhoa=c(rho1^((1:nmatch)-1))

rhob=c(rho2^(nmatch-(1:nmatch)))

OddsVector1 = oddsR\*c(1:nmatch)/(((nmatch+1-c(1:nmatch))\*rhoa\*rhob) + ((oddsR)\*c(1:nmatch)));

OddsVector2 = oddsR\*c(1:nmatch)\*((nmatch+1-c(1:nmatch))\*rhoa)/((((nmatch+1-c(1:nmatch))\*rhoa) + ((oddsR)\*c(1:nmatch))))^2;

#myeOdds is used in the calculation of conditional mean, myvOdds is used in the calculation of conditional variance

myvOdds = sum(tVector\*OddsVector2);

myeOdds =sum(tVector\*OddsVector1);

return(c(myeOdds,myvOdds))

}

ExpVarnull = function(pnot,myp1,oddsR,mycorr,nmatch)

{#function calculates the expectation and variance for chi-square test

#Input:

#pnot: probability that a sample control patient is exposed

#myp1 probability that a sampled case patient is exposed, would be derived from CalcP1

#mycorr: correlation coefficient for exposure in matched pairs of case-control patients (called phi in paper),

#nmatch: number of controls per cases in the sample

#oddsR: odds ratio

#Output:

#2 dimensional vector of expectation and variance

qnot=1-pnot;

myq1=1-myp1;

#p0plus is the probability that a control patient is exposed given that his matched case patient is exposed

#p0minus is the probability that a control patient is exposed given that his matched case patient is not exposed

myterm = mycorr\*sqrt(pnot\*qnot);

p0plus = pnot + myterm\*sqrt(myq1/myp1);

p0minus = pnot - myterm\*sqrt(myp1/myq1);

#print(p0plus);

#print(p0minus); these commands would print intermediate results if needed

#tVector is the vector of probabilities of observing i (vector index) exposed subjects among a case patient and his nmatch controls.

tVector = myp1\*dbinom(c(1:nmatch)-1,size=nmatch,prob=p0plus) + myq1\*dbinom(c(1:nmatch),size=nmatch,prob=p0minus);

OddsVector1 = oddsR\*c(1:nmatch)/(nmatch+1);

OddsVector2 = oddsR\*c(1:nmatch)\*(nmatch+1-c(1:nmatch))/(nmatch+1)^2;

#myeOdds is used in the calculation of conditional mean, myvOdds is used in the calculation of conditional variance

myvOdds = sum(tVector\*OddsVector2);

myeOdds =sum(tVector\*OddsVector1);

return(c(myeOdds,myvOdds))

}

SSCCorrected = function(alpha,power,pnot,oddsR,mycorr,nmatch)

{#function derives the sample size for a matched case control study using 'Corrected Method'

#Input:

#power: either 0.8 or 0.9

#alpha: type 2 error (2-sided),

#pnot: probability that a sample control patient is exposed ,

#mycorr: correlation coefficient for exposure in matched pairs of case-control patients (called phi in paper),

#nmatch: number of controls per cases in the sample

#oddsR: odds ratio

#output: the sample size needed

#uses functions CalcP1 and ExpVar

#uses a chi-square test, all formulas from William Dupont (1988)

###derive myp1 from pnot, oddsR and mycorr (Appendix of paper)

##myp1 is the probability that a sampled case patient is exposed

myp1=CalcP1(pnot,mycorr,oddsR);

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

myout= ExpVar(pnot,myp1,oddsR,mycorr,nmatch)

myeOdds = myout[1];

myvOdds=myout[2];

myeOdds

#p1=p0 when oddsR=1 under the null

myout1= ExpVarnull(pnot,myp1,1,mycorr,nmatch)

mye1 = myout1[1];

myv1 = myout1[2];

myzbeta = qnorm(1-power,mean=0,sd=1,lower.tail=FALSE);

myzalpha = qnorm(alpha/2,mean=0,sd=1,lower.tail=FALSE);

mynum= (myzbeta\*sqrt(myvOdds) + myzalpha\*sqrt(myv1))^2; # numerator

myden= (mye1 - myeOdds)^2;#denominator

myN = mynum/myden;

return(myN);

}

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

####this program contains functions to calculate Sample Size using Dupont's (1988) original formula, input required is alpha, beta, pnot, psi, phi and M

CalcP1 = function(pnot,mycorr,oddsR)

{#function calculates the probability that a sampled case patient is exposed

#Input:

#pnot: probability that a sample control patient is exposed ,

#mycorr: correlation coefficient for exposure in matched pairs of case-control patients (called phi in paper),

#nmatch: number of controls per cases in the sample

#oddsR: odds ratio

#Output: myp1 is the probability that a sampled case patient is exposed

#last modified: August 29th 2016

qnot = 1-pnot;

mydelta = sqrt(mycorr^2\*(oddsR-1)^2 + 4\*oddsR);

mypiece1= 2\*oddsR\*pnot\*(oddsR\*pnot+qnot);

mypiece2 = (oddsR-1)^2\*pnot\*qnot\*mycorr^2;

myp1 = 0.5\*(mypiece1 + mypiece2 - (oddsR-1)\*pnot\*qnot\*mycorr\*mydelta)/((oddsR\*pnot+qnot)^2 + mypiece2);

return(myp1)

}

ExpVarD = function(pnot,myp1,oddsR,mycorr,nmatch)

{#function calculates the expectation and variance for chi-square test

#Input:

#pnot: probability that a sample control patient is exposed

#myp1 probability that a sampled case patient is exposed, would be derived from CalcP1

#mycorr: correlation coefficient for exposure in matched pairs of case-control patients (called phi in paper),

#nmatch: number of controls per cases in the sample

#oddsR: odds ratio

#Output:

#2 dimensional vector of expectation and variance

qnot=1-pnot;

myq1=1-myp1;

#p0plus is the probability that a control patient is exposed given that his matched case patient is exposed

#p0minus is the probability that a control patient is exposed given that his matched case patient is not exposed

myterm = mycorr\*sqrt(pnot\*qnot);

p0plus = pnot + myterm\*sqrt(myq1/myp1);

p0minus = pnot - myterm\*sqrt(myp1/myq1);

#print(p0plus);

#print(p0minus); these commands would print intermediate results if needed

#tVector is the vector of probabilities of observing i (vector index) exposed subjects among a case patient and his nmatch controls.

tVector = myp1\*dbinom(c(1:nmatch)-1,size=nmatch,prob=p0plus) + myq1\*dbinom(c(1:nmatch),size=nmatch,prob=p0minus);

OddsVector1 = oddsR\*c(1:nmatch)/(((nmatch+1-c(1:nmatch))) + ((oddsR)\*c(1:nmatch)));

OddsVector2 = oddsR\*c(1:nmatch)\*((nmatch+1-c(1:nmatch)))/((((nmatch+1-c(1:nmatch))) + ((oddsR)\*c(1:nmatch))))^2;

#myeOdds is used in the calculation of conditional mean, myvOdds is used in the calculation of conditional variance

myvOdds = sum(tVector\*OddsVector2);

myeOdds =sum(tVector\*OddsVector1);

return(c(myeOdds,myvOdds))

}

ExpVarnull = function(pnot,myp1,oddsR,mycorr,nmatch)

{#function calculates the expectation and variance for chi-square test

#Input:

#pnot: probability that a sample control patient is exposed

#myp1 probability that a sampled case patient is exposed, would be derived from CalcP1

#mycorr: correlation coefficient for exposure in matched pairs of case-control patients (called phi in paper),

#nmatch: number of controls per cases in the sample

#oddsR: odds ratio

#Output:

#2 dimensional vector of expectation and variance

qnot=1-pnot;

myq1=1-myp1;

#p0plus is the probability that a control patient is exposed given that his matched case patient is exposed

#p0minus is the probability that a control patient is exposed given that his matched case patient is not exposed

myterm = mycorr\*sqrt(pnot\*qnot);

p0plus = pnot + myterm\*sqrt(myq1/myp1);

p0minus = pnot - myterm\*sqrt(myp1/myq1);

#print(p0plus);

#print(p0minus); these commands would print intermediate results if needed

#tVector is the vector of probabilities of observing i (vector index) exposed subjects among a case patient and his nmatch controls.

tVector = myp1\*dbinom(c(1:nmatch)-1,size=nmatch,prob=p0plus) + myq1\*dbinom(c(1:nmatch),size=nmatch,prob=p0minus);

OddsVector1 = oddsR\*c(1:nmatch)/(nmatch+1);

OddsVector2 = oddsR\*c(1:nmatch)\*(nmatch+1-c(1:nmatch))/(nmatch+1)^2;

#myeOdds is used in the calculation of conditional mean, myvOdds is used in the calculation of conditional variance

myvOdds = sum(tVector\*OddsVector2);

myeOdds =sum(tVector\*OddsVector1);

return(c(myeOdds,myvOdds))

}

SSCDupont = function(alpha,power,pnot,oddsR,mycorr,nmatch)

{#function derives the sample size for a matched case control study using Dupont's original method (1988)

#Input:

#power: either 0.8 or 0.9

#alpha: type 2 error (2-sided),

#pnot: probability that a sample control patient is exposed ,

#mycorr: correlation coefficient for exposure in matched pairs of case-control patients (called phi in paper),

#nmatch: number of controls per cases in the sample

#oddsR: odds ratio

#output: the sample size needed

#uses functions CalcP1 and ExpVar

#uses a chi-square test, all formulas from William Dupont (1988)

###derive myp1 from pnot, oddsR and mycorr (Appendix of paper)

##myp1 is the probability that a sampled case patient is exposed

myp1=CalcP1(pnot,mycorr,oddsR);

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

myout= ExpVarD(pnot,myp1,oddsR,mycorr,nmatch)

myeOdds = myout[1];

myvOdds=myout[2];

myeOdds

#p1=p0 when oddsR=1 under the null

myout1= ExpVarnull(pnot,myp1,1,mycorr,nmatch)

mye1 = myout1[1];

myv1 = myout1[2];

myzbeta = qnorm(1-power,mean=0,sd=1,lower.tail=FALSE);

myzalpha = qnorm(alpha/2,mean=0,sd=1,lower.tail=FALSE);

mynum= (myzbeta\*sqrt(myvOdds) + myzalpha\*sqrt(myv1))^2; # numerator

myden= (mye1 - myeOdds)^2;#denominator

myN = mynum/myden;

return(myN);

}

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

####this program contains functions to calculate Sample Size using 'New Approximation' method - approximating SS for 1:M study based on SS estimate from Dupont's formula for 1:1 study, input required is alpha, beta, pnot, psi, phi and M

source('SSCDupont.R')

#SSCNewApprox clculates sample size using the 'New Approximation', improving upon Dupont's SS estimates from M > 1'

SSCNewApprox = function(alpha,power,pnot,oddsR,mycorr,nmatch) {

#myN.Dupont clculates sample size using Dupont's (1988) original formula

myN.Dupont = ceiling(SSCDupont(alpha,power,pnot,oddsR,mycorr,1))

myN.Approx = ceiling (((nmatch+1)/(2\*nmatch))\*myN.Dupont)

return(myN.Approx)

}

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

####this program contains functions to generate simulated data using input myN,myM,myp0,myphi,mypsi

###Derive p0+ and p0-

CondPNot =function(myp0,myphi,mypsi)

{myp1 = CalcP1(myp0,myphi,mypsi);

myp0Plus = myp0+ myphi\*sqrt((1-myp1)\*(1-myp0)\*myp0/myp1);

myp0Minus = myp0 - myphi\*sqrt(myp1\*(1-myp0)\*myp0/(1-myp1));

myout=c(myp1,myp0Plus,myp0Minus);

return(myout)

}

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

#Simulation function (for one simulation)

myrunsim = function(myN,myM,myp1,myp0Plus,myp0Minus,myseed)

{ #function simulates one value of Y for simulated myN cases, each with myM controls

#parmaters: myN number of cases, myM number of matched controls, myp1 is exposure rate of cases,

#myp0Plus is probability of exposure of controls conditioned on the case being exposed,

#myp0Minus is probability of exposure of controls conditioned on the case being unexposed,

#myseedV is seed vector with seed for each simultation

set.seed(myseed[1])

#simulate myN cases

myC= rbinom(myN,1,myp1);

ExpCases = length(myC[myC==1]);

NExpCases = myN-ExpCases;

#simulate M controls for each exposed case

myCtr = matrix(rep(0,myM\*myN),nrow=myN,ncol=myM);

for(i in 1:myM)

{j=myseed[i+1];

jj=j+10;

set.seed(j);

myCtr[myC==1,i]=rbinom(ExpCases,1,myp0Plus);

set.seed(jj);

myCtr[myC==0,i]=rbinom(NExpCases,1,myp0Minus)};

myn1j = rowSums(myCtr);

#number of exposed from all controls

#y = number of matched set with case exposed and m-1 control exposed for m=1 to M

#for M=3, 0 exposed, 1 exposed and 2 exposed

myY=sum(myn1j[myC==1] %in% c(0:myM-1));

myT = matrix(nrow=myM,ncol=1);

for(i in 1:myM)

{

myT[i] = sum(myn1j[myC==1]==i-1) + sum(myn1j[myC==0]==i);

#number of matched sets with case exposed and i-1 controls exposed or case unexposed and i controls exposed

}

myout = c(myY,myT);

return(myout)

}

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

#Conditional mean and variance

myMeanVar = function(myM,myTm,mypsi)

{#Tm is a vector of size myM

mydenom = c(1:myM) \* mypsi+rep(myM,myM)-c(1:myM)+rep(1,myM);

mynum1 = c(1:myM)\*myTm \*mypsi;

mynum2=c(1:myM)\*mypsi\*myTm\*(rep(myM,myM)-c(1:myM)+rep(1,myM));

mymean = sum(mynum1/mydenom);

myvar = sum(mynum2/(mydenom)^2);

myout=c(mymean,myvar)

return(myout)

}

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

#This program takes an input data of parameter values and outputs a data with true SS, and corresponding SS estimates from all SS formulae#

install.packages('doParallel')

library('doParallel')

#change to directory where output data and code are kept

setwd("")

#Source codes which contain functions for calculating sample sizes using formulae

source('SSCDupont.R')

source('SSCCorrected.R')

source('SSCNewApprox.R')

source('SimulateData.R')

#Create a data of parameter values for which true sample size is desired

#number of matched controls

myM=c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15)

#oddsratior

mypsi=c(2)

#exposure prevalence in controls

myp0=c(0.05,0.1,0.15,0.2,0.25,0.3,0.35,0.4,0.45,0.5)

#correlation of exposure prevalences in controls and cases

myphi=c(0.1,0.2,0.3,0.4,0.5)

#create a dataset of all combinations of specified parameter values

Mpsi<-merge(myM,mypsi)

p0phi<-merge(myp0,myphi)

names(Mpsi)<-c("myM","mypsi")

names(p0phi)<-c("myp0","myphi")

Mpsip0phi<-merge(Mpsi,p0phi)

Mpsip0phi<-Mpsip0phi[order(Mpsip0phi$myp0),]

Mpsip0phi$myPower=rep(0,nrow(Mpsip0phi))

Mpsip0phi$myN=rep(0,nrow(Mpsip0phi))

Mpsip0phi$myN.Approx=rep(0,nrow(Mpsip0phi))

Mpsip0phi$myN.Dupont=rep(0,nrow(Mpsip0phi))

Mpsip0phi$myN.corrform=rep(0,nrow(Mpsip0phi))

nasim<-rep(0,nrow(Mpsip0phi))

#perform simultations

nsim=5000;

#Use parallel programming, when running in cluster

cl<-makeCluster(16)

registerDoParallel(cl)

#getDoParWorkers()

alt2<-foreach (j=1:nrow(Mpsip0phi), .combine=rbind) %dopar% {

myPower=0

myM=Mpsip0phi$myM[j]

mypsi=Mpsip0phi$mypsi[j]

myphi=Mpsip0phi$myphi[j]

myp0=Mpsip0phi$myp0[j]

myp1=CalcP1(myp0,myphi,mypsi)

###Derive p0+ and p0-

myout0 = CondPNot(myp0,myphi,mypsi)

myp0Plus = myout0[2];

myp0Minus=myout0[3];

#myN.Dupont clculates sample size using Dupont's (1988) original formula

myN.Dupont = ceiling(SSCDupont(0.05,0.8,myp0,mypsi,myphi,myM))

#myN.Approx clculates sample size using the New Approximation, improving upon Duponts SS estimates from M > 1

myN.Approx = ceiling(SSCNewApprox(0.05,0.8,myp0,mypsi,myphi,myM))

#myN.corrform clculates sample size using the Corrected method, improving upon Duponts SS estimates from M > 1

myN.corrform = ceiling(SSCCorrected(0.05,0.8,myp0,mypsi,myphi,myM))

myNnew=round(myN.Dupont)

while (myPower < 0.8) {

#hold space

myN=myNnew

SimY = matrix(nrow=nsim,ncol=1);

SimT = matrix(nrow=nsim,ncol=myM);

myMeanVarVect = matrix(nrow=nsim,ncol=2);

for(i in 1:nsim)

{ #myseed=rep(i,4)+c(1,101,202,303);

myseed=rep(i,myM+1)+100\*c(0:myM+1)+c(0:myM+1);

myout= myrunsim(myN,myM,myp1,myp0Plus,myp0Minus,myseed)

###Derive p0+ and p0-

myout0 = CondPNot(myp0,myphi,mypsi)

myp0Plus = myout0[2];

myp0Minus=myout0[3];

SimY[i] = myout[1];

SimT[i,] = myout[1:myM+1];

myMeanVarVect[i,]= myMeanVar(myM,SimT[i,],1); #mean and variance under the null

rm(myout);

rm(myseed)

}

StandY = (SimY-myMeanVarVect[,1])/sqrt(myMeanVarVect[,2]);

myPower = (sum(StandY>1.96,na.rm=T) + sum(StandY < -1.96,na.rm=T))/length(StandY[which(!is.na(StandY))]);

if (myPower >= 0.4 & myPower < 0.5 ) myNnew=myN+10

else if (myPower >= 0.5 & myPower < 0.6 ) myNnew=myN+5

else if (myPower >= 0.6) myNnew=myN+2

else myNnew=myN+20

}

nasim<-nsim-length(StandY[which(!is.na(StandY))])

myPower=myPower

myN=myN

c(myM,mypsi,myphi,myp0,myPower,myN,myN.Dupont,myN.Approx,myN.corrform)

}

Mpsip0phi.out<-as.data.frame(alt2)

names(Mpsip0phi.out)<-c("myM","mypsi","myphi","myp0","myPower","myN","myN.Dupont","myN.Approx","myN.corrform")

#output a datset with true sample size and sample size estimated using Dupont's (1988) formula, 'New Approximation' and 'Corrected Method'

write.csv(Mpsip0phi.out,"Mpsip0phiout\_trueSS.csv")