```r
# ----------------------------------------------------------------------------------------------------------------------
# Program: twoSATj.R
#  Author: Hermine Maes
#    Date: 10 22 2018
#
# Twin Bivariate Saturated model to estimate means & variances, thresholds & correlations across multiple groups
# Matrix style model - Raw data - Joint Continuous Ordinal data
# -------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|

# Load Libraries & Options
rm(list=ls())
library(OpenMx)
library(psych); library(polycor)
source("miFunctions.R")

# Create Output
filename    <- "twoSATj"
sink(paste(filename,".Ro",sep=""), append=FALSE, split=TRUE)

# ----------------------------------------------------------------------------------------------------------------------
# PREPARE DATA

# Load Data
data(twinData)
dim(twinData)
describe(twinData[,1:12], skew=F)
twinData[,'ht1'] <- twinData[,'ht1']*10
twinData[,'ht2'] <- twinData[,'ht2']*10
twinData[,'wt1'] <- twinData[,'wt1']/10
twinData[,'wt2'] <- twinData[,'wt2']/10

# Select Continuous Variables
vars      <- c('ht')                   # list of continuous variables names
nvc       <- 1                         # number of continuous variables
ntvc      <- nvc*2                     # number of total continuous variables
conVars   <- paste(vars,c(rep(1,nvc),rep(2,nvc)),sep="")

# Select Ordinal Variables
nth       <- 3                         # number of thresholds
vars      <- c('wt')                   # list of ordinal variables names
nvo       <- 1                         # number of ordinal variables
ntvo      <- nvo*2                     # number of total ordinal variables
ordVars   <- paste(vars,c(rep(1,nvo),rep(2,nvo)),sep="")
ordData   <- twinData
wtquant   <- quantile(ordData[,c('wt1','wt2')],(0:(nth+1))/(nth+1),na.rm=TRUE)
 for (i in c('wt1','wt2')) { ordData[,i] <- cut(ordData[,i], breaks=wtquant, labels=c(0:nth)) }

# Select Variables for Analysis
vars      <- c('ht','wt')              # list of variables names
nv        <- nvc+nvo                   # number of variables
ntv       <- nv*2                      # number of total variables
selVars   <- paste(vars,c(rep(1,nv),rep(2,nv)),sep="")

# Select Data for Analysis
mzData    <- subset(ordData, zyg==1, selVars)
dzData    <- subset(ordData, zyg==3, selVars)
mzDataF   <- cbind(mzData[,conVars],mxFactor( x=mzData[,ordVars], levels=c(0:nth)) )
dzDataF   <- cbind(dzData[,conVars],mxFactor( x=dzData[,ordVars], levels=c(0:nth)) )

# Generate Descriptive Statistics
apply(mzData,2,myMean)
apply(dzData,2,myMean)
sapply(mzData[,ordVars],table)
sapply(dzData[,ordVars],table)
hetcor(mzData)$cor
hetcor(dzData)$cor

# Set Starting Values
frMV      <- c(TRUE,FALSE)             # free status for variables
frCvD     <- valDiagLU(frMV,T,T,ntv)   # free status for diagonal, lower & upper elements of covariance matrix
frCv      <- matrix(as.logical(frCvD),4)
svMe      <- c(15,0)                   # start value for means
svVa      <- c(.5,1)                   # start value for variances
lbVa      <- .0001                     # lower bound for variances
svLTh     <- 0                         # start value for first threshold
svITh     <- 1                         # start value for increments
svTh      <- matrix(rep(c(svLTh,(rep(svITh,nth-1)))),nrow=nth,ncol=nv)     # start value for thresholds
lbTh      <- matrix(rep(c(-3,(rep(0.001,nth-1))),nv),nrow=nth,ncol=nv)     # lower bound for thresholds

# Create Labels
labMeMZ   <- labVars("meanMZ",selVars)
labMeDZ   <- labVars("meanDZ",selVars)
```

```
labMeZ    <- labVars("meanZ",selVars)
labCvMZ   <- labLower("covMZ",ntv)
labCvDZ   <- labLower("covDZ",ntv)
labCvZ    <- labLower("covZ",ntv)
labVaMZ   <- labDiag("covMZ",ntv)
labVaDZ   <- labDiag("covDZ",ntv)
labVaZ    <- labDiag("covZ",ntv)
labThMZ   <- labTh("MZ",ordVars,nth)
labThDZ   <- labTh("DZ",ordVars,nth)
labThZ    <- labTh("Z",ordVars,nth)

# ----------------------------------------------------------------------------------------------------------------------
# PREPARE MODEL

# Create Algebra for expected Mean & Threshold Matrices
meanMZ    <- mxMatrix( type="Full", nrow=1, ncol=ntv, free=frMV, values=svMe, labels=labMeMZ, name="meanMZ" )
meanDZ    <- mxMatrix( type="Full", nrow=1, ncol=ntv, free=frMV, values=svMe, labels=labMeDZ, name="meanDZ" )
thinMZ    <- mxMatrix( type="Full", nrow=nth, ncol=ntvo, free=TRUE, values=svTh, lbound=lbTh, labels=labThMZ, name="thinMZ" )
thinDZ    <- mxMatrix( type="Full", nrow=nth, ncol=ntvo, free=TRUE, values=svTh, lbound=lbTh, labels=labThDZ, name="thinDZ" )
inc       <- mxMatrix( type="Lower", nrow=nth, ncol=nth, free=FALSE, values=1, name="inc" )
threMZ    <- mxAlgebra( expression= inc %*% thinMZ, name="threMZ" )
threDZ    <- mxAlgebra( expression= inc %*% thinDZ, name="threDZ" )

# Create Algebra for expected Covariance Matrices
covMZ     <- mxMatrix( type="Symm", nrow=ntv, ncol=ntv, free=frCv, values=valDiag(svVa,ntv), lbound=valDiag(lbVa,ntv),
 labels=labCvMZ, name="covMZ" )
covDZ     <- mxMatrix( type="Symm", nrow=ntv, ncol=ntv, free=frCv, values=valDiag(svVa,ntv), lbound=valDiag(lbVa,ntv),
 labels=labCvDZ, name="covDZ" )

# Create Data Objects for Multiple Groups
dataMZ    <- mxData( observed=mzDataF, type="raw" )
dataDZ    <- mxData( observed=dzDataF, type="raw" )

# Create Expectation Objects for Multiple Groups
expMZ     <- mxExpectationNormal( covariance="covMZ", means="meanMZ", dimnames=selVars, thresholds="threMZ", threshnames=ordVars )
expDZ     <- mxExpectationNormal( covariance="covDZ", means="meanDZ", dimnames=selVars, thresholds="threDZ", threshnames=ordVars )
funML     <- mxFitFunctionML()

# Create Model Objects for Multiple Groups
modelMZ   <- mxModel( meanMZ, covMZ, thinMZ, inc, threMZ, dataMZ, expMZ, funML, name="MZ" )
modelDZ   <- mxModel( meanDZ, covDZ, thinDZ, inc, threDZ, dataDZ, expDZ, funML, name="DZ" )
multi     <- mxFitFunctionMultigroup( c("MZ","DZ") )

# Create Confidence Interval Objects
ciCor     <- mxCI( c('MZ.covMZ','DZ.covDZ' ))
ciThre    <- mxCI( c('MZ.threMZ','DZ.threDZ' ))

# Build Saturated Model with Confidence Intervals
modelSAT  <- mxModel( "twoSATj", modelMZ, modelDZ, multi, ciCor, ciThre )

# ----------------------------------------------------------------------------------------------------------------------
# RUN MODEL

# Run Saturated Model
fitSAT    <- mxRun( modelSAT, intervals=F )
sumSAT    <- summary( fitSAT )

# Print Goodness-of-fit Statistics & Parameter Estimates
fitGofs(fitSAT)
fitEsts(fitSAT)
mxGetExpected( fitSAT, c("means","thresholds","covariance"))

# ----------------------------------------------------------------------------------------------------------------------
# RUN SUBMODELS

# Constrain expected Thresholds to be equal across Twin Order
modelEMTVO  <- mxModel( fitSAT, name="twoEMTVOj" )
svMe <- c(15,0); svVa <- c(.5,1); svLTh <- 0; svITh <- 1;
for (i in 1:nv) {
modelEMTVO <- omxSetParameters( modelEMTVO, label=c(labMeMZ[nv+i],labMeMZ[i]), free=frMV[i], values=svMe[i],
 newlabels=labMeMZ[i] )
modelEMTVO <- omxSetParameters( modelEMTVO, label=c(labMeDZ[nv+i],labMeDZ[i]), free=frMV[i], values=svMe[i],
 newlabels=labMeDZ[i] )
modelEMTVO <- omxSetParameters( modelEMTVO, label=c(labVaMZ[nv+i],labVaMZ[i]), free=frMV[i], values=svVa[i],
 newlabels=labVaMZ[i] )
modelEMTVO <- omxSetParameters( modelEMTVO, label=c(labVaDZ[nv+i],labVaDZ[i]), free=frMV[i], values=svVa[i],
 newlabels=labVaDZ[i] ) }
modelEMTVO <- omxSetParameters( modelEMTVO, label=c(labThMZ[nvo*nth+1],labThMZ[1]), free=TRUE, values=svLTh,
 newlabels=labThMZ[1] )
modelEMTVO <- omxSetParameters( modelEMTVO, label=c(labThDZ[nvo*nth+1],labThDZ[1]), free=TRUE, values=svLTh,
 newlabels=labThDZ[1] )
for (i in 2:nth) {for (j in seq(i,nvo*nth,nth)) {
```

```
modelEMTVO <- omxSetParameters( modelEMTVO, label=c(labThMZ[nvo*nth+j],labThMZ[j]), free=TRUE, values=svITh,
 newlabels=labThMZ[j] )
modelEMTVO <- omxSetParameters( modelEMTVO, label=c(labThDZ[nvo*nth+j],labThDZ[j]), free=TRUE, values=svITh,
 newlabels=labThDZ[j] ) }}
fitEMTVO   <- mxRun( modelEMTVO, intervals=F )
fitGofs(fitEMTVO); fitEsts(fitEMTVO)

# Constrain expected Thresholds to be equal across Twin Order and Zygosity
modelEMTVZ <- mxModel( fitEMTVO, name="twoEMTVZj" )
for (i in 1:nv) {
modelEMTVZ <- omxSetParameters( modelEMTVZ, label=c(labMeMZ[i],labMeDZ[i]), free=frMV[i], values=svMe[i], newlabels=labMeZ[i] )
modelEMTVZ <- omxSetParameters( modelEMTVZ, label=c(labVaMZ[i],labVaDZ[i]), free=frMV[i], values=svVa[i], newlabels=labVaZ[i] ) }
modelEMTVZ <- omxSetParameters( modelEMTVZ, label=c(labThMZ[1],labThDZ[1]), free=TRUE, values=svLTh, newlabels=labThZ[1] )
for (i in 2:nth) {for (j in seq(i,nvo*nth,nth)) {
modelEMTVZ <- omxSetParameters( modelEMTVZ, label=c(labThMZ[j],labThDZ[j]), free=TRUE, values=svITh, newlabels=labThZ[j] ) }}
fitEMTVZ   <- mxRun( modelEMTVZ, intervals=F )
fitGofs(fitEMTVZ); fitEsts(fitEMTVZ)

# Print Comparative Fit Statistics
mxCompare( fitSAT, subs <- list(fitEMTVO, fitEMTVZ) )

# ----------------------------------------------------------------------------------------------------------------------
sink()
save.image(paste(filename,".Ri",sep=""))
```