Multivariate twin models:
Independent Pathway Model (IPM) and the Common Pathway Model (CPM)

Student version

If you repeat this at home, software you need on your computer

R (version: R 4.3.0; https://cran.r-project.org)
R Studio (version: 2022.02.2+485; https://www.rstudio.com/products/rstudio/download/)

R library OpenMx (version: 2.20; https://www.rstudio.com/products/rstudio/download/)
R library psych
R library ggplot2

Datasets: skintwindata, ntwindata

Conor Dolan & Dirk Pelt
(c.v.dolan@vu.nl, h.m.pelt@vu.nl)

VU VRIJE UNIVERSITEIT AMSTERDAM | Faculty of Behavioural and Movement Sciences

Nederlands Tweelingen Register

This document is the "practical workbook" of the 3-hour practical that accompanies the 5 PPT presentations on multivariate twin modelling, based on the Independent Pathway Model (IPM) and the Common Pathway Model (CPM). In this practical, we will analyze the following two data sets:

data set	
skintwindata	skinfold data, skinfold measured at 4 locations
ntwindata	Item scores of 4 (5 point scale) items designed to measure neuroticism (a personality trait)

The *ntwindata* and the *skintwindata* featured in the 5 PPTs. The present aim is to reproduce the results in the PPTs ourselves using the OpenMx library in R Studio. In the first part of this practical, we analyze the *skintwindata*. Before the practical, we briefly go over multivariate modelling and the IPM and CPM.

Here is an overview of what we hope to do in part 1 of this practical

1) Read the skinfold data (skintwindata) into R studio, and obtain basic descriptives in the MZ and the DZ samples
2) Calculate the MZ and DZ twin correlations and use Falconer's equations to obtain a rough guess of ACE or ADE standardized variance components
3) Consider the relationship between the covariance matrix and the correlation matrix

In the PPT presentation, we first fitted a bivariate model of skinfold measured at the biceps and triceps. Here we will proceed straight away to the 4 variate model.

4) Calculate the MZ and DZ covariance matrices and means
5) Calculate the MZ and DZ covariance matrices in OpenMx with sex as a covariate
6) Fit the 4 variate ADE twin model and consider the results

In this document, all text rendered in red font `Courier New` is R code input, e.g.,

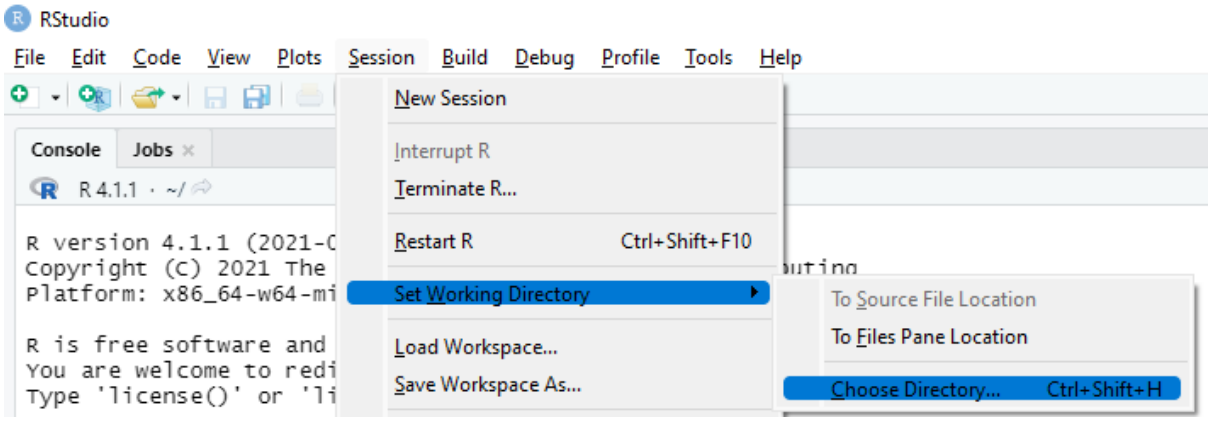
```
#
print(c("Hallo World"))
#
```

In this document, all text rendered in blue font `Courier New` is R code output, e.g.,

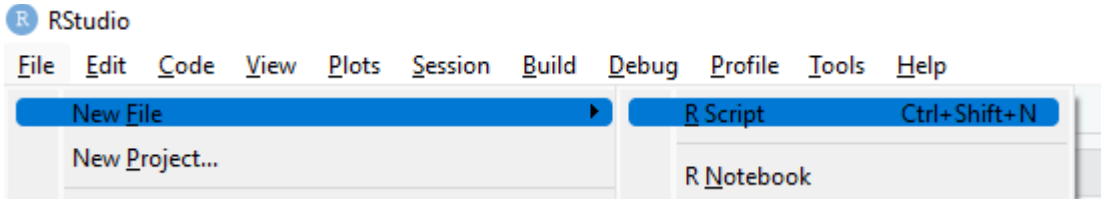
```
> print(c("Hallo World"))
[1] "Hallo World
```

How to use the workbook.

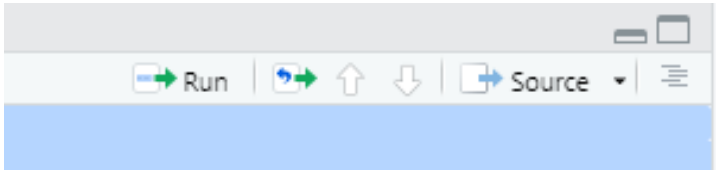
1) save the datasets (skintwindata and ntwindata) to your working directory
2) set the "working directory" to the directory that contains the data sets (skintwindata and ntwindata). In R study, you can set the working directory by

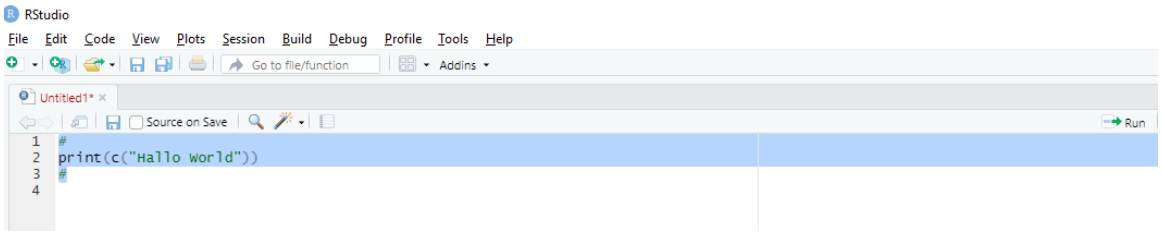


3) Open a script window in R studio



4) Cut and paste the R code from this docx to the R script window, select and execute by clicking on →Run in the top right corner of the R script window.

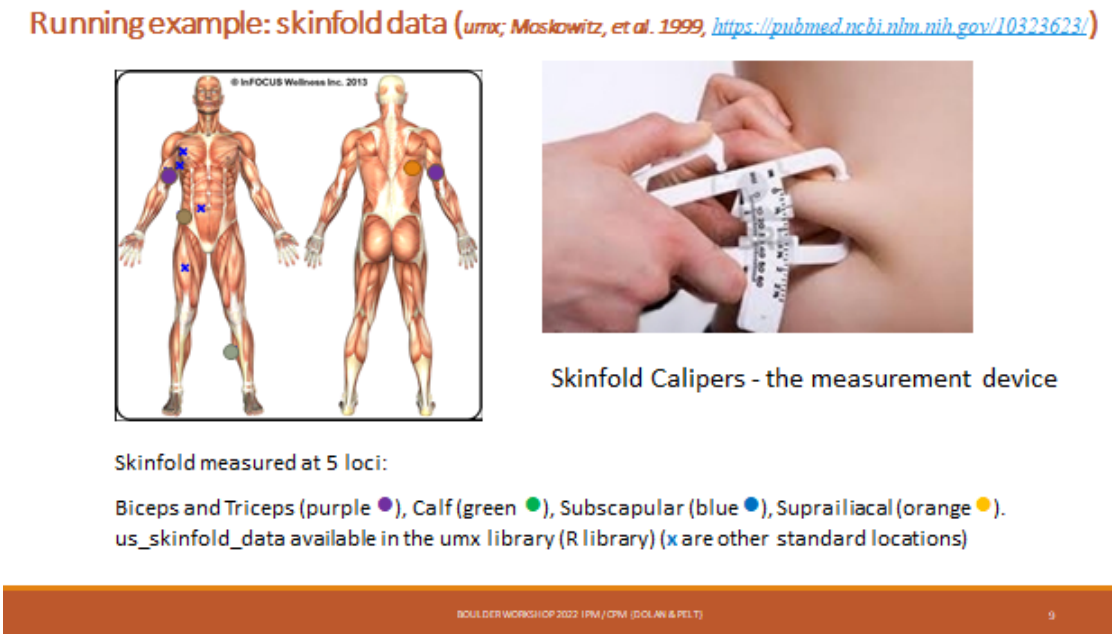




The R code that is to be cut-and-pasted is shown between #-> and #<-.

Practical part 1: Skinfold data. The 4-variate ADE twin model. 15 questions

Skinfold measured at 4 locations in 214 MZ and 181 DZ twin pairs.



Read the skinfold data (skintwindata) into R studio, and get the basic descriptives in the MZ and the DZ samples.

```
# ->
rm(list=ls(all=TRUE))   # clear the memory
library(psych)
library(OpenMx)
# mxOption(NULL, "Default optimizer","NPSOL")
mxOption(NULL, "Default optimizer","CSOLNP")
skinfold=read.table(file='skintwindata')
colnames(skinfold)
# <-
```

Check out the variable names:

```
> colnames(skinfold)
 [1] "fan"    "zyg"    "bmi_T1" "bic_T1" "ssc_T1" "sil_T1" "tri_T1"
 [8] "bmi_T2" "bic_T2" "ssc_T2" "sil_T2" "tri_T2" "sex_T1" "sex_T2"
[15] "zyg2"
```

We require these following variables in this practical (see PPT slide 9 shown above):

"bic_T1", "ssc_T1", "sil_T1", "tri_T1"	skinfold measures twin 1 biceps, suprailiacal, subscapular, triceps
"bic_T2" "ssc_T2" "sil_T2" "tri_T2"	skinfold measures twin 2 biceps, suprailiacal, subscapular, triceps
"sex_T1" "sex_T2"	sex coded 0 (males) 1 (females)
"zyg2"	zygosity coded 1 (MZ) 2 (DZ)

```
# mz dz data frames ->
selvars=c(
"bic_T1","ssc_T1","sil_T1","tri_T1",
"bic_T2","ssc_T2","sil_T2","tri_T2",
"sex_T1","sex_T2","zyg2")
# create the MZ and DZ data frames
mzskinfold=skinfold[skinfold$zyg2==1,selvars]
dzskinfold=skinfold[skinfold$zyg2==2,selvars]
# use the psych function describe () to get descriptives
describe(mzskinfold, skew=F, range=F)
describe(dzskinfold, skew=F, range=F)
# <-
```

Results descriptives:

```
> describe(mzskinfold, skew=F, range=F)
       vars   n  mean   sd   se
bic_T1    1 214 17.54 3.55 0.24
ssc_T1    2 214 20.64 4.59 0.31
sil_T1    3 214 19.31 5.43 0.37
tri_T1    4 214 22.55 3.27 0.22
bic_T2    5 213 17.08 3.44 0.24
ssc_T2    6 214 20.36 4.71 0.32
sil_T2    7 214 18.56 5.11 0.35
tri_T2    8 213 22.15 3.23 0.22
sex_T1    9 214  0.50 0.50 0.03
sex_T2   10 214  0.50 0.50 0.03
zyg2     11 214  1.00 0.00 0.00
> describe(dzskinfold, skew=F, range=F)
       vars   n  mean   sd   se
bic_T1    1 181 17.60 3.92 0.29
ssc_T1    2 181 20.73 5.03 0.37
sil_T1    3 181 19.57 5.68 0.42
tri_T1    4 181 22.70 3.72 0.28
bic_T2    5 181 18.55 3.64 0.27
ssc_T2    6 180 22.25 5.08 0.38
sil_T2    7 180 20.73 5.65 0.42
tri_T2    8 181 23.41 3.29 0.24
sex_T1    9 181  0.33 0.47 0.04
sex_T2   10 181  0.76 0.43 0.03
zyg2     11 181  2.00 0.00 0.00
```

Note that the phenotypic means are between (about) 17 and (about) 23. Let's get the covariance and correlation matrices in the two samples.

```
# ->
selvars =
c("bic_T1","tri_T1","ssc_T1","sil_T1","bic_T2","tri_T2","ssc_T2","sil_T2")
# the array with the variable names
Smz4=cov(mzskinfold[,selvars], use='complete')
Rmz4=cor(mzskinfold[,selvars], use='complete')
Sdz4=cov(dzskinfold[,selvars], use='complete')
Rdz4=cor(dzskinfold[,selvars], use='complete')
round(Smz4,3)
round(Rmz4,3)
round(Sdz4,3)
round(Rdz4,3)
# <-
```

Results covariance and correlation matrices:

```
> round(Smz4,3)
       bic_T1 tri_T1 ssc_T1 sil_T1 bic_T2 tri_T2 ssc_T2 sil_T2
bic_T1 12.478 10.379 13.047 16.254  9.571  8.586 10.661 12.264
tri_T1 10.379 10.604 11.930 14.599  8.103  8.503  9.627 11.096
ssc_T1 13.047 11.930 20.500 22.392 10.823  9.954 16.650 17.481
sil_T1 16.254 14.599 22.392 28.667 13.539 12.549 18.717 22.471
bic_T2  9.571  8.103 10.823 13.539 11.845  9.894 13.398 15.001
tri_T2  8.586  8.503  9.954 12.549  9.894 10.447 12.237 13.765
ssc_T2 10.661  9.627 16.650 18.717 13.398 12.237 22.081 21.755
sil_T2 12.264 11.096 17.481 22.471 15.001 13.765 21.755 26.022
> round(Rmz4,3)
       bic_T1 tri_T1 ssc_T1 sil_T1 bic_T2 tri_T2 ssc_T2 sil_T2
bic_T1  1.000  0.902  0.816  0.859  0.787  0.752  0.642  0.681
tri_T1  0.902  1.000  0.809  0.837  0.723  0.808  0.629  0.668
ssc_T1  0.816  0.809  1.000  0.924  0.695  0.680  0.783  0.757
sil_T1  0.859  0.837  0.924  1.000  0.735  0.725  0.744  0.823
bic_T2  0.787  0.723  0.695  0.735  1.000  0.889  0.828  0.854
tri_T2  0.752  0.808  0.680  0.725  0.889  1.000  0.806  0.835
ssc_T2  0.642  0.629  0.783  0.744  0.828  0.806  1.000  0.908
sil_T2  0.681  0.668  0.757  0.823  0.854  0.835  0.908  1.000
> round(Sdz4,3)
       bic_T1 tri_T1 ssc_T1 sil_T1 bic_T2 tri_T2 ssc_T2 sil_T2
bic_T1 15.371 13.314 17.420 19.916  4.631  3.414  5.357  5.649
tri_T1 13.314 13.924 16.800 18.706  3.817  3.347  4.631  4.587
ssc_T1 17.420 16.800 25.427 26.833  4.911  3.840  7.429  6.882
sil_T1 19.916 18.706 26.833 32.405  5.727  4.445  8.611  9.256
bic_T2  4.631  3.817  4.911  5.727 13.320 10.715 15.916 17.430
tri_T2  3.414  3.347  3.840  4.445 10.715 10.893 14.534 16.126
ssc_T2  5.357  4.631  7.429  8.611 15.916 14.534 25.819 26.827
sil_T2  5.649  4.587  6.882  9.256 17.430 16.126 26.827 31.975
> round(Rdz4,3)
       bic_T1 tri_T1 ssc_T1 sil_T1 bic_T2 tri_T2 ssc_T2 sil_T2
bic_T1  1.000  0.910  0.881  0.892  0.324  0.264  0.269  0.255
tri_T1  0.910  1.000  0.893  0.881  0.280  0.272  0.244  0.217
ssc_T1  0.881  0.893  1.000  0.935  0.267  0.231  0.290  0.241
sil_T1  0.892  0.881  0.935  1.000  0.276  0.237  0.298  0.288
bic_T2  0.324  0.280  0.267  0.276  1.000  0.890  0.858  0.845
tri_T2  0.264  0.272  0.231  0.237  0.890  1.000  0.867  0.864
ssc_T2  0.269  0.244  0.290  0.298  0.858  0.867  1.000  0.934
sil_T2  0.255  0.217  0.241  0.288  0.845  0.864  0.934  1.000
```

Question 1.1. Check out the MZ and DZ phenotypic correlation (rmz, rdz) of each phenotype. The rule of thumb is: if 2*rdz>rmz, then choose an ACE model; if 2*rdz<rmz choose a ADE model. What are the correlations and what model would you choose?

Question 1.2. Using "Falconer's equations" we can obtain preliminary estimates of the standardized A, D, E variance components, VA, VD, & VE. In the case of the ADE model, these equations are

VA = 4*rdz-rmz
VD= 2*rmz-4*rdz
VE= 1-VA-VD

Apply these to the correlations, to obtain the standardized components VA and VD. How much of the phenotypic variance is due to A (additive genetic effects)?

Question 1.3. In the output, we can see the 8x8 covariance matrix of the MZ group (called `Smz4`) and the correlation matrix of the MZ group (called `Rmz4`). We can express the covariance matrix as a function of the correlation matrix Rmz4 and the diagonal matrix containing the standard deviations. Do this in R as follows 1) get the standard deviations from the covariance matrix, and put these in the diagonal of a 8x8 matrix, called Sdmz and carry out the multiplication: `Sdmz%*%Rmz4%*%t(Sdmz)`[1]

```
# ->
# express covariance matrix using stdevs and correlation matrix
Vmz=(diag(Smz4))  # variances from diagonal to vector Vmz
Sdmz=diag(sqrt(Vmz))  # standard deviation into diagonal
Smz = Sdmz%*%Rmz4%*%t(Sdmz)  #### Useful to know
round(Smz,3)
round(Smz4-Smz,3) # compare with previous calculation
# <-
```

Given the covariance matrix Smz4 (calculated earlier), you can obtain the correlation matrix as follows:

[1] **Note.** See Video 1, Slide 4, around the 1:50 mark for more information on this.

```
# ->
Rmz=cov2cor(Smz4)
round(Rmz,3)
#
Sdmzi=diag(sqrt(1/Vmz))    # Sdmzi diagonal with 1/std
Rmz=Sdmzi%*%Smz4%*%t(Sdmzi)
round(Rmz,3)
round(Rmz4-Rmz,3) # compare with previous calculation
# <-
```

Should we take into account main effects of sex? Does sex predict the skinfold phenotypes? To address this question, we'll carry out the linear regression of the skinfold phenotypes on sex using the R function `lm()`. We do this in the data frame called `skinfold` in the twin 1 members and the twin 2 members. Here is the code to get you going:

```
# ->
r11=summary(lm(bic_T1~sex_T1, data=skinfold))
r12=summary(lm(tri_T1~sex_T1, data=skinfold))
r13=summary(lm(ssc_T1~sex_T1, data=skinfold))
r14=summary(lm(sil_T1~sex_T1, data=skinfold))
#<-
```

Question 1.4. Repeat the analyses in the twin 2 member. What do you conclude with respect to the main effect of sex? Check out the `Multiple R-squared` statistic and the p value associated with the regression coefficient in the regression of the skinfold measures on sex (`Pr(>|t|)`). You can get these as follows (e.g., `r11` output)

All p values < .001, as shown in the output `r11`, `r22`, etc.

From question 4, we know that sex explains between about 3 and 9 % of the phenotypic variance of the 4 phenotypes. We will now use OpenMx to estimate the MZ and DZ covariance and correlation matrices, but with sex as a covariate. The OpenMx script is given below.

```
# ->
# -------------------------------- SAT model with sex as covariate
#
selVars = c("bic_T1","tri_T1","ssc_T1","sil_T1","bic_T2","tri_T2","ssc_T2","sil_T2")  # the
array with the variable names
#
nv=4  # number of phenotypes
ntv=2*nv  # nv X 2 ... 4 phenotypes observed in twin pairs
svRmz=Rmz4  # starting values correlation matrix MZ
svSdmz=diag(sqrt(diag(Smz4))) ## starting values stdevs MZ
#
svRdz=Rdz4                    # starting values correlation matrix DZ
svSddz=diag(sqrt(diag(Sdz4))) ## starting values stdevs DZ
#
svb0=c(18,21,20,20)    #  intercepts b0 in skinfold=b0+b1*sex + residual
svb1=c(.01,.01,.01,.01)    # slope   b1 in skinfold=b0+b1*sex + residual
# Create regression model for expected Mean Matrices
B0_     <- mxMatrix( type="Full", nrow=1, ncol=nv, free=TRUE, values=svb0,
labels=c("b0b","b0t","b0s","b0si"), name="b0sex" )
B1_     <- mxMatrix( type="Full", nrow=1, ncol=nv, free=TRUE, values=svb1,
labels=c("b1b","b1t","b1s","b1si"), name="b1sex" )
Sex1 <- mxMatrix(type="Full", nrow=1, ncol=1, free=FALSE, labels=c('data.sex_T1'),name="sex1")
# sex twin 1
Sex2 <- mxMatrix(type="Full", nrow=1, ncol=1, free=FALSE, labels=c('data.sex_T2'),name="sex2")
# sex twin 2
#
ExpMean <- mxAlgebra(expression=cbind(b0sex+b1sex*sex1,b0sex+b1sex*sex2), name='expMean')
#
```

```
# Model covariance matrix as function of correlation matrix (RMZ RDZ) and stdevs
corMZ      <- mxMatrix( type="Stand", nrow=ntv, ncol=ntv, free=TRUE, values=svRmz,
                        name="RMZ" ) #
corDZ      <- mxMatrix( type="Stand", nrow=ntv, ncol=ntv, free=TRUE, values=svRdz,
                        name="RDZ" ) #
sdMZ      <- mxMatrix( type="Diag", nrow=ntv, ncol=ntv, free=TRUE, values=svSdmz,
                       name="sdMZ" )
sdDZ      <- mxMatrix( type="Diag", nrow=ntv, ncol=ntv, free=TRUE, values=svSddz,
                       name="sdDZ" )
#
ExpCovMZ     <- mxAlgebra( expression=sdMZ%*%RMZ%*%t(sdMZ), name="expCovMZ" )
ExpCovDZ     <- mxAlgebra( expression=sdDZ%*%RDZ%*%t(sdDZ), name="expCovDZ" )
#
dataMZ    <- mxData( observed=mzskinfold, type="raw" )
dataDZ    <- mxData( observed=dzskinfold, type="raw" )
#
expMZ     <- mxExpectationNormal( covariance="expCovMZ", means="expMean", dimnames=selVars )
expDZ     <- mxExpectationNormal( covariance="expCovDZ", means="expMean", dimnames=selVars )
funML     <- mxFitFunctionML()
# model
modelMZ   <- mxModel(B0_, B1_, corMZ, sdMZ, ExpMean,  Sex1, Sex2, ExpCovMZ, ExpMean, dataMZ,
expMZ, funML, name="MZ" )
modelDZ   <- mxModel(B0_, B1_, corDZ, sdDZ, ExpMean,  Sex1, Sex2, ExpCovDZ, ExpMean, dataDZ,
expDZ, funML, name="DZ" )
multi     <- mxFitFunctionMultigroup( c("MZ","DZ") )
#
modelSAT4  <- mxModel("SAT", modelMZ, modelDZ, multi)
# Run SAT Model
#fitSAT4    <- mxRun( modelSAT4)
fitSAT4    <- mxTryHard( modelSAT4,20)
sumSAT4    <- summary( fitSAT4 )
#
round(fitSAT4$MZ$RMZ$values,3)
round(fitSAT4$MZ$expCovMZ$result,3)
round(fitSAT4$DZ$RDZ$values,3)
round(fitSAT4$DZ$expCovDZ$result,3)
# ------------------------------end  SAT model with sex as covariate
# <-
```

Output

```
> round(fitSAT4$MZ$RMZ$values,3)
       bic_T1 tri_T1 ssc_T1 sil_T1 bic_T2 tri_T2 ssc_T2 sil_T2
bic_T1  1.000  0.900  0.806  0.856  0.770  0.737  0.621  0.662
tri_T1  0.900  1.000  0.803  0.833  0.708  0.796  0.613  0.653
ssc_T1  0.806  0.803  1.000  0.924  0.676  0.666  0.769  0.744
sil_T1  0.856  0.833  0.924  1.000  0.718  0.710  0.733  0.808
bic_T2  0.770  0.708  0.676  0.718  1.000  0.889  0.821  0.853
tri_T2  0.737  0.796  0.666  0.710  0.889  1.000  0.802  0.834
ssc_T2  0.621  0.613  0.769  0.733  0.821  0.802  1.000  0.906
sil_T2  0.662  0.653  0.744  0.808  0.853  0.834  0.906  1.000
> round(fitSAT4$MZ$expCovMZ$result,3)
       [,1]   [,2]   [,3]   [,4]   [,5]   [,6]   [,7]   [,8]
[1,] 11.807  9.886 12.275 15.628  8.989  8.133  9.730 11.565
[2,]  9.886 10.229 11.381 14.164  7.689  8.174  8.950 10.610
[3,] 12.275 11.381 19.643 21.773 10.185  9.477 15.549 16.759
[4,] 15.628 14.164 21.773 28.259 12.961 12.110 17.790 21.817
[5,]  8.989  7.689 10.185 12.961 11.541  9.690 12.735 14.715
[6,]  8.133  8.174  9.477 12.110  9.690 10.305 11.746 13.595
[7,]  9.730  8.950 15.549 17.790 12.735 11.746 20.824 21.008
[8,] 11.565 10.610 16.759 21.817 14.715 13.595 21.008 25.814
> round(fitSAT4$DZ$RDZ$values,3)
       bic_T1 tri_T1 ssc_T1 sil_T1 bic_T2 tri_T2 ssc_T2 sil_T2
bic_T1  1.000  0.902  0.873  0.888  0.309  0.250  0.263  0.250
tri_T1  0.902  1.000  0.885  0.877  0.266  0.267  0.242  0.216
ssc_T1  0.873  0.885  1.000  0.933  0.253  0.223  0.293  0.242
sil_T1  0.888  0.877  0.933  1.000  0.273  0.240  0.310  0.297
bic_T2  0.309  0.266  0.253  0.273  1.000  0.880  0.848  0.835
tri_T2  0.250  0.267  0.223  0.240  0.880  1.000  0.858  0.857
ssc_T2  0.263  0.242  0.293  0.310  0.848  0.858  1.000  0.930
sil_T2  0.250  0.216  0.242  0.297  0.835  0.857  0.930  1.000
> round(fitSAT4$DZ$expCovDZ$result,3)
       [,1]   [,2]   [,3]   [,4]   [,5]   [,6]   [,7]   [,8]
[1,] 14.383 12.435 16.222 18.859  4.124  3.039  4.905  5.225
[2,] 12.435 13.205 15.762 17.846  3.403  3.100  4.320  4.318
[3,] 16.222 15.762 24.013 25.609  4.369  3.503  7.057  6.535
[4,] 18.859 17.846 25.609 31.378  5.390  4.298  8.522  9.172
[5,]  4.124  3.403  4.369  5.390 12.385  9.914 14.670 16.206
[6,]  3.039  3.100  3.503  4.298  9.914 10.237 13.493 15.116
[7,]  4.905  4.320  7.057  8.522 14.670 13.493 24.154 25.205
[8,]  5.225  4.318  6.535  9.172 16.206 15.116 25.205 30.383
```

Question 1.5. What are the MZ and DZ correlations of the 4 phenotypes based on the OpenMx output, and why do they differ from those of question 2? The correlations were (question 2):

```
> rmz=c(.787, .808, .783, .823)
> rdz=c(.324, .272, .290, .288)
```

Question 1.6. What do you conclude on the basis of the MZ and DZ cross-trait – cross-twin correlations of biceps and triceps (underlined), given MZ cor(bic_T1, tri_T2) > given DZ cor(bic_T1, tri_T2) ?

MZ cross-twin-cross-trait correlations			DZ cross-twin-cross-trait correlations		
	bic_T1	tri_T1		bic_T1	tri_T1
bic_T2	**0.769**	0.706	bic_T2	**0.312**	0.269
tri_T2	0.737	**0.795**	tri_T2	0.254	**0.270**

The 4 variate ADE model.

We know from the application of Falconer's equations that the twin correlations suggest an ADE model. We can apply Falconer's equations to the sex-corrected correlation, but the results are about the same.

```
# ->
F_ADE = function(rmz, rdz) { c(4*rdz-rmz, 2*rmz-4*rdz) }
rmz=c(.770, .796, .769, .808)
rdz=c(.309, .267, .293, .297)
print(F_ADE(rmz,rdz))
# <-
```

```
VA                              VD
0.466 0.272 0.403 0.380         0.304 0.524 0.366 0.428
```

We'll fit the 4 variate twin model to the skinfold data. Σ_{Ph} is the 4x4 covariance matrix, and we want: $\Sigma_{Ph} = \Sigma_A + \Sigma_D + \Sigma_E$, i.e., the decomposition of the 4x4 phenotypic covariance matrix in to the 4x4 A, 4x4 D, and 4x4 E covariance matrices.

The generalization from p=1 (univariate) to p=2 (bivariate) to p>2 (multivariate).

Σ_{Ph} MZ	p phenotypes	p phenotypes
	$\Sigma_A + \Sigma_D + \Sigma_E$	$\Sigma_A + \Sigma_D$
	$\Sigma_A + \Sigma_D$	$\Sigma_A + \Sigma_D + \Sigma_E$

Σ_{Ph} DZ	p phenotypes	p phenotypes
	$\Sigma_A + \Sigma_D + \Sigma_E$	$\frac{1}{2}\Sigma_A + \frac{1}{4}\Sigma_D$
	$\frac{1}{2}\Sigma_A + \frac{1}{4}\Sigma_D$	$\Sigma_A + \Sigma_D + \Sigma_E$

Biceps and Triceps (purple), ~~Calf (green)~~,
Subscapular (blue), Suprailiacal (orange).

Boulder workshop 2022 IPM / CPM (Dolan & Pelt) 5

The twin design is the means to this end. In the OpenMx script, we estimate the correlation matrices and the standard deviations, and use those to calculate the covariance matrices. So, for instance, the 4x4 covariance Σ_A is modelled as follows (a matrix expression):

$\Sigma_A = sd_A\ R_A\ sd_A$, where sd_A is 4x4 diagonal matrix (containing the standard deviations) and R_A is 4x4 correlation matrix.

Or, in the notation of the OpenMx script:

```
VA      <- mxAlgebra( expression= sA%*%RA%*%t(sA), name="VA" )
```

So, in the notation of the OpenMx script, sA is 4x4 diagonal, containing the A standard deviations, and RA is the 4x4 additive genetic correlation matrix. The same applies to the D (sD and RD) and

the E components (sE and RE). Below is the OpenMx script. Before you run this OpenMx score, let's consider the starting values. Here is the part of the script that concerns the starting values:

```
# ->
nv=4  # number of variable
ntv=2*nv  # number of variables times 2 (twins)
# starting values based on the phenotypic cov matrix
svS=matrix(c(
  14,     12,     15,     18,
  12,     12,     14,     17,
  15,     14,     23,     25,
  18,     17,     25,     31),4,4)
#
svVA=svS*.4  # rough guess of A cov matrix
svVE=svS*.2  # rough guess of E cov matrix
svVD=svS*.4  # rough guess of D cov matrix
rvA= round(cov2cor(svVA),3) # rough guess of A correlation matrix
rvD= round(cov2cor(svVE),3) # rough guess of E correlation matrix
rvE= round(cov2cor(svVD),3) # rough guess of D correlation matrix
svsA=sqrt(diag(svVA)) # A standard deviations
svsD=sqrt(diag(svVD)) # D standard deviations
svsE=sqrt(diag(svVE)) # E standard deviations
# <-
```

The matrix svS is a rough guess based on the phenotypic covariance matrix of the 4 phenotypes.
We obtain a rough guess of VA (= Σ_A) by `svVA=svS*.4`
We obtain a rough guess of VD (= Σ_D) by `svVD=svS*.2`
We obtain a rough guess of VE (= Σ_E) by `svVE=svS*.4`

The starting values of RA:
`rvA=cov2cor(svVA)`
The starting values of the standard deviations of the A
`svsA=sqrt(diag(svVA))`
The same applies to D and E.

Question 1.7. What does the guess `svVA=svS*.4` actually imply with respect to the variance of the 4 phenotype phenotypes? Is this reasonable given your answer to question 2? (From question 2, we know that the percentages are 50.9%, 28.0%, 37.7% and 32.9%**).**

```
# ->
## ADE model with sex as covariate
# -----------------------------------------------------------------------
selVars = c("bic_T1","tri_T1","ssc_T1","sil_T1","bic_T2","tri_T2","ssc_T2","sil_T2")  # the
array with the variable names
# PREPARE MODEL
#
nv=4  # number of variable
ntv=2*nv  # number of variables times 2 (twins)
# starting values based on the phenotypic cov matrix
svS=matrix(c(
  14,     12,     15,     18,
  12,     12,     14,     17,
  15,     14,     23,     25,
  18,     17,     25,     31),4,4)
#
svVA=svS*.4  # rough guess of A cov matrix
svVE=svS*.2  # rough guess of E cov matrix
svVD=svS*.4  # rough guess of D cov matrix
rvA=round(cov2cor(svVA),2) # rough guess of A correlation matrix
rvD=round(cov2cor(svVE),2) # rough guess of E correlation matrix
rvE=round(cov2cor(svVD),2) # rough guess of D correlation matrix
svsA=sqrt(diag(svVA)) # A standard deviations
svsD=sqrt(diag(svVD)) # D standard deviations
svsE=sqrt(diag(svVE)) # E standard deviations
#
svb0=c(18,21,20,20)
svb1=c(.0,.0,.0,.0)
#
# Create regression model for expected Mean Matrices
B0_     <- mxMatrix( type="Full", nrow=1, ncol=nv, free=TRUE, values=svb0,
labels=c("b0b","b0t","b0s","b0si"), name="b0sex" )
B1_     <- mxMatrix( type="Full", nrow=1, ncol=nv, free=TRUE, values=svb1,
labels=c("b1b","b1t","b1s","b1si"), name="b1sex" )
Sex1 <- mxMatrix(type="Full", nrow=1, ncol=1, free=FALSE, labels=c('data.sex_T1'),name="sex1")
# sex twin 1
Sex2 <- mxMatrix(type="Full", nrow=1, ncol=1, free=FALSE, labels=c('data.sex_T2'),name="sex2")
# sex twin 2
#
ExpMean <- mxAlgebra(expression=cbind(b0sex+b1sex*sex1,b0sex+b1sex*sex2), name='expMean')
#
# Create Matrices for Variance Components
corA      <- mxMatrix( type="Stand", nrow=nv, ncol=nv, free=TRUE, values=rvA,
label=c("rA21","rA31","rA41","rA32","rA42","rA43"), name="RA" )
corD      <- mxMatrix( type="Stand", nrow=nv, ncol=nv, free=TRUE, values=rvD,
label=c("rD21","rD31","rD41","rD32","rD42","rD43"), name="RD" )
corE      <- mxMatrix( type="Stand", nrow=nv, ncol=nv, free=TRUE, values=rvE,
label=c("rE21","rE31","rE41","rE32","rE42","rE43"), name="RE" )
#
sdA      <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=svsA,
label=c("sA1","sA2","sA3","sA4"), name="sA" )
sdD      <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=svsD,
label=c("sD1","sD2","sD3","sD4"), name="sD" )
sdE      <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=svsE,
label=c("sE1","sE2","sE3","sE4"), name="sE" )
#
# Create Algebra for expected Variance/Covariance Matrices in MZ & DZ twins
#
VA      <- mxAlgebra( expression= sA%*%RA%*%t(sA), name="VA" )
VD      <- mxAlgebra( expression= sD%*%RD%*%t(sD), name="VD" )
VE      <- mxAlgebra( expression= sE%*%RE%*%t(sE), name="VE" )
covP      <- mxAlgebra( expression= VA+VD+VE, name="V" )
covMZ     <- mxAlgebra( expression= VA+VD, name="cMZ" )
covDZ     <- mxAlgebra( expression= 0.5%x%VA+.25%x%VD, name="cDZ" )
expCovMZ  <- mxAlgebra( expression= rbind( cbind(V, cMZ), cbind(t(cMZ), V)), name="expCovMZ" )
expCovDZ  <- mxAlgebra( expression= rbind( cbind(V, cDZ), cbind(t(cDZ), V)), name="expCovDZ" )
#
# Create Data Objects for Multiple Groups
dataMZ    <- mxData( observed=mzskinfold, type="raw" )
dataDZ    <- mxData( observed=dzskinfold, type="raw" )
#
# Create Expectation Objects for Multiple Groups
expMZ     <- mxExpectationNormal( covariance="expCovMZ", means="expMean", dimnames=selVars )
expDZ     <- mxExpectationNormal( covariance="expCovDZ", means="expMean", dimnames=selVars )
```

```
funML     <- mxFitFunctionML()
#
pars      <- list(B0_, B1_, corA, corD, corE, sdA, sdD, sdE, VA, VD, VE, covP )
modelMZ   <- mxModel( pars, ExpMean,  Sex1, Sex2, covMZ, expCovMZ, ExpMean, dataMZ, expMZ,
funML, name="MZ" )
modelDZ   <- mxModel( pars, ExpMean,  Sex1, Sex2, covDZ, expCovDZ, ExpMean, dataDZ, expDZ,
funML, name="DZ" )
multi     <- mxFitFunctionMultigroup( c("MZ","DZ") )
#
modelADE4  <- mxModel("ACE4", pars, modelMZ, modelDZ, multi)
# Run Model
fitADE4    <- mxRun( modelADE4 )
sumADE4    <- summary( fitADE4 )
mxCompare(fitSAT4, fitADE4)
# <-
```

The results of main interest are the covariance matrices of A, D, and E. Let's extract the results, pertaining to A, from the output fitADE4.

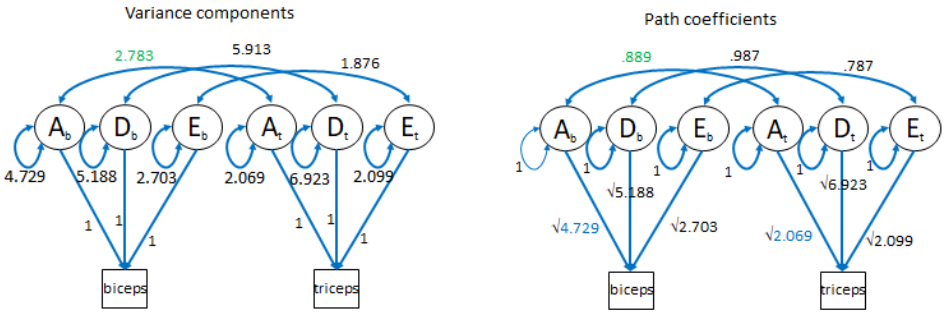
```
# ->
VA_ = fitADE4$VA$result
RA_ = fitADE4$RA$values
sA_ = fitADE4$sA$values
# <-
```

Question 1.8. Verify that VA_ = sA_%*%RA_%*%t(sA_).

Question 1.9. Extract the covariance matrices of A (already done!), D and E, call these VA_, VD_, and VE_. But here we only need the 2x2 matrices of the phenotypes biceps and triceps. Do this as follows.

```
# ->
VA_ = fitADE4$VA$result[1:2,1:2]
VD_ = fitADE4$VD$result[1:2,1:2]
VE_ = fitADE4$VE$result[1:2,1:2]
# <-
```

You can represent the results of the ADE model using a path diagram. In the PPTs, we presented three path diagrams. These are shown below just for biceps and triceps. Now we'll reproduce the parameter values in these path diagrams.

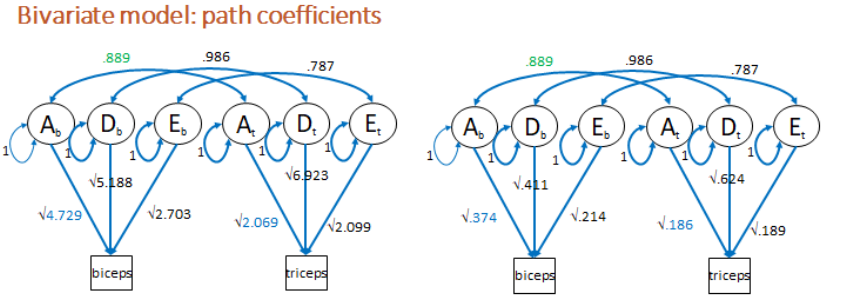


Σ_A	biceps	triceps
biceps	σ_{Ab}^2	σ_{AbAt}
triceps	$\sigma_{Ab,At}$	σ_{At}^2

Σ_A	biceps	triceps
biceps	a_b^2	$a_b\rho_{Ab,At}a_t$
triceps	$a_b\rho_{Ab,At}a_t$	a_t^2

BOULDER WORKSHOP 2022 IPM / CPM (DOLAN & PELT) 26

***Figure slide 26 left*: variance components; *Figure slide 26 right*: raw path coefficients[2].**



Phenotypes not standardized,
A,D,E standardized

Completely standardized
Phenotypes standardized, A,D,E standardized

.374 = 4.729 / (4.729+5.188+2.703) = A_b
.411= 5.188/ (4.729+5.188+2.703) = D_b
.214 = 2.703 / (4.729+5.188+2.703) = E_b

BOULDER WORKSHOP 2022 IPM / CPM (DOLAN & PELT) 27

***Figure slide 27 left*: raw path coefficients (again); *Figure slide 27 right*: standardized variance components.**

Figure 26 left depicts these variance components, the values depend only on VA_, VD_, and VE_.

[2] See Video 1 around the 21:00 mark for more information on this.

Question 1.10. Look at the values in VA_, VD_, and VE_, they should resemble approximately the values in *Figure slide 26 left*. NOTE: they are not exactly equal, because the results in the slides were obtain by fitting the bivariate model. but the present results were obtained by fitting the 4-variate model. Verify that the parameters in *Figure slide 26 left* resemble the values in VA_, VD_, and VE_.

Question 1.11. To obtain the values in *Figure slide 26 right* we need the A, D, and E correlations and the A D and E standard deviations. In this path diagram, the path coefficients are the A, D, and E standard deviations and the A, D, E variables are standardized, so that the coefficients association with the double headed arrows are correlations. Get these results by getting the standard deviations and the correlations. You can extract these from the output. E.g., for A:

```
# ->
VA_ = fitADE4$VA$result[1:2,1:2]
RA_ = fitADE4$RA$values[1:2,1:2]
sA_ = fitADE4$sA$values[1:2,1:2]
# <-
```

Alternatively, you can use `cov2cor()` applied to the covariance matrices (to obtain the correlations), and `sqrt(diag())` applied to the covariance matrices (to obtain the standard deviations). Verify that the parameter in Figure slide 26 right resemble your values.

Question 1.12. The representation in *Figure slide 27 right* is the easiest to interpret, because all variables are standardized. However, it is more complicated to calculate the path coefficients. We already have the correlations (Question 11). We now require the standardized variance components. To calculate these:

```
# ->
VPh_ = VA_ + VD_ + VE_
diag(VA_ / VPh_) # the standardized A variance component
diag(VD_ / VPh_) # the standardized D variance component
diag(VE_ / VPh_) # the standardized E variance component
# <-
```

Verify that your parameter values resemble those in *Figure slide 27 right*.

By the path tracing rules, we can calculate the contribution of A to the phenotypic correlation in *Figure slide 27 right*. In terms of our A parameters, this is

sqrt(.384)*.885*sqrt(.201) = .246

Question 1.13: If I tell you that in the model the environmental correlation equals .789, does that mean that E is contributing greatly to the phenotypic correlation? The actual expected phenotypic correlation is .893. Answer this by referring to the actual contribution of E to the phenotypic correlation.

Question 1.14: What is wrong with the 4x4 correlation matrix of D? You can extract this as follows:

```
# ->
RD_ = fitADE4$RD$values
# <-
```

The fact that some correlations are greater than 1.0 may seem problematic, but in practice this can happen. One thing we can do is test the hypothesis that the correlations are actually all equal to 1. We can do this by fixing the D correlations to one. The **covariance** matrix D is a function of the D standard deviations

```
sdD      <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=svsD,
label=c("sD1","sD2","sD3","sD4"), name="sD" )
```

and the D correlation matrix

```
corD      <- mxMatrix( type="Stand", nrow=nv, ncol=nv, free=TRUE, values=rvD,
label=c("rD21","rD31","rD41","rD32","rD42","rD43"), name="RD" )
```

```
# ->
RD_ = fitADE4$RD$values
sD_ = fitADE4$sD$values
sD_%*%RD_%*%sD_
# <-
```

Question 1.15: use the `omxSetParameters()` function to fix the D correlations to 1.0, and use the `mxCompare()` function to test whether these constraints are ok statistically. If OK, what does this result suggest (interpretation)? Why is df=6 (`diffdf=6`)?

End of part 1

Practical part 2: Four neuroticism Items. Multivariate ADE model. The AE Independent Pathway model and the Common Pathway Model. 12 questions.

In the second part of the practical, we will analyze 4 neuroticism items in the following step

1) Read the data into R studio and create the MZ and DZ data frames
2) Calculate descriptives and correlations
3) Fit the 4 variate ADE model, test the significance of the D component
4) Fit the AE IPM, interpret the results
5) Fit the AE CPM, interpret the results
6) Fit the AE model to the Neuroticism sum score, compare results with those of step 5

The data file, which is called `ntwindata`, contains the neuroticism items scores in a sample of 1528 MZ pairs and 1277 DZ pairs. The response format of the items is a 5-point scale (1 to 5), the higher the item score, the more neurotic. The dataset includes the variable `zyg` (zygosity, `zyg` = 1 for MZ; `zyg` = 2 for DZ).

Start by reading the data and creating the MZ and DZ data frames, and calculate the summary statistics.

```
# ->
rm(list=ls(all=TRUE))   # clear the memory
library(psych)
library(OpenMx)
#mxOption(NULL, "Default optimizer","NPSOL")
#mxOption(NULL, "Default optimizer","CSOLNP")
N4=read.table(file='ntwindata')
colnames(N4)
#
N4mz=subset(N4,zyg==1, select =c("N1_4","N1_6","N1_7","N1_10","N2_4","N2_6","N2_7","N2_10"))
N4dz=subset(N4,zyg==2, select =c("N1_4","N1_6","N1_7","N1_10","N2_4","N2_6","N2_7","N2_10"))
#
describe(N4mz, skew=F)
describe(N4dz, skew=F)
# <-
```

The descriptives are

```
> describe(N4mz, skew=F)
      vars    n mean   sd min max range
N1_4     1 1528 2.55 1.05   1   5     4
N1_6     2 1528 2.28 1.07   1   5     4
N1_7     3 1528 2.69 1.04   1   5     4
N1_10    4 1528 2.60 1.02   1   5     4
N2_4     5 1528 2.52 1.06   1   5     4
N2_6     6 1528 2.29 1.06   1   5     4
N2_7     7 1528 2.65 1.05   1   5     4
N2_10    8 1528 2.62 1.05   1   5     4
> describe(N4dz, skew=F)
      vars    n mean   sd min max range
N1_4     1 1277 2.49 1.05   1   5     4
N1_6     2 1277 2.30 1.07   1   5     4
N1_7     3 1277 2.61 1.04   1   5     4
N1_10    4 1277 2.57 1.04   1   5     4
N2_4     5 1277 2.52 1.08   1   5     4
N2_6     6 1277 2.26 1.06   1   5     4
N2_7     7 1277 2.60 1.05   1   5     4
N2_10    8 1277 2.57 1.01   1   5     4
```

As you can see, the range is from 1 to 5 as expected. next calculate the MZ and DZ 8x8 correlation matrices.

```
# ->
rmzs=round(cor(N4mz),3)
rdzs=round(cor(N4dz),3)
print(rmzs)
print(rdzs)
# <-
```

```
> round(cor(N4mz),3)
       N1_4  N1_6  N1_7 N1_10  N2_4  N2_6  N2_7 N2_10
N1_4  1.000 0.397 0.461 0.531 0.212 0.179 0.209 0.220
N1_6  0.397 1.000 0.376 0.438 0.203 0.330 0.216 0.218
N1_7  0.461 0.376 1.000 0.487 0.211 0.198 0.236 0.216
N1_10 0.531 0.438 0.487 1.000 0.231 0.211 0.258 0.270
N2_4  0.212 0.203 0.211 0.231 1.000 0.419 0.463 0.504
N2_6  0.179 0.330 0.198 0.211 0.419 1.000 0.405 0.448
N2_7  0.209 0.216 0.236 0.258 0.463 0.405 1.000 0.517
N2_10 0.220 0.218 0.216 0.270 0.504 0.448 0.517 1.000
> round(cor(N4dz),3)
       N1_4  N1_6  N1_7 N1_10  N2_4  N2_6  N2_7 N2_10
N1_4  1.000 0.461 0.446 0.521 0.100 0.083 0.074 0.117
N1_6  0.461 1.000 0.431 0.490 0.082 0.170 0.071 0.104
N1_7  0.446 0.431 1.000 0.508 0.094 0.113 0.114 0.109
N1_10 0.521 0.490 0.508 1.000 0.079 0.156 0.079 0.120
N2_4  0.100 0.082 0.094 0.079 1.000 0.363 0.377 0.452
N2_6  0.083 0.170 0.113 0.156 0.363 1.000 0.361 0.467
N2_7  0.074 0.071 0.114 0.079 0.377 0.361 1.000 0.439
N2_10 0.117 0.104 0.109 0.120 0.452 0.467 0.439 1.000
```

Question 2.1: Consider the twin correlations of each item. Remember that rmz>2*rdz suggests an ADE model and rmz<2*rdz suggests an ACE model. However, rmz=2*rdz suggests an AE model and rmz = rdz suggests a CE model. Based on the correlations, which model is most likely to hold?

We will now fit the saturated model to estimate the phenotypic covariance matrices.

```
# ->
selVars = c("N1_4","N1_6","N1_7","N1_10","N2_4","N2_6","N2_7","N2_10")  # the array with the
variable names
#
# unconstrained .... get good values
rmzs=round( cor(N4mz),2)   # correlations
rdzs=round( cor(N4dz),2)
sdmz=round( sqrt(diag(cov(N4mz))),2)
sddz=round( sqrt(diag(cov(N4dz))),2)
memz=apply(N4mz[,selVars[1:4]],2,mean)+apply(N4mz[,selVars[5:8]],2,mean)
memz=round( memz/2,2)
medz=apply(N4dz[,selVars[1:4]],2,mean)+apply(N4dz[,selVars[5:8]],2,mean)
medz=round( medz/2,2)
#
nv=4
ntv=2*nv
#
# Create regression model for expected Mean Matrices
#
B0_mz <- mxMatrix( type="Full", nrow=1, ncol=nv, free=TRUE, values=memz,
labels=c("b01mz","b02mz","b03mz","b04mz"), name="b0MZ" )
ExpMean_mz <- mxAlgebra(expression=cbind(b0MZ,b0MZ), name='expMeanMZ')
corPh_mz      <- mxMatrix( type="Stand", nrow=ntv, ncol=ntv, free=TRUE, values=rmzs,
name="RPhMZ" )
sdPh_mz      <- mxMatrix( type="Diag", nrow=ntv, ncol=ntv, free=TRUE, values=sdmz,
name="sPhMZ" )
evalPh_mz <- mxAlgebra(eigenval(RPhMZ), name='eval_PhMZ')
ExpCov_mz      <- mxAlgebra( expression= sPhMZ%*%RPhMZ%*%t(sPhMZ), name="expCovMZ" )
#
#
B0_dz <- mxMatrix( type="Full", nrow=1, ncol=nv, free=TRUE, values=memz,
labels=c("b01dz","b02dz","b03dz","b04dz"), name="b0DZ" )
ExpMean_dz <- mxAlgebra(expression=cbind(b0DZ,b0DZ), name='expMeanDZ')
corPh_dz      <- mxMatrix( type="Stand", nrow=ntv, ncol=ntv, free=TRUE, values=rdzs,
name="RPhDZ" )
sdPh_dz      <- mxMatrix( type="Diag", nrow=ntv, ncol=ntv, free=TRUE, values=sddz,
name="sPhDZ" )
evalPh_dz <- mxAlgebra(eigenval(RPhDZ), name='eval_PhDZ')
ExpCov_dz      <- mxAlgebra( expression= sPhDZ%*%RPhDZ%*%t(sPhDZ), name="expCovDZ" )
#
#
dataMZ    <- mxData( observed=N4mz, type="raw" )
dataDZ    <- mxData( observed=N4dz, type="raw" )
#
# Create Expectation Objects for Multiple Groups
expMZ     <- mxExpectationNormal( covariance="expCovMZ", means="expMeanMZ", dimnames=selVars )
expDZ     <- mxExpectationNormal( covariance="expCovDZ", means="expMeanDZ", dimnames=selVars )
funML     <- mxFitFunctionML()
#
modelMZ   <- mxModel(dataMZ, B0_mz, ExpMean_mz, corPh_mz, sdPh_mz, ExpCov_mz, evalPh_mz,
expMZ, funML, name="MZ" )
modelDZ   <- mxModel(dataDZ, B0_dz, ExpMean_dz, corPh_dz, sdPh_dz, ExpCov_dz, evalPh_dz,
expDZ, funML, name="DZ" )
multi     <- mxFitFunctionMultigroup( c("MZ","DZ") )
#
modelSAT4A  <- mxModel("Sat", modelMZ, modelDZ, multi)
# ------------------------------------------------------------------------------------------
--------------------------
# RUN sat MODEL
#
fitSAT4A    <- mxRun( modelSAT4A )
sfitSAT4A = summary( fitSAT4A)
#fitSAT4A    <- mxTryHard( modelSAT4A, 20)

#modelSAT4B <-
omxSetParameters(fitSAT4A,labels=c("b01mz","b02mz","b03mz","b04mz","b01dz","b02dz","b03dz","b0
4dz"),
#newlabels=c("b01","b02","b03","b04","b01","b02","b03","b04"), free=T, values=c(2))
#fitSAT4B    <- mxTryHard( modelSAT4B, 20)
#fitSAT4B    <- mxTryHard( modelSAT4B, 20)
#mxCompare(fitSAT4A, fitSAT4B)
# <-
```

Just as a check take a look at the MZ and DZ correlation matrices.

```
# ->
round(fitSAT4A$MZ$RPhMZ$values,3)
round(fitSAT4A$DZ$RPhDZ$values,3)
# <-
```

```
> round(fitSAT4A$MZ$RPhMZ$values,3)
       N1_4  N1_6  N1_7 N1_10  N2_4  N2_6  N2_7 N2_10
N1_4  1.000 0.397 0.462 0.531 0.212 0.179 0.209 0.220
N1_6  0.397 1.000 0.376 0.438 0.203 0.330 0.216 0.218
N1_7  0.462 0.376 1.000 0.487 0.211 0.198 0.236 0.216
N1_10 0.531 0.438 0.487 1.000 0.231 0.211 0.258 0.269
N2_4  0.212 0.203 0.211 0.231 1.000 0.419 0.463 0.504
N2_6  0.179 0.330 0.198 0.211 0.419 1.000 0.405 0.448
N2_7  0.209 0.216 0.236 0.258 0.463 0.405 1.000 0.517
N2_10 0.220 0.218 0.216 0.269 0.504 0.448 0.517 1.000
> round(fitSAT4A$DZ$RPhDZ$values,3)
       N1_4  N1_6  N1_7 N1_10  N2_4  N2_6  N2_7 N2_10
N1_4  1.000 0.461 0.446 0.521 0.100 0.083 0.074 0.117
N1_6  0.461 1.000 0.431 0.490 0.082 0.169 0.071 0.104
N1_7  0.446 0.431 1.000 0.508 0.094 0.113 0.114 0.109
N1_10 0.521 0.490 0.508 1.000 0.079 0.156 0.079 0.120
N2_4  0.100 0.082 0.094 0.079 1.000 0.362 0.377 0.451
N2_6  0.083 0.169 0.113 0.156 0.362 1.000 0.361 0.467
N2_7  0.074 0.071 0.114 0.079 0.377 0.361 1.000 0.439
N2_10 0.117 0.104 0.109 0.120 0.451 0.467 0.439 1.000
```

You may note a slight difference between the correlation we reported above and these correlations. This is due to fact that have constrained the mean vectors: we estimated 4 means in the MZs and 4 means in the DZ. You can test whether the means are equal in the MZs and DZs with this additional code

```
# ->
# impose equality constraints on the means
modelSAT4B <- omxSetParameters(fitSAT4A,
     labels=c("b01mz","b02mz","b03mz","b04mz","b01dz","b02dz","b03dz","b04dz"),
  newlabels=c("b01","b02","b03","b04","b01","b02","b03","b04"),
       free=T, values=c(2))
#
fitSAT4B    <- mxRun( modelSAT4B)
#fitSAT4B    <- mxTryHard( modelSAT4B, 20)
mxCompare(fitSAT4A, fitSAT4B)
# <-
```

Question 2.2: Are the means equal in the MZs and DZs? And why is df=4 (`diffdf=4`).

ADE 4 variate twin model.

Although it is unlikely that D is significant (in view of the twin correlations), we'll first fit the 4 variate ADE model and test the D component to make sure it is not significant. Here is the code:

```
# ADE model    ---------------------------------------
nv=4
ntv=2*nv
svS=matrix(.5,4,4); diag(svS)=1  # approx S
svVA=svS*.4  # starting value  SA
svVE=svS*.5  # starting values SE
svVD=svS*.1   #  starting values SD
#
svb0=c(rep(2.5,4))   # means
#
# Create regression model for expected Mean Matrices
#
B0_     <- mxMatrix( type="Full", nrow=1, ncol=nv, free=TRUE, values=svb0,
labels=c("b01","b02","b03","b04"), name="b0" )
#
ExpMean <- mxAlgebra(expression=cbind(b0,b0), name='expMean')
#
# ADE model
# Create Matrices for Variance Components
#
SA      <- mxMatrix( type="Symm", nrow=nv, ncol=nv, free=TRUE, values=svVA,
label=c("SA11","SA21","SA31","SA41","SA22","SA32","SA42","SA33","SA43","SA44"),
        name="VA" )
SE      <- mxMatrix( type="Symm", nrow=nv, ncol=nv, free=TRUE, values=svVE,
label=c("SE11","SE21","SE31","SE41","SE22","SE32","SE42","SE33","SE43","SE44"),
         name="VE" )
#
SD      <- mxMatrix( type="Symm", nrow=nv, ncol=nv, free=TRUE, values=svVD,
label=c("SD11","SD21","SD31","SD41","SD22","SD32","SD42","SD33","SD43","SD44"),
name="VD" )
#
RA <- mxAlgebra( expression = cov2cor(VA), name="RA")
RE <- mxAlgebra( expression = cov2cor(VE), name="RE")
RD <- mxAlgebra( expression = cov2cor(VD), name="RD")
#
covP      <- mxAlgebra( expression= VA+VD+VE, name="V" )
covMZ     <- mxAlgebra( expression= VA+VD, name="cMZ" )
covDZ     <- mxAlgebra( expression= 0.5%x%VA+.25%x%VD, name="cDZ" )
expCovMZ  <- mxAlgebra( expression= rbind( cbind(V, cMZ), cbind(t(cMZ), V)), name="expCovMZ" )
expCovDZ  <- mxAlgebra( expression= rbind( cbind(V, cDZ), cbind(t(cDZ), V)), name="expCovDZ" )
#ADE model
# Create Data Objects for Multiple Groups
dataMZ    <- mxData( observed=N4mz, type="raw" )
dataDZ    <- mxData( observed=N4dz, type="raw" )
# Create Expectation Objects for Multiple Groups
expMZ     <- mxExpectationNormal( covariance="expCovMZ", means="expMean", dimnames=selVars )
expDZ     <- mxExpectationNormal( covariance="expCovDZ", means="expMean", dimnames=selVars )
funML     <- mxFitFunctionML()
pars      <- list(B0_, SA, SD, SE, RA, RE, RD, covP )
modelMZ   <- mxModel( pars, ExpMean,covMZ, expCovMZ, ExpMean, dataMZ, expMZ, funML, name="MZ"
)
modelDZ   <- mxModel( pars, ExpMean, covDZ, expCovDZ, ExpMean, dataDZ, expDZ, funML, name="DZ"
)
multi     <- mxFitFunctionMultigroup( c("MZ","DZ") )
#
modelADE4V  <- mxModel("ADE4V", pars, modelMZ, modelDZ, multi)
# ------------------------
# Run ADE Model
fitADE4V    <- mxTryHard( modelADE4V, 20)
sumADE4V    <- summary( fitADE4V )
# ------------------------------------------------------------
```

Question 2.3: Does the ADE model fit relative to the saturated model?

```
# ->
mxCompare(fitSAT4B, fitADE4V)
# <-
```

Question 2.4: Use `omxSetParameters()` to drop the D component from the ADE model, and fit the AE 4 variate model. The D component parameter labels are given in the OpenMx script:

```
SD      <- mxMatrix( type="Symm", nrow=nv, ncol=nv, free=TRUE, values=svVD,
label=c("SD11","SD21","SD31","SD41","SD22","SD32","SD42","SD33","SD43","SD44"),
name="VD" )
```

Question 2.5. Use `mxCompare()` to carry out the Likelihood ratio test that the D component is zero (`fitADE4V` vs `fitAE4V`) What do you conclude? And why is df=10 (`diffdf=10`)?

The df (diffdf) equals 10, because the D covariance matrix has 10 elements (4 variances, 6 covariances). Before we move on the IPM and CPM, let's take a look at the A and the E parameters. We can extract these from the fitAE4V output as follows.

```
# ->
rA=round(fitAE4V$RA$result,3)   # A correlations
sA=diag(sqrt(round(fitAE4V$VA$values,3)))  # A standard deviations
SA=round(fitAE4V$VA$values,3) # A cov matrix
#
rE=round(fitAE4V$RE$result,3)  # E correlations
sE=diag(sqrt(round(fitAE4V$VE$values,3))) # E standard deviations
SE=round(fitAE4V$VE$values,3) # E cov matrix
rA # correlation matrix A
SA # cov matrix A
rE # correlation matrix E
SE # cov matrix E
# <-
```

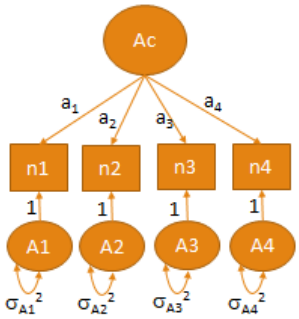
```
> rA # correlation matrix A
      [,1]  [,2]  [,3]  [,4]
[1,] 1.000 0.713 0.907 0.932
[2,] 0.713 1.000 0.734 0.762
[3,] 0.907 0.734 1.000 0.915
[4,] 0.932 0.762 0.915 1.000
> SA
      [,1]  [,2]  [,3]  [,4]
[1,] 0.232 0.211 0.221 0.238
[2,] 0.211 0.378 0.228 0.248
[3,] 0.221 0.228 0.256 0.245
[4,] 0.238 0.248 0.245 0.281
> rE
      [,1]  [,2]  [,3]  [,4]
[1,] 1.000 0.306 0.307 0.373
[2,] 0.306 1.000 0.263 0.333
[3,] 0.307 0.263 1.000 0.349
[4,] 0.373 0.333 0.349 1.000
> SE
      [,1]  [,2]  [,3]  [,4]
[1,] 0.892 0.251 0.266 0.312
[2,] 0.251 0.757 0.210 0.256
[3,] 0.266 0.210 0.840 0.283
[4,] 0.312 0.256 0.283 0.781
```

Question 2.6. You can see that the A correlations are much higher (.713 to .932) than the E correlations (.263 to .349). Does that mean that A contributes more to the phenotypic correlations that does E? Check your answer against the output of the following code. First, calculate the covariance matrices SA and SE (yes you actually already have these!). Second, get the proportions, and answer the question in the light of these proportions.

```
# ->
SA_=diag(sA)%*%rA%*% diag(sA)  # A covariance matrix = SA
SE_= diag(sE)%*%rE%*% diag(sE)     # E covariance matrix =SE
SA_
SE_
SPh=SA_+SE_  # expected phenotypic covariance matrix
round(SA_/SPh,3)       # proportions
round(SE_/SPh,3)      # proportions
# <-
```

AE 4 variate twin Independent Pathway Model (IPM)

In the AE IPM, we estimate the covariance A and the E covariance matrices, subject to a single factor model. For instance, the ΣA, the 4x4 A covariance, is modeled as shown below (the same applies to the ΣE, the E covariance matrix).



$$L_A = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{pmatrix} \quad T_A = \begin{pmatrix} \sigma_{A1}^2 & 0 & 0 & 0 \\ 0 & \sigma_{A2}^2 & 0 & 0 \\ 0 & 0 & \sigma_{A3}^2 & 0 \\ 0 & 0 & 0 & \sigma_{A4}^2 \end{pmatrix}$$

$$L_A^t = \begin{pmatrix} a_1 & a_2 & a_3 & a_4 \end{pmatrix}$$

$$\Sigma_A = L_A L_A^t + T_A = \begin{pmatrix} a_1^2+\sigma_{A1}^2 & a_1a_2 & a_1a_3 & a_1a_4 \\ a_2a_1 & a_2^2+\sigma_{A2}^2 & a_2a_3 & a_2a_4 \\ a_3a_1 & a_3a_2 & a_3^2+\sigma_{A3}^2 & a_3a_4 \\ a_4a_1 & a_4a_2 & a_4a_3 & a_4^2+\sigma_{A4}^2 \end{pmatrix}$$

4 factor loadings ($a_1\, a_2\, a_3\, a_4$) and 4 residual variances ($\sigma_{A1}^2\, \sigma_{A2}^2\, \sigma_{A3}^2\, \sigma_{A4}^2$) = 8 parameters

Let's illustrate this in R using arbitrary (but sensible) parameter values. Run the following code:

```
# ->
LA=matrix(c(.6,.5,.7,.6),4,1)
TA=diag(c(.65,.75,.55,.65))
S_ = LA%*%t(LA) + TA
#
LA   # Factor loadings
TA    #  Residuals
S_      # expected covariance matrix
cov2cor(S_)  # correlation matrix
# <-
```

Note that the unconstrained estimate of the 4x4 ΣA contains 10 parameters (4 variances and 6 covariances), but the single factor model of ΣA ($LALA^t+TA$) contains 8 (as shown in the slide above)

Let's fit the IPM model and then move on to the CPM model. The code for the IPM model is given below (see Appendix for additional annotation). Run this code.

```
# ->
# AE IPM
nv=4
#
svb0=c(2.5,2.5,2.5,2.5)
#
# Create regression model for expected Mean Matrices
B0_     <- mxMatrix( type="Full", nrow=1, ncol=nv, free=TRUE, values=svb0,
labels=c("b01","b02","b03","b04"), name="b0" )
#
ExpMean <- mxAlgebra(expression=cbind(b0,b0), name='expMean')
#AE model
# Create Matrices for Variance Components
LA      <- mxMatrix( type="Full", nrow=nv, ncol=1, free=TRUE, values=c(.5,.5,.5,.5),
label=c("LA1","LA2","LA3","LA4"), name="LA" )
LD      <- mxMatrix( type="Full", nrow=nv, ncol=1, free=FALSE, values=0,
label=c("LD1","LD2","LD3","LD4"), name="LD" )
LE      <- mxMatrix( type="Full", nrow=nv, ncol=1, free=TRUE, values=c(.7,.7,.7,.7),
label=c("LE1","LE2","LE3","LE4"), name="LE"  )
#
TA      <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=c(.5,.5,.5,.5),
label=c("tA1","tA2","tA3","tA4"), name="TA" )
TD      <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=FALSE, values=0,
label=c("tD1","tD2","tD3","tD4"), name="TD" )
TE      <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=c(.7,.7,.7,.7),
label=c("tE1","tE2","tE3","tE4"), name="TE" )
#
# Create Algebra for expected Variance/Covariance Matrices in MZ & DZ twins
#
VA      <- mxAlgebra( expression= LA%*%t(LA) + TA, name="VA" )
VD      <- mxAlgebra( expression= LD%*%t(LD) + TD, name="VD" )
VE      <- mxAlgebra( expression= LE%*%t(LE) + TE, name="VE" )
#
covP      <- mxAlgebra( expression= VA+VD+VE, name="V" )
covMZ     <- mxAlgebra( expression= VA+VD, name="cMZ" )
covDZ     <- mxAlgebra( expression= 0.5%x%VA+.25%x%VD, name="cDZ" )
expCovMZ  <- mxAlgebra( expression= rbind( cbind(V, cMZ), cbind(t(cMZ), V)), name="expCovMZ" )
expCovDZ  <- mxAlgebra( expression= rbind( cbind(V, cDZ), cbind(t(cDZ), V)), name="expCovDZ" )
#AE IPM model
dataMZ    <- mxData( observed=N4mz, type="raw" )
dataDZ    <- mxData( observed=N4dz, type="raw" )
#AE IPM model
# Create Expectation Objects for Multiple Groups
expMZ     <- mxExpectationNormal( covariance="expCovMZ", means="expMean", dimnames=selVars )
expDZ     <- mxExpectationNormal( covariance="expCovDZ", means="expMean", dimnames=selVars )
funML     <- mxFitFunctionML()
#AE IPM model
pars      <- list(B0_, LA, TA, LD, TD, LE, TE, VA, VD, VE, covP)
modelMZ   <- mxModel( pars, ExpMean,  covMZ, expCovMZ, ExpMean, dataMZ, expMZ, funML,
name="MZ" )
modelDZ   <- mxModel( pars, ExpMean,   covDZ, expCovDZ, ExpMean, dataDZ, expDZ, funML,
name="DZ" )
multi     <- mxFitFunctionMultigroup( c("MZ","DZ") )
#AE IPM model
modelAE4IPM  <- mxModel('AE_IPM', pars, modelMZ, modelDZ, multi)
# ------------------------------------------------------------------------------------------
-------------------------
# Run AE IPM Model
fitAE4IPM    <- mxTryHard( modelAE4IPM, 20)
sumAE4IPM    <- summary( fitAE4IPM )
# <-
```

Question 2.7. Does the AE IPM model fit the data relative to the 4 variate AE model? Why is df=4 (`diffdf = 4`)?

```
# ->
mxCompare(fitAE4V,fitAE4IPM)
# <-
```

Finally let's take a look at the parameters of the AE IPM model.

```
# ->
estLA=round(fitAE4IPM$LA$values,3)  # A loadings
estTA=round(fitAE4IPM$TA$values,3)  # A residual variances
estLE=round(fitAE4IPM$LE$values,3)  # E loadings
estTE=round(fitAE4IPM$TE$values,3)  # E residual variances
#
t(estLA)     # A loadings
diag(estTA) # A residual variances
t(estLE)    # E loadings
diag(estTE)  # E residual variances
SA=estLA%*%t(estLA) + estTA  # A cov matrix
SE=estLE%*%t(estLE) + estTE  # E cov matrix
SPh=SA+SE     # expected phenotypic cov matrix
round(SA/SPh,3)   # proportions
round(SE/SPh,3)  # proportions
# <-
```

Consider first the A and E factor loadings. The factor loadings are all between .45 to .57.

```
> t(estLA)     # A loadings
      [,1]  [,2]  [,3] [,4]
[1,] 0.457 0.473 0.476 0.52
> t(estLE)    # E loadings
      [,1]  [,2]  [,3]  [,4]
[1,] 0.549 0.447 0.487 0.573
```

Question 2.8. Consider the A and E item residual variances. With respect to item specific A and E effect, what do you conclude? Bear in mind that the E residual variance include measurement error. For instance in the practical part 1, we analysed skinfold data. The correlation between skinfold measures and ultrasound measures of subcutaneous fat is between .80 and .90 (the ultrasound measurement is the *golden standard).*[3] That means that the skinfold measures are not perfect measures of subcutaneous fat, i.e., they are characterized by measurement error. The same applies to the neuroticism items.

```
> diag(estTA) # A residual variances
[1] 0.024 0.154 0.028 0.010
> diag(estTE)  # E residual variances
[1] 0.590 0.557 0.603 0.453
```

[3] Ryan-Stewart, H.; O'Leary, A.; Paine, E.; Faulkner, J.; Jobson, S. The Relationship between Skinfold and Ultrasound Measures of Subcutaneous Fat in Untrained Healthy Males. Appl. Sci. 2021, 11, 10561. https://doi.org/10.3390/ app112210561)

Reporting results

Based on this model, we can calculate the narrow-sense (due to A) and broad-sense (due to A and D) heritabilities. You might also want to represent the variance due to each of the variance components (common and unique) in a nice graph[4]. To do this, it is convenient to put all the variance components into their own objects. Do this by running the code below:

```
# ->
LA_res<-fitAE4IPM$LA$values # common A factor loadings
cvarA<-LA_res%*%t(LA_res) # common A variance matrix
TA_res<-diag(fitAE4IPM$TA$values) # residual A variance vector
totalA<-cvarA+diag(TA_res) # total A variance matrix
#
LE_res<-fitAE4IPM$LE$values # common E factor loadings
TE_res<-diag(fitAE4IPM$TE$values) # common E variance matrix
cvarE<-LE_res%*%t(LE_res) # residual E variance vector
totalE<-cvarE+diag(TE_res) # total E variance matrix
#
totalPh<-totalA +totalE # total phenotypic variance matrix
# <-
```

Now calculate the variance due to A (narrow sense heritability), and E:

```
# ->
diag(totalA)/diag(totalPh) # narrow sense h2
diag(totalE)/diag(totalPh) # Env
# <-
```

```
> diag(totalA)/diag(totalPh) # narrow sense h2
[1] 0.2071790 0.3328229 0.2323188 0.2649317
> diag(totalE)/diag(totalPh)
[1] 0.7928210 0.6671771 0.7676812 0.7350683
```

Now let's make the graph. First we represent the common and unique variances (for A, D, and E separately) in percentages:

```
# ->
commonA<-floor(diag(cvarA/totalPh)*100)
uniqueA<-floor(diag(TA_res/totalPh)*100)
commonE<-floor(diag(cvarE/totalPh)*100)
uniqueE<-floor(diag(TE_res/totalPh)*100)
# <-
```

Then prepare the data for your plot:

```
# ->
Trait<-as.factor(rep(c(1,2,3,4),4))
Source<-c(rep("Common A",nv),rep("Common E",nv),
          rep("Unique A",nv),rep("Unique E",nv))
VC<-rep("Total variance",16)
Perc<-c(commonA,commonE,uniqueA,uniqueE)
data<-data.frame(Trait,Source,VC,Perc)
```

[4] see Video 4 from the 15:30 mark onwards

```
#
# put the components in the right order
data$Source <- factor(data$Source, levels = c("Unique E","Common
E","Unique A", "Common A"))
# <-
```

Now let's make the actual plot.

```
# ->
library(ggplot2)
p1 <- ggplot(data=data, aes(x=Trait, y=Perc, fill=Source )) +
  geom_bar(stat="identity")  +
  facet_wrap(~factor(VC), scales = "free_x")    +
  xlab(" ") + ylab("% of variance") +
  scale_fill_manual(name = " ",values = c("Purple","Violet","Red","tomato")) +
  ylim(0,100) +
  theme_classic(base_size = 17, base_family = "sans") +
  theme(legend.position = "bottom",
   axis.text.x = element_text(angle = 45, vjust = 1,
   size = 14, hjust = 1),legend.text=element_text(size=18))
p1
# <-
```

After running this code, you have produced a plot similar as one shown on slide 19 of Video 4.

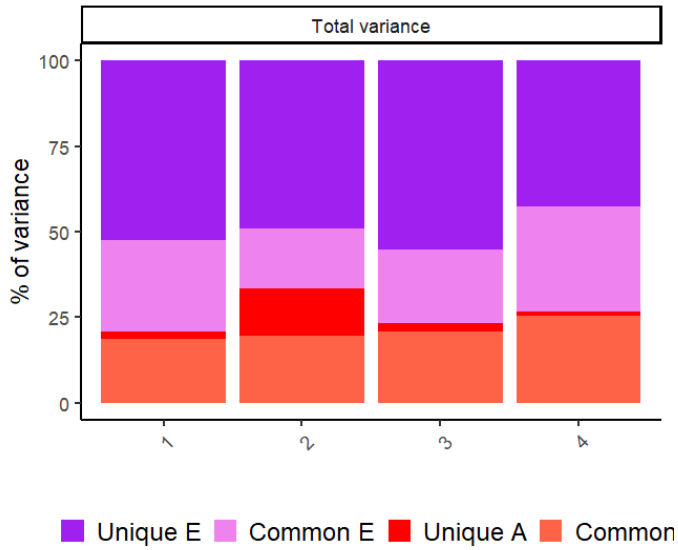
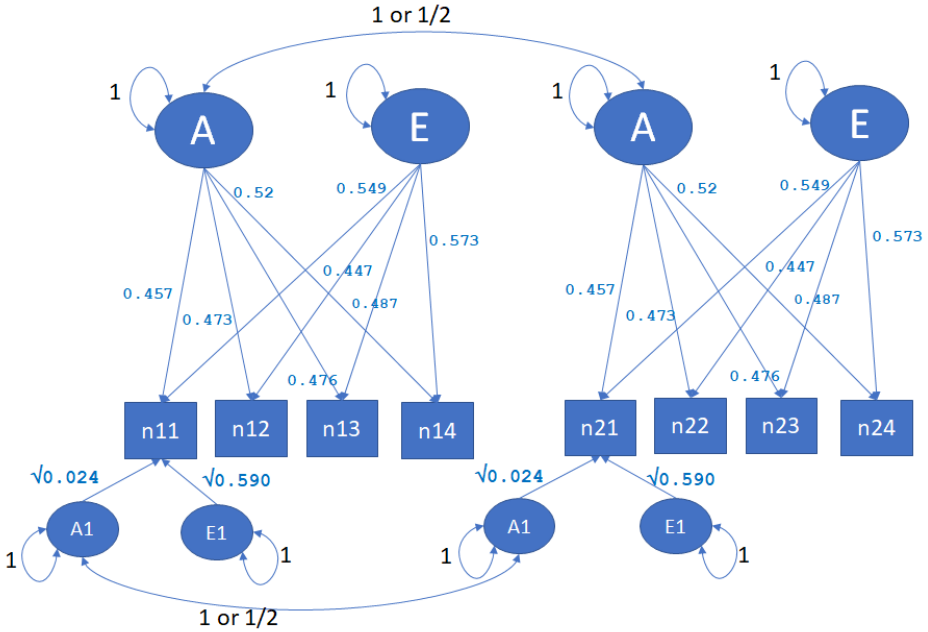


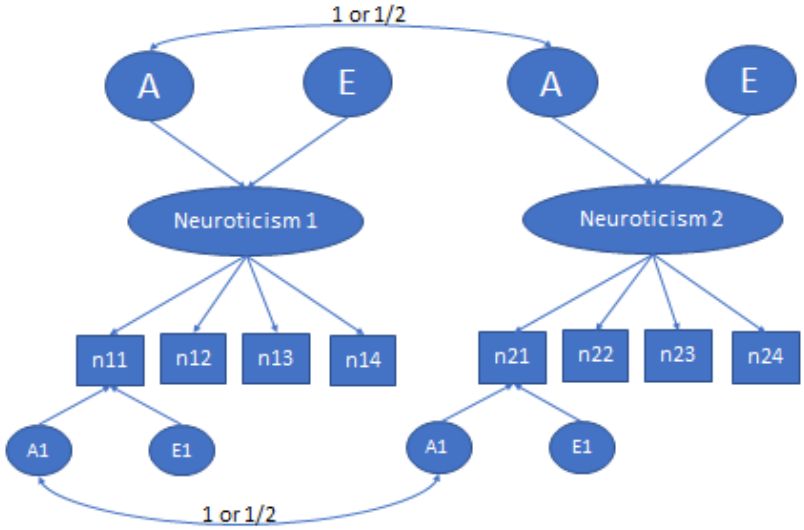
Figure: Unique E and A are item specific. "Common E" and "Common A" are due to the common A and common E factors. This Plot demonstrates the large contribution of item specific E ("Unique E") and the small contribution of item specific A ("Unique A"). Remember that the items specific E includes measurement error!

AE 4 variate twin Common Pathway Model (CPM)

The IPM path diagram is shown below, with only the residuals of item 1 shown (to avoid clutter). The parameters are taken from the fitAE4IPM output.



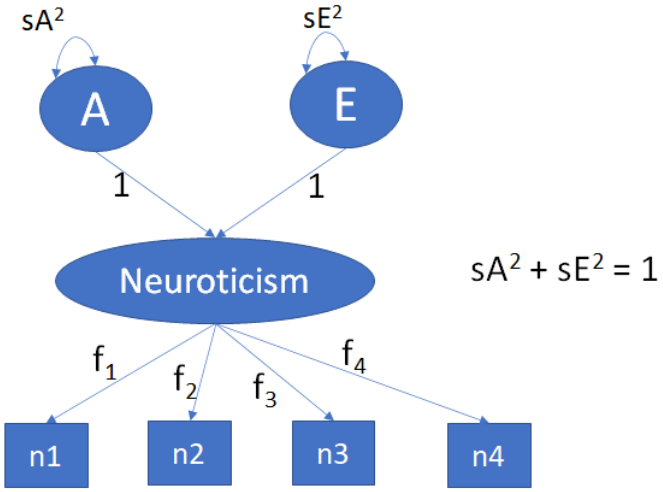
We will now fit the CPM. Here is the path diagram for twins, again only showing the residuals of item 1:



Boulder workshop 2022 IPM / CPM (Dolan & Pelt) 27

We can fit the CPM in two ways: A) using the IPM with proportionality constraints, or B) as depicted above. We explained the proportionality constraints in the ppt presentations. We 'll fit the model as depicted above. The OpenMx scripts for both specifications of CPM models (A and B) are annotated in the Appendix.

Let's fit the version B model, as shown below (residual variances not shown).



The variance of the latent variable Neuroticism equals $sA^2 + sE^2$, but to identify the factor loadings f_1, f_2, f_3, f_4, we have to scale the latent variable. That means that we have to assign a fixed values to its variance. Here we assigned the value 1, so that $sA^2 + sE^2 = 1$. Note that this model has the following parameters: 4 factor loadings, 4 A residual variances, 4 E residual variances (residual variances not shown above), 4 means and the parameter sA^2: 17 parameters. The parameter sE^2 is not counted because if can be derived as follows $sE^2 = 1 - sA^2$.

In the OpenMx code given below (see Appendix for additional annotation), the constraint is specified as follows:

```
# scaling constraint
con_stand <- mxConstraint(expression= (sA+sD+sE) == 1, name='standLF')
```

Don't be confused by `sD` in `(sA+sD+sE) == 1`: in the model specification `sD` is fixed to zero!

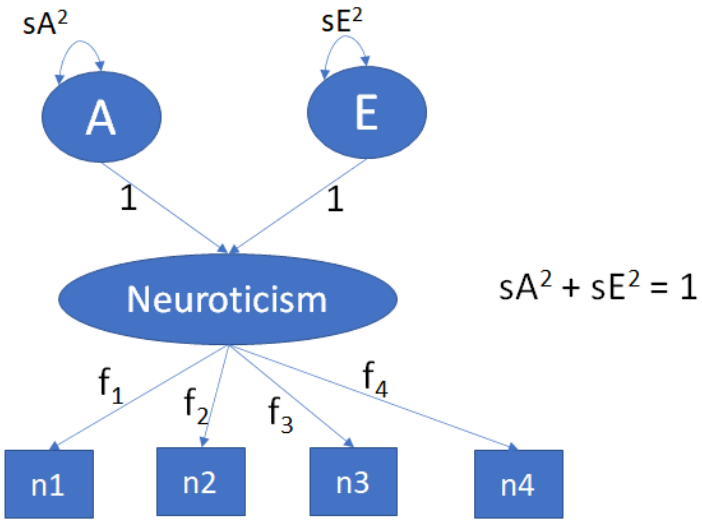
```
# ->
# CPM 2
nv=4
svb0=c(2,2,2,2)
#
# Create regression model for expected Mean Matrices
B0_      <- mxMatrix( type="Full", nrow=1, ncol=nv, free=TRUE, values=svb0,
labels=c("b01","b02","b03","b04"), name="b0" )
#
ExpMean <- mxAlgebra(expression=cbind(b0,b0), name='expMean')
# Create Matrices for Variance Components
LF       <- mxMatrix( type="Full", nrow=nv, ncol=1, free=TRUE, values=c(.5,.5,.5,.5),
label=c("Lf1","Lf2","Lf3","Lf4"), name="LF" )
SA       <- mxMatrix( type="Full", nrow=1, ncol=1, free=TRUE, values=c(.5), label=c("sA"),
name="SA" )
SD      <- mxMatrix( type="Full", nrow=1, ncol=1, free=FALSE, values=c(.0), label=c("sD"),
name="SD" )
SE       <- mxMatrix( type="Full", nrow=1, ncol=1, free=TRUE, values=c(.5), label=c("sE"),
name="SE" )
# scaling constraint
con_stand <- mxConstraint(expression= (sA+sD+sE) == 1, name='standLF')
# residual variances
TA       <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=c(.5,.5,.5,.5),
label=c("tA1","tA2","tA3","tA4"), name="TA" )
TD       <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=FALSE, values=0,
label=c("tD1","tD2","tD3","tD4"), name="TD" )
TE       <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=c(.7,.7,.7,.7),
label=c("tE1","tE2","tE3","tE4"), name="TE" )
# Create Algebra for expected Variance/Covariance Matrices in MZ & DZ twins
VA       <- mxAlgebra( expression= LF%*%(SA)%*%t(LF)  + TA, name="VA" )
VD       <- mxAlgebra( expression= LF%*%(SD)%*%t(LF) + TD, name="VD" )
VE       <- mxAlgebra( expression= LF%*%(SE)%*%t(LF) + TE, name="VE" )
#
covP       <- mxAlgebra( expression= VA+VD+VE, name="V" )
covMZ      <- mxAlgebra( expression= VA+VD, name="cMZ" )
covDZ      <- mxAlgebra( expression= 0.5%x%VA+.025%x%VD, name="cDZ" )
expCovMZ   <- mxAlgebra( expression= rbind( cbind(V, cMZ), cbind(t(cMZ), V)), name="expCovMZ" )
expCovDZ   <- mxAlgebra( expression= rbind( cbind(V, cDZ), cbind(t(cDZ), V)), name="expCovDZ" )
# Create Data Objects for Multiple Groups
#ACDE model
dataMZ     <- mxData( observed=N4mz, type="raw" )
dataDZ     <- mxData( observed=N4dz, type="raw" )
#ACDE model
# Create Expectation Objects for Multiple Groups
expMZ      <- mxExpectationNormal( covariance="expCovMZ", means="expMean", dimnames=selVars )
expDZ      <- mxExpectationNormal( covariance="expCovDZ", means="expMean", dimnames=selVars )
funML      <- mxFitFunctionML()
#
pars       <- list(B0_, LF, TA, TD, TE, SA, SD, SE, VA, VD,VE, covP)
modelMZ    <- mxModel( pars, ExpMean,  covMZ, expCovMZ, ExpMean, dataMZ, expMZ, funML,
name="MZ" )
modelDZ    <- mxModel( pars, ExpMean,   covDZ, expCovDZ, ExpMean, dataDZ, expDZ, funML,
name="DZ" )
multi      <- mxFitFunctionMultigroup( c("MZ","DZ") )
# model
modelAE4CPM2  <- mxModel("AE_CPM2", pars, modelMZ, modelDZ, multi,con_stand )
# RUN MODEL
fitAE4CPM2    <- mxTryHard( modelAE4CPM2, 20)
sumAE4CPM2    <- summary( fitAE4CPM2 )
# <-
```

Question 2.9. The AE IPM has 4 means, 2x4 (A and E) factor loadings and 2x4 (A and E) residual variances, i.e., 20 parameters. How many independent parameters does the CPM model have, and why is the df in `mxCompare(fitAE4IPM,fitAE4CPM2)` equal to 3?

Question 2.10. Does the AE CPM (version B) fit the data relative to the AE IPM?

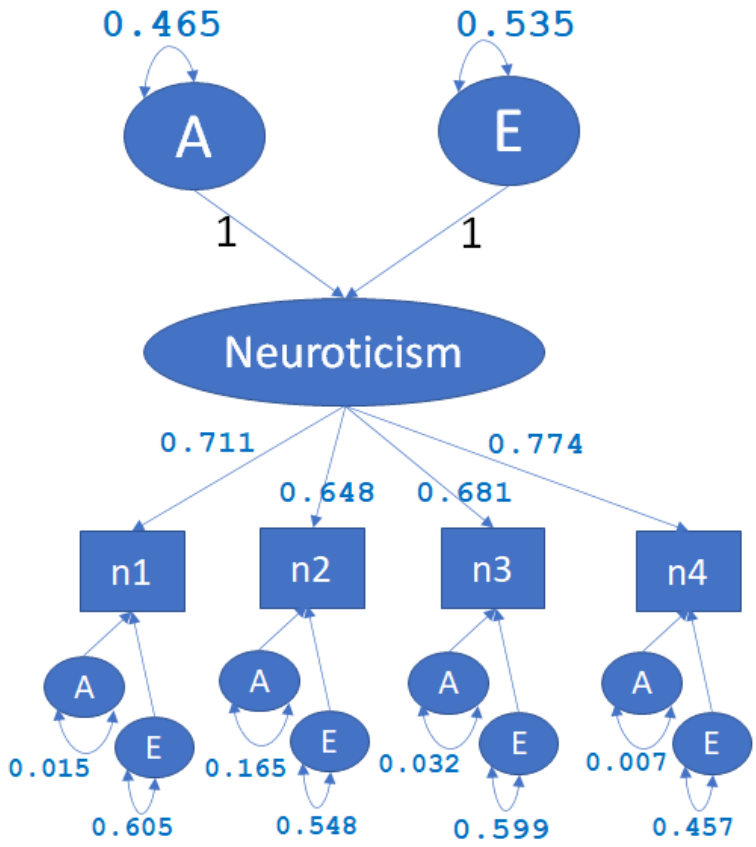
```
# ->
mxCompare(fitAE4IPM,fitAE4CPM2)
# <-
```

Extract the parameters from the output fitAECPM2,




```
# ->
FL=t(round(fitAE4CPM2$LF$values,3))
TA=diag(round(fitAE4CPM2$TA$values,3))
TE=diag(round(fitAE4CPM2$TE$values,3))
sA2=round(fitAE4CPM2$SA$values,3)
sE2=round(fitAE4CPM2$SE$values,3)
#
print(c(sA2, sE2))
print(FL)
print(TA)
print(TE)
# <-
```

```
> print(c(sA2, sE2))
[1] 0.465 0.535
> print(FL)
      [,1]  [,2]  [,3]  [,4]
[1,] 0.711 0.648 0.681 0.774
> print(TA)
[1] 0.015 0.165 0.032 0.007
> print(TE)
[1] 0.605 0.548 0.599 0.457
```



Question 2.11. What is the heritability of the latent variable Neuroticism, according to the fitAE4CPM2 output?

Using the CPM, we fitted the AE model to the latent variable Neuroticism. In practice, given a set of items comprising a test to measure neuroticism, one would often simply create the sum score based on the items scores, and conduct the analysis of the sum score. As a last step, let's fit the AE model to the sum score do that we can compare the decomposition of variance based on the sum score analysis and the CPM results. First create the sum scores and get the descriptives, and the twin correlations.

```
#->
Nmz=dim(N4mz)[1]
Ndz=dim(N4dz)[1]
sumNmz=matrix(0,Nmz,2)
sumNmz[,1]=N4mz[,1]+N4mz[,2]+N4mz[,3]+N4mz[,4]
sumNmz[,2]=N4mz[,5]+N4mz[,6]+N4mz[,7]+N4mz[,8]
sumNdz=matrix(0,Ndz,2)
sumNdz[,1]=N4dz[,1]+N4dz[,2]+N4dz[,3]+N4dz[,4]
sumNdz[,2]=N4dz[,5]+N4dz[,6]+N4dz[,7]+N4dz[,8]
#
colnames(sumNdz)=c('sumN1','sumN2')
sumNdz=as.data.frame(sumNdz)
colnames(sumNmz)=c('sumN1','sumN2')
sumNmz=as.data.frame(sumNmz)
#
describe(sumNmz, range=F, skew=F)
describe(sumNdz, range=F, skew=F)
```

```
#
round(cor(sumNmz),3)
round(cor(sumNdz),3)
#<-


> describe(sumNmz, range=F, skew=F)

      vars    n  mean   sd   se
sumN1    1 1528 10.12 3.20 0.08
sumN2    2 1528 10.08 3.26 0.08
> describe(sumNdz, range=F, skew=F)
      vars    n mean   sd   se
sumN1    1 1277 9.97 3.27 0.09
sumN2    2 1277 9.95 3.14 0.09

> round(cor(sumNmz),3)
      sumN1 sumN2
sumN1 1.000 0.383
sumN2 0.383 1.000
> round(cor(sumNdz),3)
      sumN1 sumN2
sumN1 1.000 0.179
sumN2 0.179 1.000
```

You can see that the standard deviation is about 3.2, so the variance is about 10. The means are about 10. The twin correlations still suggest an AE model as .179*2 equals .358, which is close to the MZ correlation of .383. Here is this univariate OpenMx script:

```
# ->
# univariate model AE model
# starting values
selVars=c('sumN1','sumN2')
nv=1  # number of phenotypes
ntv=2   # times number of twins
svVA=10*.4   #  A variance
svVE=10*.6   #  E variance
svVD=10*.0    # D variance = 0
svb0=10   # mean approx
# Create regression model for expected Mean Matrices
ExpMean    <- mxMatrix( type="Full", nrow=1, ncol=ntv, free=TRUE, values=svb0,
labels=c("b0","b0"), name="expMean" )
# ADE model
# Create Matrices for Variance Components
covA      <- mxMatrix( type="Symm", nrow=nv, ncol=nv, free=TRUE, values=svVA, label="VA11",
name="VA" )
covD      <- mxMatrix( type="Symm", nrow=nv, ncol=nv, free=FALSE, values=svVD, label="VD11",
name="VD" )
covE      <- mxMatrix( type="Symm", nrow=nv, ncol=nv, free=TRUE, values=svVE, label="VE11",
name="VE" )
# Create Algebra for expected Variance/Covariance Matrices in MZ & DZ twins
covP      <- mxAlgebra( expression= VA+VD+VE, name="V" )
covMZ     <- mxAlgebra( expression= VA+VD, name="cMZ" )
covDZ     <- mxAlgebra( expression= 0.5%x%VA+.25%x%VD, name="cDZ" )
expCovMZ  <- mxAlgebra( expression= rbind( cbind(V, cMZ), cbind(t(cMZ), V)), name="expCovMZ" )
expCovDZ  <- mxAlgebra( expression= rbind( cbind(V, cDZ), cbind(t(cDZ), V)), name="expCovDZ" )
# Create Data Objects for Multiple Groups
dataMZ    <- mxData( observed=sumNmz, type="raw" )
dataDZ    <- mxData( observed=sumNdz, type="raw" )
# Create Expectation Objects for Multiple Groups
expMZ     <- mxExpectationNormal( covariance="expCovMZ", means="expMean", dimnames=selVars )
expDZ     <- mxExpectationNormal( covariance="expCovDZ", means="expMean", dimnames=selVars )
funML     <- mxFitFunctionML()
pars      <- list(covA, covD, covE, covP )
modelMZ   <- mxModel( pars, ExpMean, covMZ, expCovMZ, ExpMean, dataMZ, expMZ, funML, name="MZ"
)
modelDZ   <- mxModel( pars, ExpMean,  covDZ, expCovDZ, ExpMean, dataDZ, expDZ, funML,
name="DZ" )
multi     <- mxFitFunctionMultigroup( c("MZ","DZ") )
# Create Algebra for Variance Components
rowVC     <- rep('VarComp',nv)
colVC     <- rep(c('VA','VD','VE','SA','SD','SE'),each=nv)
estVC     <- mxAlgebra( expression=cbind(VA,VD,VE,VA/V,VD/V,VE/V), name="VarComp",
dimnames=list(rowVC,colVC) )
ciAE     <- mxCI( "VarComp[1,1:8]" )
# Build Model with Confidence Intervals
modelAE  <- mxModel('univAE', pars, modelMZ, modelDZ, multi, estVC, ciAE )
# Run ADE Model
fitAE    <- mxRun( modelAE, intervals=F)
sumAE    <- summary( fitAE )
# <-
```

Question 2.12. What is the heritability of the Neuroticism sum score and why does it differ from the heritability based on the CPM (see answer to question 3.11)? The variance components can be obtained from the OpenMx output as follows `fitAE$VarComp`.

End of Practical

NOTE: Below there are 4 Appendices, in which the OpenMx scripts are annotated.

This is "homework", if you like, but not practical material!

Appendix 1: 4 variate ADE model (skinfold data): OpenMx input with additional annotation.

The OpenMx scripts in this practical all specify a model for the phenotypic means and covariance matrices. In the ADE model script below, we estimate the MZ and DZ covariance matrix subject to ADE decomposition, and, at the same time we regress the phenotypes on sex, coded 0/1. As such, we model the covariance matrices, corrected for sex differences in the phenotypic means.

```
# ->
## ADE model with sex as covariate
# ------------------------------------------------------------------------
selVars = c("bic_T1","tri_T1","ssc_T1","sil_T1","bic_T2","tri_T2","ssc_T2","sil_T2")  # the
array with the variable names
# PREPARE MODEL
#
nv=4  # number of variable
ntv=2*nv  # number of variables times 2 (twins)
# starting values based on the phenotypic cov matrix
```

Starting values as discussed in the practical.

```
svS=matrix(c(
  14,     12,     15,     18,
  12,     12,     14,     17,
  15,     14,     23,     25,
  18,     17,     25,     31),4,4)
#
svVA=svS*.4  # rough guess of A cov matrix
svVE=svS*.2  # rough guess of E cov matrix
svVD=svS*.4  # rough guess of D cov matrix
rvA=round(cov2cor(svVA),2) # rough guess of A correlation matrix
rvD=round(cov2cor(svVE),2) # rough guess of E correlation matrix
rvE=round(cov2cor(svVD),2) # rough guess of D correlation matrix
svsA=sqrt(diag(svVA)) # A standard deviations
svsD=sqrt(diag(svVD)) # D standard deviations
svsE=sqrt(diag(svVE)) # E standard deviations
```

The following code takes care of the means, that is the means are modeled as follows:

mean(biceps|sex) = b0 + b1*sex = `b0b + b1b*data.sex_T1`
So that is mean(biceps | sex= 0) = b0 and mean(biceps | sex=1) = b0 + b1.

The starting values of `b0b` and `b1b` are 18 and 0, respectively. See first values in svb0 and svb1. You can see the means in the last line:

```
ExpMean <- mxAlgebra(expression=cbind(b0sex+b1sex*sex1,b0sex+b1sex*sex2), name='expMean')


svb0=c(18,21,20,20)
svb1=c(.0,.0,.0,.0)
#
# Create regression model for expected Mean Matrices
B0_     <- mxMatrix( type="Full", nrow=1, ncol=nv, free=TRUE, values=svb0,
labels=c("b0b","b0t","b0s","b0si"), name="b0sex" )
B1_     <- mxMatrix( type="Full", nrow=1, ncol=nv, free=TRUE, values=svb1,
labels=c("b1b","b1t","b1s","b1si"), name="b1sex" )
Sex1 <- mxMatrix(type="Full", nrow=1, ncol=1, free=FALSE, labels=c('data.sex_T1'),name="sex1")
# sex twin 1
Sex2 <- mxMatrix(type="Full", nrow=1, ncol=1, free=FALSE, labels=c('data.sex_T2'),name="sex2")
# sex twin 2
#
ExpMean <- mxAlgebra(expression=cbind(b0sex+b1sex*sex1,b0sex+b1sex*sex2), name='expMean')
```

The following code takes care of the covariance matrices in the MZ and DZ groups. Remember that we're modeling the MZ and DZ covariance matrices as shown below. In the slide below, ΣA, ΣD and ΣE are the 4x4 A, D, and E matrices. However, we not estimating covariance matrices. Rather, we are estimated the A, D, and E correlation matrices (each 6 correlations RA, RD, RE) and A, D, and E standard deviations (each 4 standard deviations, sA, sD, sE). Note that the ΣA, ΣD and ΣE are called VA, VD, and VE in the OpenMx script.

```
VA      <- mxAlgebra( expression= sA%*%RA%*%t(sA), name="VA" )
VD      <- mxAlgebra( expression= sD%*%RD%*%t(sD), name="VD" )
VE      <- mxAlgebra( expression= sE%*%RE%*%t(sE), name="VE" )
```

The generalization from p=1 (univariate) to p=2 (bivariate) to p>2 (multivariate).

Σ_{Ph} MZ	p phenotypes	p phenotypes
	$\Sigma_A + \Sigma_D + \Sigma_E$	$\Sigma_A + \Sigma_D$
	$\Sigma_A + \Sigma_D$	$\Sigma_A + \Sigma_D + \Sigma_E$

Σ_{Ph} DZ	p phenotypes	p phenotypes
	$\Sigma_A + \Sigma_D + \Sigma_E$	$\frac{1}{2}\Sigma_A + \frac{1}{4}\Sigma_D$
	$\frac{1}{2}\Sigma_A + \frac{1}{4}\Sigma_D$	$\Sigma_A + \Sigma_D + \Sigma_E$

Biceps and Triceps (purple), ~~Calf (green),~~
Subscapular (blue), Suprailiacal (orange).

Boulder workshop 2022 IPM / CPM (Dolan & Pelt) 5

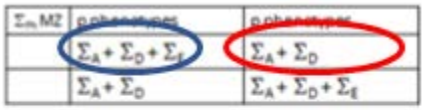
```
# Create Matrices for Variance Components
corA      <- mxMatrix( type="Stand", nrow=nv, ncol=nv, free=TRUE, values=rvA,
label=c("rA21","rA31","rA41","rA32","rA42","rA43"), name="RA" )
corD      <- mxMatrix( type="Stand", nrow=nv, ncol=nv, free=TRUE, values=rvD,
label=c("rD21","rD31","rD41","rD32","rD42","rD43"), name="RD" )
corE      <- mxMatrix( type="Stand", nrow=nv, ncol=nv, free=TRUE, values=rvE,
label=c("rE21","rE31","rE41","rE32","rE42","rE43"), name="RE" )
#
sdA      <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=svsA,
label=c("sA1","sA2","sA3","sA4"), name="sA" )
sdD      <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=svsD,
label=c("sD1","sD2","sD3","sD4"), name="sD" )
sdE      <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=svsE,
label=c("sE1","sE2","sE3","sE4"), name="sE" )
#
# Create Algebra for expected Variance/Covariance Matrices in MZ & DZ twins
#
VA      <- mxAlgebra( expression= sA%*%RA%*%t(sA), name="VA" )
VD      <- mxAlgebra( expression= sD%*%RD%*%t(sD), name="VD" )
VE      <- mxAlgebra( expression= sE%*%RE%*%t(sE), name="VE" )
```

Once we have the 4x4 covariance matrices VA, VD, and VE, we calculate the phenotype 4x4 covariance matrix:

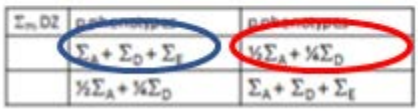
```
covP      <- mxAlgebra( expression= VA+VD+VE, name="V" )
```

and we calculated

```
covMZ     <- mxAlgebra( expression= VA+VD, name="cMZ" )
covDZ     <- mxAlgebra( expression= 0.5%x%VA+.25%x%VD, name="cDZ" )
```



Σ_{mMZ}	p phenotypes	p phenotypes
	$\Sigma_A + \Sigma_D + \Sigma_E$	$\Sigma_A + \Sigma_D$
	$\Sigma_A + \Sigma_D$	$\Sigma_A + \Sigma_D + \Sigma_E$

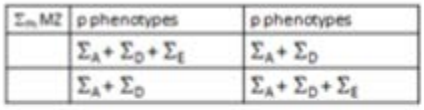


Σ_{mDZ}	p phenotypes	p phenotypes
	$\Sigma_A + \Sigma_D + \Sigma_E$	$\frac{1}{2}\Sigma_A + \frac{1}{4}\Sigma_D$
	$\frac{1}{2}\Sigma_A + \frac{1}{4}\Sigma_D$	$\Sigma_A + \Sigma_D + \Sigma_E$

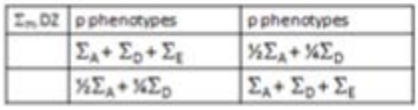
covP circled in blue, covMZ and covDZ circled in red.

We then assemble the 8x8 phenotypic covariance matrices.

```
expCovMZ  <- mxAlgebra( expression= rbind( cbind(V, cMZ), cbind(t(cMZ), V)), name="expCovMZ" )
expCovDZ  <- mxAlgebra( expression= rbind( cbind(V, cDZ), cbind(t(cDZ), V)), name="expCovDZ" )
```



Σ_{mMZ}	p phenotypes	p phenotypes
	$\Sigma_A + \Sigma_D + \Sigma_E$	$\Sigma_A + \Sigma_D$
	$\Sigma_A + \Sigma_D$	$\Sigma_A + \Sigma_D + \Sigma_E$



Σ_{mDZ}	p phenotypes	p phenotypes
	$\Sigma_A + \Sigma_D + \Sigma_E$	$\frac{1}{2}\Sigma_A + \frac{1}{4}\Sigma_D$
	$\frac{1}{2}\Sigma_A + \frac{1}{4}\Sigma_D$	$\Sigma_A + \Sigma_D + \Sigma_E$

```
# Create Data Objects for Multiple Groups
dataMZ    <- mxData( observed=mzskinfold, type="raw" )
dataDZ    <- mxData( observed=dzskinfold, type="raw" )
#
# Create Expectation Objects for Multiple Groups
expMZ     <- mxExpectationNormal( covariance="expCovMZ", means="expMean", dimnames=selVars )
expDZ     <- mxExpectationNormal( covariance="expCovDZ", means="expMean", dimnames=selVars )
funML     <- mxFitFunctionML()
#
pars      <- list(B0_, B1_, corA, corD, corE, sdA, sdD, sdE, VA, VD, VE, covP )
modelMZ   <- mxModel( pars, ExpMean,  Sex1, Sex2, covMZ, expCovMZ, ExpMean, dataMZ, expMZ,
funML, name="MZ" )
modelDZ   <- mxModel( pars, ExpMean,  Sex1, Sex2, covDZ, expCovDZ, ExpMean, dataDZ, expDZ,
funML, name="DZ" )
multi     <- mxFitFunctionMultigroup( c("MZ","DZ") )
#
modelADE4  <- mxModel("ACE4", pars, modelMZ, modelDZ, multi)
# Run Model
fitADE4    <- mxRun( modelADE4 )
sumADE4    <- summary( fitADE4 )
mxCompare(fitSAT4, fitADE4)
# <-
```

Appendix 2: 4 variate AE Independent Pathway Model (neuroticism items): OpenMx input with additional annotation.

In the AE IPM model script below, we estimate the MZ and DZ covariance matrix subject to AE decomposition, and at the same time fit a single common factor model to the 4x4 A and 4x4 E covariance matrices. We also estimate the phenotype means. We have no covariates (like sex or age), so the model for the phenotypic means is simple.

```
# ->
# AE IPM
nv=4
#
```

We have no covariates (like sex or age), so the model for the phenotypic means is simple: we just estimate the 4 means. svb0 are the starting values, which are sensible (based on the descriptives of the data).

```
svb0=c(2.5,2.5,2.5,2.5)
# Create regression model for expected Mean Matrices
B0_     <- mxMatrix( type="Full", nrow=1, ncol=nv, free=TRUE, values=svb0,
labels=c("b01","b02","b03","b04"), name="b0" )
#
ExpMean <- mxAlgebra(expression=cbind(b0,b0), name='expMean')
```

The covariance matrices are modeled as shown below. Rather than estimating ΣA and ΣE, we are estimating these subject to $\Sigma A = LAL A^t + TA$ and $\Sigma E = LEL E^t + TE$.

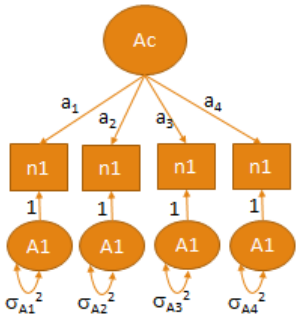
MZ	p phenotypes	p phenotypes
	$\Sigma_A + \Sigma_E$	Σ_A
	Σ_A	$\Sigma_A + \Sigma_E$

MZ	p phenotypes	p phenotypes
	$L_A L_A^t + T_A + L_E L_E^t + T_E$	$L_A L_A^t + T_A$
	$L_A L_A^t + T_A$	$L_A L_A^t + T_A + L_E L_E^t + T_E$

$$\Sigma_A = L_A L_A^t + T_A$$
$$\Sigma_E = L_E L_E^t + T_E$$

DZ	p phenotypes	p phenotypes
	$\Sigma_A + \Sigma_E$	$.5*\Sigma_A$
	$.5*\Sigma_A$	$\Sigma_A + \Sigma_E$

DZ	p phenotypes	p phenotypes
	$L_A L_A^t + T_A + L_E L_E^t + T_E$	$.5*(L_A L_A^t + T_A)$
	$.5*(L_A L_A^t + T_A)$	$L_A L_A^t + T_A + L_E L_E^t + T_E$



$$L_A = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} \quad T_A = \begin{bmatrix} \sigma_{A1}^2 & 0 & 0 & 0 \\ 0 & \sigma_{A2}^2 & 0 & 0 \\ 0 & 0 & \sigma_{A3}^2 & 0 \\ 0 & 0 & 0 & \sigma_{A4}^2 \end{bmatrix}$$

$$L_A^t = \begin{bmatrix} a_1 & a_2 & a_3 & a_4 \end{bmatrix}$$

$$\Sigma_A = L_A L_A^t + T_A = \begin{bmatrix} a_1^2+\sigma_{A1}^2 & a_1a_2 & a_1a_3 & a_1a_4 \\ a_2a_1 & a_2^2+\sigma_{A2}^2 & a_2a_3 & a_2a_4 \\ a_3a_1 & a_3a_2 & a_3^2+\sigma_{A3}^2 & a_3a_4 \\ a_4a_1 & a_4a_2 & a_4a_3 & a_4^2+\sigma_{A4}^2 \end{bmatrix}$$

4 factor loadings ($a_1\, a_2\, a_3\, a_4$) and 4 residual variances ($\sigma_{A1}^2\, \sigma_{A2}^2\, \sigma_{A3}^2\, \sigma_{A4}^2$) = 8 parameters

Here are the parameter matrices (note that the D parameters are all fixed to zero).

```
#AE model
# Create Matrices for Variance Components
LA      <- mxMatrix( type="Full", nrow=nv, ncol=1, free=TRUE, values=c(.5,.5,.5,.5),
label=c("LA1","LA2","LA3","LA4"), name="LA" )
LD      <- mxMatrix( type="Full", nrow=nv, ncol=1, free=FALSE, values=0,
label=c("LD1","LD2","LD3","LD4"), name="LD" )
LE      <- mxMatrix( type="Full", nrow=nv, ncol=1, free=TRUE, values=c(.7,.7,.7,.7),
label=c("LE1","LE2","LE3","LE4"), name="LE"  )
#
TA      <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=c(.5,.5,.5,.5),
label=c("tA1","tA2","tA3","tA4"), name="TA" )
TD      <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=FALSE, values=0,
label=c("tD1","tD2","tD3","tD4"), name="TD" )
TE      <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=c(.7,.7,.7,.7),
label=c("tE1","tE2","tE3","tE4"), name="TE" )
#
# Create Algebra for expected Variance/Covariance Matrices in MZ & DZ twins
#
```

Here the 4x4 covariance matrices VA and VE are calculated.

```
VA      <- mxAlgebra( expression= LA%*%t(LA) + TA, name="VA" )
VD      <- mxAlgebra( expression= LD%*%t(LD) + TD, name="VD" )
VE      <- mxAlgebra( expression= LE%*%t(LE) + TE, name="VE" )
#
```

Once we have the 4x4 covariance matrices VA, VD, and VE, we calculate the phenotype 4x4 covariance matrix (covP) and create the MZ and DZ covariance matrices, just like we did in the saturated model.

```
covP      <- mxAlgebra( expression= VA+VD+VE, name="V" )
covMZ     <- mxAlgebra( expression= VA+VD, name="cMZ" )
covDZ     <- mxAlgebra( expression= 0.5%x%VA+.25%x%VD, name="cDZ" )
expCovMZ  <- mxAlgebra( expression= rbind( cbind(V, cMZ), cbind(t(cMZ), V)), name="expCovMZ" )
expCovDZ  <- mxAlgebra( expression= rbind( cbind(V, cDZ), cbind(t(cDZ), V)), name="expCovDZ" )
#AE IPM model
```

Σ_{mMZ}	p phenotypes	p phenotypes
	$\Sigma_A + \Sigma_D + \Sigma_E$	$\Sigma_A + \Sigma_D$
	$\Sigma_A + \Sigma_D$	$\Sigma_A + \Sigma_D + \Sigma_E$

Σ_{mDZ}	p phenotypes	p phenotypes
	$\Sigma_A + \Sigma_D + \Sigma_E$	$\frac{1}{2}\Sigma_A + \frac{1}{4}\Sigma_D$
	$\frac{1}{2}\Sigma_A + \frac{1}{4}\Sigma_D$	$\Sigma_A + \Sigma_D + \Sigma_E$

```
dataMZ    <- mxData( observed=N4mz, type="raw" )
dataDZ    <- mxData( observed=N4dz, type="raw" )
#AE IPM model
# Create Expectation Objects for Multiple Groups
expMZ     <- mxExpectationNormal( covariance="expCovMZ", means="expMean", dimnames=selVars )
expDZ     <- mxExpectationNormal( covariance="expCovDZ", means="expMean", dimnames=selVars )
funML     <- mxFitFunctionML()
#AE IPM model
pars      <- list(B0_, LA, TA, LD, TD, LE, TE, VA, VD, VE, covP)
modelMZ   <- mxModel( pars, ExpMean,  covMZ, expCovMZ, ExpMean, dataMZ, expMZ, funML,
name="MZ" )
modelDZ   <- mxModel( pars, ExpMean,   covDZ, expCovDZ, ExpMean, dataDZ, expDZ, funML,
name="DZ" )
multi     <- mxFitFunctionMultigroup( c("MZ","DZ") )
#AE IPM model
modelAE4IPM  <- mxModel('AE_IPM', pars, modelMZ, modelDZ, multi)
# ------------------------------------------------------------
# Run AE IPM Model
fitAE4IPM    <- mxTryHard( modelAE4IPM, 20)
sumAE4IPM    <- summary( fitAE4IPM )
# <-
```

Appendix 3: 4 variate AE Common Pathway Model (neuroticism items), version 1 (proportionality constraints): OpenMx input with additional annotation.

The common pathway model can be formulated as a constrained IPM. This was explained using the following slide. The reasoning is this: Suppose that the true model is the CPM (parameters in red), but you fit the IPM (parameters in blue). Equal the parameters (blue = red) and note that the ratios a1/e1 = a2/e2 = a3/e3 = a4/e4.

constraint: $a^2 + e^2 = 1$ 5 parameters		$\sigma_A^2 = 1, \sigma_E^2 = 1$ 8 parameters	
A(CPM)	E(CPM)	A(IPM)	E(IPM)
$a*f_1$	$e*f_1$	a_1	e_1
$a*f_2$	$e*f_2$	a_2	e_2
$a*f_3$	$e*f_3$	a_3	e_3
$a*f_4$	$e*f_4$	a_4	e_4

constraints to turn IPM into CPM		proportionality constraint
A(IPM) = A(CPM)	E(IPM) = E(CPM)	if CPM model is the true model
$a_1 = a*f_1$	$e_1 = e*f_1$	$a_1/e_1 = a*f_1 / e*f_1 = a/e$
$a_2 = a*f_2$	$e_2 = e*f_2$	$a_2/e_2 = a*f_2 / e*f_2 = a/e$
$a_3 = a*f_3$	$e_3 = e*f_3$	$a_3/e_3 = a*f_3 / e*f_3 = a/e$
$a_4 = a*f_4$	$e_4 = e*f_4$	$a_4/e_4 = a*f_4 / e*f_4 = a/e$

If the CPM is the true model, but we fit the IPM model we expect: $a_1/e_1 = a_2/e_2 = a_3/e_3 = a_4/e_4$
Constrained IPM model (= CPM model) by estimating a_1 and e_1, e_2, e_3, e_4 and constraining a_2, a_3, a_4

proportionality		constraints
$a_1/e_1 = a_2/e_2$	→	$a_2 = (a_1/e_1)*e_2$
$a_1/e_1 = a_3/e_3$	→	$a_3 = (a_1/e_1)*e_3$
$a_1/e_1 = a_4/e_4$	→	$a_4 = (a_1/e_1)*e_4$

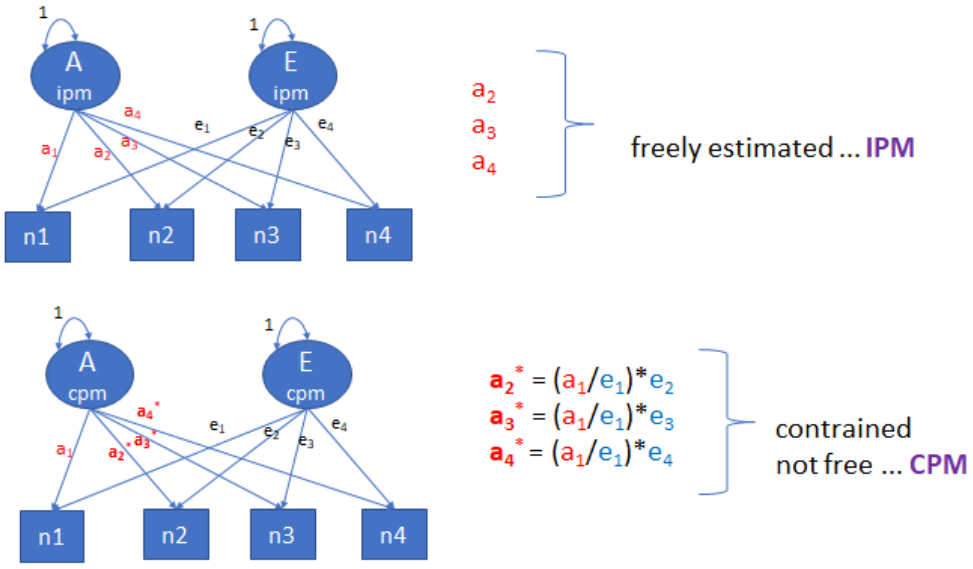
the CPM is nested under the IPM
5 (CPM) vs 8 (IPM) parameters

Boulder workshop 2022 IPM / CPM (Dolan & Pelt) 18

We introduced the constraints as follows in to the IPM script:

```
#
cLA2  <- mxConstraint(expression=LA2==(LA1/LE1)*LE2, name='conLA2')
cLA3  <- mxConstraint(expression=LA3==(LA1/LE1)*LE3, name='conLA3')
cLA4  <- mxConstraint(expression=LA4==(LA1/LE1)*LE4, name='conLA4')
#
...
#
modelAE4CPM  <- mxModel("AE_CPM1", pars, modelMZ, modelDZ, multi,cLA2,cLA3,cLA4 )
#
```

Note: in the following figures, the residual variances not shown



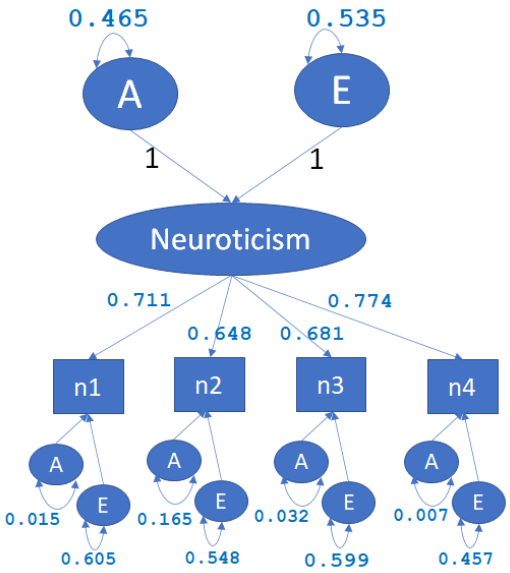
In the OpenMx script, these constraints are implemented as follows:

```
cLA2  <- mxConstraint(expression=LA2==(LA1/LE1)*LE2, name='conLA2')
cLA3  <- mxConstraint(expression=LA3==(LA1/LE1)*LE3, name='conLA3')
cLA4  <- mxConstraint(expression=LA4==(LA1/LE1)*LE4, name='conLA4')
```

The complete CPM version 1 is given below.

```
# ->
# CPM AE version 1 ------------------------------------------------------------------
nv=4
svb0=c(2,2,2,2)
# Create regression model for expected Mean Matrices
B0_     <- mxMatrix( type="Full", nrow=1, ncol=nv, free=TRUE, values=svb0,
labels=c("b01","b02","b03","b04"), name="b0" )
#
ExpMean <- mxAlgebra(expression=cbind(b0,b0), name='expMean')
#AE model
# Create Matrices for Variance Components
LA      <- mxMatrix( type="Full", nrow=nv, ncol=1, free=TRUE, values=c(.5,.5,.5,.5),
label=c("LA1","LA2","LA3","LA4"), name="LA" )
LD      <- mxMatrix( type="Full", nrow=nv, ncol=1, free=FALSE, values=0, label=c("LD1","LD2","LD3","LD4"),
name="LD" )
LE      <- mxMatrix( type="Full", nrow=nv, ncol=1, free=TRUE, values=c(.7,.7,.7,.7),
label=c("LE1","LE2","LE3","LE4"), name="LE"  )
#
TA      <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=c(.5,.5,.5,.5),
label=c("tA1","tA2","tA3","tA4"), name="TA" )
TD      <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=FALSE, values=0, label=c("tD1","tD2","tD3","tD4"),
name="TD" )
TE      <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=c(.7,.7,.7,.7),
label=c("tE1","tE2","tE3","tE4"), name="TE" )
#
cLA2  <- mxConstraint(expression=LA2==(LA1/LE1)*LE2, name='conLA2')
cLA3  <- mxConstraint(expression=LA3==(LA1/LE1)*LE3, name='conLA3')
cLA4  <- mxConstraint(expression=LA4==(LA1/LE1)*LE4, name='conLA4')
#
# Create Algebra for expected Variance/Covariance Matrices in MZ & DZ twins
VA      <- mxAlgebra( expression= LA%*%t(LA) + TA, name="VA" )
VD      <- mxAlgebra( expression= LD%*%t(LD) + TD, name="VD" )
VE      <- mxAlgebra( expression= LE%*%t(LE) + TE, name="VE" )
#
covP      <- mxAlgebra( expression= VA+VD+VE, name="V" )
covMZ     <- mxAlgebra( expression= VA+VD, name="cMZ" )
covDZ     <- mxAlgebra( expression= 0.5%x%VA+.25%x%VD, name="cDZ" )
expCovMZ  <- mxAlgebra( expression= rbind( cbind(V, cMZ), cbind(t(cMZ), V)), name="expCovMZ" )
expCovDZ  <- mxAlgebra( expression= rbind( cbind(V, cDZ), cbind(t(cDZ), V)), name="expCovDZ" )
# Create Data Objects for Multiple Groups
dataMZ    <- mxData( observed=N4mz, type="raw" )
dataDZ    <- mxData( observed=N4dz, type="raw" )
# Create Expectation Objects for Multiple Groups
expMZ     <- mxExpectationNormal( covariance="expCovMZ", means="expMean", dimnames=selVars )
expDZ     <- mxExpectationNormal( covariance="expCovDZ", means="expMean", dimnames=selVars )
funML     <- mxFitFunctionML()
#CPM AE model
pars      <- list(B0_, LA, TA, LD, TD,LE, TE, VA, VD,VE, covP)
modelMZ   <- mxModel( pars, ExpMean,  covMZ, expCovMZ, ExpMean, dataMZ, expMZ, funML, name="MZ" )
modelDZ   <- mxModel( pars, ExpMean,   covDZ, expCovDZ, ExpMean, dataDZ, expDZ, funML, name="DZ" )
multi     <- mxFitFunctionMultigroup( c("MZ","DZ") )
#CPM AE model
# Build Model with Confidence Intervals
modelAE4CPM  <- mxModel("AE_CPM1", pars, modelMZ, modelDZ, multi,cLA2,cLA3,cLA4 )
# Run Model
fitAE4CPM    <- mxTryHard( modelAE4CPM, 20)
sumAE4CPM    <- summary( fitAE4CPM )
# <-
```

Appendix 4: 4 variate AE Common Pathway Model (neuroticism items), version 2: OpenMx input with additional annotation. As used in the practical.




```
# ->
# CPM 2
nv=4
```

We have no covariates (like sex or age), so the model for the phenotypic means is simple: we just estimate the 4 means. svb0 are the starting values, which are sensible (based on the descriptives of the data).

```
svb0=c(2,2,2,2)
#
# Create regression model for expected Mean Matrices
B0_     <- mxMatrix( type="Full", nrow=1, ncol=nv, free=TRUE, values=svb0,
labels=c("b01","b02","b03","b04"), name="b0" )
#
ExpMean <- mxAlgebra(expression=cbind(b0,b0), name='expMean')
# Create Matrices for Variance Components
```

In the following LF is the 4x1 matrix of factor loadings (f1,f2,f3,f4). All D parameters are fixed to zero. SA and SE are the A and the E variances. Note that var(Neuroticism) = SA + SE = 1. This SA + SE = 1 is achieved by specificing this constraint (`con_stand <- mxConstraint(expression= (sA+sD+sE) == 1, name='standLF')`).

```
LF      <- mxMatrix( type="Full", nrow=nv, ncol=1, free=TRUE, values=c(.5,.5,.5,.5),
label=c("Lf1","Lf2","Lf3","Lf4"), name="LF" )
SA      <- mxMatrix( type="Full", nrow=1, ncol=1, free=TRUE, values=c(.5), label=c("sA"),
name="SA" )
SD      <- mxMatrix( type="Full", nrow=1, ncol=1, free=FALSE, values=c(.0), label=c("sD"),
name="SD" )
SE      <- mxMatrix( type="Full", nrow=1, ncol=1, free=TRUE, values=c(.5), label=c("sE"),
name="SE" )
# scaling constraint
con_stand <- mxConstraint(expression= (sA+sD+sE) == 1, name='standLF')
```

In the following TA and TE are 4x4 diagonal matrices which contain the residual A and E item variances.

```
# residual variances
```

```
TA      <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=c(.5,.5,.5,.5),
label=c("tA1","tA2","tA3","tA4"), name="TA" )
TD      <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=FALSE, values=0,
label=c("tD1","tD2","tD3","tD4"), name="TD" )
TE      <- mxMatrix( type="Diag", nrow=nv, ncol=nv, free=TRUE, values=c(.7,.7,.7,.7),
label=c("tE1","tE2","tE3","tE4"), name="TE" )
```

In the following, VA and VE (ΣA and ΣE) are calculated. VA equals `LF%*%(SA)%*%t(LF) + TA` . That is (using the OpenMx parameter labels):

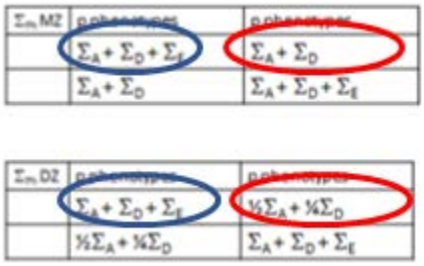
Lf1^2*SA+ tA1
Lf1*Lf2*SA Lf2^2*SA + tA2
Lf1*Lf3*SA Lf2*Lf3*SA Lf3^2*SA + tA3
Lf1*Lf4*SA Lf2*Lf4*SA Lf3*Lf4*SA Lf4^2*SA + tA4

VE equals `LF%*%(SE)%*%t(LF) + TE` . That is (using the OpenMx parameter labels):

Lf1^2*SE+ tE1
Lf1*Lf2*SE Lf2^2*SE + tE2
Lf1*Lf3*SE Lf2*Lf3*SE Lf3^2*SE + tE3
Lf1*Lf4*SE Lf2*Lf4*SE Lf3*Lf4*SE Lf4^2*SE + tAE

```
# Create Algebra for expected Variance/Covariance Matrices in MZ & DZ twins
VA      <- mxAlgebra( expression= LF%*%(SA)%*%t(LF) + TA, name="VA" )
VD      <- mxAlgebra( expression= LF%*%(SD)%*%t(LF) + TD, name="VD" )
VE      <- mxAlgebra( expression= LF%*%(SE)%*%t(LF) + TE, name="VE" )
#
```

In the following, the 4x4 phenotype covariance matrices covP (blue circle) and the twin 1 – twin 2 covariance matrices are created (red circle). These are used to assemble the 8x8 covariance matrices (expCovMZ, expCovDZ).




```
covP      <- mxAlgebra( expression= VA+VD+VE, name="V" )
covMZ     <- mxAlgebra( expression= VA+VD, name="cMZ" )
covDZ     <- mxAlgebra( expression= 0.5%x%VA+.025%x%VD, name="cDZ" )
expCovMZ  <- mxAlgebra( expression= rbind( cbind(V, cMZ), cbind(t(cMZ), V)), name="expCovMZ" )
expCovDZ  <- mxAlgebra( expression= rbind( cbind(V, cDZ), cbind(t(cDZ), V)), name="expCovDZ" )
# Create Data Objects for Multiple Groups
#ACDE model
dataMZ    <- mxData( observed=N4mz, type="raw" )
dataDZ    <- mxData( observed=N4dz, type="raw" )
#ACDE model
# Create Expectation Objects for Multiple Groups
expMZ     <- mxExpectationNormal( covariance="expCovMZ", means="expMean", dimnames=selVars )
expDZ     <- mxExpectationNormal( covariance="expCovDZ", means="expMean", dimnames=selVars )
funML     <- mxFitFunctionML()
#
pars      <- list(B0_, LF, TA, TD, TE, SA, SD, SE, VA, VD,VE, covP)
modelMZ   <- mxModel( pars, ExpMean,  covMZ, expCovMZ, ExpMean, dataMZ, expMZ, funML,
name="MZ" )
modelDZ   <- mxModel( pars, ExpMean,   covDZ, expCovDZ, ExpMean, dataDZ, expDZ, funML,
name="DZ" )
multi     <- mxFitFunctionMultigroup( c("MZ","DZ") )
# model
```

```
modelAE4CPM2  <- mxModel("AE_CPM2", pars, modelMZ, modelDZ, multi,con_stand )
# RUN MODEL
fitAE4CPM2    <- mxTryHard( modelAE4CPM2, 20)
sumAE4CPM2    <- summary( fitAE4CPM2 )
# <-
```