2020 International Statistical Genetics Workshop

Challenges and solutions

Jose Morosoli and Sarah Medland

With thanks to Katrina Grasby, Lucía Colodro-Conde and Elizabeth Prom-Wormley

\workshop\Faculty\jose\Friday

In this practical…

- We don’t cover ANY theory.
- We will learn about common coding and analytical problems we find when we work with OpenMx and twin data in real life.
- Hopefully we will make OpenMx less scary.

Finding starting values

- Most of them can be estimated outside.
- Good for checking if results makes sense.
- For saturated models (assumptions), different starting values for each twin and zygosity. In genetic models, they are equated.
- Closest starting values come from last assumption testing model. Some are exactly the same (means, thresholds, covariate effect).
- Don’t use exact value for starting values, a bit smaller to give space to the optimizer.

Finding starting values

Parameter	For saturated model
Means and Covariates β	lm(vD1con_1~AGE_1*SEX_1, data = mzData)*
Thresholds (without covariates)	qnorm(cumsum(table(mzData$var_T1))/sum(table(mzData$var_T1)))[1:nthresholds]
Vars and covars	var(mzData[,selVars])
Cors (categorical)	cor(mzData[,selVars], method = "kendall")

* For a continuous phenotype, a linear mixed model would be closest to what OpenMx does, but a standard linear regression can do the job for starting values.

Finding starting values

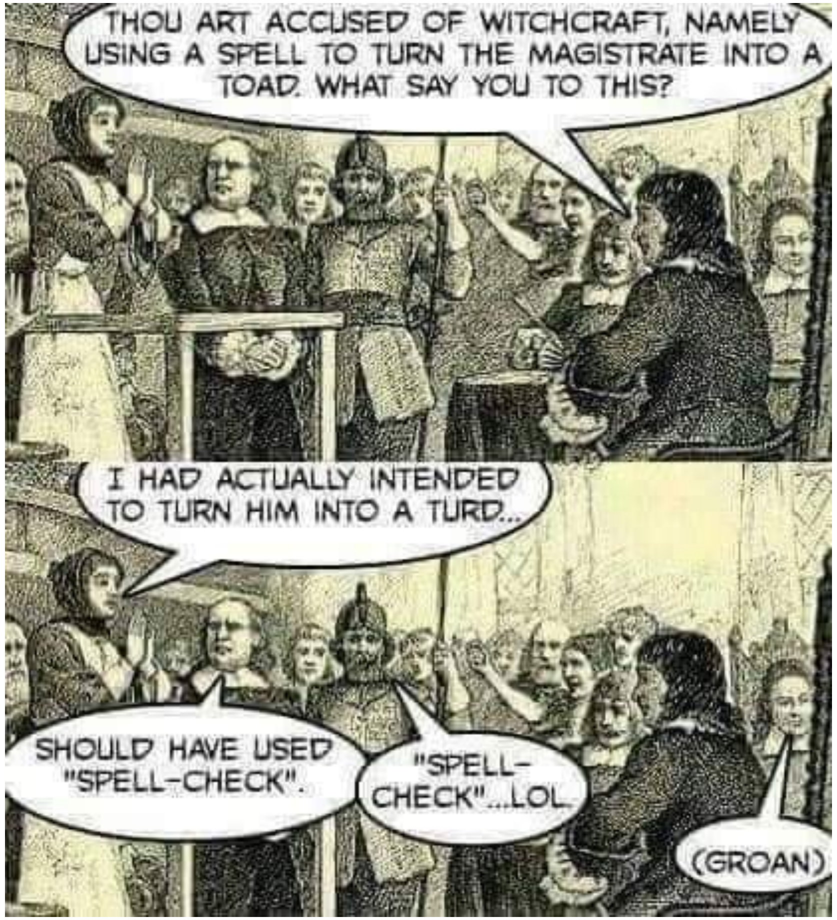
Parameter	Unstandardized components*
VA	2•(covarianceMZ – covarianceDZ)
VC	2•covarianceDZ – covarianceMZ
VD	covMZ – [2•(covMZ – covDZ)]
VE	varMZ – covMZ
COVA	2•(cross.covMZ – cross.covDZ)
COVC	2•cross.covDZ – cross.covMZ
COVE	covPh –cross.covMZ

*Based on Holzinger (1929) and Falconer (1960,) formulas.

Fantastic error codes and where to find them

Most common errors: misspelling, mismatched variable names, or forget to add an object to the model…

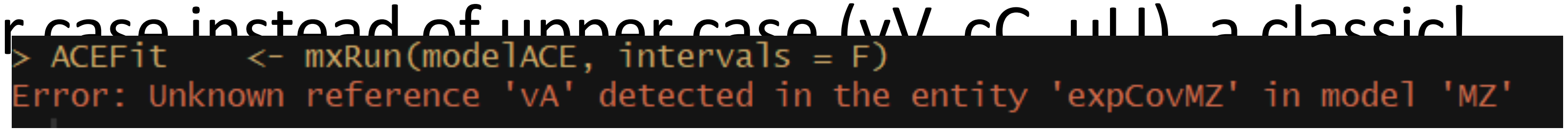
- Take a deep breath and relax, *everything is gonna be fine*.
- Run code line by line – Read the error message and locate the objects they are referring to.
- Personal experience: only 1/10 scripts I run for the first time don't give an error.


THOU ART ACCUSED OF WITCHCRAFT, NAMELY USING A SPELL TO TURN THE MAGISTRATE INTO A TOAD. WHAT SAY YOU TO THIS?
I HAD ACTUALLY INTENDED TO TURN HIM INTO A TURD...
SHOULD HAVE USED "SPELL-CHECK".
"SPELL-CHECK"...LOL.
(GROAN)

Fantastic error codes and where to find them

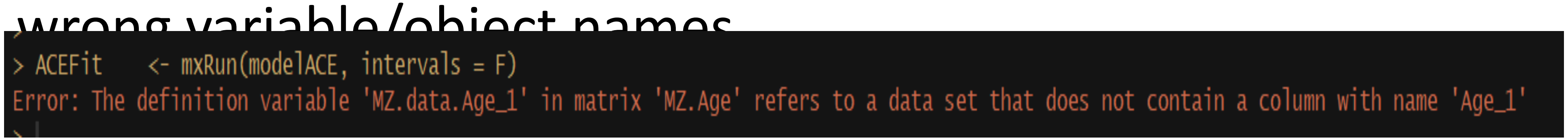
- Check variable names (mzData/dzData) and ‘names’ and ‘labels’ of mxAlgebra/mxMatrix objects.
- Check that you have included all objects in the final model.

E.g., lower case instead of upper case (vV, cC, uU), a classic!



```
> ACEFit     <- mxRun(modelACE, intervals = F)
Error: Unknown reference 'vA' detected in the entity 'expCovMZ' in model 'MZ'
```

E.g., wrong variable/object names

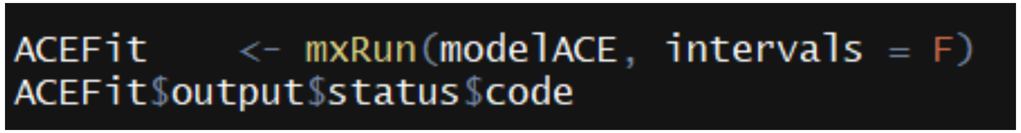


```
> ACEFit     <- mxRun(modelACE, intervals = F)
Error: The definition variable 'MZ.data.Age_1' in matrix 'MZ.Age' refers to a data set that does not contain a column with name 'Age_1'
>
```

Especial mention to *non-conformable arrays*: usually you’re multiplying matrices with incompatible dimensions (revise your matrix algebra!).

Fantastic error codes and where to find them

All mxRun(s) have a status code. They tell you if a solution has been found and its reliability.



```
ACEFit    <- mxRun(modelACE, intervals = F)
ACEFit$output$status$code
```

The Good ✓

Code 0 – All good! No probs

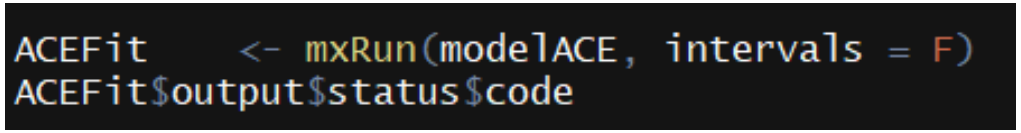
Code 1 – Probably okay but good to re-run

ACTION: MxRun(fit) or MxTryHard

https://openmx.ssri.psu.edu/wiki/errors

Fantastic error codes and where to find them

All mxRun(s) have a status code. They tell you if a solution has been found and its reliability.



```
ACEFit    <- mxRun(modelACE, intervals = F)
ACEFit$output$status$code
```

The Bad ✗

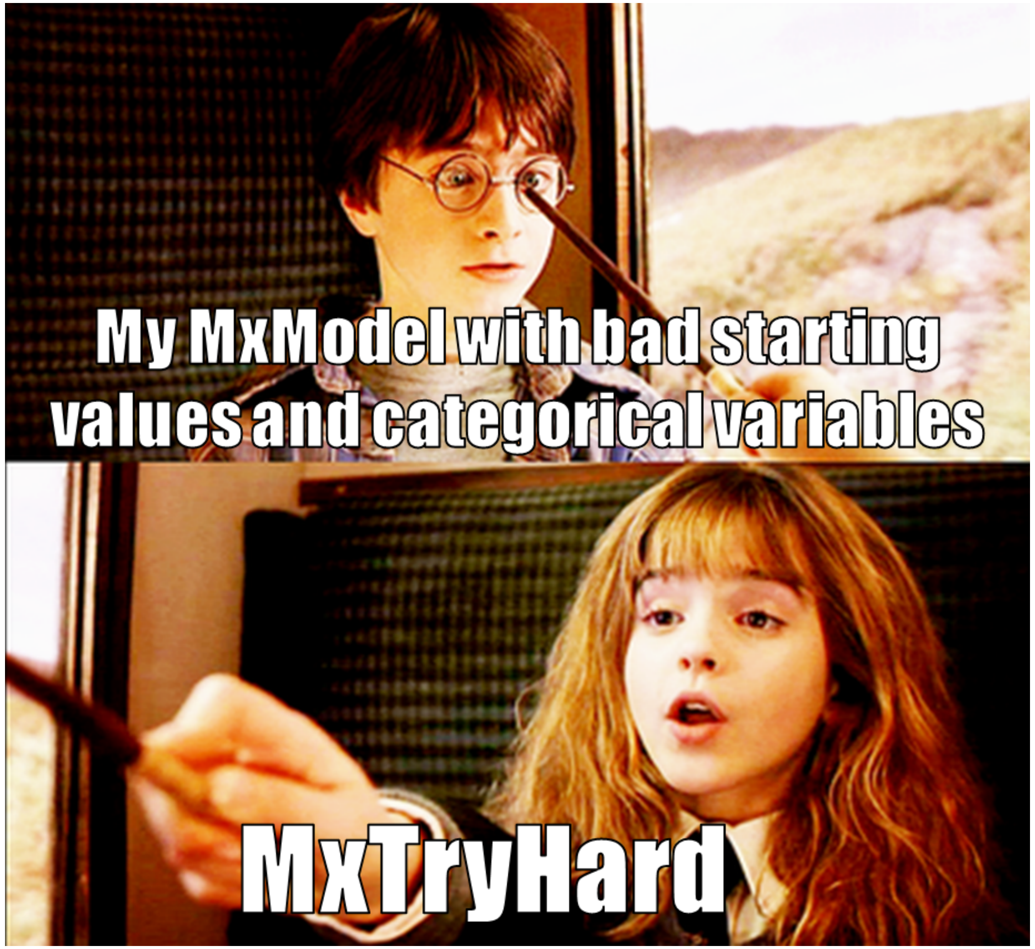
Code 5: Optimizer is stuck
Code 10: Infeasible starting values
Action: check starting values and model specification. Something is quite wrong

The Ugly ✗

Code 6: Almost there, OMX needs your help.
Action: MxTryHard(), improve starting values, rescale variables.

https://openmx.ssri.psu.edu/wiki/errors

Fantastic error codes and where to find them


My MxModel with bad starting values and categorical variables
MxTryHard

- Always check status codes
- Also status code of CIs

`summary(myRun)$CIcodes`

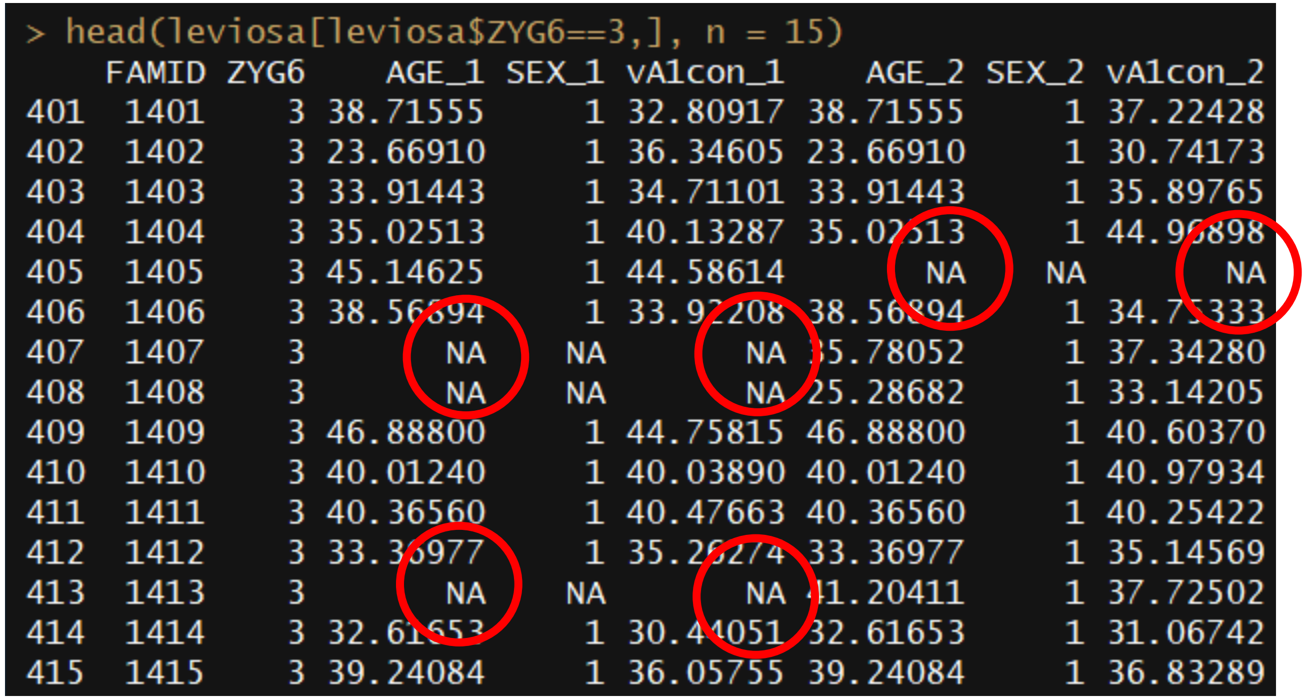
- It's OK to use **MxTryHard()** but ideally mxRun() should be enough.
- Use mxRun first, more informative error messages.
- Inspect summary() even if you get error codes: could direct you to the problem (name of the object / matrix with issues).

Missing data

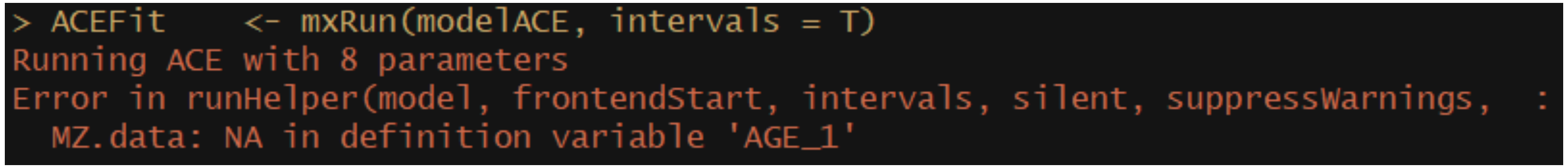
How to avoid losing the complete family when there is one twin / family member missing but not the others?

- OpenMx DON'T accept missing values in a definition variable covariate. It's okay in the phenotype.
- Full-information maximum likelihood (FIML) allows analyzing the non-missing member of the family / cluster.
- Only possible when using raw data (not cov or cor matrices).
- When only the covariate (not the phenotype) is missing, standard techniques apply (multiple imputation, substitution, elimination).

Missing data

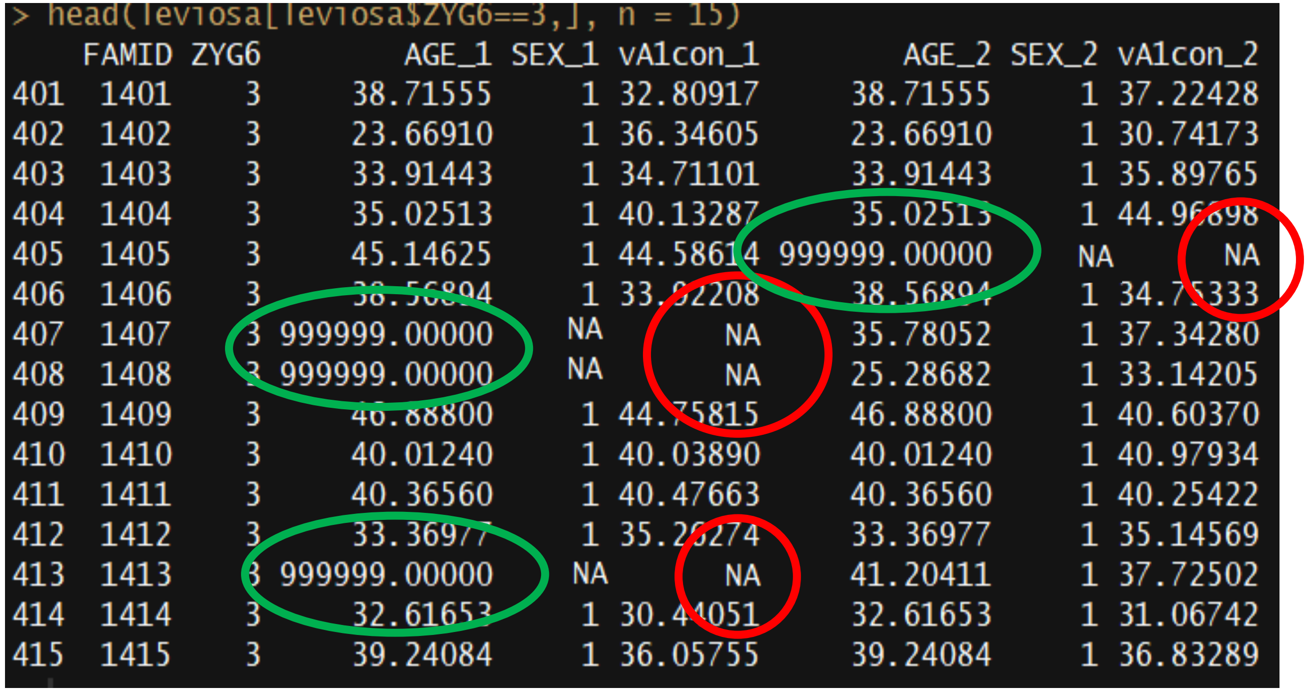


```
> head(leviosa[leviosa$ZYG6==3,], n = 15)
    FAMID ZYG6    AGE_1 SEX_1 vA1con_1    AGE_2 SEX_2 vA1con_2
401  1401    3 38.71555     1 32.80917 38.71555     1 37.22428
402  1402    3 23.66910     1 36.34605 23.66910     1 30.74173
403  1403    3 33.91443     1 34.71101 33.91443     1 35.89765
404  1404    3 35.02513     1 40.13287 35.02513     1 44.96898
405  1405    3 45.14625     1 44.58614       NA    NA       NA
406  1406    3 38.56894     1 33.92208 38.56894     1 34.75333
407  1407    3       NA    NA       NA 35.78052     1 37.34280
408  1408    3       NA    NA       NA 25.28682     1 33.14205
409  1409    3 46.88800     1 44.75815 46.88800     1 40.60370
410  1410    3 40.01240     1 40.03890 40.01240     1 40.97934
411  1411    3 40.36560     1 40.47663 40.36560     1 40.25422
412  1412    3 33.36977     1 35.26274 33.36977     1 35.14569
413  1413    3       NA    NA       NA 41.20411     1 37.72502
414  1414    3 32.61653     1 30.44051 32.61653     1 31.06742
415  1415    3 39.24084     1 36.05755 39.24084     1 36.83289
```

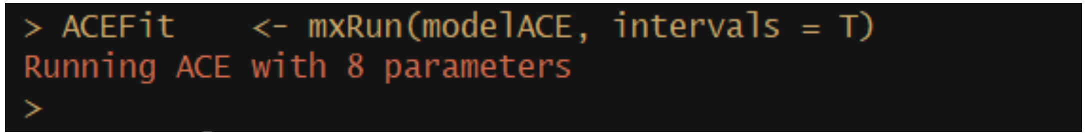


```
> ACEFit    <- mxRun(modelACE, intervals = T)
Running ACE with 8 parameters
Error in runHelper(model, frontendStart, intervals, silent, suppressWarnings,  :
  MZ.data: NA in definition variable 'AGE_1'
```

Missing data



```
> head(Teviosa[Teviosa$ZYG6==3,], n = 15)
    FAMID ZYG6        AGE_1 SEX_1 vA1con_1        AGE_2 SEX_2 vA1con_2
401  1401    3     38.71555     1 32.80917     38.71555     1 37.22428
402  1402    3     23.66910     1 36.34605     23.66910     1 30.74173
403  1403    3     33.91443     1 34.71101     33.91443     1 35.89765
404  1404    3     35.02513     1 40.13287     35.02513     1 44.96898
405  1405    3     45.14625     1 44.58614 999999.00000    NA       NA
406  1406    3     38.56894     1 33.02208     38.56894     1 34.75333
407  1407    3 999999.00000    NA       NA     35.78052     1 37.34280
408  1408    3 999999.00000    NA       NA     25.28682     1 33.14205
409  1409    3     46.88800     1 44.75815     46.88800     1 40.60370
410  1410    3     40.01240     1 40.03890     40.01240     1 40.97934
411  1411    3     40.36560     1 40.47663     40.36560     1 40.25422
412  1412    3     33.36977     1 35.26274     33.36977     1 35.14569
413  1413    3 999999.00000    NA       NA     41.20411     1 37.72502
414  1414    3     32.61653     1 30.44051     32.61653     1 31.06742
415  1415    3     39.24084     1 36.05755     39.24084     1 36.83289
```



```
> ACEFit     <- mxRun(modelACE, intervals = T)
Running ACE with 8 parameters
>
```

About optimizers

- Optimizer: algorithm programmed to find a numerical solution to an specific mathematical problem.

In our case, free parameters in expected means, thresholds, and variances.

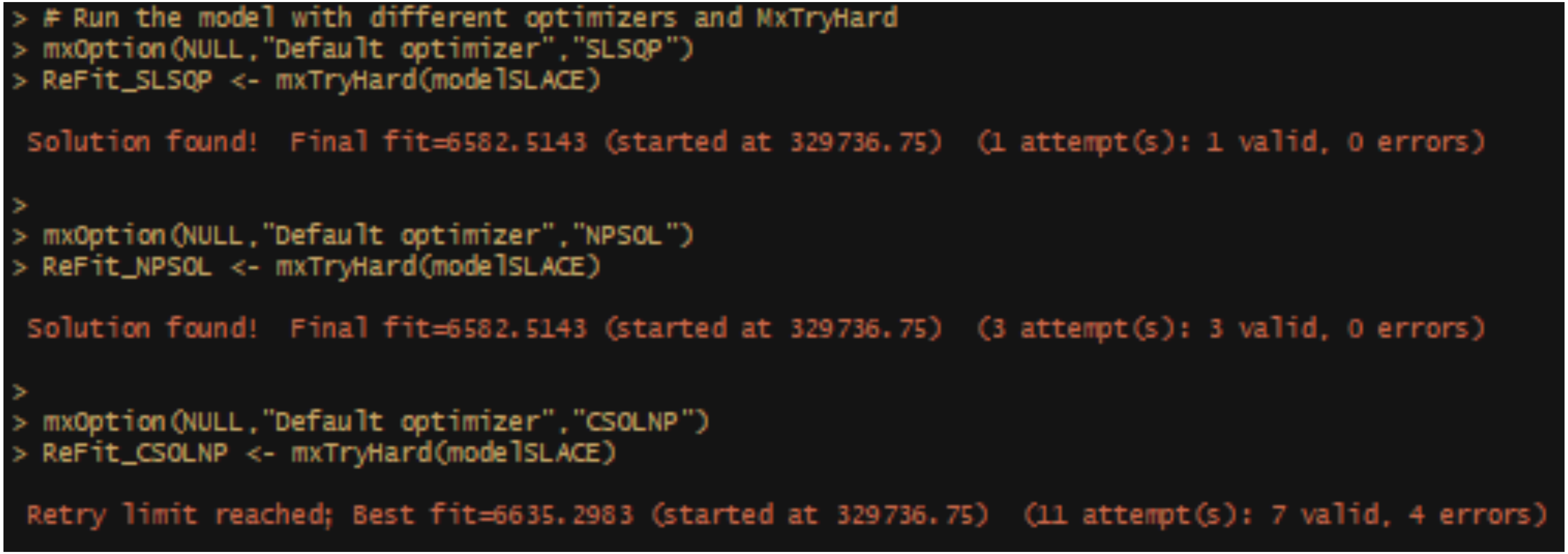
- OpenMx has three optimizers: SQSLP, CSOLNP and NPSOL (only available when you download OpenMx from https://openmx.ssri.psu.edu/installing-openmx)

About optimizers

- Different optimizers work better with different datasets. It’s worth trying a different one if your model is struggling to converge.
- Even if you get a solution, you may struggle to estimate confidence intervals. Different optimizers in different datasets may succeed where others fail.
- To change optimizers run:

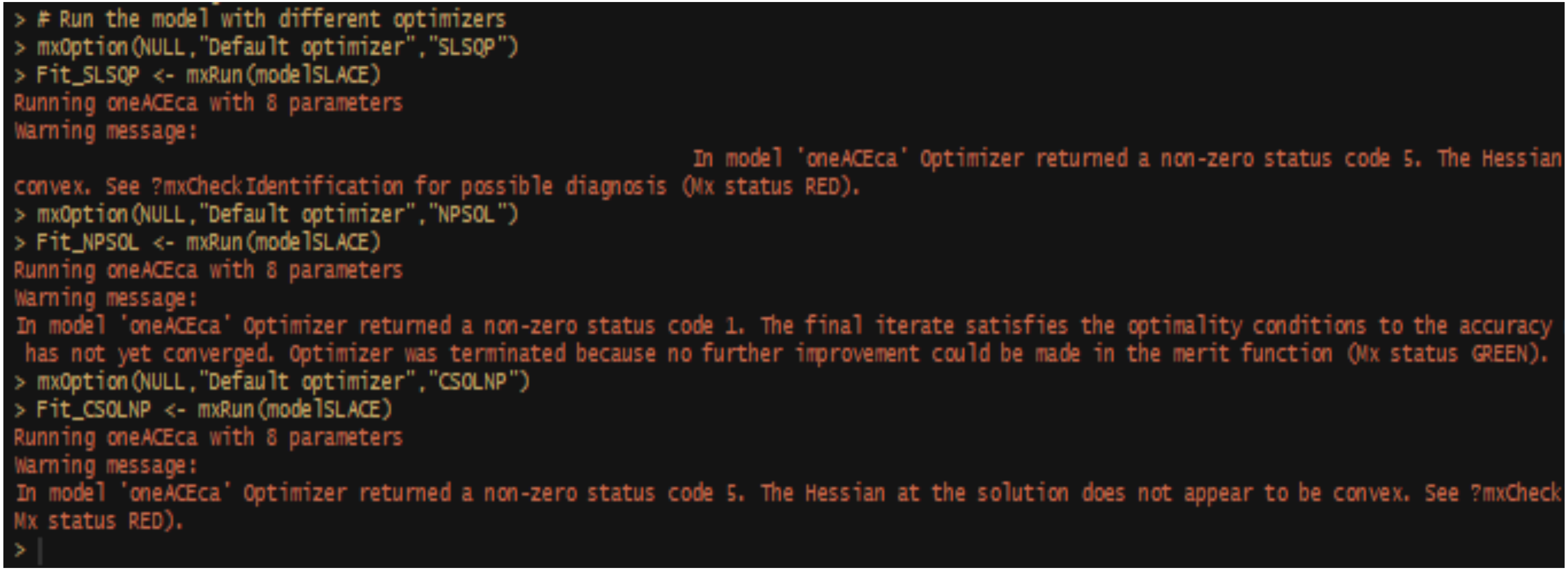
```
mxOption(NULL, “Default optimizer”, “CSOLNP”)
```

About optimizers



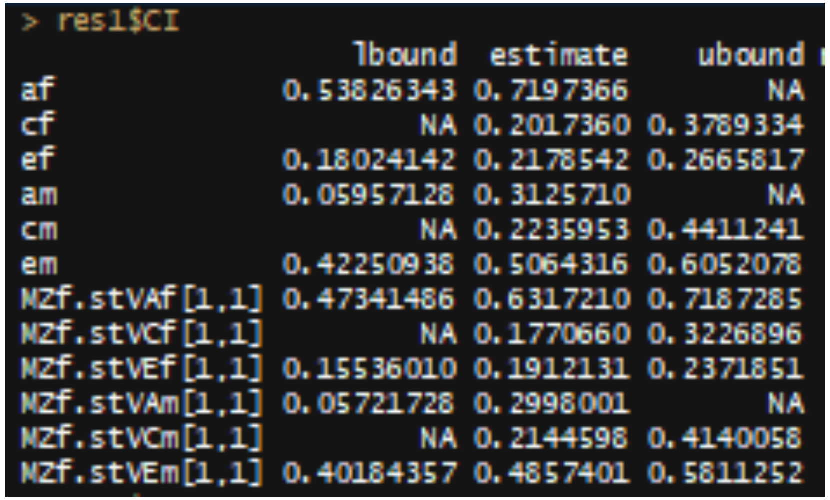
```
> # Run the model with different optimizers and MxTryHard
> mxOption(NULL,"Default optimizer","SLSQP")
> ReFit_SLSQP <- mxTryHard(modelSLACE)

 Solution found!  Final fit=6582.5143 (started at 329736.75)  (1 attempt(s): 1 valid, 0 errors)

>
> mxOption(NULL,"Default optimizer","NPSOL")
> ReFit_NPSOL <- mxTryHard(modelSLACE)

 Solution found!  Final fit=6582.5143 (started at 329736.75)  (3 attempt(s): 3 valid, 0 errors)

>
> mxOption(NULL,"Default optimizer","CSOLNP")
> ReFit_CSOLNP <- mxTryHard(modelSLACE)

 Retry limit reached; Best fit=6635.2983 (started at 329736.75)  (11 attempt(s): 7 valid, 4 errors)
```

About optimizers



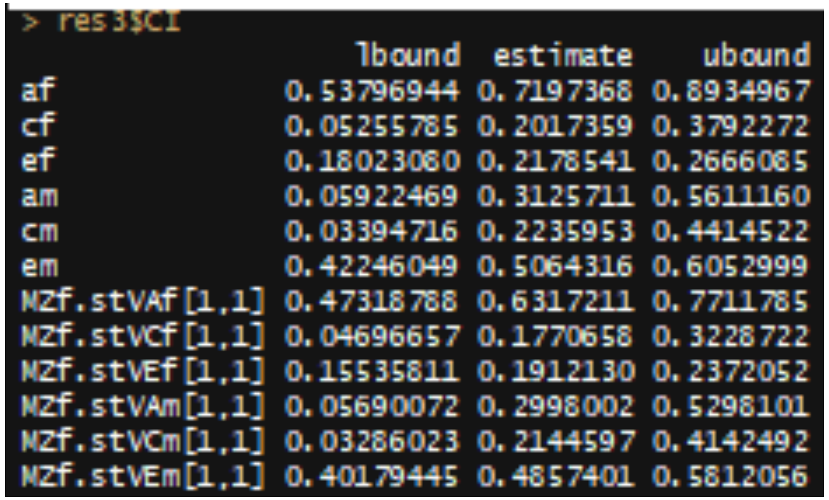
```
> # Run the model with different optimizers
> mxOption(NULL,"Default optimizer","SLSQP")
> Fit_SLSQP <- mxRun(modelSLACE)
Running oneACEca with 8 parameters
Warning message:
                                                                 In model 'oneACEca' Optimizer returned a non-zero status code 5. The Hessian
convex. See ?mxCheckIdentification for possible diagnosis (Mx status RED).
> mxOption(NULL,"Default optimizer","NPSOL")
> Fit_NPSOL <- mxRun(modelSLACE)
Running oneACEca with 8 parameters
Warning message:
In model 'oneACEca' Optimizer returned a non-zero status code 1. The final iterate satisfies the optimality conditions to the accuracy
 has not yet converged. Optimizer was terminated because no further improvement could be made in the merit function (Mx status GREEN).
> mxOption(NULL,"Default optimizer","CSOLNP")
> Fit_CSOLNP <- mxRun(modelSLACE)
Running oneACEca with 8 parameters
Warning message:
In model 'oneACEca' Optimizer returned a non-zero status code 5. The Hessian at the solution does not appear to be convex. See ?mxCheck
Mx status RED).
>
```

About optimizers



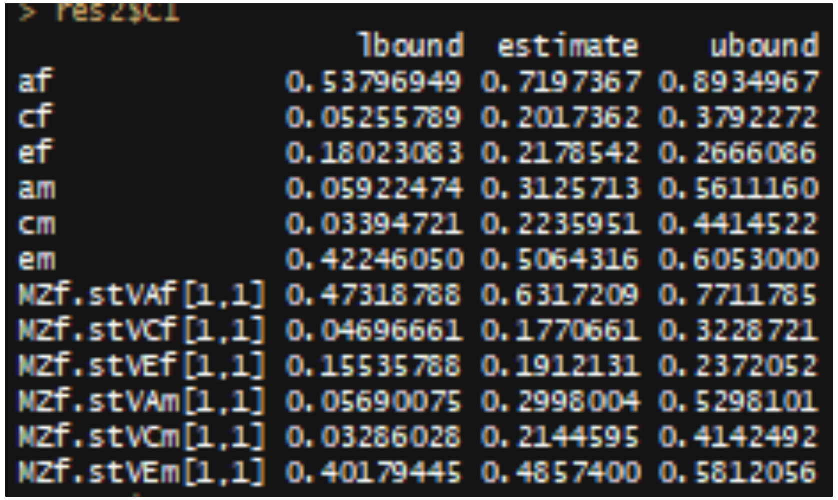
```
> res1$CI
                    lbound  estimate    ubound
af              0.53826343 0.7197366        NA
cf                      NA 0.2017360 0.3789334
ef              0.18024142 0.2178542 0.2665817
am              0.05957128 0.3125710        NA
cm                      NA 0.2235953 0.4411241
em              0.42250938 0.5064316 0.6052078
MZf.stVAf[1,1] 0.47341486 0.6317210 0.7187285
MZf.stVCf[1,1]         NA 0.1770660 0.3226896
MZf.stVEf[1,1] 0.15536010 0.1912131 0.2371851
MZf.stVAm[1,1] 0.05721728 0.2998001        NA
MZf.stVCm[1,1]         NA 0.2144598 0.4140058
MZf.stVEm[1,1] 0.40184357 0.4857401 0.5811252
```

SLSQP



```
> res3$CI
                    lbound  estimate    ubound
af              0.53796944 0.7197368 0.8934967
cf              0.05255785 0.2017359 0.3792272
ef              0.18023080 0.2178541 0.2666085
am              0.05922469 0.3125711 0.5611160
cm              0.03394716 0.2235953 0.4414522
em              0.42246049 0.5064316 0.6052999
MZf.stVAf[1,1] 0.47318788 0.6317211 0.7711785
MZf.stVCf[1,1] 0.04696657 0.1770658 0.3228722
MZf.stVEf[1,1] 0.15535811 0.1912130 0.2372052
MZf.stVAm[1,1] 0.05690072 0.2998002 0.5298101
MZf.stVCm[1,1] 0.03286023 0.2144597 0.4142492
MZf.stVEm[1,1] 0.40179445 0.4857401 0.5812056
```

CSOLNP

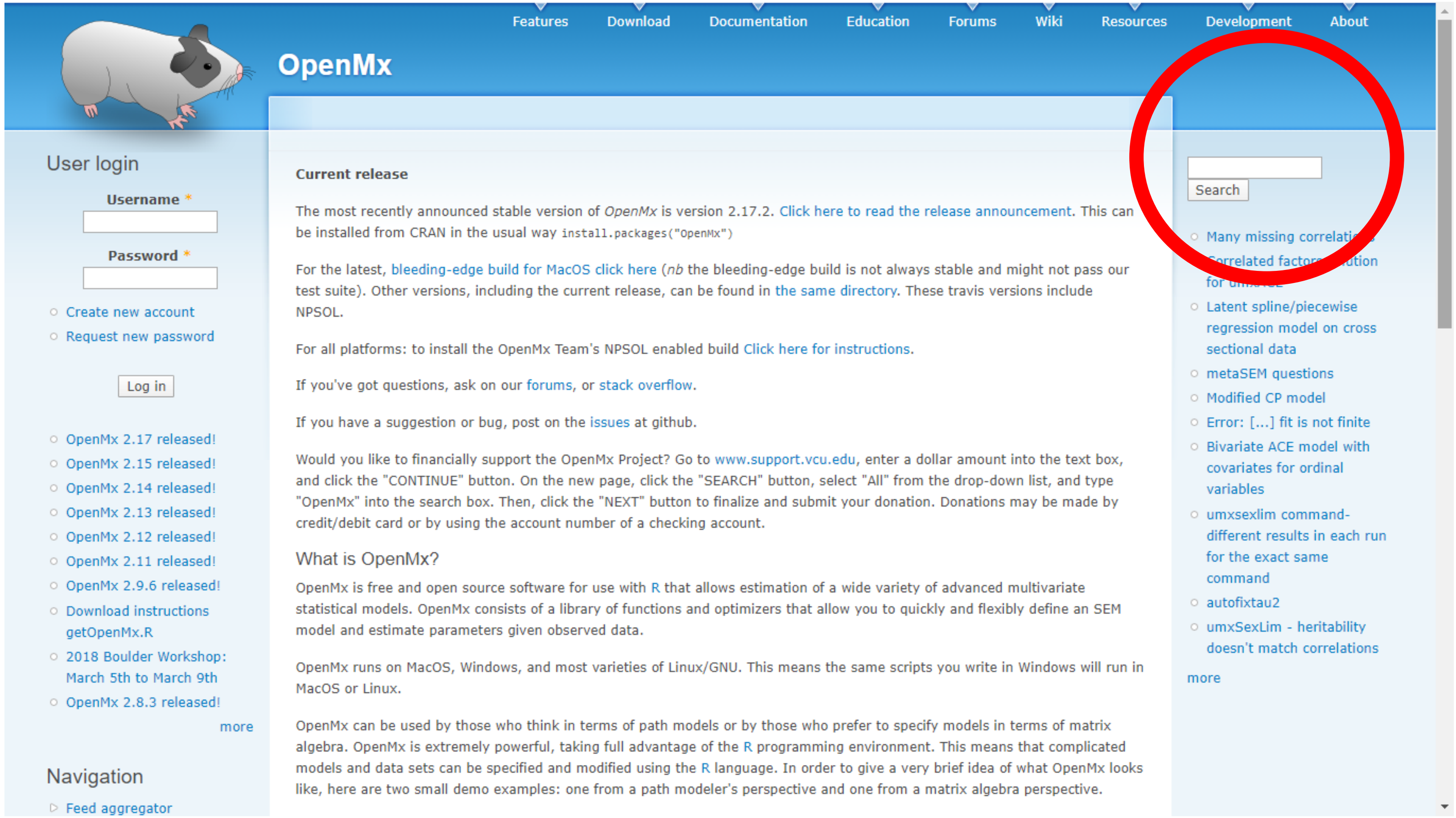


```
> res2$CI
                    lbound  estimate    ubound
af              0.53796949 0.7197367 0.8934967
cf              0.05255789 0.2017362 0.3792272
ef              0.18023083 0.2178542 0.2666086
am              0.05922474 0.3125713 0.5611160
cm              0.03394721 0.2235951 0.4414522
em              0.42246050 0.5064316 0.6053000
MZf.stVAf[1,1] 0.47318788 0.6317209 0.7711785
MZf.stVCf[1,1] 0.04696661 0.1770661 0.3228721
MZf.stVEf[1,1] 0.15535788 0.1912131 0.2372052
MZf.stVAm[1,1] 0.05690075 0.2998004 0.5298101
MZf.stVCm[1,1] 0.03286028 0.2144595 0.4142492
MZf.stVEm[1,1] 0.40179445 0.4857400 0.5812056
```

NPSOL

OpenMx Forum

https://openmx.ssri.psu.edu/



User login

Username *

Password *

- Create new account
- Request new password

Log in

- OpenMx 2.17 released!
- OpenMx 2.15 released!
- OpenMx 2.14 released!
- OpenMx 2.13 released!
- OpenMx 2.12 released!
- OpenMx 2.11 released!
- OpenMx 2.9.6 released!
- Download instructions getOpenMx.R
- 2018 Boulder Workshop: March 5th to March 9th
- OpenMx 2.8.3 released!

more

Navigation

- Feed aggregator

Current release

The most recently announced stable version of *OpenMx* is version 2.17.2. Click here to read the release announcement. This can be installed from CRAN in the usual way `install.packages("OpenMx")`

For the latest, bleeding-edge build for MacOS click here (*nb* the bleeding-edge build is not always stable and might not pass our test suite). Other versions, including the current release, can be found in the same directory. These travis versions include NPSOL.

For all platforms: to install the OpenMx Team's NPSOL enabled build Click here for instructions.

If you've got questions, ask on our forums, or stack overflow.

If you have a suggestion or bug, post on the issues at github.

Would you like to financially support the OpenMx Project? Go to www.support.vcu.edu, enter a dollar amount into the text box, and click the "CONTINUE" button. On the new page, click the "SEARCH" button, select "All" from the drop-down list, and type "OpenMx" into the search box. Then, click the "NEXT" button to finalize and submit your donation. Donations may be made by credit/debit card or by using the account number of a checking account.

What is OpenMx?

OpenMx is free and open source software for use with R that allows estimation of a wide variety of advanced multivariate statistical models. OpenMx consists of a library of functions and optimizers that allow you to quickly and flexibly define an SEM model and estimate parameters given observed data.

OpenMx runs on MacOS, Windows, and most varieties of Linux/GNU. This means the same scripts you write in Windows will run in MacOS or Linux.

OpenMx can be used by those who think in terms of path models or by those who prefer to specify models in terms of matrix algebra. OpenMx is extremely powerful, taking full advantage of the R programming environment. This means that complicated models and data sets can be specified and modified using the R language. In order to give a very brief idea of what OpenMx looks like, here are two small demo examples: one from a path modeler's perspective and one from a matrix algebra perspective.

- Many missing correlations
- Correlated factor solution for umxACE
- Latent spline/piecewise regression model on cross sectional data
- metaSEM questions
- Modified CP model
- Error: [...] fit is not finite
- Bivariate ACE model with covariates for ordinal variables
- umxsexlim command-different results in each run for the exact same command
- autofixtau2
- umxSexLim - heritability doesn't match correlations

more

Practical: The Good Samaritan

Setting the scene

- A master’s student at your lab is struggling with a bivariate twin model. You just came back from Boulder and they ask you if you could take a look to their script and help them getting it to run. They adapted someone else’s script but they keep getting errors.
- They want to analyze the relationship between internalizing anger (variable *anger*) and blood pressure (variable *mmHg*).
- They send you their script (angry2ACEc.R) and their data (goodsamaritan.txt) and promise to buy you a drink or chocolates if you fix it.

Practical: The Good Samaritan

- Assumptions were already tested and have been met.
- Final estimates for vars, covars, and means after equating across twin and zygosity from saturated model are in next slide.

NOW OPEN *score-sheet_Fri8am.docx* AND *angry2ACEc.R*

Practical: The Good Samaritan

MZ	mmHg_1	anger_1	mmHg_2	anger_2
mmHg_1	8.14			
anger_1	1.50	3.94		
mmHg_2	4.80	0.94	8.14	
anger_2	0.94	1.47	1.50	3.94

MZ	mmHg_1	anger_1	mmHg_2	anger_2
mmHg_1	8.14			
anger_1	1.50	3.94		
mmHg_2	2.01	0.42	8.14	
anger_2	0.42	1.38	1.50	3.94

means	Interc.	age	sex	a*s
anger	6.35	0.02	1.36	-0.3
mmHg	123.83	0.03	1.07	-0.2

Coding hygiene and positive thinking

1. First, write down an analysis plan and follow it: write down the research question, decide the models you need to run, don't go down rabbit holes.
2. Triangulate results: descriptives and basic tests outside OpenMx, saturated models, univariate before multivariate.
3. Don't be afraid of warning or error messages, they want to help you. Breath. Read them.
4. Relax, take another deep breath, go for a walk if stuck (coder's block is real, google it).
5. Say only nice things to OpenMx, it remembers and can strike back.

Speeding up confidence intervals

- Only for confidence intervals, not the base model.
- Try changing optimizer if confident intervals don’t converge.

```
# Computing CIs in parallel
library(snowfall)
library(parallel)
(ncores <- detectCores()) # How many cores
sfInit(parallel=TRUE, cpus = ncores)
sfLibrary(OpenMx)

EqZygFit_Ci <- omxParallelCI( EqZygFit )
```