1. RAM (the kind of model we are using), is implemented with Symmetric (S), Asymmetric (A), means (M) and a filter (F) matrix
2. Build our old friend m1: the model of mpg from `?umxRAM`
3. Type: `m1@matrices$S`
    - What is this for?
        - This is the object that stores Symmetric (2-headed) paths
    - What do you see (~5 things to note)?
        - Is has a values matrix, which is where path values live
        - It has a matrix storing whether a cell is free or fixed
        - It has matrices for the lower (lbound) and upper (ubound) limits on values a cell can take (may be NA)
        - It has a matrix for labels each cell takes (maybe NA)
    - Which cells (paths) in `S` are free for our model?
        - residuals (on the diagonal)
        $free

	mpg	wt	disp
mpg	TRUE	FALSE	FALSE
wt	FALSE	TRUE	TRUE
disp	FALSE	TRUE	TRUE

4. Type: `m1@matrices$A`
    - What do you see?
        - This is the matrix of Asymmetric paths
    - Which cells (paths) in `A` are free for our model?
        - paths from the latent variable to each manifest
        $free

	mpg	wt	disp
mpg	FALSE	TRUE	TRUE
wt	FALSE	FALSE	FALSE
disp	FALSE	FALSE	FALSE

5. Look up the help for `?mxConstraint`
6. Run the example model: What is it doing?
    - It is forcing the matrix “K” to be equal to the matrix “limit”
7. We haven’t covered it yet, but look up `?mxAlgebra`
    - Why might you need an algebra in a model?
        - To compute new variables: like a %*% = A
        - To compute the expected covariances of our model
        - To compute the expected covariances of our model
8. Look up the help for `?mxCheckIdentification`

9. Run the example model
    - What was the effect of setting the latent variable’s variance (model2n)?
        - Model no longer identified
    - What happens if you run the model
        - Some models will still run, but can give different results each time, others can’t be run
10. Use path tracing rules
    - What is the effect of a two-path connection .5 and .3?
        - .15 (the product of the connections in the pathway)
    - Draw a model a bit like the one for reading on the poster on the wall, but simpler :-)
    - put estimated values on it.
    - Use the path tracing rules to show the total influence of a latent variable in your model on a particular manifest
    - https://en.wikipedia.org/wiki/Path_analysis_(statistics)#Path_tracing_rules
11. Run the example in `?umxMI` (umx’s modification indices function)
    - The output shows paths that would appear to lead to a better fit.
    - In this case, none are very important, but try adding one of them to a copy of this model, and compare the fit
    - *note*: This is COMPLETELY POST HOC
12. Read the warnings in ?mxMI
    - Explain to your lab mates why modification indices risk unreplicable models
    - Can you think of a situtation where they might be of value?
    - How would you test a modified model?
13. Note: MI is a very limited, automated method which looks at evidence for paths given the model. It can’t invent new (potentially MUCH better) models and suggest those. That is what AI tries to do.

Extra credit

1. Run the 1-factor model (first model in examples in ?umx)
2. What does m1$F show you?
    - Hint “F” stands for filter
        - The latent variables appear only as columns, not rows
        - This allows the model to know what variables are latent
3. Build a twin model to practice for next week
    - ?umxACE

Refs

1. David Kenny page
    - http://davidakenny.net/cm/fit.htm
2. umx home page
    - http://tbates.github.io
3. OpenMx home page
    - http://openmx.ssri.psu.edu