An Introduction to Structural Equation Modelling (SEM)

Robin Beaumont 02/01/2018

An updated version to the introduction to SEM chapter in my book: Health science statistics using R and R Commander by Robin Beaumont http://amzn.eu/g76aTP8

Books SEM webpage: http://robin-beaumont.co.uk/rbook/sem/index.html

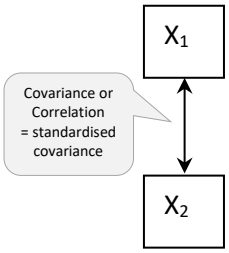
Table of Contents

65 Structural Equation Modelling (SEM)

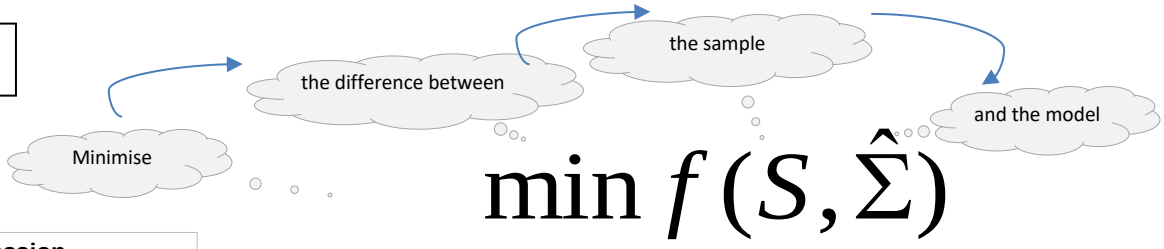
Self test questions
Whereas relatively few people actually carry out SEM analyses a much large number read books and articles reporting such analyses. Therefore, for those of you who only want to learn about SEM rather than carry out a SEM analysis I have included several self test questions in this chapter.

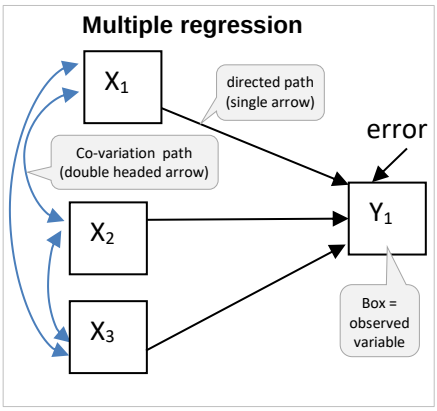
Structural Equation modelling, SEM for short, allows you to develop and test models that consist of regressions, correlations and differences in means between groups. SEM is a statistical technique that has developed from the concepts of covariance and correlation, therefore all the facts you know about correlation, including its limitations and pitfalls apply to SEM. Correlations themselves form the basis of path analysis (PA) and confirmatory factor analysis (CFA), which are forms of SEM. This will be explained diagrammatically on the next few pages.

65.1 The background to SEM

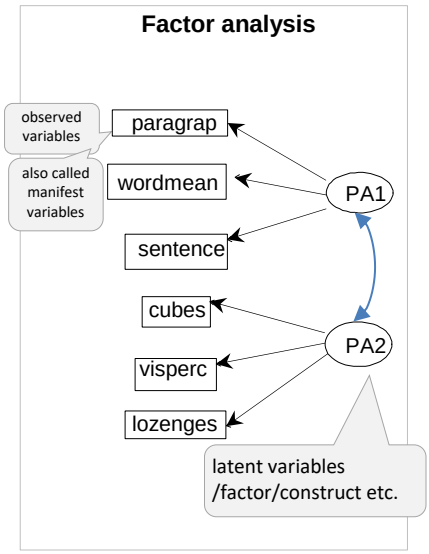


The statistical technique SEM attempts to minimise the difference between two sets of values (matrices of covariances=unstandardized correlations). One set is calculated from the sample data while the other is generated from a hypothesised theoretical model defined by the researcher. This idea can be expressed succinctly in the following equation.





Paradoxically while this model comparison approach might appear very different from the traditional hypothesis testing approach (here you are hoping for the least difference between the two) is can also incorporate the traditional approach (where you are hoping for a big a possible difference between the observed and zero effect models). For details see the online chapter *SEM equivalent to basic statistical procedures*.



Schumacher and Lomax's excellent book, *A Beginner's guide to Structural Equation Modeling*, 2010, provides details of what SEM is, from which I have compiled the following list:

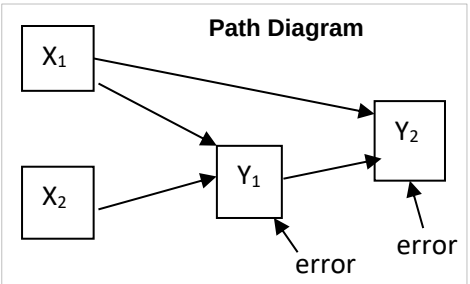
1. Based on Correlations (Covariance)
2. Complex mathematical approach only made widely available with the use of suitable software
3. Allows the definition of complex relationships using models (mathematically using covariance matrices which can be partially represented by diagrams)
4. Extends Regression (Path models)
5. Extends Confirmatory Factor Analysis (CFA)
6. Combines the two to form very complex models = Structural Equation Models (SEM)
7. Allows assessment of the degree to which a proposed model fits the sample data
8. Allows comparison of models

There has been a long-ongoing debate as to what degree SEM models can help establish causal rather than correlational relations (Bollen & Pearl, 2013). Possibly the best approach is to consider the model as correlational and then re-interpret it in terms of context to decide which are causal associations.

Let's take a look at the two building builds of SEM, Path modelling and (Confirmatory) factor analysis (CFA) in a little more detail.

65.2 Path diagrams

In SEM speak when the diagrams only contain observed variables they are called **path diagrams.** The regression model on the previous page can be developed to produce more complex configurations of observed variables where we no longer just have several inputs (independent variables) along with a single output (dependent variable), opposite there is additionally an intervening (i.e. mediating) variable as well. Measured variables are also called manifest variables. The Error represents a residual variance as the predictors only partially predict the value, think of it as a variable that mops up the inaccuracy in the prediction.



In contrast to the above regression and path models, models that contain both observed and latent variables (discussed below) are called SEM models, such an example is the CPA model above.

If the diagram has no numerical values on the lines it can either be a **standardised** or **unstandardized** diagram, as in the above examples.

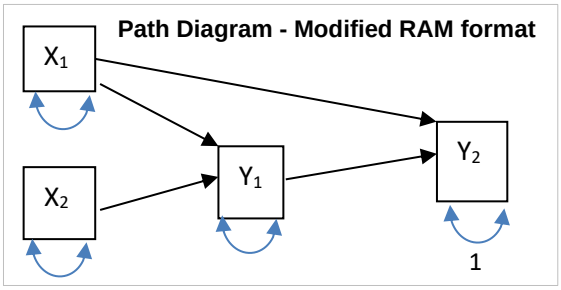
Each square represents a single **observed variable** that we specify. Traditionally the variables would be of interval/ratio scale but software can now accommodate nominal/ordinal data.

Standardized partial regression coefficients are also called beta weights (ß)

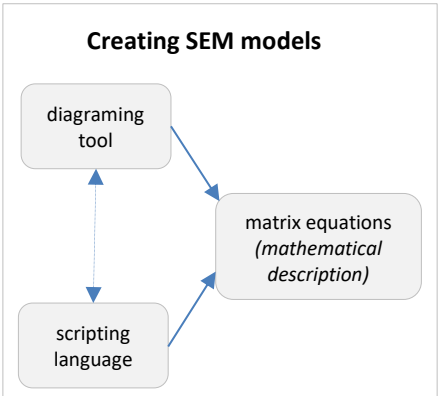
The single arrowed lines are called [**directed**] **paths** and indicate that the variable at the origin of the arrow has some influence on the target variable. These can be interpreted as (unstandardized/standardised) partial regression coefficients or factor loadings depending where the lines appear on the diagram or if the diagram is unstandardized/standardised.

The double arrowed lines represent **co-variation.** If the values on the diagram are standardised these lines represent standardised covariances = **correlations** or if the diagram is unstandardized **covariances**. The multiple regression design shown on the previous page shows correlations/covariances between the three inputs (predictors) (also see Kline, 2016 p.39).

Unfortunately, there are several different ways of drawing SEM diagrams and each SEM software developer shows or ignores different aspects. A common aspect that is omitted is the estimate of the variance of each observed/latent variable. Because a variable's estimated variance is equivalent to the estimate of its covariance with itself, i.e. $cov(x,x) = var(x)$ we can use the double-headed arrows again to represent this; they simply turn back on themselves.



The Modified RAM-format path diagram does this, (Fox, 2006 p.476; Kline, 2016 p.130) as shown opposite. The error 'variables' now simply become another co-variation arrow. Furthermore, including this visual aspect aids interpretation of the model output, specifically the parameter estimates. You can now calculate exactly the number of parameters that are going to be estimated by adding up all the single and double headed arrows omitting those that have an identification constraint, distinguished by having a value attached to them (e.g. y_2 has a constraint of 1, in other words it is a fixed value and not estimated). Eventually, whichever style of diagram you use, it will ultimately be converted into a set of matrices, which we will now briefly discuss. Alternatively, you can also specify a SEM model using various scripting languages which we will look at in the practical section in this chapter.



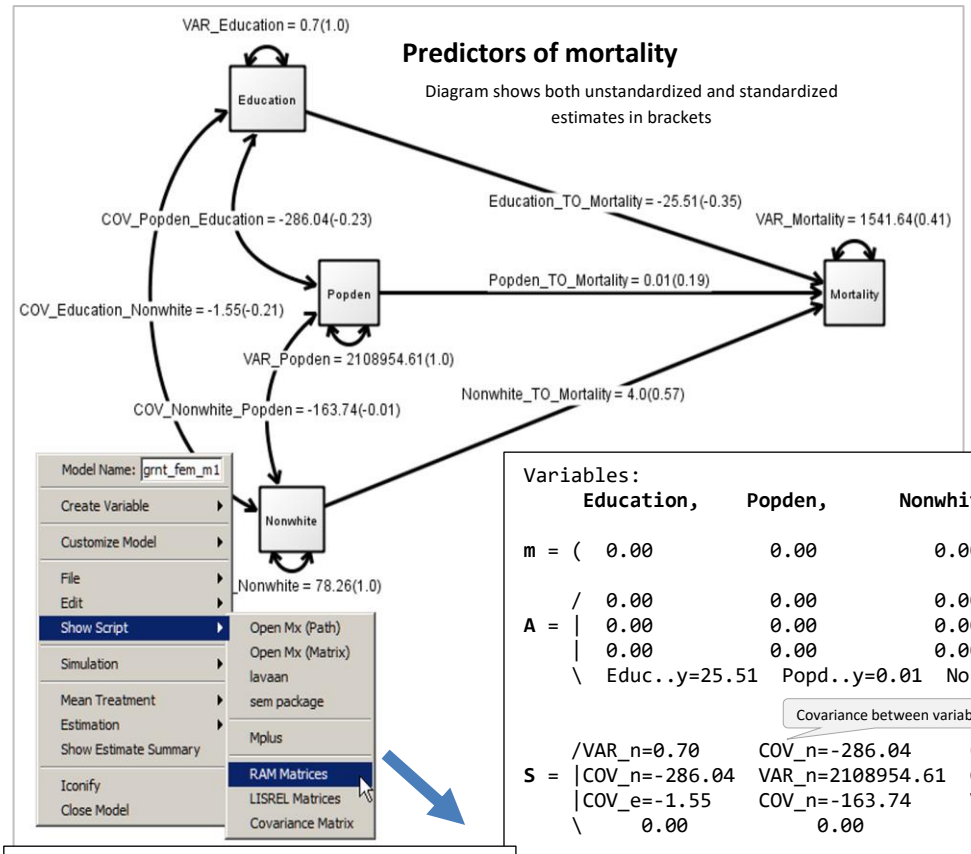
65.2.1 The model equations

There are two main ways of expressing the SEM model as a set of matrices. Firstly those developed by Joreskog & Van Thillo, 1972 culminated in the development of the LISREL software which specifies a set of matrixes by Greek letters (http://web.pdx.edu/~newsomj/semclass/ho_lisrel%20notation.pdf). LISREL is still one of the most popular commercial SEM software packages. Schumacker & Lomax, 2010 provide a chapter describing the LISREL matrix approach

In 1980, McArdle suggested an alternative formulation, simplifying the 8 matrices needed by LISREL to just 4 to create the Reticular action model (RAM). Fox, 2006 provides details of the RAM approach along with how this is implemented in the R *SEM* package.

If you consider only those variables which have one or more single arrowed paths pointing towards them, each group can be considered to be a separate equation (actually a regression equation). In the diagram below, we have one such group.

Let's look at the 'predictors of mortality' regression example below produced using the free Ωnyx program. When you right click on the diagram you can view either the RAM or LISREL matrices (menu option below). I have shown the RAM option.



While it is not necessary to understand the underlying matrices, it is sometimes useful to inspect them when either the model does not produce the expected results, or you are unable to obtain a result.

```
Variables:
     Education,     Popden,      Nonwhite,       Mortality

m = (  0.00           0.00           0.00             0.00      )

    /  0.00           0.00           0.00             0.00      \
A = |  0.00           0.00           0.00             0.00      |
    |  0.00           0.00           0.00             0.00      |
    \  Educ..y=25.51  Popd..y=0.01   Nonw..y=4.0      0.00      /

    /VAR_n=0.70      COV_n=-286.04     COV_e=-1.55     0.00      \
S = |COV_n=-286.04   VAR_n=2108954.61  COV_n=-163.74   0.00      |
    |COV_e=-1.55     COV_n=-163.74     VAR_e=78.26     0.00      |
    \     0.00            0.00           0.00       VAR_y=1541.64 /

    /  1              0              0              0            \
F = |  0              1              0              0            |
    |  0              0              1              0            |
    \  0              0              0              1            /
```

Means (defaults to zero)

Asymmetric arrows

Covariance between variables

Symmetric arrows arrows

Var = 'error' variance for a variable

Filter of observed variables

The four RAM matrices
Defines the values of all the variables, constants, latents and their relations:

- **m** = Means. By default all the values are set to zero the model using mean centred scores.
- **A** = Asymmetrical arrows = regression weights (single arrowed lines)
- **S** = Symmetrical arrows = the covariance matrix among the variables in the model.
- **F** = Filter = index of variables in the model where, observed =1; constants and latent's=0. There are only observed variables in this model. (this matrix is also called **J** by Fox, 2006)

The above matrices are manipulated to produce the various results shown in a SEM output. Specifically, the **S** matrix of OBSERVED sample values are compared against those PREDICTED by the model stored in a matrix called **C**, which is calculated from all the above matrices.

The computer program then attempts to minimise the difference between the two sets of values. This description leading to two questions:

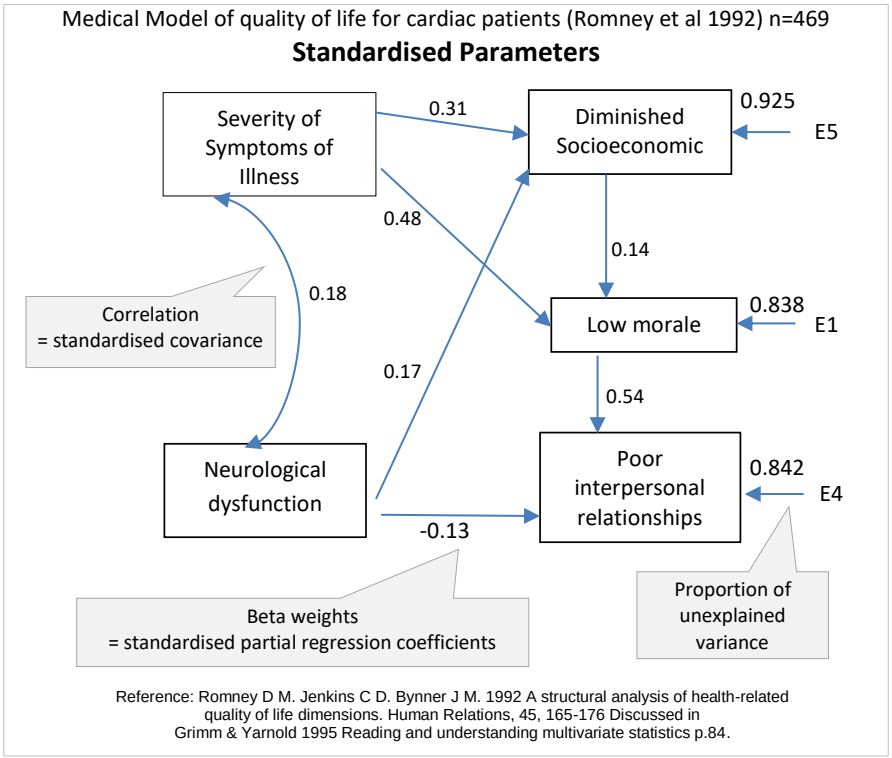
1. **What is meant by difference between?**
   There are various ways of measuring the difference between the two matrices and the computed values are often called fit or discrepancy functions. Such functions go by the name of ordinary least squares (OLS), asymptotically distribution free criterion (ADF), generalised least squares (GLS) and maximum Likelihood (ML) where ML is the standard technique in SEM. In ordinary least squares the result is the sum of the squared difference between the two corresponding items in each matrix. In all these criteria the minimum value is zero when both matrices are the same.

2. **Is there any strategy to finding an improved fit between the two matrices each time around?**
   By making use of one of the above computed discrepancy function values we can compare these for difference guesses of our parameters and hopefully gradually home in until we get a sufficiently small value. There are special search algorithms build into SEM computer programs to help it gradually 'home in' but because the search program may go widely astray we can usually also set a maximum number of times (i.e. iterations) it can have a shot at finding a sufficiently close answer.

We will now look at a typical SEM diagram showing a set of model path estimates.

65.2.2 Interpreting a Path model (i.e. no circles)

The diagram below contains more information than the previous ones so I will discuss each aspect separately:



The results are taken from a correlational study concerning 469 cardiac patients where the investigators were keen to find out the associations between various measures.

SEM is usually carried out on large samples and 469 is small compared to many studies. Also usually, the study is of a retrospective/correlational design although often there are attempts in the discussion of such papers to interpret the relationships as causal. We will return to this issue later.

Title of Standardised Parameters - this indicates that the variables are standardised (i.e. z scores). This allows the various parameters in the diagram to be compared and interpreted in a specific way described below.

Double arrow lines – These values indicate a correlation where the modelling process estimates the value. Variables that do not have lines between them specify that the correlation between them is set to zero in the model (i.e. they are independent).

Single arrow lines - These are beta weights (standardised partial regression coefficients) which are identical to those in multiple regression. With this knowledge, it is possible to interpret them as you would do in a regression equation. For example, a level of *Severity of symptoms of illness* one full standard deviation above the mean predicts an increase in low morale by 0.48 standard deviations above the mean controlling for Neurological dysfunction.

Error terms (E1, E4, E5)– This is the unexplained variance, that which is either measurement error (random) or not explained by the model. This diagram uses the 'e' style to show them rather than the self directed double arrows.

Proportion of unexplained variance values – these are also called the *standardised residual variances* **(SRV).** One minus these values gives the R^2 value (proportion of variance explained). So the SRV x100 gives the % of the variance that is not predicted by the specific inputs that have arrows pointing to that variable. In some diagrams and software (e.g. EQS) the square root value is given instead.

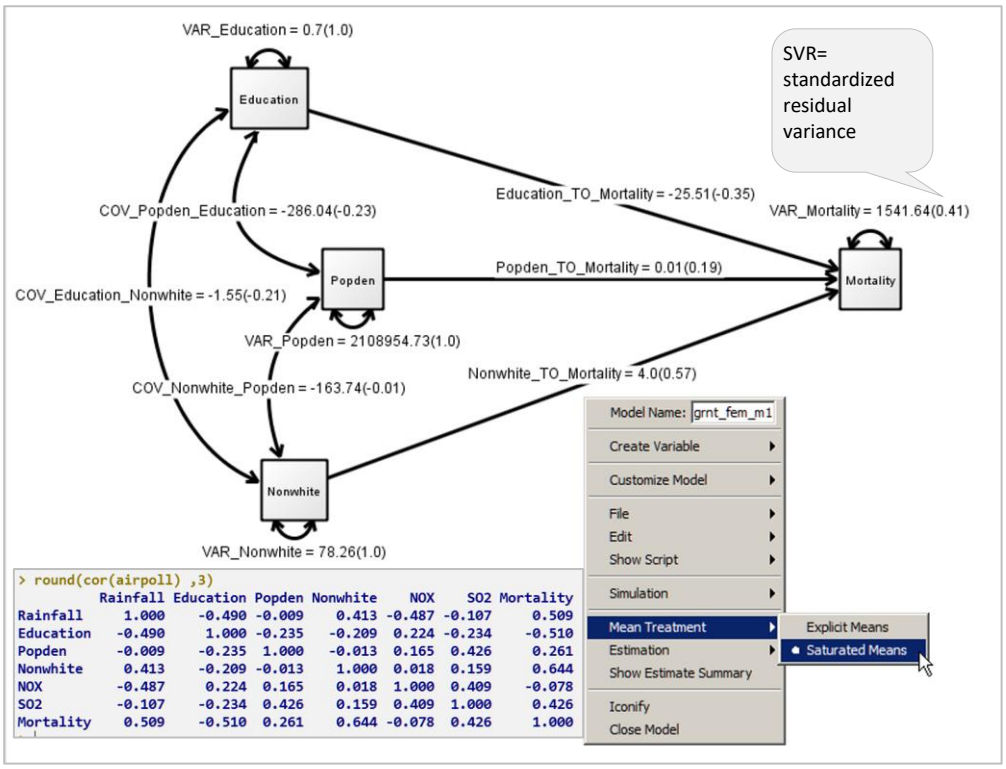
Direction of the single arrow - this does matter as it defines which variable will have the error term (the variable that the arrow points to) and also how to interpret the path coefficient.

To check your understanding I have included some self-test questions opposite.

Self-test:

1. Please complete this sentence. A level of Neurological dysfunction one full standard deviation above the mean predicts an increase in poor interpersonal relationships by ____ Standard _____ below the mean controlling for Symptoms of illness.
2. What does the value 0.54 represent between 'Low morale' and 'Poor relationships'?
3. What do the values from E5, E1 and E4 suggest about the model? How do they relate to R^2.
4. What does the double pointed arrow (<-->) indicate.
5. There is no path between 'Neurological dysfunction' and 'Low morale' in the model. What does this say about the model specification?
6. Does the above model more closely represent a Regression or Factor analysis? Give reasons for your answer.
7. How do you think you might calculate the correlation between *Severity of illness* and *Poor interpersonal relationships* (discussed further below).

65.2.3 Interpreting a Standardised SEM model (i.e. circles and squares)



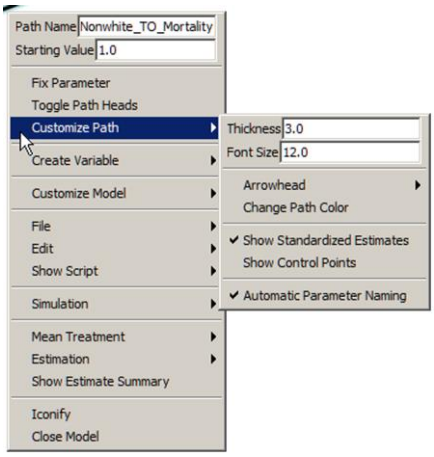

```
> round(cor(airpoll) ,3)
          Rainfall Education Popden Nonwhite    NOX    SO2 Mortality
Rainfall     1.000    -0.490 -0.009    0.413 -0.487 -0.107     0.509
Education   -0.490     1.000 -0.235   -0.209  0.224 -0.234    -0.510
Popden      -0.009    -0.235  1.000   -0.013  0.165  0.426     0.261
Nonwhite     0.413    -0.209 -0.013    1.000  0.018  0.159     0.644
NOX         -0.487     0.224  0.165    0.018  1.000  0.409    -0.078
SO2         -0.107    -0.234  0.426    0.159  0.409  1.000     0.426
Mortality    0.509    -0.510  0.261    0.644 -0.078  0.426     1.000
```

Interpreting an SEM model containing both observed and latent variables (discussed latter) is the same as for the above path model.

65.2.4 Computing R squared from a regression using SEM

Utilising the standardized path coefficients in the SEM diagram opposite along with the simple correlations (below) we can obtain R^2 by using the equation discussed in the multiple linear regression chapter.

$$R^2 = \beta_1 r_{YX_1} + \beta_2 r_{YX_2} + \beta_3 r_{YX_3}$$
$$= (-.35)(-.510) + (.19)(.261) + (.57)(.644)$$
$$= .1770741 + .0488818 + .3694264$$
$$= .595 \ (3 decimal \ places)$$



Alternatively, we could have simply calculated:

`(1 - the standardised residual variance (SRV) of Mortality) = 1-.41 = 0.59.`

I hope that this strengthens your belief that the SRV is a measure of what remains unexplained after the regression analysis, as well as the idea that the proportion of variance explained can be calculated from the beta weights and the simple correlations between each input and the output.

To display the standardised path estimates alongside the unstandardised ones in Ωnyx you need to select each path and then select the menu option shown opposite.

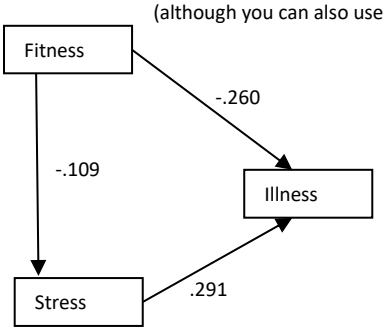
65.2.5 Calculating Direct, Indirect and Total Effects

In a structural equation model it is possible to calculate the various effects one or more variables have upon another in the model via other variables as well as directly.

From the example opposite it might appear easy to calculate such effects, however this is not the case with complex models as the indirect path may follow tortuous routes. To help simplify this problem Sewell Wright in the 1920's specified three rules:

Standardised Estimates

(although you can also use unstandardized ones to work out the various effects)



Direct effect of Fitness upon illness = -0.260

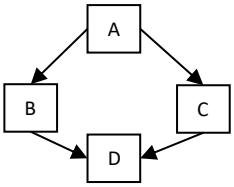
Indirect effect of Fitness upon illness = (-0.109)(0.291) = -0.031719

Total effects are the sum of all direct and indirect effects of one variable on another

Total effect of Fitness upon illness = -.260 + -0.031719 = -0.291719

Taken from Kline 2005p. 125 -129 adapted

Admissible tracing rule example



Indirect effect of **B** to **C** = BAC admissible (rule 1)
In contrast BDC is not admissible

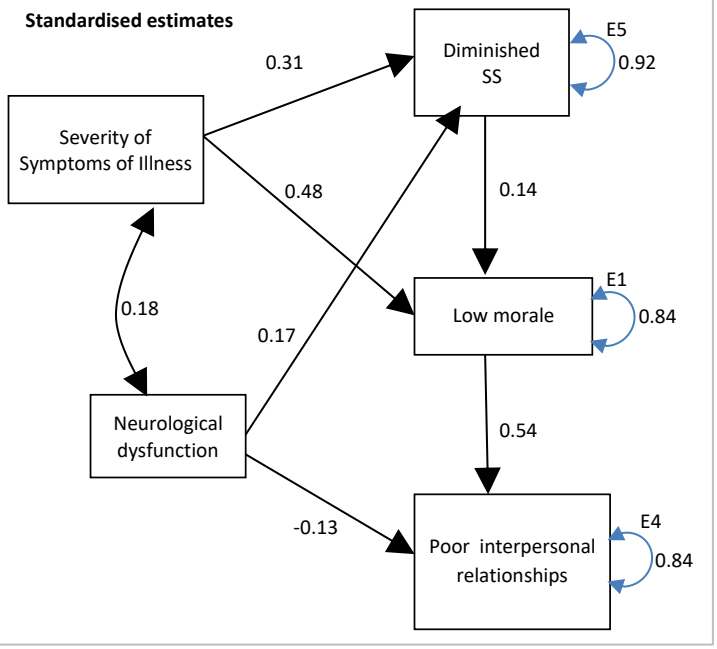
Taken from Loehlin 2004 p.9 (adapted)

1. You can trace backward up an arrow and then forward along the next, or forwards from one variable to another, but never forward and then back.
2. You can pass through each variable only once in a given chain of paths.
3. No more than one bi-directional arrow can be included in each path-chain.

Remember that the SEM model you have created will undoubtedly not contain all the possible paths between each variable so that there will also be **unanalysed effects.** Similarly, there may also be **spurious effects** in the model where you create a path that goes against the flow of the arrows at some point.

Once again, because you are more likely to need to interpret and calculate these measures from a diagram rather than carrying out a whole analysis yourself I have included an exercise below.

Adapted from Grimm & Yarnold, 1995 p.84



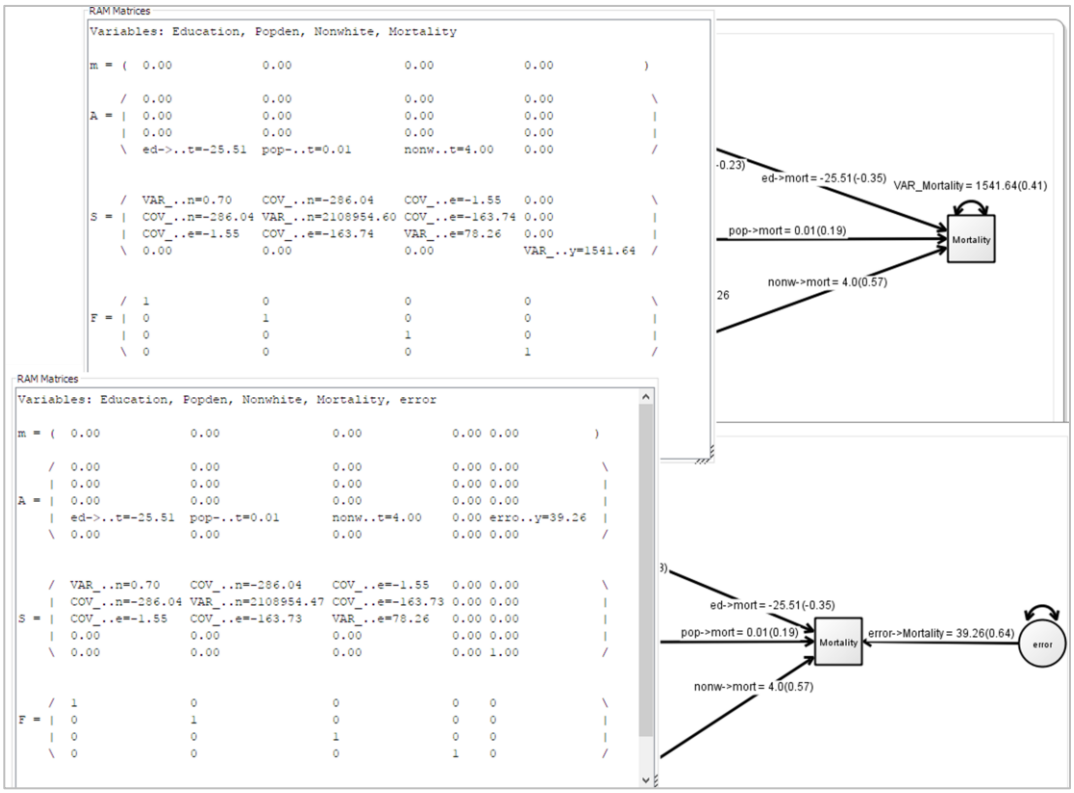
Self test:

1. What is the total effect of Severity of Symptoms of Illness on Poor Interpersonal relationships?
2. What is the total effect of Neurological dysfunction upon Diminished Socioeconomic Status?
3. What is the direct effect of Diminished Socioeconomic Status on Poor interpersonal relationships?

Hint: in reality often pathways which have very small coefficients (i.e. less than 0.1) are ignored.

65.3 Latent variables

So far all the variables in the models we have discussed in detail only contain observed variables (boxes). However we can also model latent variables. These are variables that are created from the data and mop up variation in the model that is not directly taken into account by the directed (regression) paths. I can demonstrate this by considering a regression model which I have depicted two ways below.



In the first the dependent variable is modelled with a error variance producing a unstandardized value of 1541.64. (*SD = 39.25*) Alternatively I have also modelled the regression where the dependant variable has no variance itself but is associated with a latent variable which I have called error. This latent variable consists of the residuals scores, that is the difference between the expected and predicted score for each case. The first table below shows the values for the first ten cases along with the residual and predicted (fitted) values from a standard regression analysis. The second table shows the values from the error latent variable. You can see that the residuals column from the traditional analysis is equivalent to this.

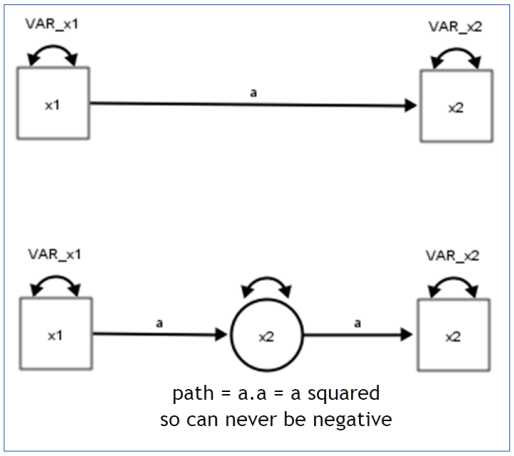
From traditional regression analysis:

rowname	Education	Popden	Nonwhite	Mortality	fitted	residuals
akronOH	11.4	3243	8.8	921.9	912.262883030003	9.63711696999752
albanyNY	11	4281	3.5	997.9	909.522843048244	88.3771569517563
allenPA	9.8	4260	0.8	962.4	929.164384657685	33.2356153423151
atlantGA	11.1	3125	27.1	982.3	992.17493049223	-9.87493049223009
baltimMD	9.6	6441	24.4	1071	1046.01268876674	24.9873112332572
birmhmAL	10.2	3325	38.5	1030	1062.32122750566	-32.3212275056638
bostonMA	12.1	4679	3.5	934.7	884.631050132788	50.068949867212
bridgeCT	10.6	2140	5.3	899.5	909.894939626843	-10.3949396268433
bufaloNY	10.5	6582	8.1	1002	958.979251244517	43.0207487554831
cantonOH	10.7	4213	6.7	912.3	929.433784297092	-17.1337842970919

From Onyx:

	Education	Popden	Nonwhite	Mortality	error
	Standardised scores				*Raw score*
akronOH	0.4266666666666641	-623.0500000000002	-3.070000000000004	-18.481666666666456	9.637116969994196
albanyNY	0.02666666666666373	414.9499999999998	-8.370000000000005	57.518333333333544	88.37715695175669
allenPA	-1.1733333333333356	393.9499999999998	-11.070000000000004	22.018333333333544	33.23561534231747
atlantGA	0.12666666666666337	-741.0500000000002	15.229999999999997	41.91833333333352	-9.874930492230668
baltimMD	-1.3733333333333366	2574.95	12.529999999999994	130.61833333333357	24.987311233272635
birmhmAL	-0.7733333333333337	-541.0500000000002	26.629999999999995	89.61833333333357	-32.32122750565975
bostonMA	1.1266666666666634	812.9499999999998	-8.370000000000005	-5.681666666666388	50.06894986721164
bridgeCT	-0.3733333333333366	-1726.0500000000002	-6.570000000000005	-40.88166666666643	-10.394939626850245
bufaloNY	-0.47333333333333627	2715.95	-3.770000000000005	61.61833333333357	43.020748755494466
cantonOH	-0.273333333333337	346.9499999999998	-5.170000000000004	-28.08166666666648	-17.13378429709057

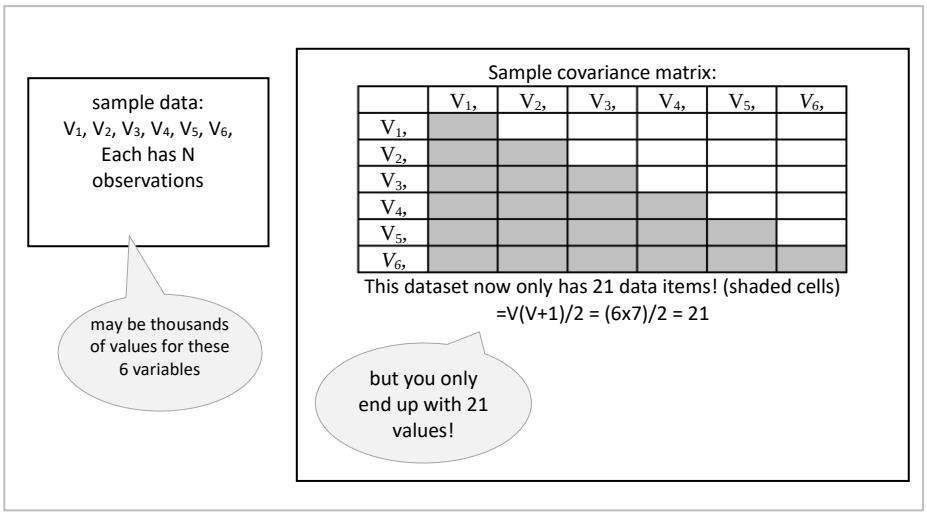
Latent variables therefore provide a method of modelling concepts such as measurement error and shared variance as we do in factor analysis and similar techniques. You can even use this approach to model repeated measures, multilevel models and Meta-analysis (Cheung 2015).



Another use of a latent variable is to add what is called a phantom variable (cheung 2016 p.38). The variance of the phantom latent variable is set to zero and adds nothing to the model. What is does is allow the modeller to impose a constraint such as making is non-negative.

We will now see how it is not always necessary to use the raw data to carry out a SEM analysis.

65.4 Covariances rather than raw data are used in model development

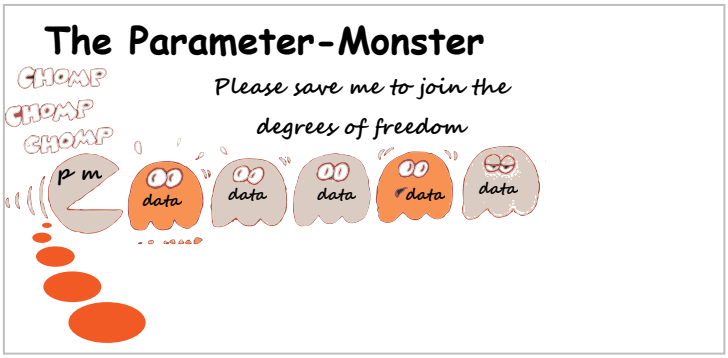


From the above discussion it should now be clear that calculation of the various path values (coefficients) only makes use of summary statistics (i.e. covariances or correlations). You may remember in the chapter on factor analysis we saw how a matrix of correlations along with a value specifying the sample size could be used instead of the raw data; the same goes for SEM analysis.

Related to this is the fact that authors are encouraged to publish the observed covariance matrix, to enable others to repeat at least part of the analysis when presenting a SEM analysis.

Just because the 'raw data' is not used in the calculation of the path values, it should not be assumed that sample size is not important, in contrast, it is extremely important. This is because we always need to assess if the various estimated values are any different from zero. This means it is necessary to either produce p-values or confidence intervals, both of which rely upon the calculation of standard errors (unless we use bootstrapping techniques), which itself is directly dependent upon the raw sample size. Furthermore, because SEM models often have a substantial number of variables and paths, a large sample size (as a rule of thumb, of at least 200) is needed to begin to make any sense of the data.

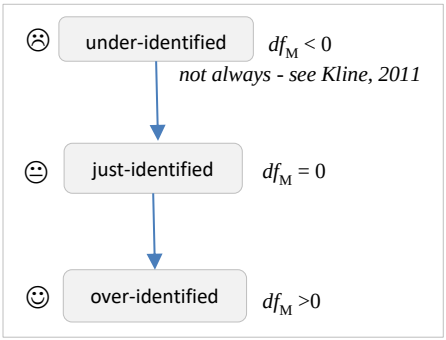
65.5 The identification problem and sample size



The identification aspect is basically asking, do we have enough data to allow us to develop a unique solution for our model. The following discussion, adapted from Schumacker & Lomax, 2010 p.63, provides an explanation. Suppose our model has two values we need to estimate (i.e. **free parameters**) and one data item. We can express this in the form of an equation:

$$X + Y = 10$$

If this is the case, X and Y can take on any number of values (e.g. x=1;y=9; x=2.5 y=7.5 etc.). There is no unique combination.



Imagine that each time the computer runs a program to find a solution to this problem it just randomly assigns a value between 0.1 and 9.9 to X and therefore provides a different solution for Y each time. When it is impossible to define a unique value to each of our parameters in our model the model is said to be **under-identified** and is useless. When we talk about identification, we are talking about the degrees of freedom for the model (df_m) - in other words the amount of data that can move freely after accounting for the parameter estimates. A under-identified model possesses negative degrees of freedom and no unique solution can be found.

In contrast, the desirable situation is to have a model which is preferably over-identified or, failing that, just identified. Ideally we want a model that estimates the smallest number of parameters but at the same time provides the best fit, thus maximising the accuracy of the estimates of our parameters (or more traditionally we say maximising the power of the study).

This ratio of the of sample size to number of parameters to be estimated in an SEM is important. Quoting Bagozzi & Yi, 2012 p.29: *As Bentler and Chou (1987, p. 91) state that the ratio "may be able to go as low as 5:1 under normal and elliptical theory, especially when there are many indicators of latent variables and the associated factor loadings are large". But they also believe that "a ratio of at least 10:1 may be more appropriate for arbitrary distributions."*. I will revisit this issue when discussing model fit indices.

65.6 Constraining parameters

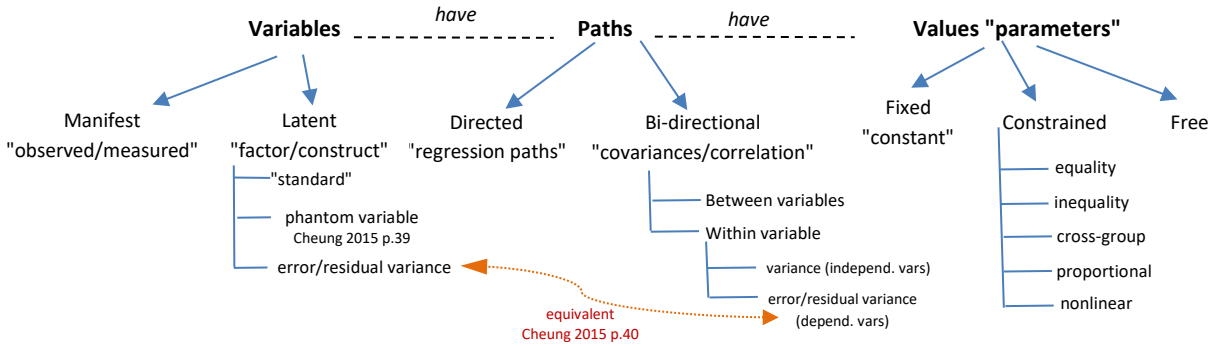
One solution to the identification problem is to set one or more of the parameters to a constant value, which is then known as a **fixed parameter**. An example of fixed parameters are all the hidden paths in the model indicating a correlation fixed to zero. Looking back at the model matrices section you can see these hidden values are represented by the zeros. Anther example of this is known as an **identification constraint**. We saw this in the factor models chapter where one of the loadings was set to 1 for each of the factors.

Types of constrained parameters (Kline 2016 p. 129)

1. **Equality** – two or more paths are set to equal one another
2. **Inequality** – this is where a value is specified to be within a specified range
3. **Cross-group** – This is where two of more paths across independent groups in a model are set to be equal.
4. **Proportional** – one parameter is proportional to another path estimate
5. **Nonlinear** – This is where one parameter is a nonlinear relation to another such as its square.

A constrained parameter is one that does not equal a constant but has some type of restriction imposed upon it. Kline 2016 p.129 mentions five types listed opposite. A good example of one of these can be found in my youtube video of the two independent samples *t* test as a SEM model https://www.youtube.com/watch?v=Hak-fyDc8fg

I have provided a graphical summary of some of the aspects discussed in the above sections, which you can use as a revision add.



Along with the identification problem comes the problem of assessing how well overall the model fits the observed data.

65.7 Model fit

There are many problems with taking an often complex model and obtaining a single value from it to represent the idea of fit. As there is no acceptable solution to this problem every SEM program provides over two dozen fit measures, and most research papers offer a selection! Because single fit measures provide an overall averaged value they unfortunately stop researchers looking at each individual parameter and its residual. There is always the possibility that the model may fit very well for the majority of the parameters but then have a large residual for one. If this is the case the researcher would then need to consider if the particular variable was correctly specified in the model and also how important is was in comparison to the other variables. I describe this as the *does my bum look big in this dress aspect.* Because of this I recommend that you always start by looking at the standardised residual matrix (Loehlin, 2004 p.69) which will highlight particular hotspots.

Schumacker & Lomax then suggest a three stage strategy (2010 p. 74):

1. Inspect two overall fit measures:
    - Chi-squared - you are aiming for an insignificant result i.e. high p-value (usually above .95) resulting from a small actual value. This value is problematic both when you have df=0 or a large initial sample (number of subjects, not the number of covariances). Various versions of this statistic such as GFI and AGFI have been developed to take these problems into account.
    - RMSEA (Root Mean Square Error of Approximation) - is at present the most popular measure, you can use it to produce confidence intervals or an associated p-value.
2. Inspect significance of individual parameters. The critical ratio (CR) for each parameter estimate is computed by dividing the estimate by its standard error. In most SEM programs it is compared to a z or t value of 1.96 at the .05 level of significance. Some software (i.e. EQS) indicate parameters that are statistically significant by placing an asterisk (*) beside them. Often in the model development process insignificant paths are removed from the model.

3. Inspect magnitude and direction of parameter estimates - does the value make sense in the proposed theoretical model? Has the computer provided impossible values such as negative variances (called Heywood cases) or correlations >1.

Observed Statistics	: 14.0
estimated Parameters	: 10
Restricted Degrees of Freedom	: 4.0
Minus Two Log Likelihood	: 620.5323387476835
Number of Observations	: 60.0
χ^2	: 1.1368683772161603E-13
AIC	: 640.5323387476835
AICc	: 644.5323387476835
BIC	: 661.4757843699045
RMSEA	: 0.0
SRMR (covariances only)	: 3.769947920989143E-9
CFI (to independent model)	: 1.079116807697788
TLI (to independent model)	: 0.9999999999999953

A typical set of model fitting statistics is shown opposite for the morality model obtained from the free Ωnyx software.

The chi-squared actual value is very small (E-13) meaning that we would have a *p*-value approaching 1, Indicating a good fit. The AIC and BIC measures are used to compare models where smaller values indicate a better fit.

Overall model fit metrics – Be wary indications of a good fit!

- Chi-squared p-value ≥ .95
- CFI≥0.95
- SRMR≤.08
- RMSEA ≤.10 "good" ≤ .05 "very good" fit

Baggozzi, 2012 quoting various writers recommends the following standards for assessing adequate fitness of SEM models: CFI≥0.95, and SRMR≤.08, also see Schumacker & Lomax, 2010. Again both the values reported opposite on the previous page indicate a good model fit. I'll discuss the RMSEA next.

65.7.1 RMSEA Root Mean Square Error of Approximation

The RMSEA is the golden boy of fit indices having many attractive attributes. It uses a value *d*, based the non-centrality parameter (Lambda =λ) of the chi-squared (χ^2) distribution which is simply the obtained chi-squared value of the model (i.e. Lambda =λ = obtained chi-squared).

$$RMSEA = \sqrt{\frac{d}{df}}$$

where

$$d = \frac{\chi^2 - df}{N - 1}$$

Taking the mortality example shown earlier we have a χ^2 = approximately zero. With 4 df in a sample of 60 cases, producing a *d* value of (0 - 4)/59 = -.066, and a RMSEA equal to √(.066/59), or -.001186. Because this value is less than zero we set the RMSEA value to zero. The maximum value RMSEA can obtain is 1.

The RMSEA measure was developed by Steiger (1998) who considers values below .10 "good" and below .05 indicative of "very good" fit. It is also possible to obtain a confidence limit for RMSEA using the noncentral χ^2, distribution, described in the *confidence intervals for effect sizes* chapter. The development of the RMSEA stems from Steiger's (Steiger, 1989) belief that the appropriate question for statistical analysis in SEM is not assessing how perfect the fit is but rather, the three following questions.

1. How well does my model fit my statistical population?

2. How precisely have we determined population fit from our sample data?

3. Does the fit still appear good when we take into account the complexity of the model and its number of free parameters?

By developing a confidence interval for the RMSEA we can take into account the three above issues testing a null hypothesis of poor, instead of perfect, fit. If the upper limit of the 90% confidence interval lies above the desired cutoff, for example .10, we can then reject the hypothesis that the fit of the model in the population is worse than our desired level. Similarly, if the CI does not extend beyond this cut-off we can conclude our model is of

adequate fit. The width of the confidence interval will also provide information about the accuracy of the estimate which is always useful. Because the RMSEA is based upon the non-central chi-squared distribution it can also form the basis of a power analysis discussed next.

65.8 Power and sample size determination

The previous section described the general rule of the ratio of the sample size to the number of parameters. However this rather questionably naïve approach has been improved upon by several writters.

The easiest approach is to consider the RMSEA for two models from which you can readily calculate the power. Schoemann, Preacher & Coffman, 2010 provide an online calculator along with several other comparable measures, along with R code. Similarly the R package SEM Tools contains a function *plotRMSEApower()* which allows the plotting of power across a range of sample sizes much like Gpower. The investigation of power during the planning stages of a study that intends to use SEM techniques can be a long and arduous process as it requires not least the investigators to define their proposed models at the start of the process and also make a guess (quite literally) of what the RMSEA value might be.

Therefore detailed power analysis is not for the faint-hearted and anyone planning to undertake this I would recommend they start by reviewing the literature. I would advise MacCallum, Browne & Sugawara, 1996; Hancock & Freeman, 2001; Lee, Cai & MacCallum, 2012 and Miles, 2003. Professor David Kaplan, University of Wisconsin Madison, provides a brief review of these at: http://www2.gsu.edu/~mkteer/power.html

Rex Kline who has written a popular introduction to SEM (Kline, 2016) and provides a set of notes concerning power estimation for a variety of SEM models, including some R code, at

http://psychology.concordia.ca/fac/kline/SEM/qicss2/qicss2setA.pdf

Professor Timo Gnambs at the Leibniz Institute for Educational Trajectories in Bamberg provides an online R code generator for SEM power analysis at:
http://timo.gnambs.at/en/scripts/powerforsem

65.9 Why and When to use Structural Equation Modelling

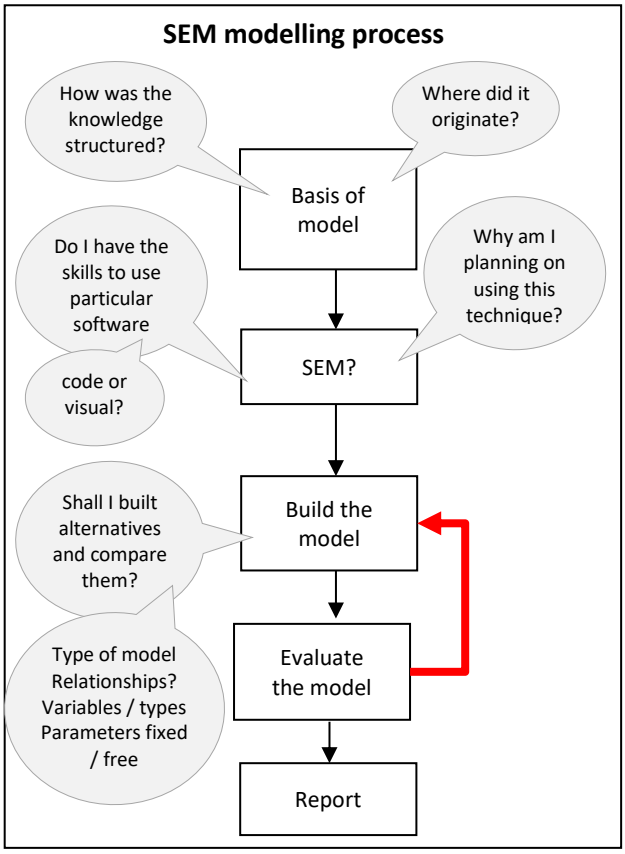
The following is a list of the common reasons given for using SEM techniques:

1. Confirm Relationships / hypotheses
2. Test complexity (multivariate)
3. Longitudinal / Multiple groups (panel designs)
4. Test Specific relationships
5. Multiple observed variables
6. Take measurement error into account

To use Structural Equation Modelling successfully often requires time to dedicate oneself to the SEM literature and also specific SEM software with the result that frequently those researchers that become competent tend to apply it to almost all problems. Because of the paradox between the complexity of the mathematics and interpretation of the results compared to the relative ease of being able to draw the diagrams for a structural equation model it is often inappropriately used.

One common problem often encountered with developing SEM models is that of identification, which is discussed next.

65.10 Stages of Structural Equation Modelling



After having discussed identification issues, model fit indices and power analysis it is clear that the development of SEM models is far more complex than just drawing a diagram. Complex models also require the analyst to know the research area in depth and be able to make sensible informed decisions concerning prospective models.

Once you have a working model that the program is able to estimate then you often need to change it to perform comparisons with various other models. Often only after you have done this several times do you finally come to a decision deciding which is the 'best' possible one.

I'll now demonstrate how you can create a simple SEM model by repeating the analysis described in the factor analysis chapter. I'll also demonstrate a more complex model.

65.11 A basic SEM model – the Hozinger & Swineford 1939 data analysed using Onyx (Ωnyx)

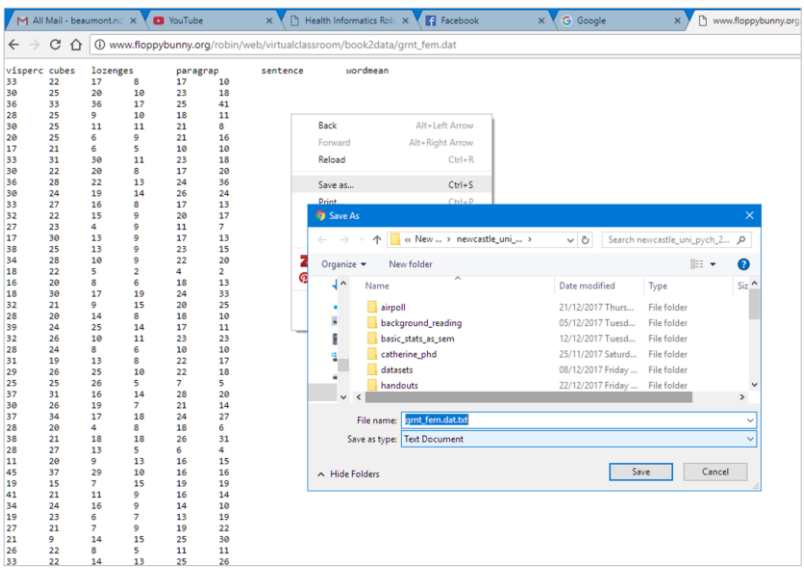
Details of the Hozinger & Swineford 1939 dataset were discussed in the factor analysis chapter. In this section we will use a separate program that produces R code called Ωnyx (Ωnyx) to carry out the SEM analysis. This allows us to specify the model using a set of graphical symbols which you may recognise from the factor analysis chapter.

65.11.1 Preliminaries

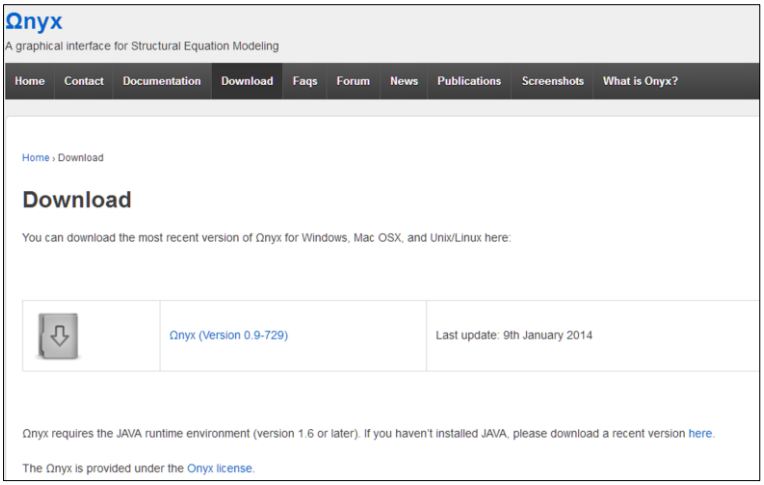
Watch my youtube video demonstrating the steps described below:

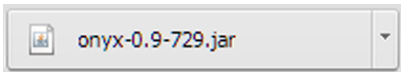
https://youtu.be/GN-HOWd31Ao (this link is case sensitive, the last character is lower case 'o' as in 'on')

You first need to download the dataset (tab-delimited format) from the link below and save it locally: http://www.robin-beaumont.co.uk/virtualclassroom/book2data/grnt_fem.dat you can also find it at the books website.



If you go to the link above the page opposite will appear showing the data, while on the page right mouse click and save the data locally.





Before you can use the Ωnyx program you need to have Java installed which you can find at:

http://www.java.com/en/

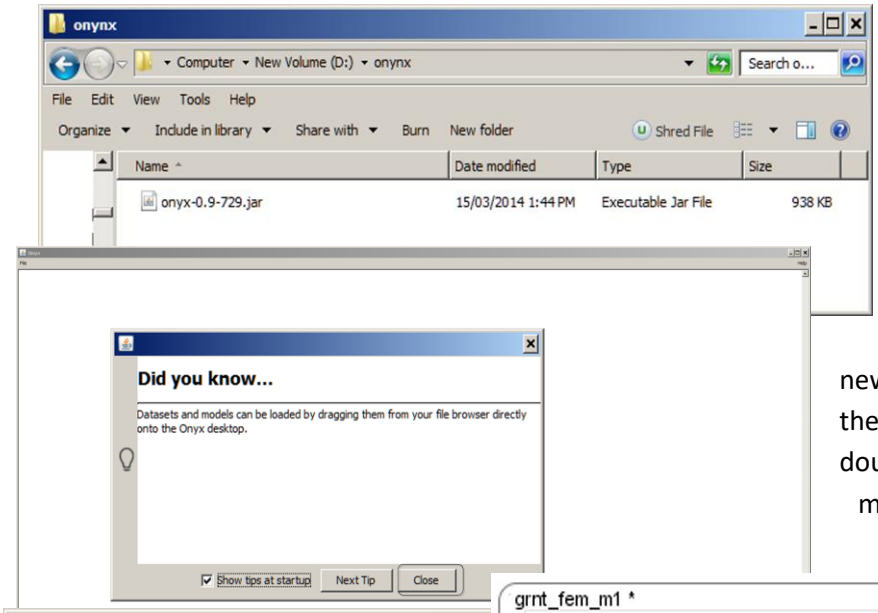
Then download Ωnyx from:

http://onyx.brandmaier.de/download/

The Ωnyx program, being a java file, has a jar extension, which you save to a local folder.

As shown opposite, I created a folder on my d: drive called onyx.

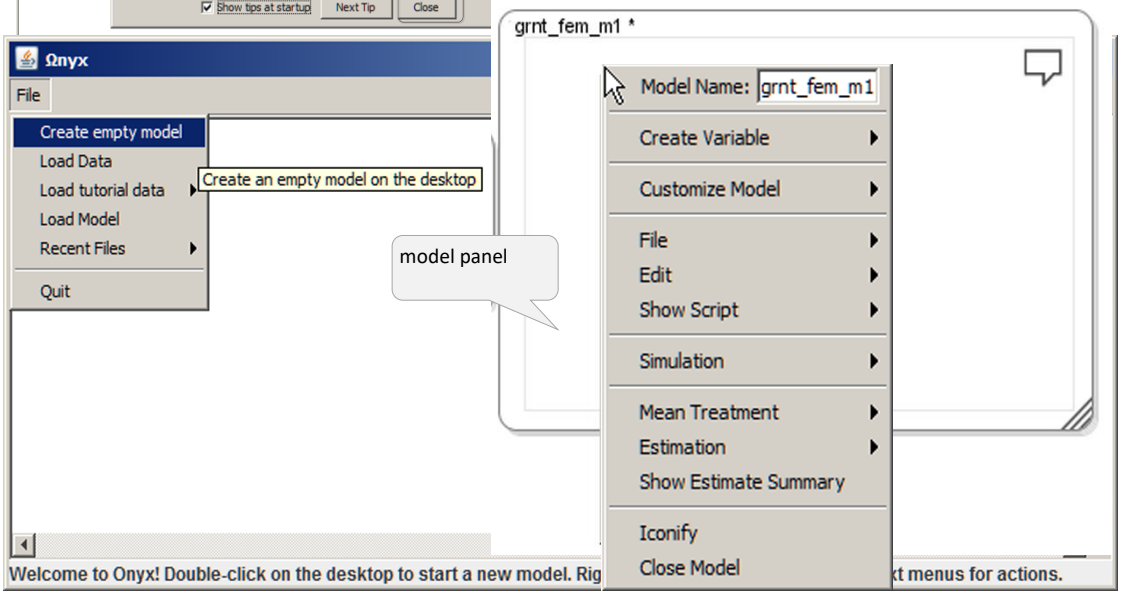
To run Ωnyx *Double click* on the jar file. This brings up the main window of Ωnyx.

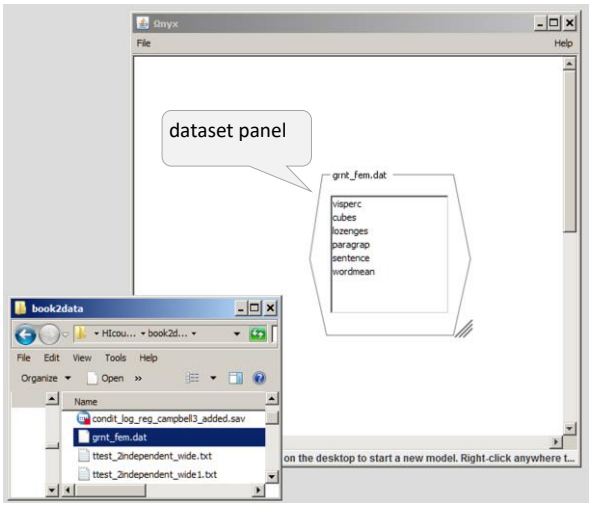


65.11.2 Creating an SEM model in Ωnyx

The main window of Ωnyx contains a tip of the day box which you can close by clicking on the **close** button. The tip also appears on the bottom bar of the Ωnyx window.

The first thing you need to do is create a new model. You can do this either by selecting the option from the dropdown menu or by double left mouse clicking in the desktop. The model panel now appears. You can give the model a name by *right mouse clicking* anywhere in the window. I have called it *grnt_fem_m1*.





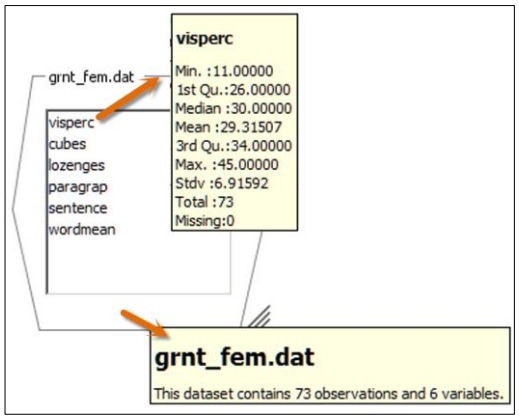
We can now proceed to specify the model by either drawing the symbols and then linking them to the fields in your dataset or, the technique we are going to use now, loading the dataset into Ωnyx and dragging the fields across to the model panel.

To load the dataset into Ωnyx, either select the following menu option or simply drag the data file from the file browser window.

File->load data

A hexagonal window (called the dataset panel) appears which represents your dataset.

Ωnyx has context-sensitive popup menus, which means that if you hold the mouse cursor over various elements you get information about them



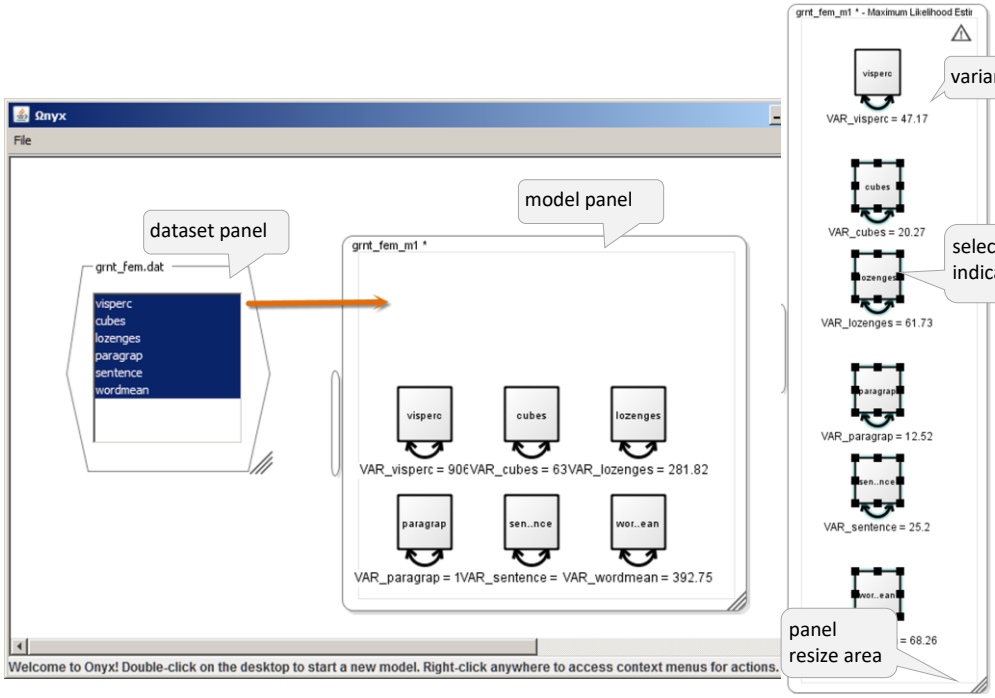
To add one or more variables to your model window simply drag them from the dataset panel to the model panel. You can select more than one using either the *shift* (continuous blocks) or *CTRL* keys (discontinuous blocks).

Ωnyx gives each graphical representation of the variable the same name as the variable in the dataset.

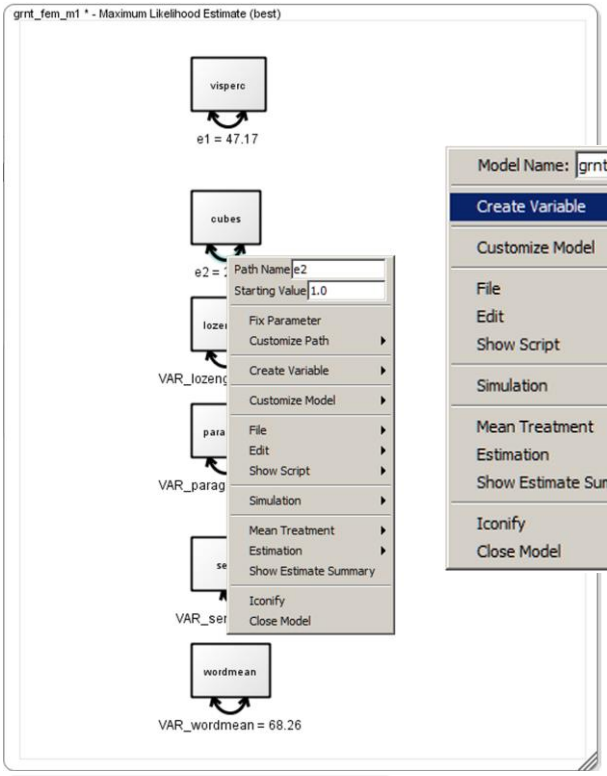
You can move and reshape the elements by selecting and then clicking and dragging.

Similarly, you can move the model panel by dragging its top or bottom border (a hand appears) or resize it by dragging the right hand lower corner .

We are going to create a type of SEM model known as a Confirmatory Factor Analysis (CFA) and to interpret the findings of such an analysis is it best to have the variables standardised. Luckily, you don't need to convert the variables as Ωnyx can do it for you.



The error variance for each variable is often simply labelled *E1* to *En* where *n* is the number of variables in the model requiring error variances.



To edit them we follow a similar process to that above. First select the error variance (*left mouse click*), that is the little loop attached to each variable then bring up the menu (*right mouse click*). This menu allows you to change the path name, where I have gone through each in turn and renamed them *e1* to *e6*.

We now need to add the two latent variables (called factors in the last chapter).

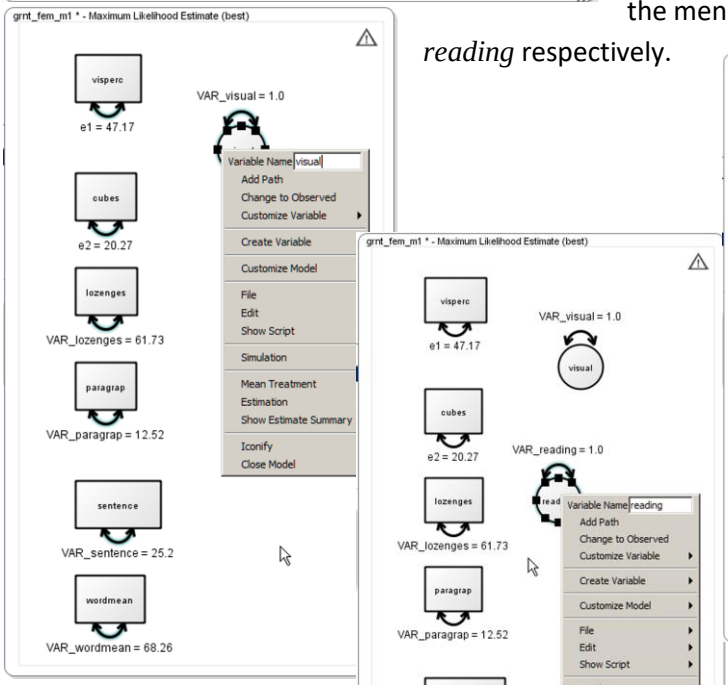
Either double right mouse click in an empty part of the model panel

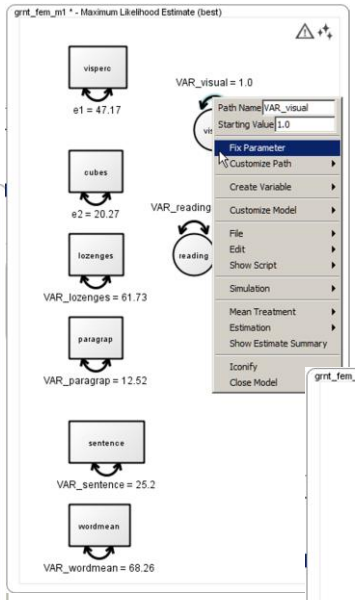
or

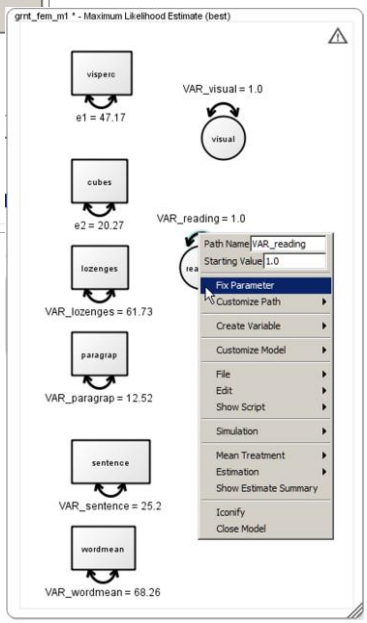
Right mouse click on the model panel to bring up the menu shown opposite, then select the following:

Create Variable->Latent

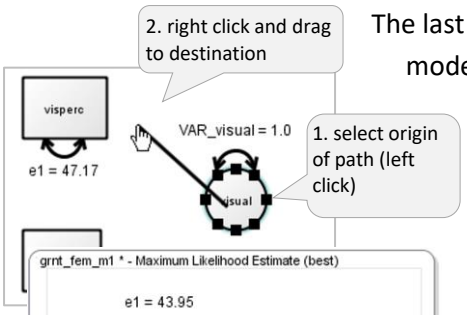
We select each latent variable (factor) in turn and then bring up the menu (*right mouse click*) to change the names to *visual* and *reading* respectively.







We now need to fix both the error variances associated with these latent variables . Select the variance paths (*left mouse click*) and then bring up the menu (*right mouse click*). Each time select the **Fix Parameter** option from the menu.



The last thing to do is to specify the relationships between the variables in the model. First the paths between the observed variables and the latent variables.

To draw a path select the variable for the paths origin (*left mouse click*) then:

Right mouse click on it and drag to the target variable.

I repeat this several times to create the model shown on the left.

Important: note that the arrows go from the latent variables to the observed ones.

The style described above is the traditional way of defining a factor model using what is known as reflective measurement. An alternative, more contentious approach is to have the arrows pointing in the other direction producing a formative measurement model. However there are questions concerning the validity of this latter approach and I suggest that anyone who believes otherwise consult various articles in Psychological Methods, 2007, 12(2), specifically Howell, Breivik, & Wilcox, 2007.

The only other path to add is that for the correlation (i.e. covariance) between the two latent variables.

Select one of the latent variables by left mouse clicking on it and hold down the *shift key + right mouse click* then drag to target variable.

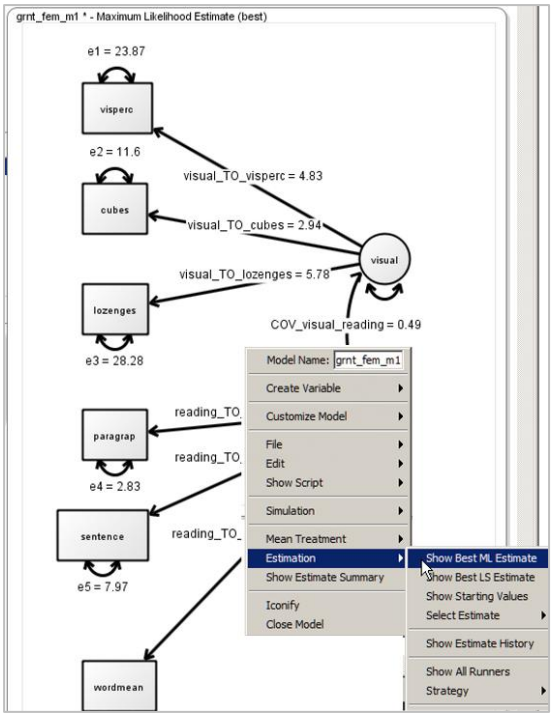
The far left screenshot shows that by hovering the mouse cursor over this line we discover it is a fixed parameter estimate whereas we want the model to estimate its values. To change it:

Select the path (*left mouse click*) then *right mouse click* on it to bring up the menu and select the following menu option:

Free Parameter

You need to carry out this process for all the paths directed from the observed variables to the latent ones as well. Once again, you can check to see which ones are fixed or allowed to be estimated by hovering over each path.

65.11.3 Model estimation

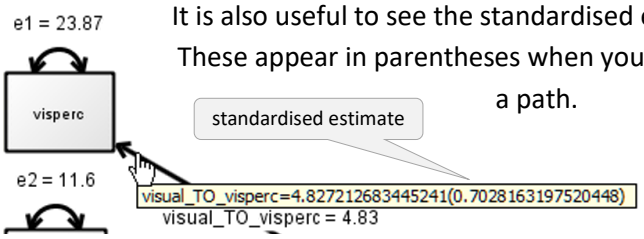


While we have been specifying the model, the Ωnyx program has been calculating the parameter estimates along with several model fit values.

You can update the diagram to show the 'best' parameter estimates by right mouse clicking on an empty part of the model panel and then selecting

Estimation-> Show Best ML Estimate

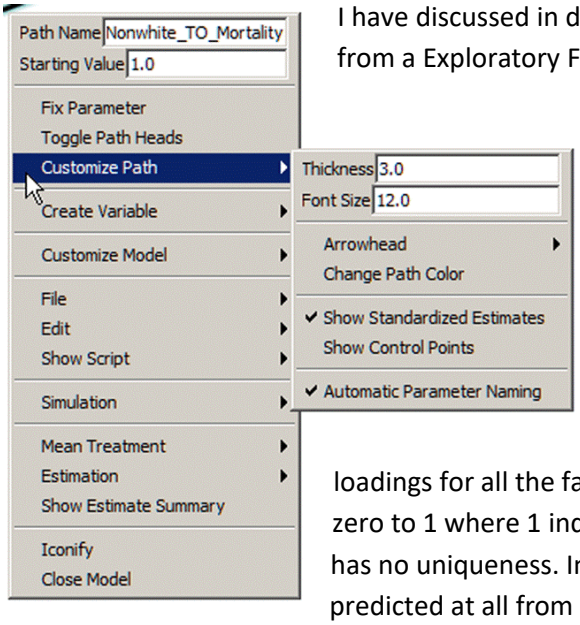
65.11.3.1 Displaying standardised values



It is also useful to see the standardised estimates. These appear in parentheses when you hover over a path.

To display the standardised path estimates alongside the unstandardized ones in an Ωnyx diagram you need to select each path and then select the menu option shown opposite. You can select multiple paths by using the *shift+ left* mouse click combination.

65.11.4 The results



I have discussed in detail in the previous chapterl the meaning of the various results from a Exploratory Factor Analysis, and the same interpretations apply here.

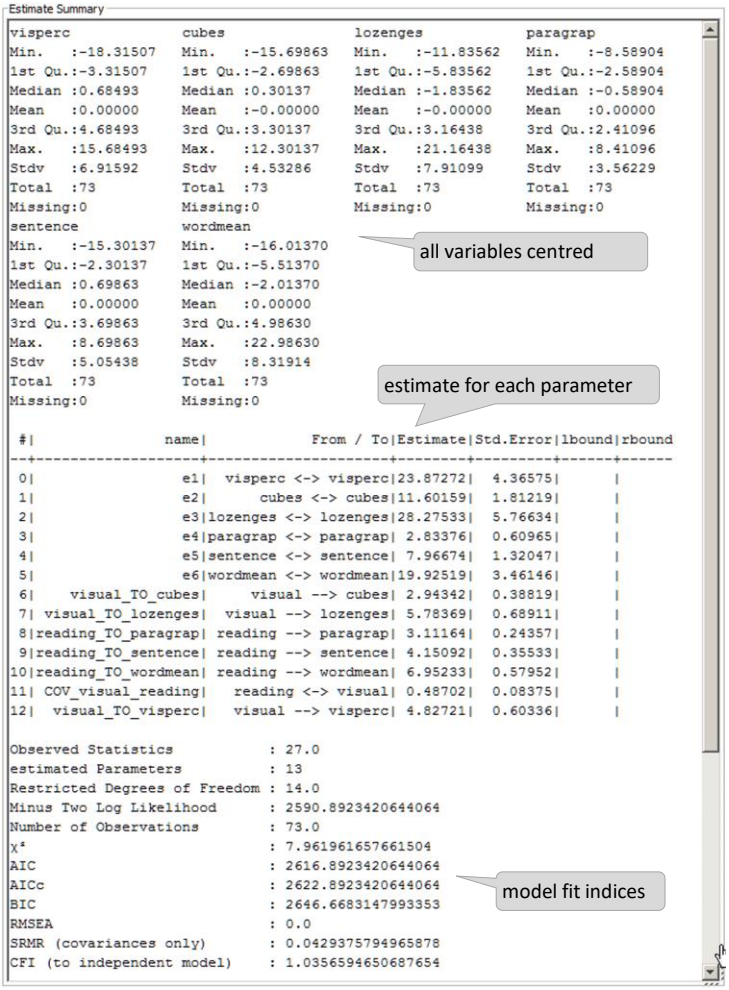
Factor loadings These are the paths coefficients between a specific observed variable ('indicator') and a specific factor. Higher values mean a closer relationship. They are equivalent to standardised regression coefficients (β weights) in multiple regression. **Higher the value the better.**

Communality (h2) This is the total influence on a single observed variable from all the factors associated with it. It is equal to the sum of all the squared [standardised] factor loadings for all the factors related to the observed variable. The value ranges from zero to 1 where 1 indicates that the variable can be fully defined by the factors and has no uniqueness. In contrast a value of 0 indicates that the variable cannot be predicted at all from any of the factors. The communality (h2) can be derived for each variable by taking the sum of the squared factor loadings for each of the factors associated with the variable. As these values are equivalent to the R squared values in multiple regression, we can interpret them the same way; they represent the % of variability taken into account by the model. Because we are hoping that the observed dataset is reflected in the model we **want** these values to be as **high** as possible, **nearer to one the better**.

Uniqueness (u2) for each observed variable this is that portion of the variable that cannot be predicted from the other variables (i.e. the latent variables). As the communality can be interpreted as the % of the variability that is predicted by the model we can say that uniqueness is the % of variability in a specific observed variable that is NOT predicted by the model. This means that we **want** this value for each observed variable to be as **low** as possible. **Lower the better**.

65.11.5 Obtaining the results

In Ωnyx the results are obtained by right mouse clicking on an empty part of the model panel and then selecting **Estimate summary**.



Estimate Summary

```
visperc             cubes               lozenges            paragrap
Min.   :-18.31507   Min.   :-15.69863   Min.   :-11.83562   Min.   :-8.58904
1st Qu.:-3.31507    1st Qu.:-2.69863    1st Qu.:-5.83562    1st Qu.:-2.58904
Median :0.68493     Median :0.30137     Median :-1.83562    Median :-0.58904
Mean   :0.00000     Mean   :-0.00000    Mean   :-0.00000    Mean   :0.00000
3rd Qu.:4.68493     3rd Qu.:3.30137     3rd Qu.:3.16438     3rd Qu.:2.41096
Max.   :15.68493    Max.   :12.30137    Max.   :21.16438    Max.   :8.41096
Stdv   :6.91592     Stdv   :4.53286     Stdv   :7.91099     Stdv   :3.56229
Total  :73          Total  :73          Total  :73          Total  :73
Missing:0           Missing:0           Missing:0           Missing:0
sentence            wordmean
Min.   :-15.30137   Min.   :-16.01370
1st Qu.:-2.30137    1st Qu.:-5.51370
Median :0.69863     Median :-2.01370
Mean   :0.00000     Mean   :0.00000
3rd Qu.:3.69863     3rd Qu.:4.98630
Max.   :8.69863     Max.   :22.98630
Stdv   :5.05438     Stdv   :8.31914
Total  :73          Total  :73
Missing:0           Missing:0
```

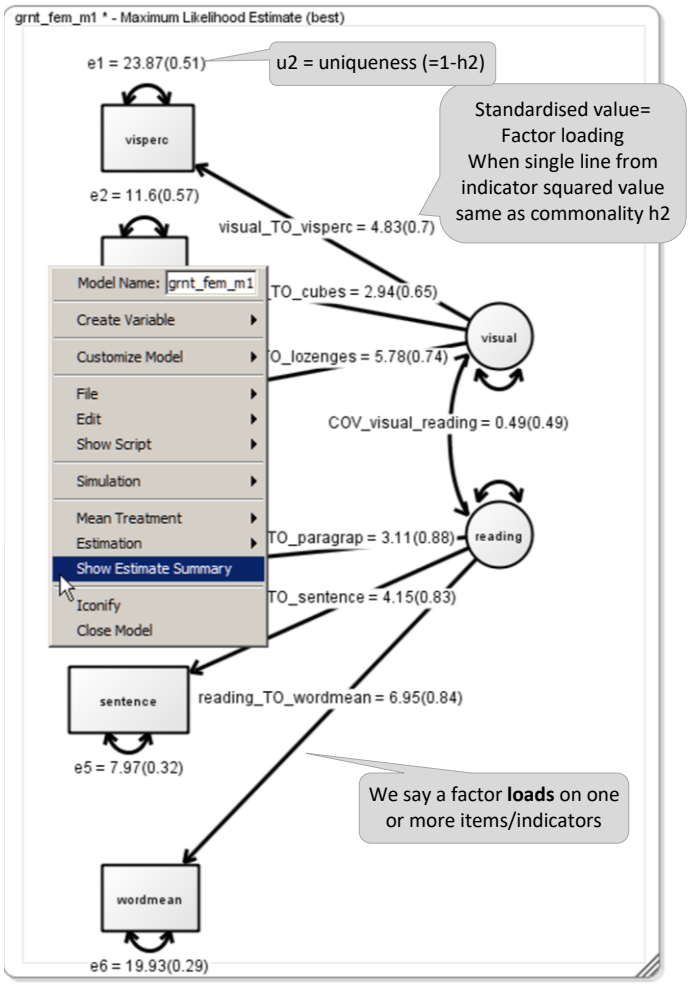
all variables centred

estimate for each parameter

#	name	From / To	Estimate	Std.Error	lbound	rbound
0	e1	visperc <-> visperc	23.87272	4.36575		
1	e2	cubes <-> cubes	11.60159	1.81219		
2	e3	lozenges <-> lozenges	28.27533	5.76634		
3	e4	paragrap <-> paragrap	2.83376	0.60965		
4	e5	sentence <-> sentence	7.96674	1.32047		
5	e6	wordmean <-> wordmean	19.92519	3.46146		
6	visual_TO_cubes	visual --> cubes	2.94342	0.38819		
7	visual_TO_lozenges	visual --> lozenges	5.78369	0.68911		
8	reading_TO_paragrap	reading --> paragrap	3.11164	0.24357		
9	reading_TO_sentence	reading --> sentence	4.15092	0.35533		
10	reading_TO_wordmean	reading --> wordmean	6.95233	0.57952		
11	COV_visual_reading	reading <-> visual	0.48702	0.08375		
12	visual_TO_visperc	visual --> visperc	4.82721	0.60336		

```
Observed Statistics           : 27.0
estimated Parameters          : 13
Restricted Degrees of Freedom : 14.0
Minus Two Log Likelihood      : 2590.8923420644064
Number of Observations        : 73.0
χ²                            : 7.961961657661504
AIC                           : 2616.8923420644064
AICc                          : 2622.8923420644064
BIC                           : 2646.6683147993353
RMSEA                         : 0.0
SRMR (covariances only)       : 0.0429375794965878
CFI (to independent model)    : 1.0356594650687654
```

model fit indices



The results are shown above. It is important to realise that Ωnyx calculates the results 200 times and the final part of the estimate summary provides details of the results from which we can see that luckily the results stabilised.

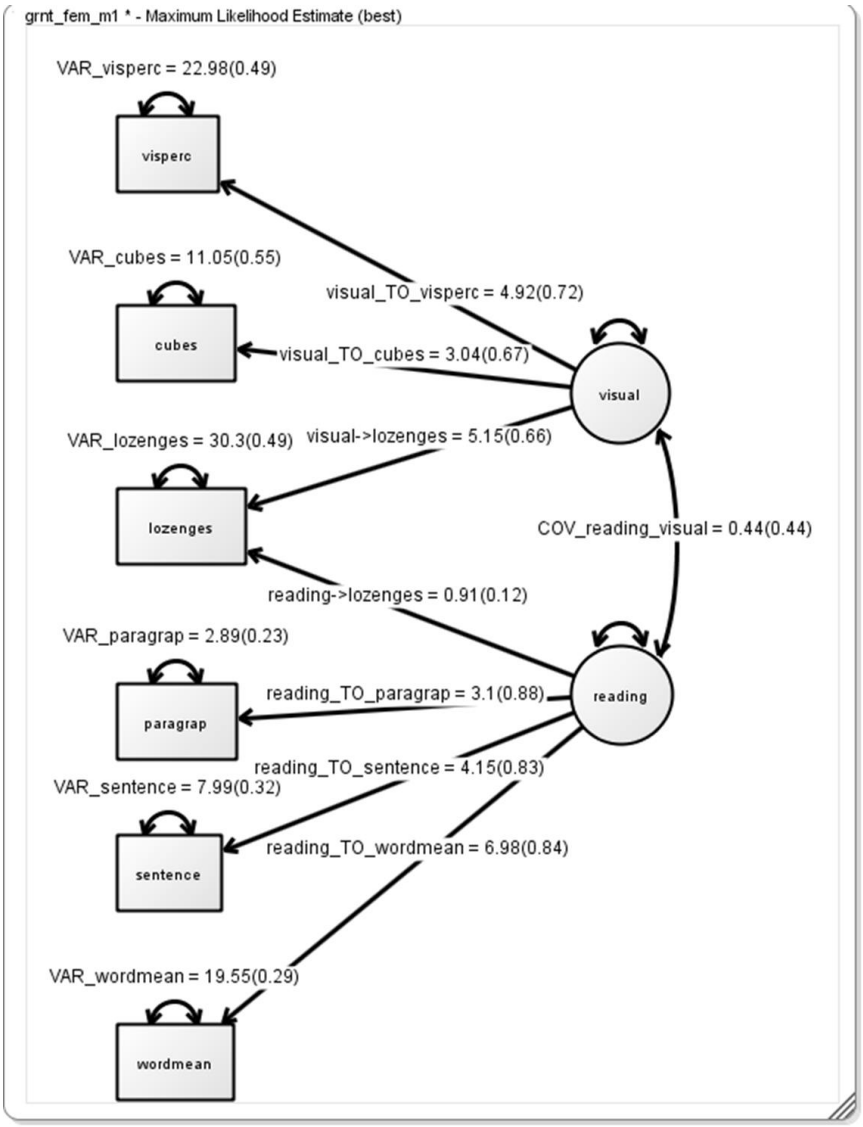
```
This estimate is a local optimum, better estimates exist.
This estimate is reliably converged.
There are 20 local maxium likelihood optima found so far, 14 of them reliable.
This estimate has been found with 7 of 200 starting value sets converged in total
The overall estimation situation has stabilized.
```

Overall model fit metrics – Be wary indications of a good fit!

- Chi-squared p-value ≥ .95
- CFI≥0.95
- SRMR≤.08
- RMSEA ≤.10 "good" ≤ .05 "very good" fit

Because the above model is very similar to that presented in the factor analysis chapter, mimicking most closely the Promax rotation result, we can interpret the paths between the latent variables and the observed variables as commonality values and the error terms as being the measurement errors. The slight differences between these values and those in the factor analysis chapter, where u2i + h2i =1, is because here we have constrained several of the paths to zero (i.e. there is no path between lozenges and verbal). This is known as a Confirmatory Factor Analysis (CFA) approach in contrast to the previous chapter where we followed what is known as an Exploratory Factor (EFA) approach.

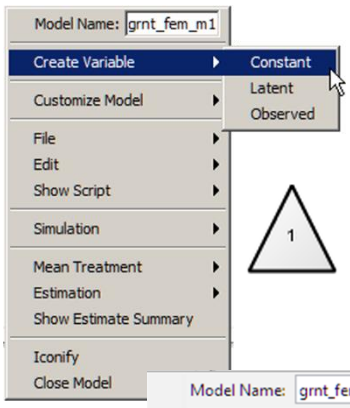
As mentioned above when there is a single path from a observed variable (i.e. indicator) to the factor (i.e. latent variable) the communality is equal to the squared factor loading which is the same as the squared standardised path coefficient.



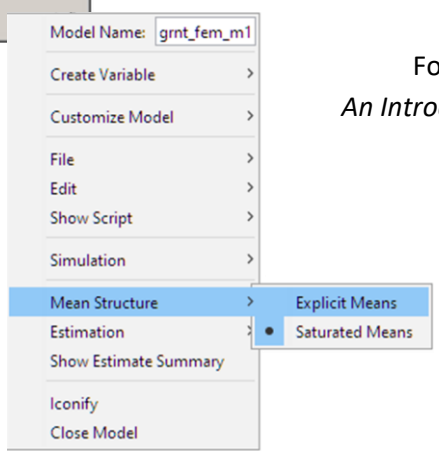
In the example opposite I have now added what is known as cross factor loading where both factors load on the *Lozenges* indicator. Here the communality for lozenges is now $0.66^2 + .12^2 = 0.45$ which is different from the previous value of $74^2 = 0.5476$ in other words this is a less desirable value than we previously had. In the section looking at the lavaan output we will look in more depth at the two models.

65.12 Adding intercepts - Mean structures

For more complex SEM models, specifically where we wish to investigate possible differences between means in several groups, it is necessary to add this information to the model. This is because so far we have been working with centred variables (i.e. mean = zero), the so-called saturated means model.



Graphically to explicitly specify the means in a SEM model we add a special graphical symbol, a triangle, to the model and then draw lines from it, pointing to those variables for which you want the means to be included in the model. Adding this symbol implicitly sets the menu option:



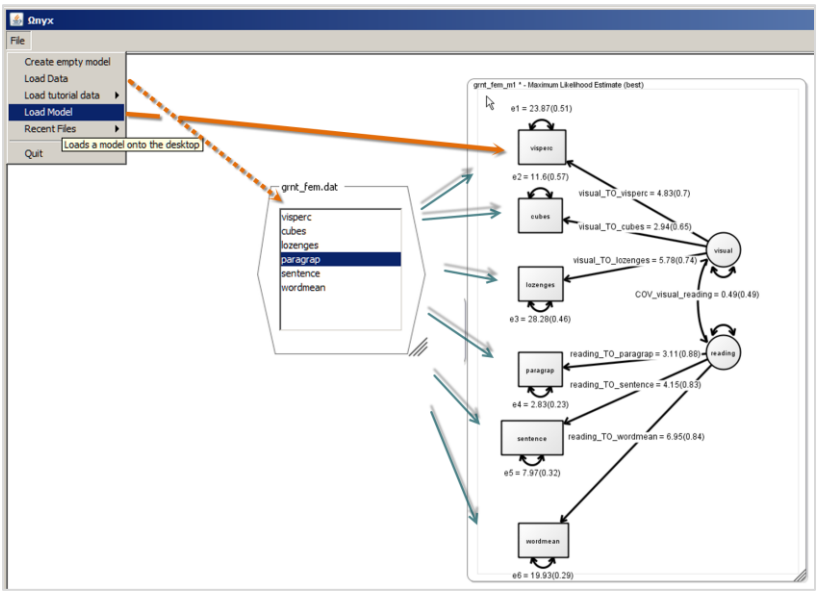
Mean Structure -> Explicit Means.

For further details see the online chapters at the books website:
An Introduction to Regression Modelling using Structural Equation Modelling (SEM)

and

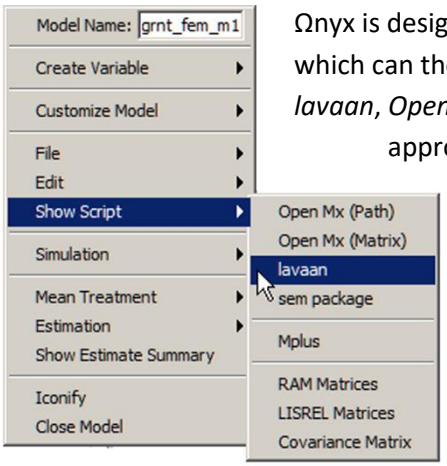
SEM equivalent to basic statistical procedures

65.13 Reloading a saved model



Once you have saved a model in Ωnyx when you reload it all the connections to the data will have disappeared. To reform these connections you simply open the data file again, then drag the appropriate field name into the correct object in the model panel. Repeat this process for each field.

65.14 Linking Ωnyx models to R

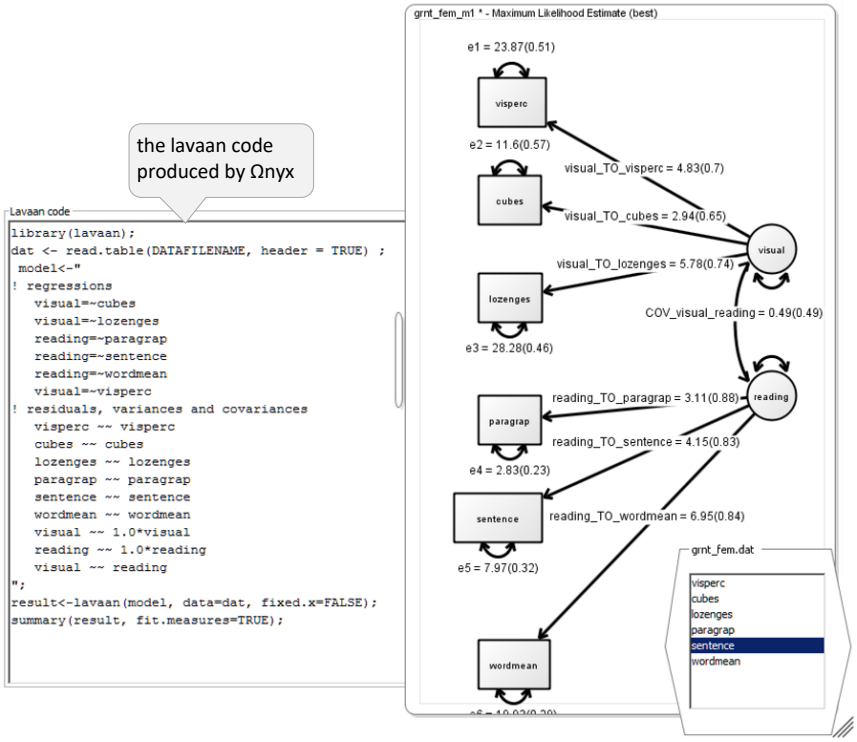


Ωnyx is designed primarily as a pedagogical tool allowing you to specify models graphically which can then be run using actual R packages. Specifically it supports three R packages *lavaan*, *OpenMx* and *SEM*. The way it supports these R packages is by producing the appropriate R code for each. We will begin by looking at the *Lavaan* code.

65.14.1 Lavaan

I will demonstrate the use of Lavaan using the code generated by Ωnyx for the Confirmatory Factor Analysis (CFA) described above. You first need to obtain and load the data to go with the model into R. Copy the following code into the R Console window.

```
hozdata<- read.table("http://www.robin-beaumont.co.uk/virtualclassroom/book2data/grnt_fem.dat",
 sep="\t", header=TRUE);        names(hozdata)
```



Within R you first need to install and load the lavaan package:

```
install.packages("lavaan", dependencies =
TRUE); library(lavaan)
```

The *Lavaan* package makes use of what it calls lavaan model syntax which describes a latent variable model, and luckily Ωnyx produces this code for us. You can see this syntax by selecting the **lavaan** option in the menu shown above, this appears when you *right mouse click* in an empty area of the model pane. The resulting lavaan code window is shown opposite.

Lavaan code generated (2018)

```
library(lavaan);
modelData <- read.table(DATAFILENAME, header = TRUE) ;
 model<-"
! regressions
  visual=~visual_TO_cubes*cubes
  visual=~visual_TO_lozenges*lozenges
  reading=~reading_TO_paragrap*paragrap
  reading=~reading_TO_sentence*sentence
  reading=~reading_TO_wordmean*wordmean
  visual=~visual_TO_visperc*visperc
! residuals, variances and covariances
  visperc ~~ VAR_visperc*visperc
  cubes ~~ VAR_cubes*cubes
  lozenges ~~ VAR_lozenges*lozenges
  paragrap ~~ VAR_paragrap*paragrap
  sentence ~~ VAR_sentence*sentence
  wordmean ~~ VAR_wordmean*wordmean
  visual ~~ 1.0*visual
  reading ~~ 1.0*reading
  reading ~~ COV_reading_visual*visual
! observed means
  visperc~1;
  cubes~1;
  lozenges~1;
  paragrap~1;
  sentence~1;
  wordmean~1;
";
 result<-lavaan(model, data=modelData, fixed.x=FALSE,
missing="FIML");
summary(result, fit.measures=TRUE);
```

A 'label' indicated by the * to the right of it

A mean structure appears to be added!

If you delete the mean structure you MUST remove the missing=FIML option as well

Since the book was written the code generated has become more verbose and includes all the path labels and an additional observed means section. The path labels appear just before the '*' symbol and in this context they should be seen as a way of identifying modifiers not multiplication. The lavaan website provides further details http://lavaan.ugent.be/tutorial/syntax2.html

As with R Commander this auto-generated code is more complex than need be, these automated tools lack the aesthetics and creative originality of a human! But it does provide a very useful starting point.

The code is punctuated by several comments lines (!) where each subsection defines a particular aspect of the model. I have reproduced the code below with detailed comments. To obtain the equivalent analysis to Ωnyx in R using the *lavaan* package requires slight modification of the code which we will now undertake. You can safely remove the path labels, the observed means section and the missing=FIML option, as I have done below:

specify the data filename (text file) you wish to use for the model –by editing this value or remove this line. I usually remove it. See below

load the lavaan package, we have already done that, so you can remove this line

```
library(lavaan);
dat <- read.table(DATAFILENAME, header =
TRUE) ;
 model<-"
! regressions
   visual=~cubes
   visual=~lozenges
   reading=~paragrap
   reading=~sentence
   reading=~wordmean
   visual=~visperc
! residuals, variances and covariances
   visperc ~~ visperc
   cubes ~~ cubes
   lozenges ~~ lozenges
   paragrap ~~ paragrap
   sentence ~~ sentence
   wordmean ~~ wordmean
   visual ~~ 1.0*visual
   reading ~~ 1.0*reading
   visual ~~ reading
";
result<-lavaan(model, data=dat,
fixed.x=FALSE);
summary(result, fit.measures=TRUE);
```

" or '
we now begin to specify the lavaan model, basically we have model_name <-"
note the double quote that is important

a comment line, you could use the # symbol instead

=~
Latent variable (continuous variables) definitions. The name of the latent variable is on the left of the "=~" operator, while the terms on the right, separated by "+" operators, are the indicators of the latent variable. Alternatively, as here each is on a separate line. Read "=~" as "is measured by"

~~
Variances and covariances. The "~~" ('double tilde') operator specifies (residual) variances of an observed or latent variable, or a set of covariances between one variable, and several other variables. Several variables, separated by "+" operators can appear on the right. This way, several pairwise (co)variances involving the same left-hand variable can be expressed in a single expression. The distinction between variances and residual variances is made automatically.

variances

"*" are referred to as modifiers. 1 * sets the variance to one, it becoming a fixed parameter

covariances - only one in this model

you need to specify the dataframe here. Change "data=data" to "data = hozdata"

now send the model definition to the lavaan function

produce a set of summary results for our model including fit indices

" or '
To indicate we have finished specifying the lavaan model. Followed either by a SEMi colon (;) or a blank line

~ (not used in above model)
Regression definitions. The output (dependent) variable is on the left of a "~" operator and the input (independent) variables, separated by "+" operators, are on the right.

Taking into account the annotations described about we now have:

```
lavaan (0.5-16) converged normally after  51 iterations

  Number of observations                            73

  Estimator                                         ML
  Minimum Function Test Statistic                7.962
  Degrees of freedom                                 8
  P-value (Chi-square)                           0.437

Model test baseline model:

  Minimum Function Test Statistic              190.325
  Degrees of freedom                                15
  P-value                                        0.000

User model versus baseline model:

  Comparative Fit Index (CFI)                    1.000
  Tucker-Lewis Index (TLI)                       1.000

Loglikelihood and Information Criteria:

  Loglikelihood user model (H0)              -1295.446
  Loglikelihood unrestricted model (H1)      -1291.465

  Number of free parameters                         13
  Akaike (AIC)                                2616.892
  Bayesian (BIC)                              2646.668
  Sample-size adjusted Bayesian (BIC)         2605.705

Root Mean Square Error of Approximation:

  RMSEA                                          0.000
  90 Percent Confidence Interval          0.000  0.137
  P-value RMSEA <= 0.05                          0.569
```

```
library(lavaan);
dat <- hozdata
 model<-"
! regressions
  visual=~cubes
  visual=~lozenges
  visual=~visperc
  reading=~paragrap
  reading=~sentence
  reading=~wordmean
! residuals, variances and covariances
  visperc ~~ visperc
  cubes ~~ cubes
  lozenges ~~ lozenges
  paragrap ~~ paragrap
  sentence ~~ sentence
  wordmean ~~ wordmean
  visual ~~ 1.0*visual
  reading ~~ 1.0*reading
  visual ~~ reading
";
result<-lavaan(model, data=dat, fixed.x=FALSE);
summary(result, fit.measures=TRUE);
```

The above summary function produces the output on the left showing a range of model fit indices. To gain R squared and standardised values you can edit the summary function to:

```
summary(result, fit.measures=TRUE, standardized=TRUE, rsq=TRUE)
```

```
R-Square:

    visperc          0.494
    cubes            0.428
    lozenges         0.542
    paragrap         0.774
    sentence         0.684
    wordmean         0.708
```

To gain further information about the parameter estimates we use the *parameterEstimates()* function which also provides confidence intervals, given below. I have shaded out two irrelevant columns.

```
parameterEstimates(result, standardized = TRUE)
```

standardized estimate either a standardised path value, covariance or a variance

```
> parameterEstimates(result, standardized = TRUE)
        lhs op      rhs     est    se     z pvalue ci.lower ci.upper std.lv std.all std.nox
1    visual =~    cubes  2.943 0.549 5.357  0.000    1.867    4.020  2.943   0.654   0.654
2    visual =~ lozenges  5.784 0.955 6.057  0.000    3.912    7.655  5.784   0.736   0.736
3   reading =~ paragrap  3.112 0.344 9.033  0.000    2.436    3.787  3.112   0.880   0.880
4   reading =~ sentence  4.151 0.502 8.270  0.000    3.167    5.135  4.151   0.827   0.827
5   reading =~ wordmean  6.952 0.820 8.476  0.000    5.345    8.560  6.952   0.841   0.841
6    visual =~  visperc  4.827 0.836 5.777  0.000    3.189    6.465  4.827   0.703   0.703
7   visperc ~~  visperc 23.873 5.945 4.016  0.000   12.221   35.525 23.873   0.506   0.506
8     cubes ~~    cubes 11.602 2.566 4.521  0.000    6.572   16.631 11.602   0.572   0.572
9  lozenges ~~ lozenges 28.275 7.837 3.608  0.000   12.914   43.636 28.275   0.458   0.458
10 paragrap ~~ paragrap  2.834 0.863 3.286  0.001    1.143    4.524  2.834   0.226   0.226
11 sentence ~~ sentence  7.967 1.856 4.292  0.000    4.329   11.605  7.967   0.316   0.316
12 wordmean ~~ wordmean 19.925 4.917 4.052  0.000   10.288   29.563 19.925   0.292   0.292
13   visual ~~   visual  1.000 0.000    NA     NA    1.000    1.000  1.000   1.000   1.000
14  reading ~~  reading  1.000 0.000    NA     NA    1.000    1.000  1.000   1.000   1.000
15   visual ~~  reading  0.487 0.117 4.145  0.000    0.257    0.717  0.487   0.487   0.487
```

indicating regression weight or variance/covariance

estimated value

standard error

p-value given that it equals zero in the population

95% ci

I have also added below some of the results from the model with the cross factor loading where both factors loaded on the *Lozenges* indicator. As would be expected the path from the *Reading* factor to is statistically insignificant indicating that its population value is equal to zero.

Unfortunately lavaan does not automatically produce indirect/total effect values but the following tutorial explains how you can do it

https://www.youtube.com/watch?v=-B37sK9NTfI

```
> parameterEstimates(result5, standardized = TRUE)
        lhs op      rhs               label    est    se      z pvalue ci.lower ci.upper std.lv std.all std.nox
1    visual =~  visperc   visual_TO_visperc  4.919 0.867  5.673  0.000    3.219    6.618  4.919   0.716   0.716
2    visual =~    cubes     visual_TO_cubes  3.036 0.564  5.384  0.000    1.931    4.140  3.036   0.674   0.674
3    visual =~ lozenges visual_TO_lozenges  5.150 1.152  4.471  0.000    2.892    7.408  5.150   0.655   0.655
4   reading =~ paragrap reading_TO_paragrap  3.103 0.345  8.999  0.000    2.427    3.779  3.103   0.877   0.877
5   reading =~ sentence reading_TO_sentence  4.148 0.502  8.260  0.000    3.164    5.133  4.148   0.826   0.826
6   reading =~ wordmean reading_TO_wordmean  6.979 0.819  8.520  0.000    5.374    8.585  6.979   0.845   0.845
7   reading =~ lozenges   reading__lozenges  0.909 1.032  0.881  0.378   -1.114    2.932  0.909   0.116   0.116
```

65.14.1.1 Modification indices (MI)

There are been several attempts to automate both model creation and model modification. Unfortunately both approaches can result in nonsense models where the optimum fitting model (usually several) does not reflect a sensible theory in the particular research area. As with factor analysis, SEM modelling is a mixture of both art and science.

Modification measures come in two main varieties:

1. Adding paths - Langrange multiplier tests (LM) provide information about how much better the model would fit (i.e. χ_2 statistic would be reduced) if a particular parameter were added to the model. That is if the parameter were allowed to be estimated rather than being constrained/fixed.

2. Removing paths - Wald tests (backward search) indicate which parameters can be dropped without affecting model fit.

The *lavaan* package takes the first approach with the *modificationIndices()* function, using the output from our model *modificationIndices(result)* the output of which is shown below.

improvement in model fit if parameter added

Expected Parameter Change (EPC). Estimated parameter value if it were added to the model

standardized EPC value

```
> modificationIndices(result)
       lhs op      rhs    mi     epc sepc.lv sepc.all sepc.nox
1   visual =~    cubes 0.000   0.000   0.000    0.000    0.000
2   visual =~ lozenges 0.000   0.000   0.000    0.000    0.000
3   visual =~ paragrap 0.001  -0.013  -0.013   -0.004   -0.004
4   visual =~ sentence 0.244   0.273   0.273    0.054    0.054
5   visual =~ wordmean 0.197  -0.398  -0.398   -0.048   -0.048
6   visual =~  visperc 0.000   0.000   0.000    0.000    0.000
7  reading =~    cubes 1.984  -0.890  -0.890   -0.198   -0.198
8  reading =~ lozenges 0.724   0.989   0.989    0.126    0.126
9  reading =~ paragrap 0.000   0.000   0.000    0.000    0.000
10 reading =~ sentence 0.000   0.000   0.000    0.000    0.000
11 reading =~ wordmean 0.000   0.000   0.000    0.000    0.000
12 reading =~  visperc 0.228   0.472   0.472    0.069    0.069
13   cubes ~~    cubes 0.000   0.000   0.000    0.000    0.000
14   cubes ~~ lozenges 0.228   2.609   2.609    0.074    0.074
15   cubes ~~ paragrap 0.000   0.016   0.016    0.001    0.001
16   cubes ~~ sentence 0.681  -1.179  -1.179   -0.052   -0.052
17   cubes ~~ wordmean 0.037  -0.446  -0.446   -0.012   -0.012
18   cubes ~~  visperc 0.724   3.804   3.804    0.123    0.123
19 lozenges ~~ lozenges 0.000   0.000   0.000    0.000    0.000
20 lozenges ~~ paragrap 0.689  -1.324  -1.324   -0.048   -0.048
21 lozenges ~~ sentence 0.026  -0.386  -0.386   -0.010   -0.010
22 lozenges ~~ wordmean 2.541   6.180   6.180    0.095    0.095
23 lozenges ~~  visperc 1.984 -13.219 -13.219   -0.245   -0.245
24 paragrap ~~ paragrap 0.000   0.000   0.000    0.000    0.000
25 paragrap ~~ sentence 0.197  -1.159  -1.159   -0.065   -0.065
26 paragrap ~~ wordmean 0.244   2.226   2.226    0.076    0.076
27 paragrap ~~  visperc 0.690   1.173   1.173    0.048    0.048
28 sentence ~~ sentence 0.000   0.000   0.000    0.000    0.000
29 sentence ~~ wordmean 0.001  -0.185  -0.185   -0.004   -0.004
30 sentence ~~  visperc 2.067   3.059   3.059    0.089    0.089
31 wordmean ~~ wordmean 0.000   0.000   0.000    0.000    0.000
32 wordmean ~~  visperc 3.916  -6.799  -6.799   -0.120   -0.120
33  visperc ~~  visperc 0.000   0.000   0.000    0.000    0.000
34   visual ~~   visual    NA      NA      NA       NA       NA
35   visual ~~  reading 0.000   0.000   0.000    0.000    0.000
36  reading ~~  reading    NA      NA      NA       NA       NA
```

While blindly looking through the indices and purely deciding on 'improving the model' by adding one or two paths with the highest *mi* values might be tempting this is not really the best strategy as the modelling process should also be informed by knowledge of the research area.

There are no standard rules concerning a specific cut off value of the *mi* to indicate necessary inclusion in the model. Therefore it is often more sensible to only display those values where the change is greater than a specific value, e.g. 10, which can be achieved with the following code.

```
change_results <-modificationIndices(result)

subset(change_results, mi > 10)
```

Looking at the results opposite, we see that the greatest improvement in model fit would be achieved by adding a covariance between *visperc* and *wordmean*. Whether this makes sense in terms of the theory of verbal and spatial intelligence would need knowledge of the research area.

65.14.1.2 Residual analysis – analysing local fit

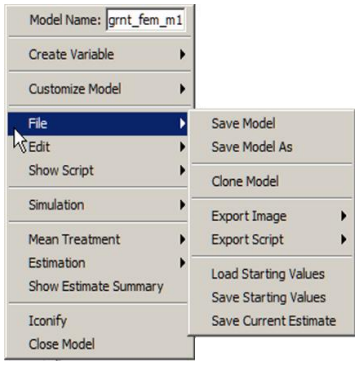
```
> resid(result, type="raw")
$cov
         cubes  lozngs pargrp sentnc wordmn visprc
cubes     0.000
lozenges  0.392  0.000
paragrap -1.105  0.291  0.000
sentence -1.914  1.536 -0.053  0.000
wordmean -3.126  4.364  0.085 -0.013  0.000
visperc   0.722 -1.388  1.020  2.886 -3.308  0.000

$mean
   cubes lozenges paragrap sentence wordmean  visperc
       0        0        0        0        0        0

> resid(result, type="normalized")
$cov
         cubes  lozngs pargrp sentnc wordmn visprc
cubes     0.000
lozenges  0.085  0.000
paragrap -0.580  0.085  0.000
sentence -0.713  0.315 -0.020  0.000
wordmean -0.706  0.539  0.020 -0.002  0.000
visperc   0.180 -0.197  0.339  0.672 -0.485  0.000

$mean
   cubes lozenges paragrap sentence wordmean  visperc
       0        0        0        0        0        0

> resid(result, type="standardized") # like z scores
$cov
         cubes  lozngs pargrp sentnc wordmn visprc
cubes     0.000
lozenges  0.423     NA
paragrap -1.055  0.179  0.000
sentence -1.184  0.572     NA  0.003
wordmean -1.203  0.986  0.300 -0.035     NA
visperc   0.719     NA  0.642  1.122 -0.905  0.003
```

As mentioned earlier it is all well and good obtaining overall (i.e. global) model fit measures but, as is the case with the simple contingency table chi-squared introduced many chapters previously, often the desire is to see where the actual model fits best and least well. As we did in that chapter, a residual analysis of the values, this time the covariance residual matrix, provides such information.

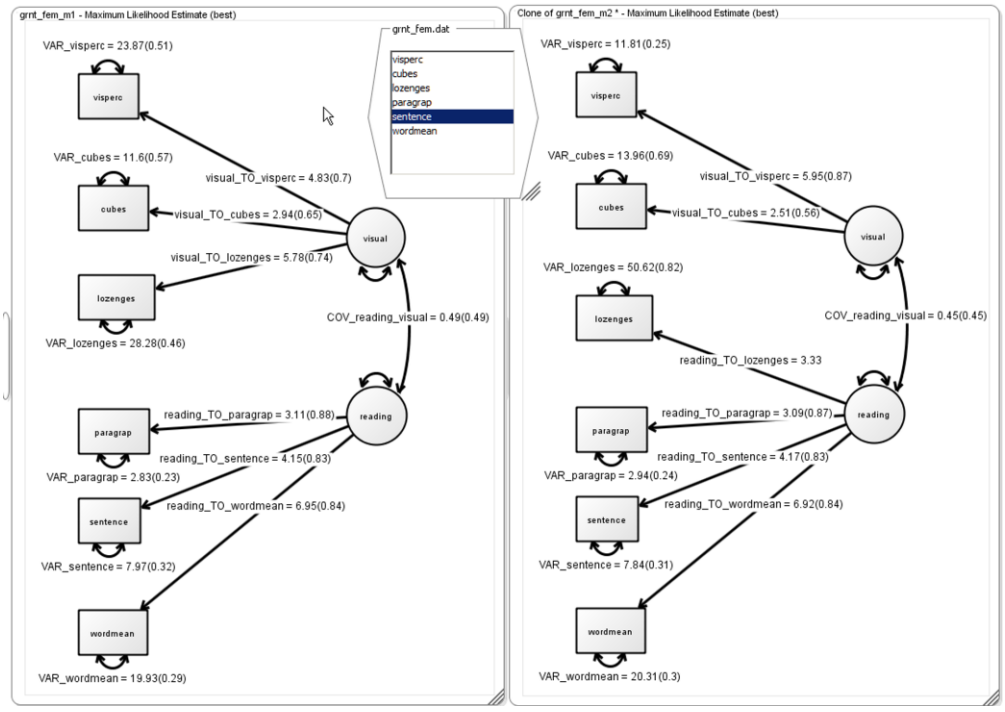
In effect, each value in the residual matrix reflects the difference between the observed and modal covariance matrix. You can obtain three sets of values, raw difference, standardised scores (which are similar to z scores), and normalised scores which are more difficult to interpret (see Bollen, 1989 p.259).

```
resid(result, type="raw")
resid(result, type="normalized")
resid(result, type="standardized")
# like z scores
```

If we look at the standardized residuals, we can see that the largest is for the covariance between cubes and sentence which misfits by around three quarters of a standard deviation. Given that none of them is greater than 1.96 reinforces the belief that this is not only a good model fit at the global level (from the RMSEA value) but also a good model fit from the individual parameter level as well.



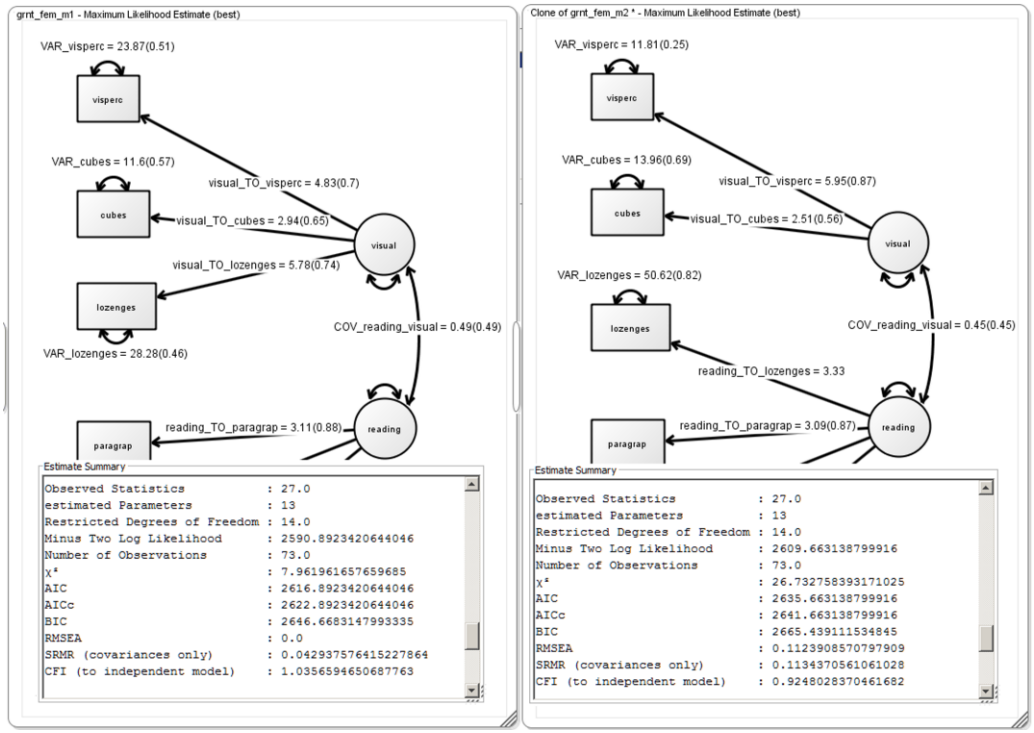
65.14.1.3 Comparing models



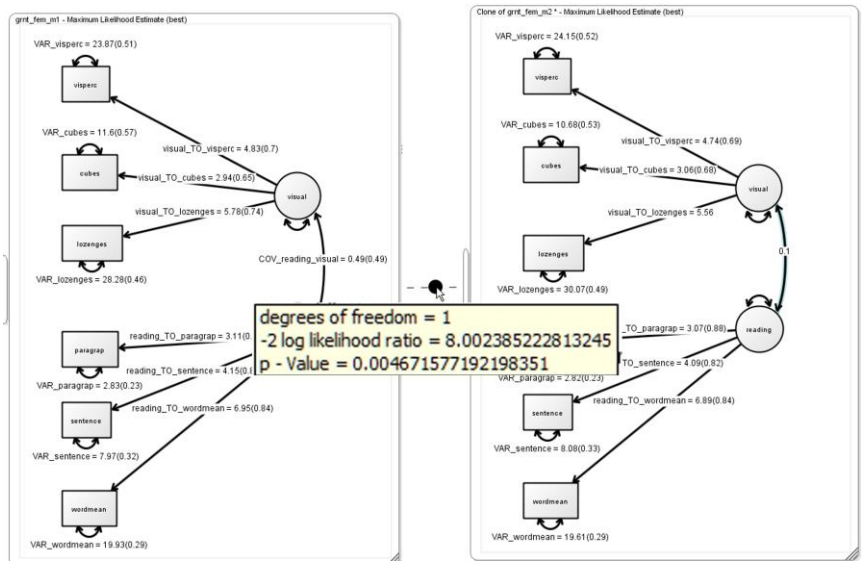
Ωnyx provides an interesting graphical approach to model comparison.

You start by defining two, or more, models. An easy way to do this is to clone one from the other, achieved by selecting the **Edit->Clone Model** popup menu option which appears when you *right mouse click* within a model panel. In the example below I have cloned the model then changed the path from *visual_to_lozenges* to *reading_to_lozenges*.

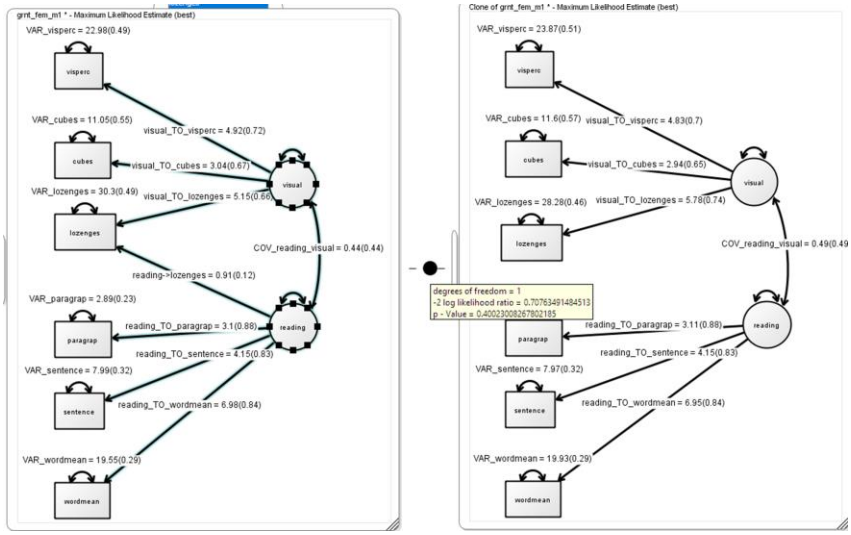
To compare these models, where they both have the same number of parameters, we simply bring up the estimate summary box for both and compare the various fit indices. As shown below, the left hand side one has a smaller RMSEA value (☺) and smaller AIC and BIC values (☺) all indicating that it is a better fit.



To compare models where in one of them one or more of the paths is constrained to a particular value (called a nested model = number of estimated parameters is different for each model) we use a likelihood ratio test, as discussed in the *Logistic Regression* chapter. To obtain this value in Ωnyx involves using a unique method, we simply right mouse click on one model and then drag to the comparator model. In the example below, I have constrained the covariance between the two latent variables to 0.1 and then compared the model to that where the covariance is estimated. Hovering over the black dot informs us that we have a -2log likelihood value of 8.002 with an associated p-value of 0.004; we can conclude from this that the two models are indeed different with the additional free parameter providing a better fit.



As I mentioned before these global fit measures are not always to be trusted for example the following is a comparison between our usual model and the one where both factors load on lozenges; using this approach we would think both models were equally well fitted, which we know from looking at the individual parameter estimates is not the case



Ωnyx has many more features which I do not have room to discuss in this chapter, for further information visit the Ωnyx website.

65.14.2 OpenMx (formally Mx)

OpenMx is another SEM package. However this originally was a standalone application called Mx which has been redeveloped as an R library. The *OpenMx* code generated by Ωnyx for our CFA is shown below. I do not intend to discuss this, just noting that it is very different syntactically. The OpenMx website has extensive tutorials, and because people tend to learn either *lavaan* or *OpenMx* I will leave it at that.

```
# This model specification was automatically generated by Ωnyx
require("OpenMx");
modelData <- read.table(DATAFILENAME, header = TRUE)
manifests<-c("visperc","cubes","lozenges","paragrap","sentence","wordmean")
latents<-c("visual","reading")
model <- mxModel("grnt_fem_m1",
type="RAM",
manifestVars = manifests,
latentVars = latents,
mxPath(from="visual",to=c("visperc","cubes","lozenges"), free=c(TRUE,TRUE,TRUE), value=c(1.0,1.0,1.0) , arrows=1,
label=c("visual_TO_visperc","visual_TO_cubes","visual_TO_lozenges") ),
mxPath(from="reading",to=c("paragrap","sentence","wordmean"), free=c(TRUE,TRUE,TRUE), value=c(1.0,1.0,1.0) , arrows=1,
label=c("reading_TO_paragrap","reading_TO_sentence","reading_TO_wordmean") ),
mxPath(from="visperc",to=c("visperc"), free=c(TRUE), value=c(1.0) , arrows=2, label=c("VAR_visperc") ),
mxPath(from="cubes",to=c("cubes"), free=c(TRUE), value=c(1.0) , arrows=2, label=c("VAR_cubes") ),
mxPath(from="lozenges",to=c("lozenges"), free=c(TRUE), value=c(1.0) , arrows=2, label=c("VAR_lozenges") ),
mxPath(from="paragrap",to=c("paragrap"), free=c(TRUE), value=c(1.0) , arrows=2, label=c("VAR_paragrap") ),
mxPath(from="sentence",to=c("sentence"), free=c(TRUE), value=c(1.0) , arrows=2, label=c("VAR_sentence") ),
mxPath(from="wordmean",to=c("wordmean"), free=c(TRUE), value=c(1.0) , arrows=2, label=c("VAR_wordmean") ),
mxPath(from="visual",to=c("visual"), free=c(FALSE), value=c(1.0) , arrows=2, label=c("VAR_visual") ),
mxPath(from="reading",to=c("reading","visual"), free=c(FALSE,TRUE), value=c(1.0,1.0) , arrows=2, label=c("VAR_reading","COV_reading_visual") ),
mxPath(from="one",to=c("visperc","cubes","lozenges","paragrap","sentence","wordmean"), free=T, value=1 , arrows=1),
mxData(modelData, type = "raw")
);

result <- mxRun(model)
summary(result)
```

65.15 Structural Equation Modelling directly in R using the *sem* package

From the above lavaan example, I'm sure you realise that you could have also directly written the *lavaan* code in R and bypassed Ωnyx but at the same time I'm sure you can see the advantage of using Ωnyx. An R package originally developed long before *lavaan*, in 2001, called *sem* also allows the creation of SEM models in R directly. The *sem* package just uses two types of relation: **<->** to indicate variances/covariances and **->** to indicate regression paths. You then need to give each a name or specify it as being a fixed value using *NA* and also indicate the estimation starting/fixed value or allow the program to estimate it freely (*NA*). An example should help make this clearer. Because SEM model specification code is similar to the *lavaan* approach I have put them side by side below.

```
# lavaan code
onyx.model<-"
! regressions
  visual=~visperc
  visual=~cubes
  visual=~lozenges
  reading=~paragrap
  reading=~sentence
  reading=~wordmean
! residuals, variances and
covariances
  visperc ~~ visperc
  cubes ~~ cubes
  lozenges ~~ lozenges
  paragrap ~~ paragrap
  sentence ~~ sentence
  wordmean ~~ wordmean
  visual ~~ 1*visual
  reading ~~ 1*reading
  visual ~~ reading
"
onyx.model
result<-lavaan(onyx.model,
data=hozdata)
summary (result,
fit.measures= TRUE)
```

```
install.packages("sem") ; library(sem)  # to install and load the sem package
hozdatamodel <- specifyModel()
visual -> visperc,         visual_TO_visperc,     NA #first factor
visual -> cubes,           visual_TO_cubes,       NA
visual -> lozenges,        visual_TO_lozenges,    NA
reading ->  paragrap,      reading_TO_paragrap,   NA # second factor
reading -> sentence,       reading_TO_sentence,   NA
reading -> wordmean,       reading_TO_wordmean,   NA
visperc <-> visperc,       err_visperc,           NA # errors
cubes <-> cubes,           err_cubes,             NA
lozenges <-> lozenges,     err_lozenges,          NA
paragrap <-> paragrap,     err_paragrap,          NA
sentence <-> sentence,     err_sentence,          NA
wordmean <-> wordmean,     err_wordmean,          NA
visual <-> reading,        COV_visual_reading,    NA # covariances
visual <-> visual,         NA,                     1  # fixed parameters
reading <-> reading,       NA,                     1

hozmodel1 <- SEM(hozdatamodel, data = hozdata)
summary(hozmodel1)
standardizedCoefficients(hozmodel1)
```

```
> summary(hozmodel1)

 Model Chisquare =  7.852894   Df =  8 Pr(>Chisq) = 0.4479704
 AIC =  33.85289
 BIC =  -26.47078

 Normalized Residuals
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-0.69510 -0.19380  0.00000 -0.02143  0.18000  0.68350

 R-square for Endogenous Variables
 visperc    cubes lozenges paragrap sentence wordmean
  0.4940   0.4275   0.5419   0.7736   0.6838   0.7081

 Parameter Estimates
                  Estimate   Std Error z value  Pr(>|z|)
visual_TO_visperc    4.8606192 0.8472456 5.736966 9.638767e-09 visperc <--- visual
visual_TO_cubes      2.9637940 0.5570414 5.320599 1.034261e-07 cubes <--- visual
visual_TO_lozenges   5.8237161 0.9681471 6.015321 1.795300e-09 lozenges <--- visual
reading_TO_paragrap  3.1331703 0.3492597 8.970890 2.941287e-19 paragrap <--- reading
reading_TO_sentence  4.1796433 0.5088761 8.213479 2.148706e-16 sentence <--- reading
reading_TO_wordmean  7.0004445 0.8316418 8.417620 3.842094e-17 wordmean <--- reading
err_visperc         24.2042982 6.0692766 3.988004 6.663159e-05 visperc <--> visperc
err_cubes           11.7627275 2.6196207 4.490241 7.114270e-06 cubes <--> cubes
err_lozenges        28.6680567 8.0011697 3.582983 3.396925e-04 lozenges <--> lozenges
err_paragrap         2.8731229 0.8805372 3.262920 1.102705e-03 paragrap <--> paragrap
err_sentence         8.0773857 1.8949870 4.262502 2.021508e-05 sentence <--> sentence
err_wordmean        20.2019195 5.0200027 4.024285 5.714877e-05 wordmean <--> wordmean
COV_visual_reading   0.4870169 0.1183007 4.116771 3.842174e-05 reading <--> visual

 Iterations =  83
```

There are many similarities between the two and I would recommend anyone who is thinking about using the *sem* package to begin by adapting the *lavaan* output from the Ωnyx application. The *SEM* package is very unhelpful when it comes to reporting errors, providing what I find to be, very cryptic messages. You can waste many hours only to discover the problem was a missing comma.

The results are provided opposite to demonstrate their equivalence to the other packages/ programs used.

There are also similar functions in the *SEM* package to produce the various residues and matrices:

```
> standardizedCoefficients(hozmodel1)
                    Std. Estimate
1    visual_TO_visperc     0.7028162    visperc <--- visual
2      visual_TO_cubes     0.6538466      cubes <--- visual
3   visual_TO_lozenges     0.7361554   lozenges <--- visual
4  reading_TO_paragrap     0.8795393  paragrap <--- reading
5  reading_TO_sentence     0.8269342  sentence <--- reading
6  reading_TO_wordmean     0.8414862  wordmean <--- reading
7          err_visperc     0.5060493   visperc <--> visperc
8            err_cubes     0.5724846     cubes <--> cubes
9         err_lozenges     0.4580753 lozenges <--> lozenges
10        err_paragrap     0.2264106 paragrap <--> paragrap
11        err_sentence     0.3161799 sentence <--> sentence
12        err_wordmean     0.2919009 wordmean <--> wordmean
13  COV_visual_reading     0.4870169    reading <--> visual
14                         1.0000000     visual <--> visual
15                         1.0000000   reading <--> reading
```

```
effects(hozmodel1)
# gives total/direct/indirect effects

hozmodel1$C # model-reproduced covariance matrix

hozmodel1$A # RAM A matrix

hozmodel1$P

residuals(hozmodel1)

standardizedResiduals(hozmodel1)

vcov(hozmodel1)
```

RAM P matrix
covariances amongst elements
same as S in Ωnyx

covariances amongst all parameters

I have not shown all the results for the above options as they would produce results very similar to those shown previously.

Another useful command in the *sem()* function, is *objective =objectiveGLS* which instructs the program to produce generalized least squares estimates instead of the default maximum likelihood ones. This can be useful if there is a problem with the initial estimation process. Secondly you can get the *sem()* function to provide details for each iteration of the estimation process by adding the option *debug=TRUE*. The use of these options is shown below.

```
hozmodel2 <- sem(hozdatamodel, data = hozdata, objective=objectiveGLS, debug=TRUE)

hozmodel2
```

```
> modIndices(hozmodel1)

 5 largest modification indices, A matrix:
 wordmean<-visperc lozenges<-wordmean    cubes<-sentence
          2.485672           2.353646          2.141871
     visual<-cubes     reading<-cubes
          1.956953           1.956948

  5 largest modification indices, P matrix:
 wordmean<->visperc wordmean<->lozenges  sentence<->visperc
          3.861991           2.506210          2.038260
     visual<->cubes     reading<->cubes
          1.956955           1.956949
```

same as the mi measures in the lavaan package

The *modIndices()* function provides modification indices. By setting the *n.largest=n* you will obtain the nth largest values, the default is 5, as shown opposite.

The *sem* package has what are known as wrapper functions for the *sem()* function, this basically means that these other functions set a number of defaults so that you can specify the model by providing less information. One such wrapper function is *cfa()* standing for Confirmatory Factor Analysis. I have used this function below to create an equivalent cfa model to that produced above, called *cfa_hoz*, but with much less code. When you run this code the *sem* package informs us that additional error variances will automatically be added along with a covariance between the factors.

```
cfa_hoz <- cfa(reference.indicators=FALSE)

F1: visperc, cubes, lozenges

F2: paragrap, sentence, wordmean

hozmodel3 <- SEM(cfa_hoz, data = hozdata)

summary(hozmodel3)
```

I personally prefer to use the *sem()* function as you then know exactly what you are getting.

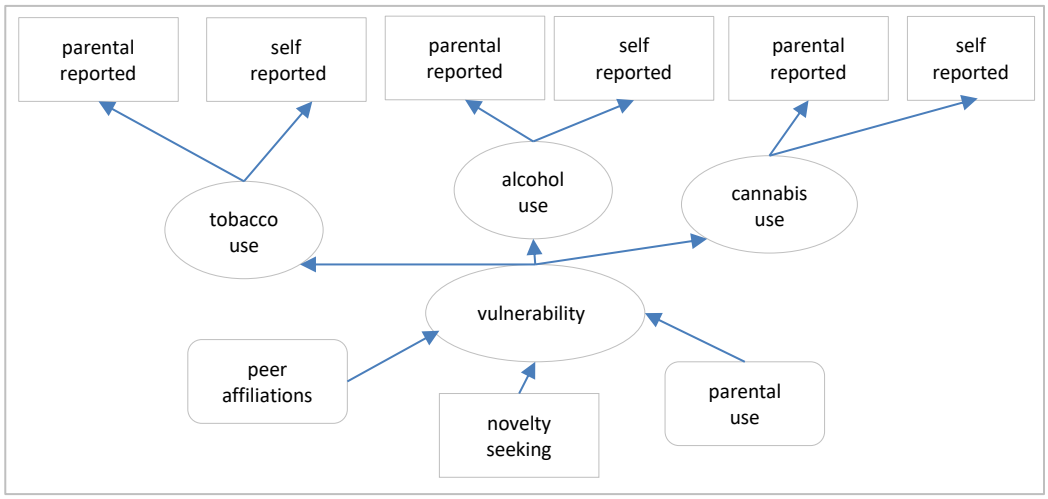
We have now considered both creating SEM models directly in Ωnyx, specifying them initially in Ωnyx, then repeating the analysis in R using the automatic code generated and finally specifying the model directly in R. I deliberately kept the example as simple as possible but now it is time to see a more typical example to demonstrate some of the representative characteristics of an SEM analysis.

The latest version of Ωnyx (v1.00) provides an export to the *SEM* package, producing the following code on the left, as with the previous export code you need to edit it by adding the dataframe you wish to use. Also the first call to the *sem()* function seems to create an error for this particular model.

Ωnyx export	**Edited**
require("SEM"); paths <- c("Education <-> Education", "Popden <-> Popden", "Nonwhite <-> Nonwhite", "Mortality <-> Mortality", "Education -> Mortality", "Nonwhite -> Mortality", "Nonwhite <-> Popden", "Education <-> Popden", "Education <-> Nonwhite", "Popden -> Mortality") parameter <- c("VAR_Education", "VAR_Popden", "VAR_Nonwhite", "VAR_Mortality", "Education_TO_Mortality", "Nonwhite_TO_Mortality", "COV_Nonwhite_Popden", "COV_Education_Popden", "COV_Education_Nonwhite", "Popden_TO_Mortality") values <- c("0.7026222048487266", "2108953.5033314778", "78.26043332764722", "1541.636507142136", "-25.50698060512576", "3.9999225532362965", "-163.73509363742735", "-286.0402052689569", "-1.5481333432299311", "0.007954486809572933") model <- array(c(paths, parameter, values), dim = c(10,3)) colnames(model) <- c("col1","col2","col3") dat <- read.table(DATAFILENAME, header = TRUE) result <- SEM(model = model, data = dat, fixed.x ="Intercept", raw = TRUE) result <- SEM(model = model, data = dat)	require("SEM"); paths <- c("Education <-> Education", "Popden <-> Popden", "Nonwhite <-> Nonwhite", "Mortality <-> Mortality", "Education -> Mortality", "Nonwhite -> Mortality", "Nonwhite <-> Popden", "Education <-> Popden", "Education <-> Nonwhite", "Popden -> Mortality") parameter <- c("VAR_Education", "VAR_Popden", "VAR_Nonwhite", "VAR_Mortality", "Education_TO_Mortality", "Nonwhite_TO_Mortality", "COV_Nonwhite_Popden", "COV_Education_Popden", "COV_Education_Nonwhite", "Popden_TO_Mortality") values <- c("0.7026222048487266", "2108953.5033314778", "78.26043332764722", "1541.636507142136", "-25.50698060512576", "3.9999225532362965", "-163.73509363742735", "-286.0402052689569", "-1.5481333432299311", "0.007954486809572933") model <- array(c(paths, parameter, values), dim = c(10,3)) colnames(model) <- c("col1","col2","col3") dat <- hozdata result <- SEM(model = model, data = dat) summary(result) standardizedCoefficients(result)

65.16 A complex example

Lynsky, Fergusson & Horwood, 1998, analysed the relationships between reports of tobacco, alcohol, and cannabis use in a birth cohort of New Zealand children studied until the age of 16 (n=1265 dropping to 913 for complete cases). Comparable information from their parents was also obtained. The researchers were interested in assessing the validity of a particular model known as Jessor and Jessor's Problem Behaviour Theory, which posits that proclivity to problem behaviours are due to three aspects; demographic social structure (peer afflictions, novelty seeking and parental illicit drug use); the perceived social environment and the personality system. This is in contrast to the stage or gateway theory where there is a progression from one substance to another. (Lynsky, Fergusson & Horwood, 1998 p.1004). They had therefore a general idea of something like that shown above.



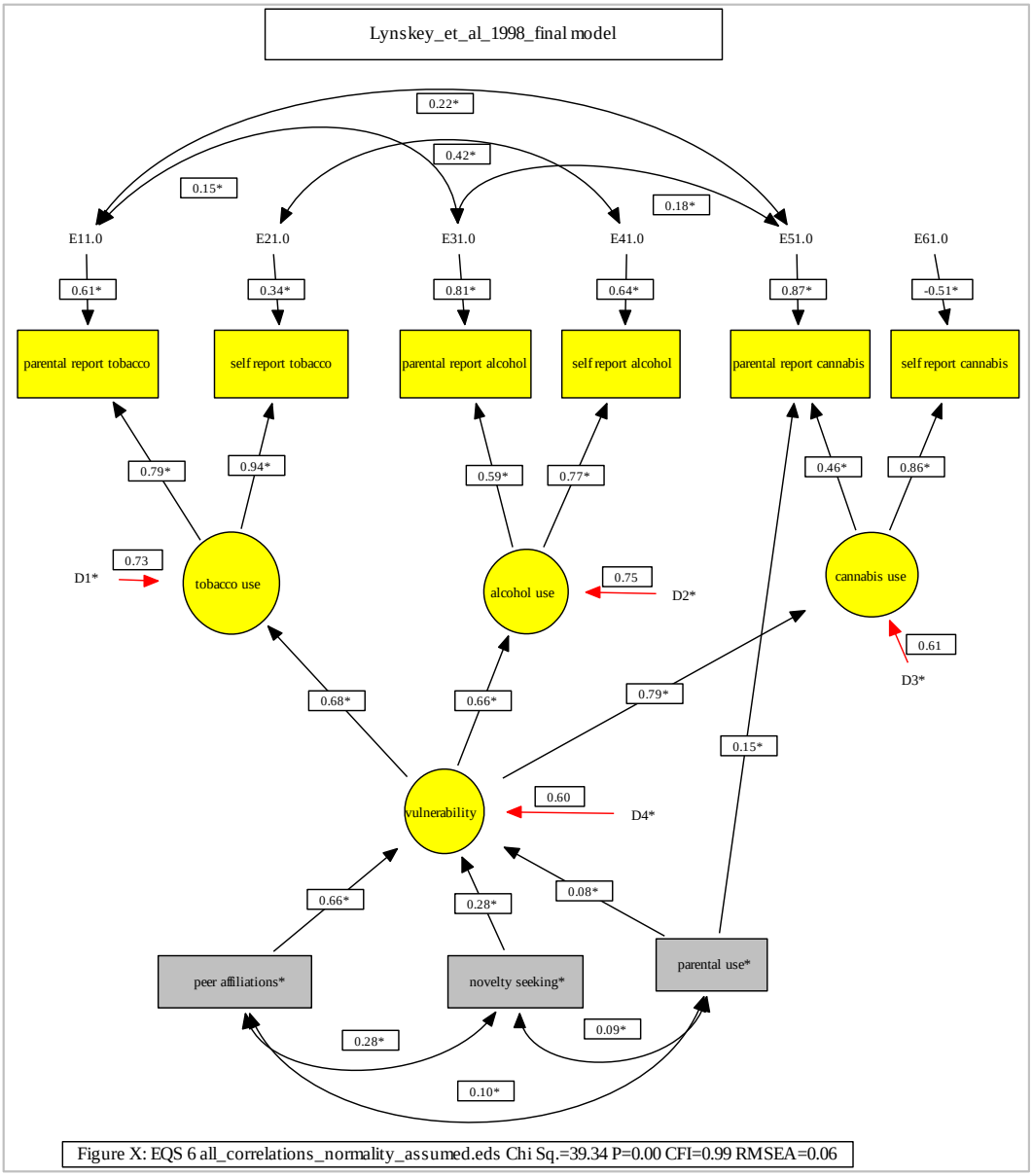


Figure X: EQS 6 all_correlations_normality_assumed.eds Chi Sq.=39.34 P=0.00 CFI=0.99 RMSEA=0.06

The final model produced from the study is shown opposite. It must be remembered that this is the final model of a long development process and I strongly advise you to read the original article. The parameter values are slightly different but comparable to the original as I have produced it using a subset of the original data (n=868) in the commercial SEM program EQS.

In EQS the error values are the square roots of those produced in Ωnyx.
Therefore: 1 -(SEM error value)^2 = R^2.

There is a lot to take in concerning this complex model. Possible key points are:

1. Parental reports of tobacco, alcohol and cannabis use are correlated in contrast to self-reported cannabis use, which does not correlate with parental or other self reported measures.
2. Self-reported measures of tobacco, alcohol and cannabis use have higher standardised factor loadings (.77 to .94) compared to parental values (.46 to .79). Remember the higher the loading the greater the degree to which the measure is reflected in the factor.

3. Relationship between tobacco, alcohol and cannabis use (latent variable) and vulnerability - the standardised loading factor values here (.66 to .79) indicates a strong relationship between the three specific latent variables and the general vulnerability latent variable.

```
 STANDARDIZED SOLUTION:                                                R-SQUARED

PRSMK16 =V1  =    .792*F1    +  .610*E1                                    .628
SRSMK16 =V2  =    .941*F1    +  .338*E2                                    .886
PRALC16 =V3  =    .590*F2    +  .808*E3                                    .348
SRALC16 =V4  =    .771*F2    +  .637*E4                                    .594
PRCANN16=V5  =    .457*F3    +  .148*V12   +  .866*E5                      .250
SRCANN16=V6  =    .862*F3    -  .506*E6                                    .744
   F1    =F1  =    .683*F4    +  .730 D1                                    .467
   F2    =F2  =    .665*F4    +  .747 D2                                    .442
   F3    =F3  =    .794*F4    +  .608 D3                                    .631
   F4    =F4  =    .659*V7    +  .281*V8    +  .084*V12   +  .600 D4        .639
```

4. The error value associated with the vulnerability (.6) indicates how well the three predictors (peer affiliations, novelty seeking and parental drug) predict vulnerability. As R^2 = 1-.6^2 = .64 indicting that 64% of the variability in vulnerability can be accounted for by the predictors. For the full data set (Lynsky, Fergusson & Horwood, 1998) this value was reduced to 54%.

5. The asterisks (*) beside the parameter estimate indicates that the value is statistically significant.

6. The overall model fit measures indicate a well-fitting model with CFI =0.99 (good fit CFI≥0.95) and RMSEA=0.06 (RMSEA≤.05 very good fit).

65.17 Extending SEM

SEM techniques have been extended in many ways:

- Non-normal distributions
- Bayesian approaches
- Ordinal and binary data, mimicking factorial ANOVA and Logistic regression.
- Time series (also called latent growth models)
- Comparisons of multiple groups (means)
- moderation & mediation effects (see https://www.youtube.com/watch?v=mirI5ETQRTA)
- Meta-analysis (Cheung 2015)

A good introduction to the above techniques, except moderation and mediation, can be found in Schumacker & Lomax, 2010.

65.18 Reviewing SEM research

The following page provides a checklist, of aspects to consider when reviewing a SEM article. While all the points are valid, one or two of them, such as cross validation of the modal using a subset of the data is a rare occurrence.

Criteria for reviewing a article using SEM (adapted from Schumacker & Lomax 2006, p.251-5)

1. Data Preparation

1. Have you adequately described the population from which the random sample data was drawn?
2. Did you report the measurement level of your variables?
3. Did you report the descriptive statistics on your variables? [+ normality]
4. Did you test for multivariate normality?
5. Did you create a table with correlations, means, and standard deviations?
6. Did you have missing data if so why and what strategy did you use to overcome the problem.
7. Did you have outliers and did you resolve this by using robust statistics or deletion methods?
8. Did you resolve non-normality of a variable by some form of transformation?
9. Did you have problems with multi-collinearity among variables? If so how did you resolve it?
10. Did you specify the input data used (raw, correlations, covariances etc.)
11. Did you include the set of commands as an appendix so that others could carry out a similar analysis?

2. Model Specification

1. Did you provide a rationale for your study?
2. Did you explain why SEM rather than another approach was required?
3. Did you describe your latent variables
4. Did you provide a theoretical foundation for your model?
5. Did you theoretically justify alternative models?
6. Did you justify your sample size?
7. Did you clearly state your statistical hypotheses?
8. Did you discuss the expected magnitude and direction of expected parameter estimates?
9. Did you include a figure or diagram of your proposed model(s)?

3. Model Identification Issues

1. Did you calculate the number of distinct values in your sample matrix?
2. Did you calculate the number of parameters needing to be estimated (free parameters) [and constraints]
3. [Did you software provide you with appropriate messages indicating there was no identification problem.]

4. Model Estimation

1. How will you consider power and sample size?
2. What is the ratio of sample size to number of parameters?
3. What estimation technique is most suitable given your sample size and normality situation?
4. Did you encounter Heywood cases (negative variance) or other impossible values in the output?
5. Did the software use raw data or a matrix as input?
6. How did you scale the latent variable (reference variable or fix the variance of the latent variable).
7. Which SEM program and version did you use?
8. Did you encounter any convergence problems?
9. Did you report the R^2 values to indicate the total effect the independent variables had on each dependent variable.

5. Model Testing

1. Did you include a website providing more information?
2. Did you provide tables (American Psychological Association style, Schumacker p.241) providing details for each factor (i.e. Reliability rho) and for each indicator Loading and R^2
3. Did you use single items or composite scale scores?

Several other issues in the original list not considered here.

6. Model Modification

1. Did you compare alternative or equivalent models?
2. Did you clearly indicate how you modified the initial model?
3. Did you provide a theoretical justification for the respecified model?
4. Did you add or delete one path/ parameter at a time? [recommended]
5. Which statistics helped guide you through the process (Wald, Lagrange [good] or simple *t* tests [bad])
6. Did you provide parameter estimates and model fit indices for both the initial model and the respecified one(s)?
7. Did you report expected change statistics.
8. Your model is not the only model that fits the sample data, so did you check for equivalent models or theoretically justify your final model?
9. How did you evaluate and select the best model?

7. Model Validation

1. Did you replicate your SEM model analysis using another sample of data?
2. Did you cross-validate your SEM model by splitting your original sample of data?
3. Did you use bootstrapping [simulate the samples] to determine bias in your parameter estimates?
4. Did you report effect sizes and confidence intervals in addition to statistical significance testing?
5. Did you evaluate your results with regard to your original theoretical framework?

65.19 Tricks and Tips

This chapter has provided only an introduction but hopefully has helped you understand the basics of SEM and provided enough information to enable you to interpret findings from such an analysis. People who undertake SEM tend to learn a particular SEM program or R package and then stick to it. I hope by introducing a range of approaches in this chapter will have helped you decide which is the best one for you.

Understandably, much of the information available is aimed at using a certain package, particularly the many useful youtube videos. Notable exceptions are Loehin, 2004 and Kline, 2016. The companion website to Kline's book provides datasets and syntax files for several chapters and an excellent set of web links to other SEM sites at:

http://www.guilford.com/cgi-bin/cartscript.cgi?page=etc/kline.html

There are many websites devoted to SEM techniques. Professor Jason Newsom, at Portland University, runs an excellent course on SEM with online resources and links to several SEM sites at: www.upa.pdx.edu/IOA/newsom . Another is David A. Kenny (Emeritus Professor, University of Connecticut, Department of Psychology) who keeps a useful, succinct webpage listing and explaining most of the model fit indices http://davidakenny.net/cm/fit.htm

There are the webpages associated with each of the R packages used in this chapter which provide further details and examples. The Ωnyx website provides some information which hopefully will be developed in future.

Finally I have augmented the material presented in this chapter with several online chapters and YouTube videos, all available from the books website (http://robin-beaumont.co.uk/rbook/sem/index.html) two of which are:

- SEM equivalents to basic statistical procedures
- Regression models in SEM