Seeing the Impossible: Visualizing Latent Variable Models

Brief Introduction

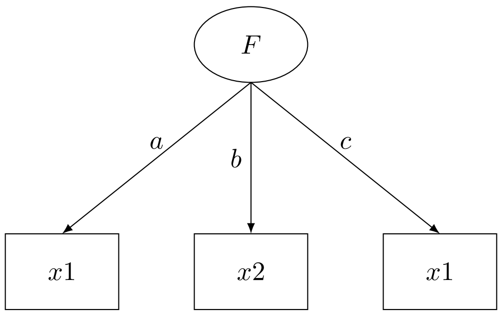
SEM is a powerful tool that is infinitely flexible and able to handle diverse sorts of modeling procedures. However, it has two fatal flaws. First, latent variables are, by definition, unobserved. As such, data visualization, a critical component of model building is not an easy task. Although some have advocated for the use of scatterplots of latent variables (Hallgren et al. 2019), these plots assume the model has accurately estimated the parameters, which may or may not be the case. Second, standard latent variable models are estimated from summary statistics, such as means and covariances. These summary statistics may be misleading if model assumptions have been violated (e.g., if the data are not normally distributed or if relationships are nonlinear). Although there are tools that allow users to *model* nonnormal data and nonlinear patterns, few *diagnostic* tools exist that allow users to assess the degree to which assumptions have been met (and/or whether violations of these assumptions are problematic).

To overcome both of these limitation, we develop visualization tools designed for latent variable models. These tools will not only provide diagnostic checks, but they will allow both researchers and lay audiences to intuitively understand the fit of latent variable models.

Diagnostic Plots

Trace Plots: Estimating the Model-Implied Slope

Suppose we have the factor analysis model shown below.



Assume both indicators and the latent variable are standard normal variables. In this situation, we define the observed variables as follows:

$$
\begin{aligned}
x_1 &= aF + e_1 \\
x_2 &= bF + e_2 \\
x_3 &= cF + e_3
\end{aligned}
$$

These can be re-expressed in terms of F as follows:

$$F = \frac{x_1 - e_1}{a}$$
$$F = \frac{x_2 - e_2}{b}$$
$$F = \frac{x_3 - e_3}{c}$$

Suppose we wish to plot the fitted line between x_1 and x_2 implied by the latent variable model. To do so, we can re-express x_2 in terms of x_1:

$$\begin{aligned} x_2 &= bF + e_2 \\ &= b(\frac{x_1 - e_1}{a}) + e_2 \end{aligned} \tag{1}$$

Computing the expectation, we get

$$\begin{aligned} E[x_2|x_1] &= E\left[b(\frac{x_1 - e_1}{a}) + e_2\right] \\ &= \frac{b}{a}E[(x_1 - e_1)] + E[e_2] \\ &= \frac{b}{a}E[x_1] - E[e_1] + E[e_2] \\ &= \frac{b}{a}E[x_1] \end{aligned} \tag{2}$$

Also, since these variables are standardized, Equation 2 is both the slope of the line predicting x_2 from x_1, as well as the correlation coefficient, ρ.

Recall that latent variable models explicitly model the unreliability in indicators. In other words, Equation 2 is the "corrected" estimate of the correlation between x_1 and x_2. As such, the regression line generated from Equation 2 will visually overestimate the strength of the relationship, as demonstrated in Figure 1.For example, iff b and a are both 0.5, Equation 2 suggests the slope is one (indicating a perfect correlation between these standardized variances). This is clearly not the case. As such, the slope needs to be "uncorrected" for unreliability.

```r
require(lavaan)
require(tidyverse)
set.seed(1212)
n = 500
slopes = c(.75, .75, .75)
latent = rnorm(n)

# create three indicators
x1 = slopes[1]*latent + rnorm(n,0, sqrt(1-slopes[1]^2))
x2 = slopes[2]*latent + rnorm(n,0, sqrt(1-slopes[2]^2))
x3 = slopes[3]*latent + rnorm(n,0, sqrt(1-slopes[3]^2))
d = data.frame(x1=x1, x2=x2, x3=x3)

# model in lavaan
model.linear = '
  A =~ x1 + x2 + x3
```

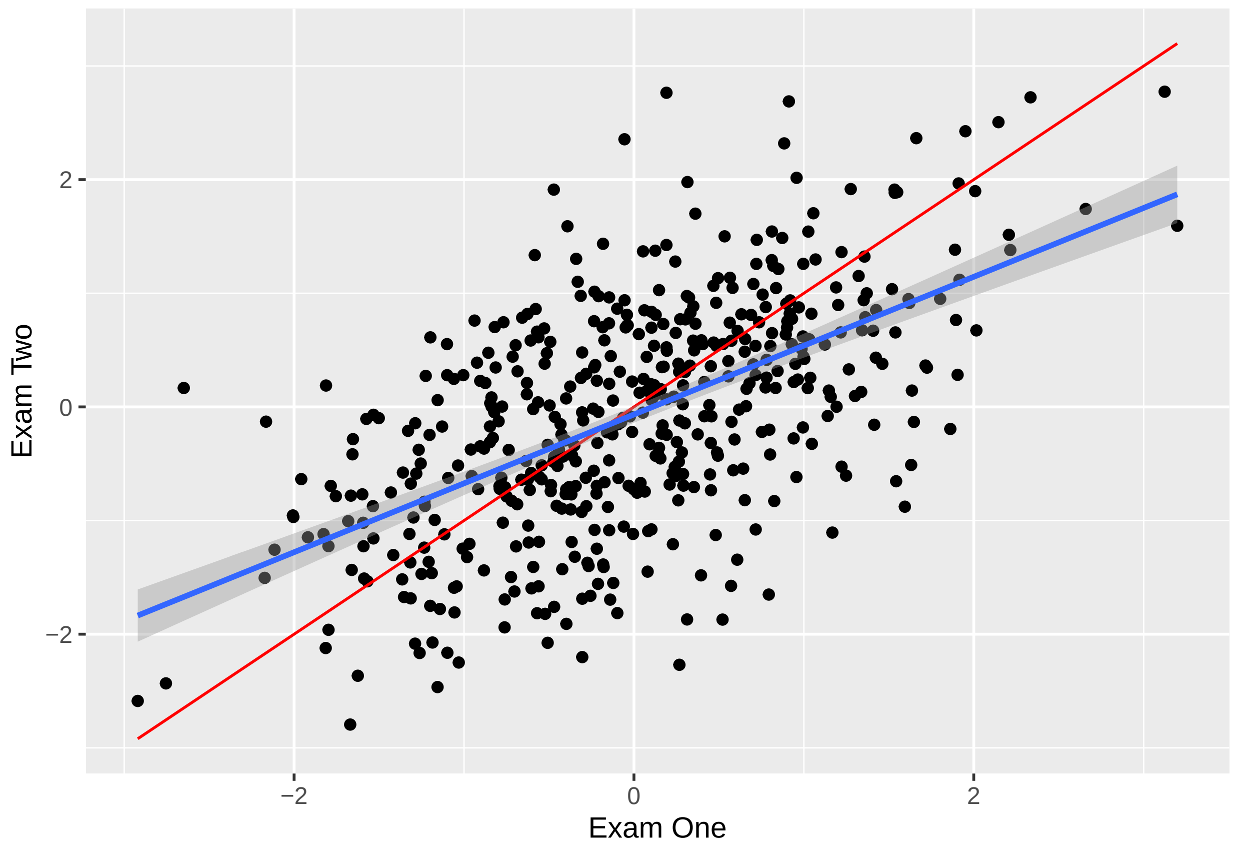


Figure 1: Scatterplot of the x1/x2 relationship. The red line shows the implied fit between the two variables under the assumption of perfect reliability. The blue line shows the actual fitted relationship.

```
  A ~~ A
'
fitted = cfa(model.linear, data=d)

  # create trace plot based on the assumption of perfect reliability
m = data.frame(x1 = seq(from=min(d$x1), to=max(d$x1), length.out=20))
m$latent = (slopes[2]/slopes[1])*m$x1  ### from Equation 1
ggplot(data=d, aes(x=x1,y=x2)) +
  geom_point() +
  geom_line(data=m, aes(x1, latent), col="red") +
  geom_smooth(method="lm") +
  labs(x="Exam One", y="Exam Two")
```

Recall the standard correction for unreliability:

$$\rho = \frac{r}{\sqrt{\rho_{xx}\rho_{yy}}} \tag{3}$$

To "uncorrect" this estimate, we simply solve for r

$$r = \rho\sqrt{\rho_{xx}\rho_{yy}} \tag{4}$$

Recall the relationship between r and the slope (β)

$$\beta = r\frac{\sigma_x}{\sigma_y} \tag{5}$$

Or, under perfect reliability,

$$\beta = \frac{\sigma_x}{\sigma_y} \tag{6}$$

We can then multiply Equation 9 by Equation 2:

$$\begin{aligned} E[x_2|x_1] &= \sqrt{\rho_{xx}\rho_{yy}}\frac{b}{a}E[x_1] \\ &= \sqrt{a^2b^2}\left(\frac{b}{a}\right)E[x_1] \\ &= a*b\left(\frac{b}{a}\right)E[x_1] \\ &= b^2E[x_1] \end{aligned} \tag{7}$$

Using Equation 7, we now get a line that closely approximates the actual relationship between the two variables (Figure **??**.

```r
m$latent = (slopes[2]^2)*m$x1  ### from Equation 1
# ggplot(data=d, aes(x=x1,y=x2)) +
#   geom_point() +
#   geom_line(data=m, aes(x1, latent), col="red") +
#   geom_smooth(method="lm") +
#   labs(x="Exam One", y="Exam Two") +
#   theme_bw()
```

We could then apply this command continuously to all observed variables in a scatterplot matrix (using, say, the upper diagonal), as in Figure **??**. As before, the red line indicates the model-implied fit. This time, however, the blue line is the fit of a loess line (which will allow for the detection of nonlinear patterns).

Disturbance Dependence Plots: Removing the Effect of the Latent Variable

It may also be interesting to utilize "disturbance dependence plots," or to plot the relationship between the variables on interest, once the effect of the latent variable has been removed. This is similar to a residual dependence plot in regression models, but we use different terminology to avoid confusion (since residuals in SEM often refer to the residual correlation matrix).

To generate these disturbance terms, we need only to generate predicted x_2 scores from Equation 7, then subtract those from the observed x_2 scores:

$$E[x_2|F, x_1] = x_2 - b^2E[x_1] \tag{8}$$

We can then plot the relationship between x_1 and x_2 after removing the effect of the latent variable. If the data are locally independent, we should observe no relationship. The expected relationship (i.e., a line centered on zero with no slope) is colored as red. We could then add this plot to the scatterplot matrix in, say, the lower diagonals, as in Figure **??**.

Detecting Deviations

These diagnostic plots are designed to detect deviations from the model's assumptions. In the example below, I'm going to simulate a situation where a) x_1 and x_2 are locally dependent, and b) x_3 has a nonlinear relationship with the latent variable. Notice from the plot below that the upper diagonals indicate our model's fit (red line) is failing to capture important nonlinear relationships with x_3 and it overestimates the relationship between x_1 and x_2.

Model Plots

Once one has iterated through the diagnostic plots, they can be more confident the factor scores are accurately estimated. At that point, the user may choose to plot the relationship between each indicator and the latent variable(s), or they may choose to plot the relationship between multiple latent variables. However, representing these as a scatterplot can be misleading because these are estimated scores, not observed scores. As such, I recommend adding some visual representation of uncertainty (e.g., each point is an ellipse whose size is an indicator of our uncertainty). This could, of course, become very busy very quickly. However, one could sample datapoints to represent latent scores or reduce opacity. These options are available and easy to use in my R package, `flexplot`.

References

Stats Jedi Dataset

Harry Potter Dataset

Suppose we have the following nonlinear relationship between observed variable X and latent $F

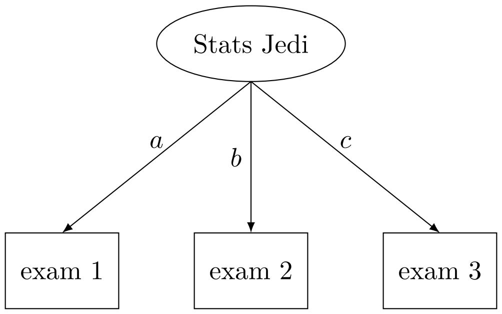
$$X = \frac{M}{1 + exp^{-x_1(F-x_0)}} \tag{9}$$

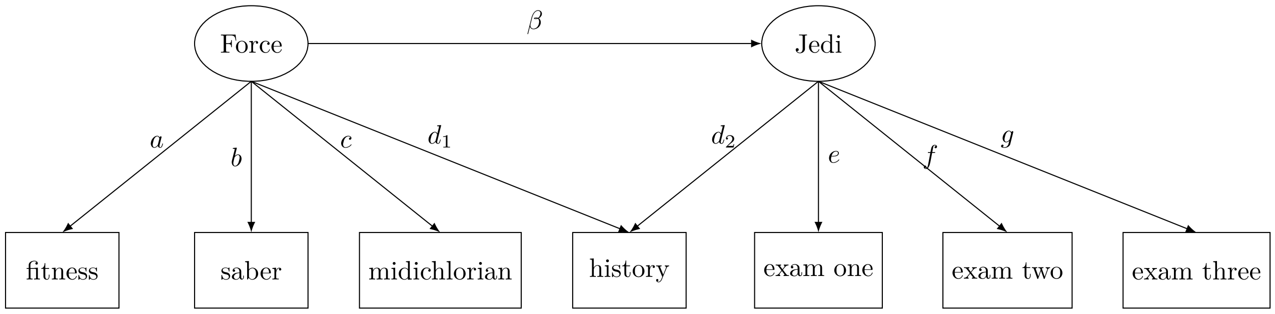
Solving for F, we get

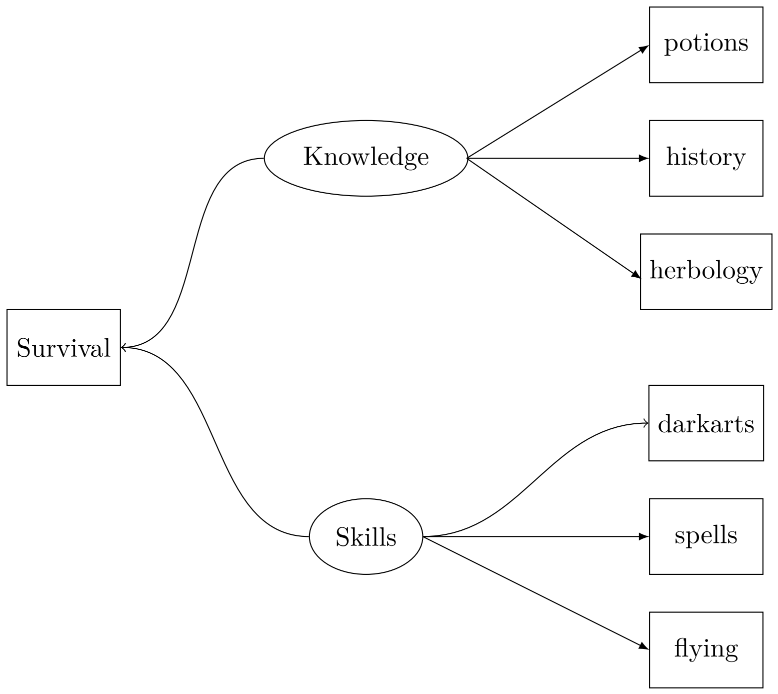
$$F = \frac{-1}{x_1} log\left(\frac{M}{X}\right) - x_1 \tag{10}$$

Let's assume we want to comput

Scratch







Hallgren, Kevin A., Connor J. McCabe, Kevin M. King, and David C. Atkins. 2019. “Beyond path diagrams: Enhancing applied structural equation modeling research through data visualization.” *Addictive Behaviors* 94 (July): 74–82. https://doi.org/10.1016/j.addbeh.2018.08.030.