Seeing the Impossible: Visualizing Latent Variable Models

Brief Introduction

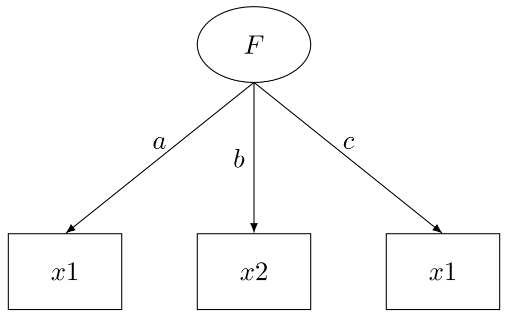
SEM is a powerful tool that is infinitely flexible and able to handle diverse sorts of modeling procedures. However, it has two fatal flaws. First, latent variables are, by definition, unobserved. As such, data visualization, a critical component of model building is not an easy task. Although some have 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.

To make the exercise more illustrative, I have simulated these data:

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

Using Wright's tracing rules, we can reproduce the model-implied correlation matrix:

$$\begin{pmatrix} 1 & ab & ac \\ ab & 1 & bc \\ ac & bc & 1 \end{pmatrix}$$

Such matrices are standard output in any SEM software, including lavaan and blavaan. Recall the following relationship between the correlation and the slope,

$$\beta = r\frac{\sigma_Y}{\sigma_X}$$

This allows us to calculate the model-implied slope for each bivariate relationship:

$$E[Y|X] = r_{ab}\frac{\sigma_Y}{\sigma_X}X \tag{1}$$

Overlaying Equation 1 on the raw data, we get a line that closely approximates the actual relationship between the two variables (Figure 1), indicating the model fits well.

```
  # compute the model-implied slope
implied.cor = lavInspect(fitted, what="cor.ov")
implied.cov = lavInspect(fitted, what="cov.ov")
stdev_ov = sqrt(diag(implied.cov))
estimated.slope = implied.cor["x1","x2"]*(stdev_ov["x2"]/stdev_ov["x1"])

ggplot(data=d, aes(x=x1,y=x2)) +
  geom_point() +
  geom_abline(slope = estimated.slope, intercept=0, col="red") +
  geom_smooth(method="lm") +
  labs(x="Exam One", y="Exam Two") +
  theme_bw()
```

We could also apply this command continuously to all observed variables in a scatterplot matrix (using, say, the upper diagonal), as in Figure 2. Here, I am utilizing the `visualize` function in the `flexplavaan` package. 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). Notice the two lines are quite similar (as they should be because these are simulated data).

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 observed variables on interest, once the effect of the latent variable(s) have been removed. This is similar to a

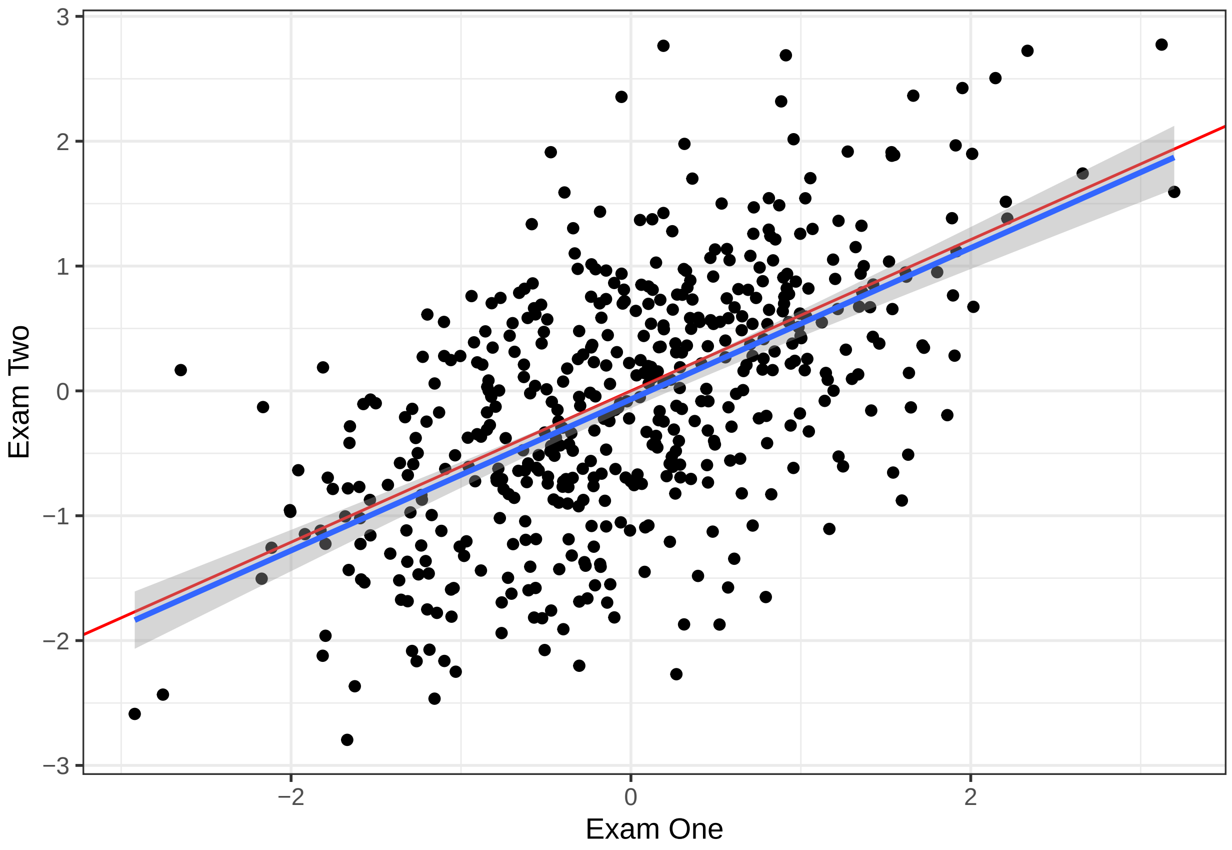


Figure 1: Scatterplot of the x1/x2 relationship, where the red line shows the implied fit between the two variables. The blue line shows the actual fitted relationship, while the red line shows the model-implied fit.

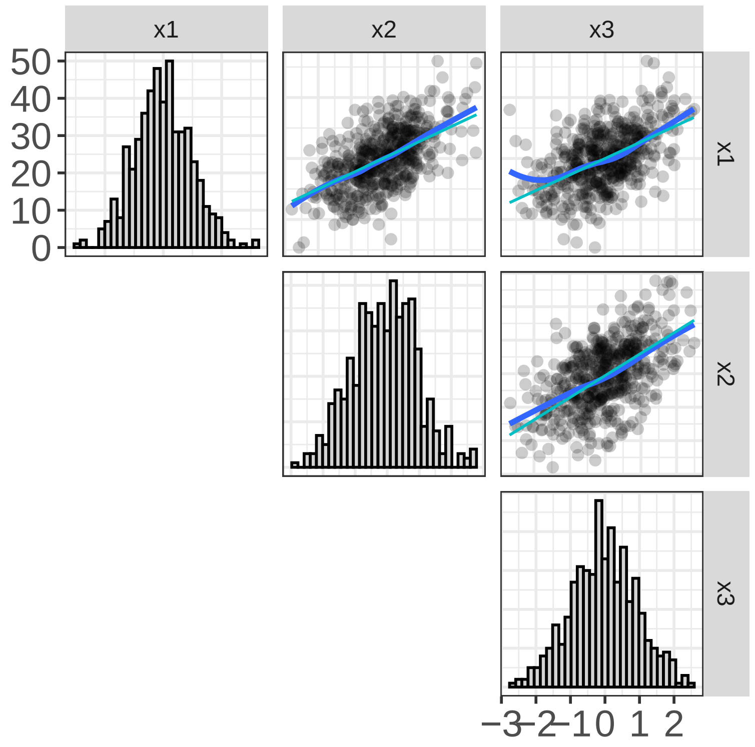


Figure 2: Scatterplot matrix of the indicator variables. As before, the red line shows the implied fit between the two variables and the blue line shows the actual fitted relationship (measured via loess lines).

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 Y scores from Equation **??** for each individual, then subtract those from the observed Y scores. We can then plot the relationship between Y and X 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 as in Figure 3. The default for the visualize function displays the model-implied plots in the upper triangle and the disturbance dependence plots in the lower triangle. As expected (because the data were simulated), the loess lines in the lower triangle are very similar to the model-implied line (i.e., the line where y=0).

Detecting Cross-Loadings

These diagnostic plots are designed to detect deviations from the model's implied relationship. In the example below, I'm going to utilize a simulated dataset of Jedi training, where two latent variables are posited (Force and Jedi), each with 3 indicators (fitness, saber, midichlorian for Force and exam one, exam two, and exam three for Jedi). In addition, one variable (history) has cross loadings on both factors. However, I will first fit a model where history is only modeled on the Force latent variable, then subsequently fit a model where the variable loads on both latent variables.

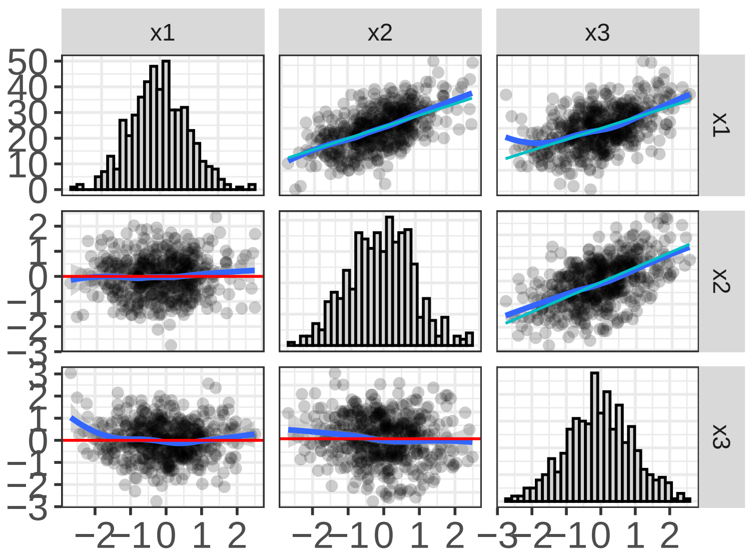
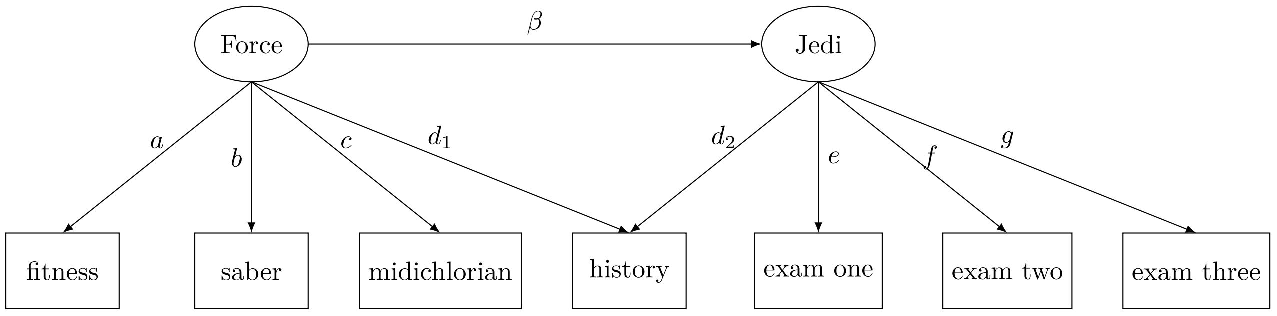


Figure 3: Scatterplot matrix of the indicator variables. The upper triangle show the model-implied fit between the indicators, while the lower triangle shows the disturbance dependence plots. As before, the red line shows the implied fit between the two variables and the blue line shows the actual fitted relationship (measured via loess lines).

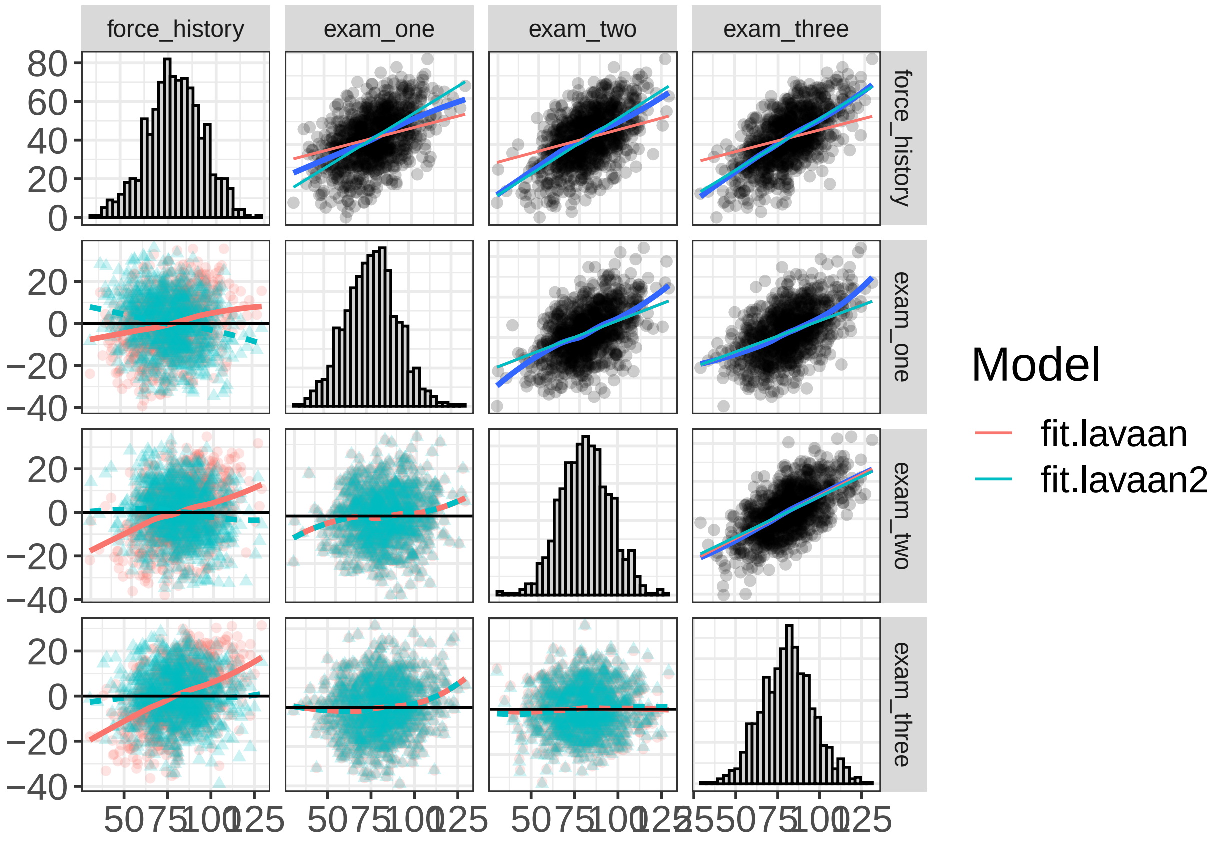


The table below shows the result of both models and various measures of fit. If one were not to fit the second model, it would be unclear where the misfit occurs. One could, of course, study modification indices, but they have a multitude of problems. One could also study the residual matrix, but even these do not suggest how the model could be fixed. Rather, the data could be plotted, as they are in Figure **??**. This makes it very apparent where the misfit comes from.

	No Crossloadings	Crossloadings Included
npar	15.00	16.00
fmin	0.18	0.01
chisq	337.28	16.10
df	13.00	12.00
pvalue	0.00	0.19
baseline.chisq	2178.11	2178.11
baseline.df	21.00	21.00
baseline.pvalue	0.00	0.00
cfi	0.85	1.00
tli	0.76	1.00
logl	-26255.66	-26095.07
unrestricted.logl	-26087.02	-26087.02

	No Crossloadings	Crossloadings Included
aic	52541.32	52222.13
bic	52613.55	52299.18
ntotal	912.00	912.00
bic2	52565.91	52248.37
rmsea	0.17	0.02
rmsea.ci.lower	0.15	0.00
rmsea.ci.upper	0.18	0.04
rmsea.pvalue	0.00	0.99
srmr	0.10	0.01

```
visualize(force_fit, force_cross, subset=4:7)
```



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:

$$X = \frac{M}{1 + exp^{-x_1(F-x_0)}} \tag{2}$$

Solving for F, we get

$$F = \frac{-1}{x_1} log\left(\frac{M}{X}\right) - x_1 \tag{3}$$

Let's assume the variable Y is a linear function of F:

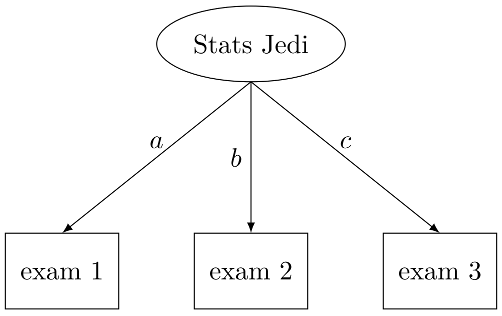
$$Y = y_0 + y_1 F \tag{4}$$

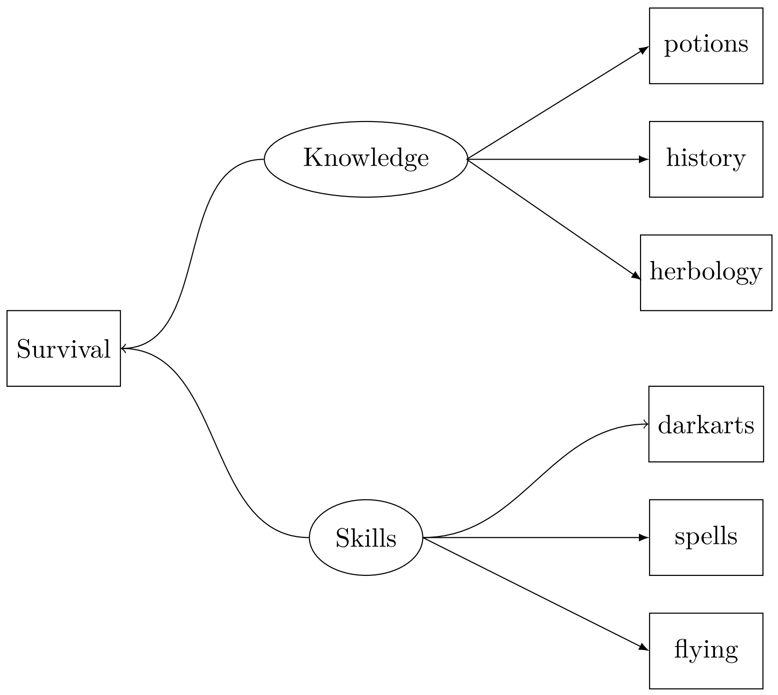
Solving for F, we get

$$F = \frac{Y - y_0}{y_1} \tag{5}$$

Now suppose we wish to estimate the value of Y for a given X. To do so, we can use substitution:

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.