Title: Seeing the Impossible: Visualizing Latent Variable Models in flexplavaan

Latent variable models (LVMs) are powerful and flexible tools that allow users to model sophisticated relationships involving variables that are not observed directly. Unfortunately, LVMs suffer from several serious limitations. Foremost among these limitations is that there is no intuitive way to visualize the data from LVMs because the variables of interest are unobserved. This makes it hard to encode information from the model, diagnose local and global misfit, and evaluate model assumptions. In this paper, we introduce flexplavaan, an R package designed to augment analyses in lavaan and blavaan. This package provides easy-to-use visuals that make it easy to encode LVMs, evaluate model assumptions, and diagnose misfit. In this paper, we develop the logic behind LVM visualizations and illustrate how flexplavaan is able to detect misfit and violated assumptions, as well as provide guidance on how best to modify models.

**Introduction**

It is currently an unprecedented time in the social sciences; multiple scientific disciplines are reeling from a “replication crisis” (Camerer et al., 2018; Ioannidis, 2005; Pashler & Wagenmakers, 2012), new norms for credibility are becoming more prevalent (Nelson, Simmons, & Simonsohn, 2018; Nosek, Ebersole, DeHaven, & Mellor, 2018), and the push for open science is accelerating at a rapid pace (Nosek et al., 2018). Amidst this push for open science practices, some have called for greater use of visualization techniques (Fife, 2020; Fife & Rodgers, 2019; Tay, Parrigon, Huang, & LeBreton, 2016). As noted by Tay, et al. (2016), “﻿[visualizations]...can strengthen the quality of research by further increasing the transparency of data...” (p. 694). In other words, one of the best, and most efficient ways of making data analysis open and transparent is to *display* each and every datapoint through visualization techniques.

Not only do visualizations adhere to the principles of openness and transparency, but they offer several additional advantages; they vastly improve encoding of information (Correll, 2015), they highlight model misfit (Healy & Moody, 2014), and they are an essential component in evaluating model assumptions (Levine, 2018; Tay et al., 2016). As such, we (as well as others, e.g., Fife, 2019, 2020; Wilkinson & Task Force on Statistical Inference, 1999) insist that *every* statistical model ought to be accompanied by a graphic.

Unfortunately, this visualization requirement is easier said than done. While visualizing some statistical models is trivial (e.g., regressions, t-tests, ANOVAs, multiple regression), visualizing others is not. One particularly troublesome class of models to visualize is latent variable models (LVMs). While researcher routinely visualize *conceptual* models (e.g., via path diagrams), visualizing the *statistical* models is not so easy. The former visualizations are common, while the latter are not (Hallgren, McCabe, King, & Atkins, 2019a). The reason statistical visualizations of LVM are not intuitive is because they rely on *unobserved* variables (Bollen, 1989). If the variables of interest are unobserved, how can we possibly visualize them?

Though it is not, at first glance, easy to visualize unobserved variables, that does not mean visualizing them is any less important. On the contrary, visualizing latent variables is, perhaps *more* important *because* their presence is unobserved. In the following section, we elaborate on why visualizations are particularly crucial for LVMs. We then review previous approaches others have used for visualizing LVMs, and note their strengths and weaknesses. We then introduce our approach and the corresponding R package flexplavaan, which allows users to visualize both lavaan and blavaan objects in R. We then conclude with several examples that highlight how visualizations assisted in identifying appropriate statistical models.

**Why Visualizations are Critical for LVMs**

**Previous Approaches to Visualizing LVMs**

**Our Approach (Linear LVMs)**

**Diagnostic Plots: Trail Plots**

To begin how to conceptualize LVMs, let us first consider how typical linear models are visualized. In a standard regression, each dot in a scatterplot represents scores on the *observed* variables. Often, analysts overlay additional symbols to represent the fit of the model (e.g., a line to represented the fitted regression model, or large dots to represent the mean). Sometimes additional symbols are overlaid to represent uncertainty (e.g., confidence bands for a regression line or standard error bars). See Figure 1 as an example. In either case, the dots represent observed information, while the fitted information is conveyed using other symbols.



Figure 1: Example figure that shows how standard statistical models are visualized. Dots represent scores on observed variables, while other symbols (e.g., regression line, large dots) represent the fit of the model.

Likewise, visualizing LVMs ought to follow similar conventions; the dots should represent the observed information, as in Bauer (2005). In his visuals, pairwise relationships between observed variables are represented in a scatterplot. However, Bauer’s approach did not overlay a model-implied fit, as we seek to do. When the line represents the model-implied fit, it denotes the *trail* left behind by the unobserved latent variable. As such, we call these plots “trail plots.”

How then does one identify the slope/intercept of the LVM’s model-implied fit? It is quite easy to do so when standard linear LVMs are used. Suppose we have a factor (*T*) with three indicators (e.g., , and ), and we wish to visualize the pairwise traceplot between and . To do so, we can simply utilize the model-implied correlation matrix:

where is the model-implied correlation between and , and are the standard deviations of the two variables. One can then estimate the intercept using basic algebra:

Figure 2 shows the LVM model-implied fit in red with a regression line in blue for simulated data. Because the regression line minimizes the sum of squared errors, we would hope that the LVM fitted line (red) closely approximate the regression line. In this case, the two overlap quite extensively. On the other hand, if the two lines differ, we can be certain the LVM fails to capture the entire relationship between the two observed variables.

****

Figure 2: The LVM-implied fit between X1 and X2, shown in red. The blue line represents the regression line between the two variables. The more closely the model-implied fit line resembles the regresson line, the better the fit of the LVM.

Of course, Figure 2 only shows one pairwise relationship between variables. If we wished to visualize all the variables in our model, we would have to utilize a scatterplot matrix, as in Figure 3. Naturally, this becomes quite cumbersome when users have more than seven or eight variables. In this case, it is best to visualize only a subset of variables. We will later discuss strategies for how best to select appropriate subsets.

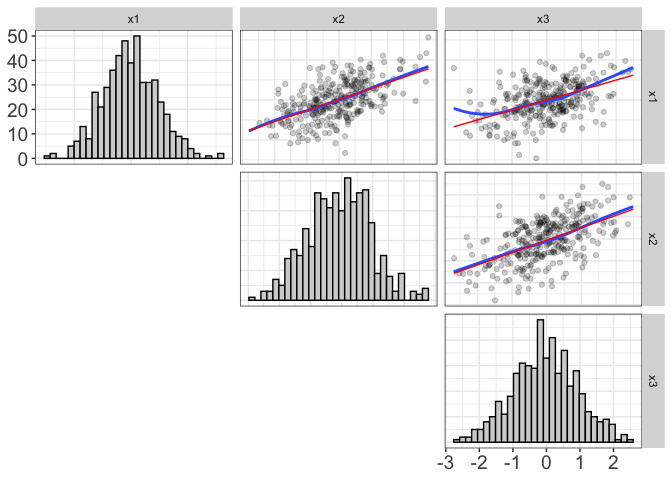


Figure 3: Scatterplot matrix showing the model-implied fit (red) and regression-implied fit (blue) between three simulated indicator variables. The diagonals show the histograms.

The advantage of trail plots is that they easily show misfit in LVMs. Misfit in measurement models reveal themselves by showing misfit between two indicators of the same variable. For example, the left plot in Figure 4 shows a trail plot for a model where the user has specified a three-factor model, but the third variable () is unrelated to the latent variable. Trail plots will reveal structural misfit by showing differences in fit between indicators of different variables. For example, the right plot shows a model where x1-x2 load onto one latent variable, x4-x5 load onto another, and x3 loads onto both; however, the specified model assumes x3 only loads onto the first model.



Figure 4: Two plots showing different types of misfit. The left plot shows measurement misfit, while the right plot shows structural misfit.

**Disturbance-Dependence Plots**

One common technique for visualizing the adequacy of statistical models in classic regression is residual-dependence plots. With these graphics, one simply plots the residuals of the model (*Y* axis) against the predicted values (*X* axis). The rationale behind this is simple: the model *should* have extracted any association between the prediction and the outcome. The residuals represent the remaining information after extracting the signal from the model. If there is a clear trend remaining in the data (e.g., a nonlinear pattern or a “megaphone” shape in the residuals), this indicates the model failed to capture important information.

Likewise, in LVMs, we can apply this same idea to determine whether the fit implied by the LVM has successfully extracted any association between any pair of predictors. However, in LVMs, residuals refer to the discrepancy between the model-implied and the actual variance/covariance matrix (or correlation matrix). As such, naming these plots “residual-dependence plots” would be a misnomer. Rather, misfit at the raw data level is typically called a disturbance[[1]](#footnote-1). As such, we call these plots disturbance-dependence plots.

Like trace plots, we visualize disturbance-dependence plots for each pair of observed variables. To do so, flexplavaan subtracts the fit implied by the trace plots from the observed scores. For example, a disturbance dependence plot for an *X*1/*X*2 relationship would subtract the “fit” of *X*2 implied by the trace plot from the actual *X*2 scores (and vice versa for the *X*2/*X*1 relationship). If the trace-plot fit actually extracts all association between the pair of observed variables, we would expect to see a scatterplot that shows no remaining association between the two. If there is a pattern in the scatterplot remaining, we know the fit of the model misses important information about that specific relationship. To aid in interpreting these plots, we can overlay the plot with a flat line (with a slope of zero), as well as a regression (or loess) line. The first line indicates what signal *should* remain after fitting the model, while the second line shows what *actually* remains.

Figure xx shows an example of trace plots in the upper triangle and disturbance-dependence plots in the lower triangle of a scatterplot matrix. These plots are for the same data shown in the right image of Figure 4. Notice how ….

Together, both of these plots (trace plots and diagnostic-dependence plots) serve as a critical diagnostic check. Both these plots will signal misfit both in the measurement and structural components of the model. Conversely, these models will also help users determine whether the model is to be believed. If they show the model adequately fits the data, the user can then proceed to plot two different types of plots: measurement plots and structural plots.

**Measurement Plots**

One of the primary purposes of the diagnostics is to determine whether one’s conceptualization of the latent variables is to be believed. If the trace plots and disturbance dependence plots indicate the LVM is a good representation of the data, one can be more confident the latent variables are properly estimated. If this is the case, we can now make a step toward visualizing the latent variables themselves.

The approach we suggest is to plot the factor scores on the $Y$ axis and the observed scores on the $X$ axis. Naturally, this means one could really only visualize one indicator at a time, which is a serious limitation for most (if not all) latent variable models. To overcome this problem, we recommend paneling each indicator, as in Figure xxx. Notice each variable is converted to z-scores to make it easier to compare the observed/latent relationship across indicators.

Another alteration from a standard scatterplot is the use of color gradients.

* Diagnostics tells us whether the estimated factor scores can be believed
* If they are to be believed, we can then develop visuals for the measurement model
* Approach
  + Estimated factor scores on X axis
    - symbol plotted must reflect uncertainty because these are not observed scores
    - we use a line that reflects the upper/lower limits of a 95% confidence or credible interval
  + observed scores on Y axis
    - these have no height (because there’s no uncertainty about their observed scores)
  + to condense visuals, we can convert from wide to long format
    - put observed scores on the same scale
    - panel by the observed variable indicator
* These improve encoding
  + make it apparent which observed variables have highest reliability

**Structural Plots**

* Often in LVMs, the visuals of interest are not the observed, but the latent variables
* Measurement model is ancillary
* As before, we need to reflect uncertainty in the estimate of the latent scores
  + now we show our uncertainty via ellipses
  + uncertainty is estimated using prediction intervals (for frequentist LVMs) or using information from the posterior distribution (for Bayesian LVMs).
* Type of plot will depend on the structural relationships hypothesized
  + e.g., scatterplot for two latent continuous variables, coplots for three latent continuous variables, beeswarm plots for a categorical versus continuous latent variable.
  + for a review, see (Fife, 2019)

**Nonlinear LVMs**

* diagnostic plots will work to *diagnose* nonlinearities
  + will not remedy these
* to remedy nonlinearity, a different approach is needed

**Bayesian LVMs**

* linear LVMs utilize covariance algebra to model fit
* Bayesian LVMs, on the other hand, use raw data
* relationships are specified explicitly
  + as well as residual distributions
* Bayesian LVMs offer several advantages
  + can estimate models impossible to estimate in linear models
  + more flexible (e.g., non-Gaussian latent factors specified in straightforward way)
  + allows users to augment analysis with priors
  + latent scores are estimated as part of the modeling procedure (not a secondary step)
  + missing data handled intuitively
  + after computing factor scores and uncertainty intervals, can compare individual cases more readily than in frequentist
    - e.g., is France (specific case) higher on liberal democracy (latent factor) than Poland?
* blavaan provides an easy-to-use interface for Bayesian LVMs
  + utilizes same syntax as lavaan
  + fitted objects are lavaan objects, so same functions can be utilized for both models
* blavaan generates JAGs or STAN syntax to make it easier to modify MCMC

**Model-Implied Fit For Nonlinear Bayesian LVMs**

* earlier derivation utilized model-implied correlations
  + these assume the fit is actually linear
* if nonlinear, the model-implied fit must be modified
* our approach relies on the estimated factor scores to derive the trail plots
* basic approach
  + model the relationship between the latent variable (x axis) and observed variable (y axis)
    - utilize a smoothed cubic spline function to allow nonlinear patterns
    - store those predictions (call these )
    - do the same for the other variable and store those predictions (call these )
  + serve as the basis of the coordinates of X/Y
    - but are the fit of the model when reliability has been removed
    - it will tend to overestimate the fit of the X/Y relationship
    - must be attenuated proportional to reliability
  + Estimate reliability
    - serves as our best estimate of the true scores of *X*
    - Reliability can be estimated as
  + Attenuate based on reliability
    - Recall that reliability provides upper bound to validity
    - With reliability of zero, the prediction of Y for a given X is the mean of Y
    - With perfect reliability, the relationship between X and Y will be equal to the relationship between
    - To weaken the line, we can adjust the predicted scores to be closer to the mean of *Y*
  + Once we have a model-implied relationship between *X/Y*, we can develop trace plots/DDPs as before

**Examples**

**Well-Fitting Model**

* show a model where rmsea is poor, but the model fits quite well

Omitted Cross-Loading

* show a model (Jedi) where the rmsea is good, but the model fits poorly based on visuals

Nonlinear model

* again, rmsea is good, but the model misses

1. This too is a bit of a misnomer. Technically, disturbances refer to random errors for endogenous variables. Our definition of disturbance includes both exogenous and endogenous variables. However, the term used is mere semantics. [↑](#footnote-ref-1)