Examples of potential time-varying covariates include amount of exposure to intervention, length of time spent studying, or number of friends. The idea here is that time-varying covariates are potential confounding variables that can change (vary) at each measurement occasion.

If you are undecided or aren't sure which way to go in terms of how to control for covariates, I suggest you use the full partial approach. It is the fullest of the control methods and does not depend on the type of model that you are trying to specify. That being said, the most common question I get on the use of covariates is "When should I include them?" The answer to this question is "It depends." If the covariates are known confounds of the relationships you are examining, I recommend including them in each model at each step of the model-building process. If the covariates are secondary concerns, I think you can wait to include the covariate influences at later stages of the model-building process. At this stage, you just want to be sure the potential covariates don't have an impact on the primary results. You should also be careful not to throw too many covariates into a model. Covariates should still be selected because they have a theoretically meaningful potential influence on the modeled relationships. When a large number of covariates are thrown into a model, the estimation can be cumbersome, the patterns of effects are difficult to interpret, and the modeled relationships are overcontrolled. By overcontrolled I am referring to what can happen due to effect of random variation. By chance alone, a covariate might happen to be associated with the focal constructs; when this source of random information is removed from the focal construct, overcontrol has occurred. Given that SEM is a theory-driven enterprise, throwing a bunch of covariates into a model as an atheoretical afterthought simply does not make sense.

RESCALING VARIABLES

As a general measurement issue, it is sometimes necessary to rescale variables before including them in an SEM model. One of the most important assumptions of SEM is that the variables used are multivariate normal. When the individual variables are all univariate normal, the likelihood of their being multivariate normal is pretty good. When some of the variables are skewed or kurtotic, the likelihood of their being multivariate normal is not so good. Maximum likelihood estimation is robust enough to tolerate moderate violations of the multivariate normal assumption, and the so-called robust estimation methods can also handle non-normal data. Some scholars suggest performing a transformation on the variables that violate univariate normal assumptions to help in the process of stabilizing your model and its estimates (Tabachnick & Fidell, 2007). Here, square-root, log, or inverse transformations are sometimes recommended to make the variables more univariate normal. In my experience, however, such transformations have little influence on the overall conclusions that are drawn from the results of the models. Granted, some

point estimates differ, some standard errors are different, the model fit information is slightly different, but the big-picture conclusions and generalizations are not substantially different. A bigger concern is that the metric of the measured variables is dramatically changed and that the interpretation of the scores on the latent variables, as well as their associations, becomes quite muddied. Moreover, as robust estimation procedures are readily available, the need to transform the raw data has become mostly a moot issue. My recommendation, therefore, is to resist the temptation to use transformations that attempt to make the distribution more normal (bell-shaped). Instead, apply a robust estimation approach, such as bootstrap estimation.

Another type of rescaling is simply to change the metric of the variables to be on comparable scales. This type of transformation does not change the shape of the distribution but instead rescales the scores to a different metric from the original one. I use such transformations when I have variables that are on very different metrics (e.g., age in years vs. millisecond reaction times). I do this type of transformation primarily because some SEM software packages seem to have a harder time converging on estimates when the metrics are quite different and an easier time when the metrics are roughly similar. A secondary reason is that I have a better feel for the magnitudes of the associations among variables when they are on a similar metric. Most modelers are able to see the patterns of associations in a correlation matrix pretty easily. When the covariance metric is similar across variables, you can see the same kinds of relative patterns that you can see in a correlation matrix, which helps with model evaluation and in troubleshooting estimation problems.

As mentioned, most SEM software will have estimation difficulties when the variables used in a model are on very different metrics. The reason for this estimation problem is that the covariances that are calculated and the parameters used to recreate the covariances (i.e., the model-implied covariance matrix) differ by orders of magnitude. For example, a covariance between two variables coded on a 0–1 metric might have meaningful differences in its estimates in the second decimal place, whereas a covariance between two variables coded on a 0–100 scale would have meaningful differences in the units place. If all the variables are put on a roughly similar metric, the meaningful information is in the same place, and the programs will find a solution to the specified model more expeditiously.

Dividing or multiplying the variables by a constant value is a common procedure to put them on a roughly common metric. Such transformations are monotonic in that they don't affect the individual-differences standings or the associations among the variables; monotonic transformations only influence the metric and the resulting meaning of the metric of a variable. Typically, you don't want to do a full *z*-score standardization of each variable, because then you lose the covariance metric that is needed for the SEM procedures, and you lose any information about mean-level changes over time. With longitudinal data, keeping the integrity of relative mean-level changes over time is crucial, and you want to use a metric that would still satisfy the need to estimate models on covariances and not correlations.

The percentage or proportion of maximum scoring (or POMS) transformation is one that I typically recommend. In Table 1.5, I give two versions of POMS: 1.5A is the proportion of maximum scoring using the theoretical maximum score, and 1.5B is the percentage of maximum scoring using the observed maximum score (in these examples I use an arbitrarily chosen 5-point Likert scale for illustration purposes). The theoretical versus observed maximum score can be swapped between formulas 1.5A and 1.5B; you can multiply by 100, which would make the score a percentage, or not multiply by 100, which would keep the score a proportion. For longitudinal models it is important to remember that the max score used for the transformation must be the same values used to transform each time of measurement. This common value is needed in order to see any longitudinal changes on the construct. If the max score is different for each time point, the meaning of the change information is lost.

Another situation where rescaling is useful is when variables are recorded on different Likert-type scales; for example, a measure may be given to respondents on a 5-point Likert scale at one age group but given at another age group on a 7-point scale. In a situation like this, a transformation is needed. That is, to compare any growth or change on the instrument, the scores on the measures must be put on a comparable metric across the two age groups. There are some possible rescaling approaches that you can employ: convert all of the measures to a 0–1 scale, convert all of the measures to a percentage or proportion of maximum scale, or convert the narrower scale (e.g., 1–5 in the preceding example) to the metric of the wider scale (e.g., 1–7; see Table 1.5C).

Another transformation that can be done is to standardize all the variables in an analysis across time to be on a similar metric. Here, you would need to calculate the grand mean over time and the grand standard deviation over time and use that mean and standard deviation to transform the scores within each time point (i.e.,

TABLE 1.5. Several alternative rescaling formulae to put all variables on similar metrics

A) $R1 = (O - 1) / S_{max}$	B) $R100 = ((O - 1) / O_{max}) \cdot 100$	C) $R7 = (((O5 - 1) / 4) \cdot 6) + 1$
• $R1$ is the rescaled variable, which we want to become a 0 to 1 scale.	• $R100$ is the rescaled variable as a percent of the maximum score.	• $R7$ is the rescaled variable, which we want to become a 1 to 7 scale.
• O is the original scale; in this example assume a 5-point scale from 1 to 5.	• O is the original scale, in this example assume a 5-point scale from 1 to 5.	• $O5$ is the original scale; in this case it is on a 5-point scale from 1 to 5.
• 1st: subtract 1 to make the scale go from 0 to 4. If the scale already starts at 0, don't subtract 1.	• 1st: subtract 1 to make the scale go from 0 to 4. If the scale already starts at 0, don't subtract 1.	• 1st: subtract 1 to make the scale go from 0 to 4. If the scale is already on a 0 to 4 scale, you don't need to subtract 1.
• S_{max} is the new scale maximum.	• O_{max} is the maximum scale value that was observed in the data after subtracting 1 or the max if it already started at 0.	• 2nd: dividing by 4 makes the scale go from 0 to 1.
• 2nd: dividing by S_{max} makes the scale from 0 to 1 or less than 1 if the scale maximum is not observed.	• 2nd: dividing by O_{max} makes the scale go from 0 to 1.	• 3rd: multiplying by 6 makes the scale go from 0 to 6.
• This rescoring method is a proportion of maximum scoring (you can convert to a percent if you multiply by 100).	• 3rd: multipyling by 100 makes it a percent, which ranges from 0 to 100.	• 4th: adding 1 makes the scale go from 1 to 7.

Note. These transformations do not change the distribution of the variables. They are monotonic transformations, like a *z*-score transformation. The original scale is put on a metric that is relatively more interpretable, or at least similar, across some or all of the variables in the analysis.

don't use the mean and standard deviation within a given time point, because you lose any information about change in means and variances). If you have multiple groups and longitudinal data, you would calculate the grand mean across time and groups, as well as the grand standard deviation across time and groups. With this method you can leave the scores in a grand *z*-score metric (again, however, the mean and standard deviation used must be the overall mean and standard deviation so you don't lose change information). You can also add a grand constant and multiply by a selected value to put the indicators on a so-called *t*-score metric (e.g., mean of 100 and a standard deviation of 15, or a mean of 10 and a standard deviation of 2, or a mean of 50 and a standard deviation of 10—there is no official definition of the mean and standard deviation of a *t*-score as far as I know). The choice of the rescaling method here does not "normalize" the data, nor does it change the shape of the distribution or the strength of an association between any of the variables. Rescaling simply provides a metric that makes the estimated values more interpretable (and can help with convergence problems). Of course, rescaling variables when they are already in meaningful and interpretable metrics may not be desired unless you are experiencing convergence problems.

The final recoding issue is reverse-coding of variables. I generally recommend that all indicators be coded so that a high score on each indicator has the same interpretation (e.g., more *X* or less *Y*). For example, if I have one indicator of anxiety that is worded as something like "I don't get nervous" with a Likert scale of *disagree* to *agree*, a high score would mean less anxious. If the other indicators are worded such that a high score means more anxious ("I am easily aroused"), I would reverse-code the "I don't get nervous" item. If the Likert scale is a 1–7 scale, I can simply create a new variable that is 8 minus the values of the original variable (8 – 7 = 1 and 8 – 1 = 7). It is also a good habit to name the latent construct by what the higher values mean. For example, if all my indicators are coded such that a high score means less anxiety, I would call the construct "lack of anxiety" or "nonanxiousness." If the indicators are coded such that a high score means more anxiety, I would label the construct "anxiety" or "anxiousness." In a similar vein, when creating dummy codes to represent gender or ethnic categories, I recommend labeling the variable with the meaning of a high score. For example, if gender is coded 0 = female and 1 = male, then I recommend calling the variable "male." Then you don't have to ask "Which gender is coded as 1?"

PARCELING

Parceling refers to taking two or more items and packaging them together (i.e., averaging them), much like a parcel you would take to the post office. The parcel (instead of the original items) is then used as the manifest indicator of the latent construct. Parceling is a premodeling step that is done before the data are fed into the SEM software. When packaging items to form a parcel (or a scale score for that matter), I