1

Simulation Variants in Mplus

Localized Simulations for Complex Mediation Models: Non-normal Data

Non-Normality for External Simulations in Mplus

INTRODUCTION

This supplement elaborates the simulation strategies for sample size analysis described in Chapter 28. I assume you have read Chapter 28. The first section of the document shows you how to conduct a power analysis simulation for more complex RET designs with multiple mediators and covariates. The second section shows you how to pursue power analysis simulations for the chi square global test of fit. The third section shows you how to conduct a power analysis simulation for the joint significance test of an omnibus indirect effect. I then cover a wide variety of localized simulations of many different forms. Additional examples of SEM-based simulation design for sample size decisions are in Muthén and Muthén (2002) and Muthén and Curran (1997). Every numerical example in the Mplus User's Guide is accompanied by a simulation program to generate their example data. These programs can be found in the Mplus example folder and can be adapted for power analysis simulations. In general, you can read the different sections of the current document on a need-to-know basis although at times, I refer you to the material in another section that you should study first.

For many of the simulations I use the example from Chapter 28 about an intervention to improve study skills and, in turn, performance on an end-of-year math exam. This is a simplified RET with a treatment and control group, a single mediator, and a single outcome. The idea is that you can easily generalize what you learn to more complex designs once you understand the core logic. The model is reproduced in Figure 1. To refresh your

memory, the study evaluates an intervention that teaches middle school students study skills for math. There are two treatment conditions, an intervention group and a control group. The mediator is student study skills measured four weeks after the intervention. The measure is scored from 0 to 100 with higher scores indicating better skills. Students typically score about 60 on the test, with a standard deviation of 15 or so. The outcome is performance on the final math exam. Scores on the exam range from 0 to 100 with higher scores indicating better performance. Like many school exams, a score of 90 is excellent, a score of 80 is above average, a score of 70 is average, and so on. Path p_3 in the figure is not included in the model, but I show it with the dashed arrow for referencing when I want to make points about its inclusion or exclusion. The assumption is that the effects of the intervention on exam performance are completely mediated by the study skills the program targets for improvement.

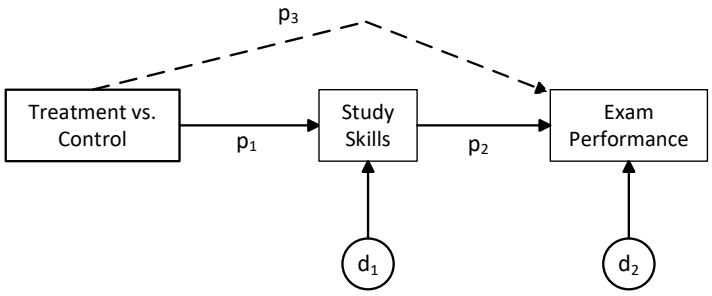


FIGURE 1. Simulation Example with Single Mediator

WORKING WITH COMPLEX MODELS: PART II

In this section, I show you heuristics for sample size analysis for complex RET models with multiple mediators. I develop heuristics using the raw metric approach described in Chapter 28. I then show you how to use the standardized metric approach from Chapter 28.

In Chapter 28, I presented three sets of equations to help determine population values for a simulation. One set of equations focused on variance decomposition and eta squared:

$$var_{TOTAL} = var_{BETWEEN} + var_{WITHIN} \quad [1]$$

$$var_{TOTAL} = var_{REGRESSION} + var_{ERROR} \quad [2]$$

$$var_{TOTAL} = var_{SYSTEMATIC} + var_{ERROR} \quad [3]$$

$$var_{TOTAL} = var_{EXPLAINED} + var_{UNEXPLAINED} \quad [4]$$

$$Eta^2 = var_{BETWEEN} / var_{TOTAL} = var_{REGRESSION} / var_{TOTAL} = var_{SYSTEMATIC} / var_{TOTAL} = var_{EXPLAINED} / var_{TOTAL} \quad [5]$$

Another set was for the relationship between covariances and correlations, expressed here for simplicity using sample notation:

$$r_{XY} = (cov_{XY}) / [(SD_X)(SD_Y)] \quad [6]$$

$$cov_{XY} = (r_{XY})(SD_X)(SD_Y) \quad [7]$$

The third set was for the variance of a linear combination. For two predictors in the model $Y = \alpha + \beta_1 X_1 + \beta_2 X_2 + \varepsilon$, the equation is:

$$var(Y) = [\beta_1^2 \, var(X_1) + \beta_2^2 \, var(X_2)] + [(2)(\beta_1)(\beta_2)cov(X_1,X_2)] + var(\varepsilon) \quad [8]$$

and for *k* predictors, it is

$$\sigma_Y^2 = [\sum_{j=1}^{k} \beta_j^2 \sigma_j^2] + [2 \sum_{m=1}^{k} \sum_{j>1}^{k} \beta_m \beta_j \, cov(X_m, X_j)] + \sigma_\varepsilon^2 \quad [9]$$

See Chapter 28 for a discussion of these equation sets.

The model I work with has a treatment versus control group (0 = control, 1 = treatment), three posttest mediators (M1, M2, M3) each with a baseline counterpart (M1B, M2B, M3B), and a follow-up outcome Y with a baseline Y (YB). For purposes of pedagogy, I will assume all of the mediator and outcome measures have a metric from 0 to 100 with a standard deviation of 10, as do their baseline counterparts. In practice, the metrics likely will vary from one variable to the next.

Figure 2 shows the model structure but omits the presumed correlations between the exogenous variables to reduce clutter. The path coefficients for covariates are signified by *b*s and the primary causal paths of substantive interest are signified by *p*s.

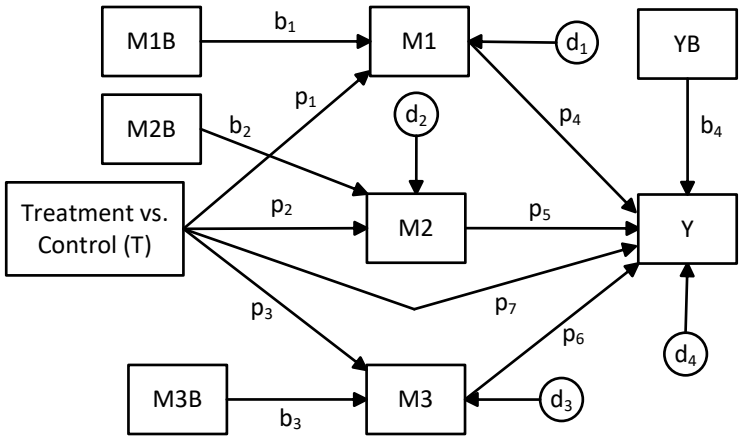


FIGURE 2. Complex RET Model

There are four endogenous variables in the model, yielding four core equations:

$M1 = a_1 + b_1\ M1B + p_1\ T + d_1$

$M2 = a_2 + b_2\ M2B + p_2\ T + d_2$

$M3 = a_3 + b_3\ M3B + p_3\ T + d_3$

$Y = a_4 + b_4\ YB + p_4\ M1 + p_5\ M2 + p_6\ M3 + p_7\ T + d_4$

I now develop how to construct simulations for sample size analysis of this model.

The Raw Metric Approach

I begin by specifying the population covariances between the exogenous variables, which are the baseline mediators, the baseline outcome, and the treatment condition dummy variable. Because the treatment condition is randomized, it is uncorrelated with all the other exogenous variables. Suppose based on my knowledge of the exogenous variables from past research I decide to set the correlations between all the other exogenous variables to be 0.30. You can vary them if you want, but I will use these values in the current example. I must translate these expected correlations into covariances for purposes of Mplus programming, for which I make use of Equation 7. Equation 7 multiplies each desired correlation by the relevant standard deviations. Because the standard deviations are 10.0 for all the exogenous variables except T, the covariances between any two of the exogenous variables (excluding T) will equal (0.30)(10)(10) = 30. Here is the Mplus syntax I ultimately will use in the larger Mplus program to specify the population covariances:

```
m1b WITH m2b*30 m3b*30 yb*30 ;
m2b WITH m3b*30 yb*30 ;
m3b WITH yb*30 ;
t WITH m1b*0 m2b*0 m3b*0 yb*0 ;
```

Next, I specify the population effects of the treatment condition on the posttest mediators, i.e., T→M. I decide to make one of them (p_1) equal to zero to evaluate Type I errors, the other (p_2) to equal what I judge to be a medium effect of 5.0, and the third (p_3) to equal what I judge to be a large effect of 8.0. These are covariate adjusted mean differences for each mediator as a function of the treatment group minus the control group. I selected the values of 5 and 8 based on my knowledge of the measures, prior research, and the context in which the RET is conducted (see Chapter 10). You will need to select the relevant covariate adjusted mean differences that you want to target.

The three posttest mediators also are impacted by their respective baseline mediators. The three posttest mediators have a standard deviation of 10 as do the baseline mediators. Based on past research of the measures/constructs, I decide to set the path coefficient from the baseline covariate to its posttest counterpart to 0.30 for each mediator, i.e., a one unit change in the pretest mediator is associated with a 0.30 unit change in the mean of the posttest mediator. This yields the following Mplus syntax that I ultimately will use in the larger Mplus program to set the values of the path coefficients to the mediators:

```
m1 ON m1b*0.30 t*0 ;
m2 ON m2b*0.30 t*5.0 ;
m3 ON m3b*0.30 t*8.0 ;
```

I next define the disturbance variances for M1, M2, and M3. The variance of M1 is 100 and it is decomposed via Equation 8 to be:

$$\text{var(M1)} = 100 = [b_1^2\ \text{var(M1B)} + p_1^2\ \text{var(T)}] + [(2)(b_1)(p_1)\text{cov(M1B,T)}] + \text{var}(d_1)$$

If I substitute the known values of the terms on the right hand side of the equation or the parameter values I have already specified, I obtain

$$100 = [(.30^2)(100) + (0^2)(.25)] + [(2)(.30)(0)(0)] + \text{var}(d_1)$$

$$= [9 + 0] + [0] + \text{var}(d_1)$$

and $\text{var}(d_1) = 100 - 9 = 91.00$

I repeat this process for M2 and find

$$\text{var(M2)} = 100 = [b_2^2\ \text{var(M2B)} + p_2^2\ \text{var(T)}] + [(2)(b_2)(p2)\text{cov(M2B,T)}] + \text{var}(d_2)$$

which yields

$100 = [(.30^2)(100) + (5^2)(.25)] + [(2)(.30)(5)(0)] + var(d_2)$

$= [9 + 6.25] + [0] + var(d_2)$

and $var(d_2) = 100 – 15.25 = 84.75$

For M3, I find

$var(M3) = 100 = [b_3^2\ var(M3B) + p_3^2\ var(T)] + [(2)(b3)(p_3)cov(M3B,T)] + var(d_3)$

which yields

$100 = [(.30^2)(100) + (8^2)(.25)] + [(2)(.30)(8)(0)] + var(d_3)$

$= [9 + 16] + [0] + var(d_3)$

and $var(d_3) = 100 – 25 = 75.0$

The Mplus syntax I will ultimately use to specify the population values for the mediator disturbance variances is

```
m1*91.0 ;
m2*84.75 ;
m3*75.0 ;
```

Next, I turn my attention to the path and regression coefficients for the predictors of the outcome, Y. I set the coefficient from YB to Y to 0.30, again, based on past research and theory. I decide to set p_4 for M1→Y to zero so I can evaluate Type I errors. I set p_5 for M2→Y to 0.20 to represent what I judge to be a medium effect and p_6 for M3→Y to 0.40 to represent what I judge to be a large effect. I set p_7 to 5.0 to reflect a medium direct effect of T on Y. Here is the Mplus syntax I ultimately will use to represent these population values:

```
y ON yb*.30 m1*0 m2*0.20 m3*0.40 t*5 ;
```

The final step is to specify the disturbance variance for Y. This is complicated because there are 5 correlated predictors whose intercorrelations are determined by other parts of the model structure that must be taken into account. To define the disturbance variance in a coherent way, I need to invoke a decomposition analysis using Equation 2:

$var_{TOTAL} = var_{REGRESSION} + var_{ERROR}$

where var_{ERROR} is the disturbance variance. I previously defined var_{TOTAL} as 100, but to work with the above equation, I need to know the value of $var_{REGRESSION}$ based on the other model population values I have already specified. Rather than derive the value of $var_{REGRESSION}$ mathematically using Equation 9, I can use an Mplus shortcut/heuristic instead. To execute the shortcut, I first program the simulation using all the values I have derived thus far but I set the disturbance variance for Y to zero. Table 1 presents the relevant Mplus syntax for doing so.

Table 1: Generate $var_{REGRESSION}$ for Raw Metric Approach

```
1.  TITLE: Generate variance regression ;
2.  MONTECARLO:
3.  NAMES ARE t m1 m2 m3 m1b m2b m3b y yb ;
4.  CUTPOINTS = t(0);
5.  NOBS = 2000000 ;     !sample size
6.  NREPS = 1 ;          !number of replicates
7.  !NOBS = 100 ;        !sample size
8.  !NREPS = 20000 ;     !number of replicates
9.  SEED = 2222 ;        !random seed
10. SAVE = temp.dat;
11. ANALYSIS:
12. ESTIMATOR = MLR ;
13. MODEL POPULATION:     !specify population model
14. [t*0] ;               !set mean when generating original continuous t
15. t*1 ;                 !set var when generating original continuous t
16. [m1b*0]; [m2b*0]; [m3b*0]; [yb*0];    !set means
17. m1b*100; m2b*100; m3b*100; yb*100 ;   !set variances
18. [y*0]; [m1*0]; [m2*0]; [m3*0];        !set intercepts
19. m1b WITH m2b*30 m3b*30 yb*30 ;        !set covariances
20. m2b WITH m3b*30 yb*30 ;
21. m3b WITH yb*30 ;
22. t WITH m1b*0 m2b*0 m3b*0 yb*0 ;
23. m1 ON m1b*0.30 t*0 ;               !define equations
24. m2 ON m2b*0.30 t*5.0 ;
25. m3 ON m3b*0.30 t*8.0 ;
26. y ON yb*.30 m1*0 m2*0.20 m3*0.40 t*5 ;
27. m1*91.0 ;    !define disturbance variances
28. m2*84.75 ;
29. m3*75.0 ;
30. y*.001 ;
31. !y*48.923 ;
32. MODEL:       !specify analysis model; don't mention exogenous
33. m1 ON m1b*0.30 t*0 ;
34. m2 ON m2b*0.30 t*5.0 ;
35. m3 ON m3b*0.30 t*8.0 ;
36. y ON yb*.30 m1*0 m2*0.20 m3*0.40 t*5 ;
37. m1*91.0 ;
```

```
38. m2*84.75 ;
39. m3*75.0 ;
40. y*.001 ;
41. !y*48.923 ;
42. MODEL INDIRECT:
43. y IND t ;         !evaluate omnibus mediation effect
44. OUTPUT: TECH9 ;
```

Most of this syntax should be familiar based on Chapter 28. On Lines 5 and 6, I create a single sample with an N of 2,000,000. When analyzed, this sample should yield parameter values that are close to the true population values because the sample size is so large. The underlying logic is similar to the "Population Check" strategy I described in the main text of Chapter 28. On Line 30, I specify the disturbance variance for Y. I set this to zero so that the computed total variance of Y on the Mplus output will then equal $var_{REGRESSION}$. However, setting it to zero will create an error in the Mplus computational algorithms so I instead set it to a very small number that is close to zero but that will not create havoc with the Mplus algorithms. In this case, I use the value of 0.001, which is functionally zero. I do the same operation on Line 40, which must parallel Line 30.

Once the above syntax is executed, the generated output prints the covariance matrix for the first replicate. Because there is only one replicate with an N=2,000,000, the printed covariance matrix is for this sample. Here is the first portion of the printed covariance matrix on the output:

```
Covariances
              M1             M2             M3             Y              T
              ________       ________       ________       ________       ________
M1            100.011
M2              2.761        100.107
M3              2.787         12.837        100.143
Y               4.334         34.104         55.343         51.077
T               0.000          1.255          2.004          2.303          0.250
M1B            29.841          9.022          9.030         14.401          0.002
M2B             8.904         30.113          9.103         18.673          0.005
M3B             8.978          9.052         30.188         22.908          0.008
YB              8.886          8.916          8.987         35.342          0.001
```

The diagonal for the Y variable contains the variance of Y, which actually is $var_{REGRESSION}$ because var_{ERROR} was set to (near) zero. It equals 51.077 and takes into account all of the model implied variances and covariances among the predictors of Y. If var_{TOTAL} equals 100 then var_{ERROR} should be 100 – 51.077 = 48.923 and I have successfully defined the disturbance variance for Y. Note that using a disturbance variance of 48.923 implies that the squared correlation predicting Y from its predictors is 51.077/100.0 = 0.510. If I feel this is too high, I can increase the disturbance variance to a value larger than

48.923 but this will change the metric of var_{TOTAL} (and the population standard deviation of Y), which may affect how I define the values of the other parameters in the model. The alternative is to change the values of the path coefficients of any of the predictors of Y to bring them closer to zero so as to lower $var_{REGRESSION}$. In the current case, I believe a squared R of 0.510 is substantively reasonable and I move forward with it accordingly.

Given the above, I next comment out Lines 5 and 6 and replace them with Lines 7 and 8, which set the true sample size I want to simulate (N = 100) and the true number of simulation replications I want to analyze. I comment out Line 10 so I do not overwrite the prior generated temp.dat file (for reasons I mention shortly) and I also comment out Lines 30 and 40 and replace them by Lines 31 and 41 to use the disturbance variance for Y that I want to define in the population. The revised syntax is in Table 2.

Table 2: Raw Metric Syntax for Complex Model

```
1.  TITLE: Monte Carlo analysis of SEM model ;
2.  MONTECARLO:
3.  NAMES ARE t m1 m2 m3 m1b m2b m3b y yb ;
4.  CUTPOINTS = t(0);
5.  !NOBS = 2000000 ;     !sample size
6.  !NREPS = 1 ;          !number of replicates
7.  NOBS = 100 ;        !sample size
8.  NREPS = 20000 ;     !number of replicates
9.  SEED = 2222 ;        !random seed
10. !SAVE = temp.dat;
11. ANALYSIS:
12. ESTIMATOR = MLR ;
13. MODEL POPULATION:     !specify population model
14. [t*0] ;               !set mean when generating original continuous t
15. t*1 ;                 !set var when generating original continuous t
16. [m1b*0]; [m2b*0]; [m3b*0]; [yb*0];     !set means
17. m1b*100; m2b*100; m3b*100; yb*100 ;    !set variances
18. [y*0]; [m1*0]; [m2*0]; [m3*0];         !set intercepts
19. m1b WITH m2b*30 m3b*30 yb*30 ;         !set covariances
20. m2b WITH m3b*30 yb*30 ;
21. m3b WITH yb*30 ;
22. t WITH m1b*0 m2b*0 m3b*0 yb*0 ;
23. m1 ON m1b*0.30 t*0 ;                !define equations
24. m2 ON m2b*0.30 t*5.0 ;
25. m3 ON m3b*0.30 t*8.0 ;
26. y ON yb*.30 m1*0 m2*0.20 m3*0.40 t*5 ;
27. m1*91.0 ;     !define disturbance variances
28. m2*84.75 ;
29. m3*75.0 ;
30. !y*.001 ;
31. y*48.923 ;
```

```
32. MODEL:       !specify analysis model; don't mention exogenous
33. m1 ON m1b*0.30 t*0 ;
34. m2 ON m2b*0.30 t*5.0 ;
35. m3 ON m3b*0.30 t*8.0 ;
36. y ON yb*.30 m1*0 m2*0.20 m3*0.40 t*5 ;
37. m1*91.0 ;
38. m2*84.75 ;
39. m3*75.0 ;
40. !y*.001 ;
41. y*48.923 ;
42. MODEL INDIRECT:
43. y IND t ;        !evaluate omnibus mediation effect
44. OUTPUT: TECH9 ;
```

This syntax then simulates the model in Figure 2 for N = 100 with 20,000 simulated replicates.

There are interesting features of this simulation that I now highlight. Many methodologists argue that a sample size of 100 is too small for a complex model using an analysis strategy based on asymptotic theory. Let's see what the results of the simulation say about this proposition in the context of the current RET. Here is the simulation output for the chi square test of global fit:

```
Chi-Square Test of Model Fit

       Degrees of freedom                     15

       Mean                                16.437
       Std Dev                              6.033
       Number of successful computations    20000

            Proportions                  Percentiles
       Expected    Observed         Expected       Observed
          0.990       0.994            5.229          5.716
          0.980       0.987            5.985          6.515
          0.950       0.966            7.261          7.877
          0.900       0.931            8.547          9.382
          0.800       0.855           10.307         11.271
          0.700       0.771           11.721         12.808
          0.500       0.592           14.339         15.719
          0.300       0.394           17.322         18.966
          0.200       0.282           19.311         21.238
          0.100       0.160           22.307         24.548
          0.050       0.090           24.996         27.513
          0.020       0.041           28.259         30.964
          0.010       0.022           30.578         33.746
```

Recall from Chapter 28 that the average chi square value (16.437) should approximate the degrees of freedom, 15. The two values seem reasonably close. The standard deviation of the chi square values should equal the square root of double the degrees of freedom. The square root of 30 is 5.48, which roughly equals the reported standard deviation value of 6.033 on the output. In the column `Proportions Expected` at the row for a theoretical p value of 0.05, the entry in the `Proportions Observed` column is 0.09. This indicates that the chi square test is rejecting about 9% of the models when it should be rejecting only 5% of the models, the alpha level expressed in percent from. As such, the test shows a tendency to over-reject correctly specified models. Looking further at the above output, I also note that in a chi square distribution with 15 degrees of freedom, one expects the critical value of chi square that rejects 5% of the models to equal 24.996. The critical value in the observed data that did so was 27.513. Based on this result, I might consider using as a cut off a value of 27.513 in my study; or I might decide I can live with the slightly inflated tendency of the chi square test to reject models of the type I am using with N = 100. Or perhaps not in which case I would decide to increase my sample size.

Here are the simulation results for the core model parameters:

			ESTIMATES		S. E.	M. S. E.	95%	% Sig
		Population	Average	Std. Dev.	Average		Cover	Coeff
M1	ON							
M1B		0.300	0.2988	0.0978	0.0941	0.0096	0.933	0.871
T		0.000	-0.0050	1.9334	1.8937	3.7379	0.943	0.056
M2	ON							
M2B		0.300	0.3006	0.0940	0.0907	0.0088	0.936	0.898
T		5.000	5.0105	1.8709	1.8270	3.5004	0.941	0.774
M3	ON							
M3B		0.300	0.3011	0.0882	0.0853	0.0078	0.938	0.927
T		8.000	8.0075	1.7243	1.7182	2.9731	0.946	0.996
Y	ON							
YB		0.300	0.3006	0.0731	0.0698	0.0053	0.934	0.982
M1		0.000	-0.0002	0.0726	0.0694	0.0053	0.933	0.067
M2		0.200	0.1999	0.0753	0.0715	0.0057	0.931	0.782
M3		0.400	0.3999	0.0795	0.0757	0.0063	0.933	0.998
T		5.000	5.0087	1.6112	1.5519	2.5958	0.936	0.884

Scanning down the `Estimates` column, there is no disconcerting parameter bias, which is affirming. For the effect of T on M1, the true path coefficient is 0, so the `% Sig Coeff` entry reflects Type I errors. Given an alpha of 0.05, the entry should be close to 0.05. It was 0.056, which seems reasonable. The effect of M1 on Y also has a population coefficient of 0 and its `% Sig Coeff` value was 0.067 when it should be 0.05. This is a bit inflated but

also does not seem unreasonable. The remaining entries in the `% Sig Coeff` column all reflect statistical power, which generally appears to be adequate. An N of 100 thus seems sufficient in the present case *given the assumptions I made in the simulation* and given a correctly specified model. This statement takes into account asymptotic theory, bias in the parameter estimates, confidence interval coverage, Type I error rates, and statistical power. Although I did not highlight it, the simulation also provides perspectives on margins of error for an N = 100, per my discussion in Chapter 28 and on omnibus tests of the indirect effects. For the latter, the results for the current analysis were similar to those for a bootstrap analysis, with one exception; for the indirect effect T→M2→Y, the statistical power was 0.39 in the original analysis but for the bootstrap analysis it was 0.52 with slightly more accurate confidence interval coverage.

There are other features of the output I could discuss but my main focus is on showing you the mechanics of conducting a simulation for more complex models than what I considered in Chapter 28. However, there is one additional point I want to make. It may be helpful on occasions to conduct a population check on the simulated data using the method discussed in Chapter 28 so that you can explore the population data to gain a better sense of the operative dynamics in it. This involves running the syntax in Table 2 but using a single replicate with 2,000,000 cases (see Lines 5 and 6) and saving that data in the file temp.dat (see Line 10). I check at the end of the output produced by these changes for the order of the saved variables and find it to be:

```
SAVEDATA INFORMATION

  Order of variables

    M1
    M2
    M3
    Y
    T
    M1B
    M2B
    M3B
    YB
```

I then analyze the saved data using the following syntax:

```
1.  TITLE: POPULATION CHECK ;
2.  DATA:
3.  FILE IS temp.dat ;
4.  VARIABLE:
5.  NAMES ARE m1 m2 m3 y t
6.    m1b m2b m3b yb ;
```

```
ANALYSIS:
ESTIMATOR = MLR ;
MODEL:           !specify analysis model
  m1 ON m1b t ;
  m2 ON m2b t ;
  m3 ON m3b t ;
  m1 ; m2 ; m3 ;
  y ON yb m1 m2 m3 t ;
  y ;
MODEL INDIRECT:
  y IND t ;
OUTPUT: SAMP STDYX TECH4 ;
```

I can use the output from this check to gain additional perspectives on the population data. For example, the unique explained variance (squared semi-part correlation) for a predictor in a given equation, assuming large N, is

$sr^2 = (CR^2 (1-R^2))/N$

where sr^2 is the squared semi-part correlation for the predictor, CR is the critical ratio for the predictor coefficient, R^2 is the overall squared multiple correlation and N is the sample size (see Chapter 10). For the effect of T on M2 (whose statistical power was 0.77), the unique explained variance was

$[(384.930^2)(1-0.153)]/2{,}000{,}000 = 0.063$

or 6.3%. For the effect of T on M3 (whose statistical power was 0.99), the unique explained variance was

$[(653.813^2)(1-0.251)]/(2{,}000{,}000) = 0.160$

or 16.0%. For the effect of M2 on Y (whose statistical power was 0.78), the unique explained variance was

$[(389.901^2)(1-0.511)]/(2{,}000{,}000) = 0.037$

or 3.7%. And so on.

The Standardized Metric Approach

In this section, I develop Mplus syntax for the model in Figure 2 but using the standardized metric approach. The approach treats all continuous variables as having a standard deviation of 1.0. This does not mean you literally standardize your data before analyzing it. Rather, we treat all the continuous variables as having standard deviations of 1.0 as a

matter of convenience because sometimes it is easier for us to think in such a standardized metric in terms of specifying population values. I assume you have read and digested the raw metric approach described above.

I begin by specifying the population covariances between the exogenous variables, which are the baseline mediators (M1B, M2B, M3B), the baseline outcome (YB), and the treatment condition dummy variable (T). Because the treatment condition is randomized, it is uncorrelated with all other exogenous variables. I again set the correlations between all the other exogenous variables to be 0.30. I must translate these into covariances for Mplus, but because they all have standard deviations of 1.0, the covariances equal the correlations. Here is the Mplus syntax I ultimately will use for the population covariances:

```
m1b WITH m2b*0.30 m3b*0.30 yb*0.30 ;
m2b WITH m3b*0.30 yb*0.30 ;
m3b WITH yb*0.30 ;
t WITH m1b*0 m2b*0 m3b*0 yb*0 ;
```

For the effects of the treatment condition on the mediators, I again decide to make one of them (p_1) equal to zero to evaluate Type I errors, the other (p_2) to what I judge to be a medium effect, and the third (p_3) to what I judge to be a large effect. These are covariate adjusted mean differences for each mediator as a function of the treatment group minus the control group. The three posttest mediators each have a standard deviation of 1.0 as do the baseline mediators. I decide to set the regression coefficient from the baseline covariates to their posttest counterparts to 0.30 for each mediator. This is analogous to using a standardized regression coefficient of 0.30. For p_2 (the effect of T on M2), I set the value of p_2 to 0.30. Because p_2 is the mean difference between the treatment and control groups, this value corresponds to a 0.30 M2 standard deviation difference between the groups, after covariate adjustment. For p_3 (the effect of T on M3), I set its value to 0.50. This corresponds to a 0.50 M3 standard deviation difference between the groups, after covariate adjustment. The above yields the following Mplus syntax that I ultimately will use to set the values of the coefficients

```
m1 ON m1b*0.30 t*0 ;
m2 ON m2b*0.30 t*0.30 ;
m3 ON m3b*0.30 t*0.50 ;
```

I next define the disturbance variances for M1, M2, and M3. The variance of M1 is 1.0 and it is decomposed via Equation 8 to be:

$$\text{var(M1)} = 1.0 = [b_1^2\ \text{var(M1B)} + p_1^2\ \text{var(T)}] + [(2)(b_1)(p_1)\text{cov(M1B,T)}] + \text{var}(d_1)$$

If I substitute the known values of the terms on the right hand side of the equation, I obtain

$1.0 = [(.30^2)(1.0) + (0^2)(.25)] + [(2)(.30)(0)(0)] + var(d_1)$

$= [.09 + 0] + [0] + var(d_1)$

and $var(d_1) = 1.00 - .09 = 0.91$

I repeat this process for M2 and find

$var(M2) = 1.0 = [b_2^2\ var(M2B) + p_2^2\ var(T)] + [(2)(b_2)(p2)cov(M2B,T)] + var(d_2)$

which yields

$100 = [(.30^2)(1.0) + (.30^2)(.25)] + [(2)(.30)(.30)(0)] + var(d_2)$

$= [0.09 + 0.0225] + [0] + var(d_2)$

and $var(d_2) = 1.0 - 0.1125 = 0.8875$

For M3, I find

$var(M3) = 100 = [b_3^2\ var(M3B) + p_3^2\ var(T)] + [(2)(b3)(p_3)cov(M3B,T)] + var(d_3)$

which yields

$100 = [(.30^2)(1.0) + (.50^2)(.25)] + [(2)(.30)(.50)(0)] + var(d_3)$

$= [0.09 + 0.0625] + [0] + var(d_3)$

and $var(d_3) = 1.0 - 0.1525 = 0.8475$

The Mplus syntax I ultimately use to specify the population values for the disturbance variances are

```
m1*0.910 ;
m2*0.8875 ;
m3*0.8475 ;
```

Next, I turn my attention to the coefficients for the outcome, Y. I set the coefficient from YB to Y to 0.30. I set p_4 for M1→Y to zero so I can evaluate Type I errors. I set p_5 for M2→Y to 0.20 to represent what I judge to be a medium effect and p_6 for M3→Y to 0.40 to represent what I judge to be a large effect. Thus, for every one standard deviation that M2 increases, Y is predicted to increase 0.20 standard deviations and for every one standard deviation that M3 increases, Y is predicted to increase 0.40 standard deviations,

holding constant the other predictors in the equation. I set p_7 to 0.30 to reflect a medium direct effect of T on Y independent of the mediators. Here is the Mplus syntax I ultimately will use to represent these population values:

```
y ON yb*.30 m1*0 m2*0.20 m3*0.40 t*0.30 ;
```

The final step is to specify the disturbance variance for Y. As with the raw metric approach, this is complicated because there are 5 correlated predictors whose intercorrelations are determined by other parts of the model structure. To define the disturbance variance in a coherent fashion, I need to know the variance of the predicted Y scores, i.e., I need to work with the decomposition from Equation 2:

$$var_{TOTAL} = var_{REGRESSION} + var_{ERROR}$$

I previously defined var_{TOTAL} as 1.0, but I need to know the value of $var_{REGRESSION}$ given the other model population values I have chosen. Rather than derive this mathematically, I use the same Mplus shortcut I used for the raw metric approach. Table 3 presents the relevant Mplus syntax, which follows the format of Table 1.

Table 3: Generate $var_{REGRESSION}$ for Standardized Metric

```
1.  TITLE: Generate variance regression ;
2.  MONTECARLO:
3.  NAMES ARE t m1 m2 m3 m1b m2b m3b y yb ;
4.  CUTPOINTS = t(0);
5.  NOBS = 2000000 ;     !sample size
6.  NREPS = 1 ;          !number of replicates
7.  !NOBS = 100 ;        !sample size
8.  !NREPS = 20000 ;     !number of replicates
9.  SEED = 2222 ;        !random seed
10. SAVE = temp.dat;
11. ANALYSIS:
12. ESTIMATOR = MLR ;
13. MODEL POPULATION:     !specify population model
14. [t*0] ;               !set mean when generating original continuous t
15. t*1 ;                 !set var when generating original continuous t
16. [m1b*0]; [m2b*0]; [m3b*0]; [yb*0];        !set means
17. m1b*1.00; m2b*1.00; m3b*1.00; yb*1.00 ;   !set variances
18. [y*0]; [m1*0]; [m2*0]; [m3*0];            !set intercepts
19. m1b WITH m2b*0.30 m3b*0.30 yb*0.30 ;      !set covariances
20. m2b WITH m3b*0.30 yb*0.30 ;
21. m3b WITH yb*0.30 ;
22. t WITH m1b*0 m2b*0 m3b*0 yb*0 ;
23. m1 ON m1b*0.30 t*0 ;              !define equations
24. m2 ON m2b*0.30 t*0.30 ;
```

```
m3 ON m3b*0.30 t*0.50 ;
y ON yb*.30 m1*0 m2*0.20 m3*0.40 t*0.30 ;
m1*0.910 ;    !define disturbance variances
m2*0.8875 ;
m3*0.8475 ;
y*.001 ;
!y*0.694 ;
MODEL:       !specify analysis model; don't mention exogenous
m1 ON m1b*0.30 t*0 ;
m2 ON m2b*0.30 t*0.30 ;
m3 ON m3b*0.30 t*0.50 ;
y ON yb*.30 m1*0 m2*0.20 m3*0.40 t*0.30 ;
m1*0.910 ;
m2*0.8875 ;
m3*0.8475 ;
y*.001 ;
!y*0.694 ;
MODEL INDIRECT:
y IND t ;        !evaluate omnibus mediation effect
OUTPUT: TECH9 ;
```

On the output, I find that $var_{REGRESSION}$ is 0.396. If var_{TOTAL} equals 1.00 then var_{ERROR} equals 1.00 – 0.396 = 0.694.

At this point, I have all the population parameters I need and it is straightforward to execute the final simulation syntax analogous to that of Table 2 to conduct the simulation. I leave that as an exercise for you.

POWER ANALYSIS FOR GLOBAL CHI SQUARE TEST

To evaluate the statistical power of the chi square test, the model I evaluate needs to be misspecified relative to the true data-generating model in the population. To illustrate the approach, I work with the model in Figure 1 on study skills and exam performance as the true generating model but where path *c* is part of the model and is non-zero. The model I will incorrectly fit to the data omits path *c*. I will use the same population values as in my exposition of power analysis in the main text of Chapter 28 for this model, but I define the true population model so that the strength of path *c* is 7.0 (the same as the value of T→M) instead of zero. Here are the population parameter values I used in the Mplus syntax:

var(T)=0.25, var(M)=225, var(Y)=225, var(d_1)=212.75, var(d_2) = 204.75, p_1=7.0, and p_2 = 0.030, p_3=7.0

Note that everything is the same as the values I used in Chapter 28 but (a) I added a value for p_3 equal to 7.0, and (b) the value of the disturbance variance for Y, var(d_2), is reduced

from 204.75 to 186.124. This reduction is necessary because I am now explaining more of the population variance in Y by virtue of p_3 being meaningful. I used the Mplus heuristic method I described in the section on working with complex models to calculate the new value. Table 4 presents the syntax that implements the above values and that I ultimately will use to calculate the power of the chi square test.

Table 4: Step 1 of Power Analysis for Chi Square Test

```
1.  TITLE: STEP 1 CHI SQUARE POWER ANALYSIS ;
2.  MONTECARLO:
3.  NAMES ARE t m y ;
4.  CUTPOINTS = t(0);
5.  NOBS = 150 ;          !sample size
6.  NREPS = 20000 ;       !number of replicates
7.  SEED = 2222 ;         !random seed
8.  RESULTS = results.txt;
9.  ANALYSIS:
10. ESTIMATOR = MLR ;
11. MODEL POPULATION:     !specify population model
12. [t*0] ;               !set mean when generating original continuous t
13. t*1 ;                 !set var when generating original continuous t
14. [y*0]; [m*0];         !set intercepts to 0
15. y ON m*.30 t*7.0 ;    !set effect of m and t on y
16. m ON t*7.0 ;          !set effect of t on M
17. y*186.124 ;           !disturbance variance for y
18. m*212.75 ;            !disturbance variance for m
19. MODEL:                !specify analysis model
20. y ON m*.30 ;          !outcome equation - note misspecification
21. m ON t*7.0 ;          !mediation equation
23. y*186.124 ;           !disturbance variance for y
24. m*212.75 ;            !disturbance variance for m16.
25. MODEL INDIRECT:
26. y IND t ;
27. OUTPUT: TECH1 TECH9 ;
```

You should be familiar with all of the syntax except Line 8, which I explain shortly. A noteworthy feature of the syntax is that the analysis model in Lines 19 to 24 is not the same as the true generating model in Lines 11 to 18. The model in the former lines excludes path *c* whereas the true generating model includes path *c*. This introduces the specification error.

Line 8 asks Mplus to store the results of the analysis model for each of the 20,000 simulation replicates in a file that I called `results.txt` (you can use any filename you want). The data in the file are stored in free field ASCII format. The entries include for each replication the replication number, the parameter estimates, the standard errors, and a set of global fit statistics, one of which is the chi square fit statistic.

You will next execute a second program in Mplus that I show you the syntax for shortly. The program locates the p value for the chi square test for each of the 20,000 replicates stored in the `results.txt` file and then calculates the proportion of them whose p value was less than or equal to 0.05, which signifies a correct rejection of the null hypothesis. This proportion represents the power of the chi square test relative to the misspecified model.

To execute the second program, you need to know the location of the p value for the chi square test in the `results.txt` file. This can vary depending on your model. To determine the location, you first need to know the number of parameters that were estimated in the model. This is shown in the section of the output for the syntax in Table 4 that looks like this:

```
MODEL FIT INFORMATION

Number of Free Parameters                       6
```

For the current model, there were 6 parameters estimated. You then find on the output the section that describes the order in which the results are saved in the `results.txt` file. In the current case, it looks like this:

```
RESULTS SAVING INFORMATION

  Order of data

    Replication number
    Parameter estimates
     (saved in order shown in Technical 1 output)
    Standard errors
     (saved in order shown in Technical 1 output)
    Number of Free Parameters
    H0 Loglikelihood
    H1 Loglikelihood
    Akaike (AIC)
    Bayesian (BIC)
    Sample-Size Adjusted BIC
    Chi-square : Value
    Chi-square : Degrees of Freedom
    Chi-square : P-Value
    CFI
    TLI
    RMSEA : Estimate
    SRMR
```

The `Replication number` is the first entry for a given simulation replicate and is an

arbitrary sequential integer; the `Parameter estimates` are the next six entries (one for each free parameter); the `Standard errors` are the next six entries (one for each free parameter), and then after these 12 entries (not counting the replicate number), you count down the above list sequentially until you reach `Chi-square : P-Value`, which is entry number 21. Note also that the total number of entries are the repetition number, plus these 21 entries, plus the remainder of entries in the list of fit indices. For the current simulation, there are a total of 26 entries per replication, one of which is the replication number. With this information, you can program and execute the second Mplus program shown in Table 5 and that will give you the power estimate you seek.

Table 5: Step 2 of Power Analysis for Chi Square Test

```
1.  TITLE: STEP 2 CHI SQUARE POWER ANALYSIS ;
2.  DATA:
3.  FILE IS results.txt ;
4.  DEFINE:
5.  IF (f21 GT 0.05) THEN chipow = 0 ;
6.  IF (f21 LE 0.05) THEN chipow = 1 ;
7.  VARIABLE:
8.  NAMES ARE rep f1-f25 ;
9.  USEVARIABLES chipow  ;
10. ANALYSIS: TYPE = BASIC ;
11. OUTPUT:  ;
```

Line 8 tells Mplus to read in 26 entries per "case," calling the first entry (the replication number) `rep` and the remaining ones `f1`, `f2`, `f3`…`f25`. Lines 5 and 6 create a new variable that I call `chipow` based on an if statement for the chi square p value which is in the variable called `f21`. If the p value is greater than 0.05, a score of 0 is assigned to `chipow` to indicate the null hypothesis was not rejected for the replicate in question. If the p value is less than or equal to 0.05, a score of 1 is assigned to `chipow` to indicate the null hypothesis was rejected for the replicate. Line 10 tells Mplus to calculate descriptive statistics for `chipow`, with the mean value being the statistical power for the chi square test. Here is the output:

```
Means
   CHIPOW
   ________
     0.856
```

The statistical power of the global chi square test of fit for an N of 150 for the model in Figure 1 in which the misspecified model erroneously omits path *c* and where the strength of p_3 is 7 in the data generating population model is 0.856.

Parenthetically, the population RMSEA fit statistic for the misspecified model was

0.243. If I run the program on my website called *Power: SEM chi square test* and enter the desired power as 0.856 and the RMSEA index of misfit as 0.243, I obtain a required sample size of 154, which is close to what the simulation suggests. The problem with the canned program on my website (variants of which are widely used in power analysis for RMSEAs) is that I would have a hard time knowing *a priori* to use an RMSEA misfit index of 0.243 to represent the degree of misfit in my misspecified model. In fact, RMSEA rules of thumb suggest that any model with misfit greater than 0.08 should not be trusted but it turns out this standard reflects a trivial and non-consequential amount of misfit in the current case. The simulation approach is better because it allows you to zero in with greater precision on specific types of misspecification you want to explore.

POWER ANALYSIS FOR THE JOINT SIGNFICANCE TEST

In Chapter 28, I stated that it is possible to conduct a power analysis for the joint significance test using Mplus. This section shows you how to do so. It is a two-step process; One first conducts a standard simulation for one's model while saving the results in a file called `results.txt` (as I did for the chi square power analysis in the previous section). Then one gathers up and summarizes relevant information for the joint significance test in Step 2.

For Step 1, I conduct the analysis for the model in Figure 1 using the same population parameter values I used in Chapter 28:

var(T)=0.25, var(M)=225, var(Y)=225, var(d_1)=212.75, var(d_2) = 204.75, p_1=7.0, p_2 = 0.30

Table 6 presents the simulation program from Chapter 28 but with two exceptions. On Line 9, I ask Mplus to save results of the simulation in the file `results.txt`. I explained how this command works in the previous section on power analysis of the chi square test, so I do not repeat myself here. If you did not read that section, do so now. The other exception is the output line where I add `TECH1` which I explain shortly.

Table 6: Step 1 for Joint Significance Test Power Analysis

```
1.  TITLE: STEP 1 JST POWER ANALYSIS;
2.  MONTECARLO:
3.  NAMES ARE t m y ;
4.  CUTPOINTS = t(0);
5.  NOBS = 150 ;          !sample size
6.  NREPS = 20000 ;       !number of replicates
7.  SEED = 2222 ;         !random seed
8.  SAVE = temp.dat;
```

```
9.  RESULTS = results.txt;   !save results
10. ANALYSIS:
11. ESTIMATOR = MLR ;
12. MODEL POPULATION:   !specify population model
13. [t*0] ;             !set mean when generating original continuous t
14. t*1 ;               !set var when generating original continuous t
15. [y*0]; [m*0];       !set intercepts to 0
16. y ON m*.30 ;        !set effect of m on y
17. m ON t*7.0 ;        !set effect of t on M
18. y*204.75 ;          !disturbance variance for y
19. m*212.75 ;          !disturbance variance for m
20. MODEL:              !specify analysis model
21. y ON m*.30 ;        !outcome equation
22. m ON t*7.0 ;        !mediation equation
23. y*204.75 ;          !disturbance variance for y
24. m*212.75 ;          !disturbance variance for m
25. MODEL INDIRECT:
26. y IND t ;         !evaluate omnibus mediation effect
27. OUTPUT: TECH1 TECH9 ;
```

Here is the output that describes the contents of the `results.txt` file for each simulation replicate:

```
Number of Free Parameters                        6

RESULTS SAVING INFORMATION

  Order of data

    Replication number
    Parameter estimates
     (saved in order shown in Technical 1 output)
    Standard errors
     (saved in order shown in Technical 1 output)
    Number of Free Parameters
    H0 Loglikelihood
    H1 Loglikelihood
    Akaike (AIC)
    Bayesian (BIC)
    Sample-Size Adjusted BIC
    Chi-square : Value
    Chi-square : Degrees of Freedom
    Chi-square : P-Value
    CFI
    TLI
    RMSEA : Estimate
    SRMR
```

Counting up the entries in this list, there are a total of 26 entries per replication that I will input into my Step 2 Mplus program with the names `rep` and `f1` to `f25`. From this list, I need to determine the names for the two coefficients that are part of mediational chain of interest, in this case p_1 and p_2. I also need to find in the file their estimated standard errors. The `TECH1` option on the `OUTPUT` line produces output that helps me accomplish this.

There are 6 free parameters in the model and the values for these parameters, based on the above listing of the contents of the `results.txt` file, are somewhere between `f1` and `f6`, inclusive. The `TECH1` output shows a series of technical matrices in which the estimated free parameters are consecutively numbered, in this case, from 1 to 6. We need to locate a matrix on the output called the `BETA` matrix. It lists potential "causes" from the model in the columns and "effects" from the model in the rows. Here is the `BETA` matrix from the `TECH1 OUTPUT` in the current example:

```
           BETA
               M             Y             T
               ________      ________      ________
M                   0             0             3
Y                   4             0             0
T                   0             0             0
```

Parameter number 3 is T→M because it intersects the column T (causes) with the row M (effects). Parameter number 4 is M→Y because it intersects the column M with the row Y. These numbers are the parameter numbers for p_1 and p_2, which means the values of the coefficients will be in variables `f3` and `f4` in my Step 2 program when I read in the data from the `results.txt` file. Their corresponding standard errors are found by adding the number of free parameters, 6, to these numbers, `f9` and `f10`.[1]

Here is the Step 2 program to calculate power for the joint significance test:

Table 7: Step 2 for Joint Significance Test Power Analysis

```
1.  TITLE: STEP 2 JST POWER ANALYSIS ;
2.  DATA:
3.  FILE IS results.txt ;
4.  DEFINE:
5.  jst11=0 ;
6.  jst00=0 ;
```

[1] The Beta matrix is used in maximum likelihood estimation. There is another matrix called the Gamma matrix that is used in weighted least squares estimation. Working with these matrices can be complex; careful study of the Mplus user guide and technical appendix is required. I provide you in the Appendix with a work-around to using these matrices that is simpler but tedious.

```
7.  jst10=0 ;
8.  jst01=0 ;
9.  if (abs(f3/f9) GE 1.96 AND abs(f4/f10) GE 1.96) THEN jst11 = 1 ;
10. if (abs(f3/f9) LT 1.96 AND abs(f4/f10) LT 1.96) THEN jst00 = 1 ;
11. if (abs(f3/f9) GE 1.96 AND abs(f4/f10) LT 1.96) THEN jst10 = 1 ;
12. if (abs(f3/f9) LT 1.96 AND abs(f4/f10) GE 1.96) THEN jst01 = 1 ;
13. VARIABLE:
14. NAMES ARE rep f1-f25 ;
15. USEVARIABLES jst11 jst00 jst10 jst01 ;
16. ANALYSIS: TYPE = BASIC ;
17. OUTPUT:    ;
```

Lines 5 to 8 create four new variables whose names begin with `jst` (for "joint significance test") and sets each of them to a value of zero for each replicate. Lines 9 to 12 modify these 0 values depending on certain conditions. Each line (a) divides the p_1 value by its standard error, takes the absolute value of the result, and then determines if this critical ratio is larger or smaller than the critical value for a two tailed z test (alpha = 0.05) and (b) also determines if the corresponding critical ratio for p_2 is larger or smaller than its corresponding critical value. In Line 9, if both results are statistically significant, then `jst11` is changed from a value of 0 to a value of 1 for the replicate in question. In Line 10, if both results are statistically non-significant, then `jst00` is changed to 1 for the replicate. In Line 11, if the result for p_1 is statistically significant but not p_2, then `jst10` is changed to 1 for the replicate. In Line 12, if the result for p_1 is statistically non-significant but p_2 is statistically significant, then `jst01` is changed to 1 for the replicate. All other syntax should be self-explanatory.

Here is the core output that results from the syntax:

```
SAMPLE STATISTICS

       Means
          JST11         JST00         JST10         JST01
          _______       _______       _______       _______
            0.814         0.006         0.028         0.152
```

Both of the joint significance null hypotheses were rejected in the same analysis 81.4% of the time across the 20,000 simulation replicates, indicating the power of the joint significance test was 0.814. On 15.2% of the replicates, only p_2 was statistically significant, while on 2.8% of the replicates, only p_1 was statistically significant. Neither p_1 nor p_2 was statistically significant for fewer than 1% of the replicates (.6%).

LOCALIZED SIMULATIONS FOR PROPORTIONS

In this section, I show how to conduct a localized simulation for the case of a binary outcome. I illustrate the approach first using logistic regression, including a design with a continuous mediator, and then using probit regression. I also consider the case where I have a binary mediator and a binary outcome.

Comparing Two Groups on a Binary Outcome Using Logistic Analysis

Here, I show you how to conduct a localized simulation using the model in Figure 1, but where I alter the exam performance outcome to be dichotomous, 1 = student received a high pass on the exam, 0 = student did not receive a high pass. I illustrate the general logic by first omitting the mediator and then I repeat the example with the mediator. Discussing the former scenario first makes it easier to understand the latter scenario.

The simulation uses limited information SEM to compare the treatment and control groups on the proportion of students receiving a high pass. The logistic model, using sample notation, takes the form:

$$\text{Ln}[\text{Odds}(HP=1)] = a + b\,T \quad [10]$$

where HP=1 indicates obtaining a high pass, T is the treatment condition, *a* is the intercept and *b* is the logistic coefficient. In this equation, *a* is the log odds of a high pass for the control group because when T = 0, the intercept references the control group. The exponent of *a* is the odds of a high pass for the control group. The coefficient *b* is the difference between the log odds of having a high pass for the intervention group minus the log odds of having a high pass for the control group. The exponent of it is the odds ratio for the effect of the treatment group relative to the control group (see Chapter 5 for details about logistic regression)

A formula I will make use of is one that converts an odds to a probability:

$$P(HP=1) = \text{Odds}(HP=1) / (1 + \text{Odds}(HP=1)) \quad [11]$$

where P(HP=1) is the probability of a high pass and Odds(HP=1) is the odds of a high pass. The equation states that the probability of a high pass equals the odds of a high pass divided by 1 plus the odds of a high pass. A second formula converts a probability to an odds:

$$\text{Odds}(HP=1) = P(HP=1) / (1 - P(HP=1)) \quad [12]$$

where all terms are as previously defined; the odds of a high pass equals the probability of a high pass divided by one minus the probability of a high pass.

I conduct a simulation using an initial sample size of N=150 (approximately 75 per group for the treatment versus control conditions) and where the population effect size is a proportion of 0.30 for a high pass in the intervention group minus a proportion of 0.20 in the control group. To write the syntax, I need to convert the intervention probability/proportion and the control group probability/proportion to log odds. Using Equation 12 in conjunction with a natural logarithm, they are

Log odds for control group = ln[0.20/(1-0.20)] = -1.38629

Log odds for intervention group = ln[0.30/(1-0.30)] = -0.847298

and the intercept (which is the control group log odds) and the slope (which is the difference in the two log odds) are

a = -1.38629

b = -0.847298 - (-1.38629) = 0.538992

For binary outcomes, Mplus works with a threshold instead of an intercept, which is the same value as the intercept but opposite in sign. The threshold value is thus 1.38629. See Chapter 5 for elaboration.

Table 8 presents the Mplus syntax for the simulation.

Table 8: Two Group Test of Proportions

```
1. TITLE: PROPORTION SIMULATION ;
2. MONTECARLO:
3. NAMES ARE t y ;
4. CUTPOINTS = t(0);
5. NOBS = 800 ;            !sample size
6. NREPS = 20000 ;         !number of replicates
7. SEED = 2222 ;           !random seed
8. GENERATE = y(1 l) ;     !binary DV, 1 threshold with logit link
9. CATEGORICAL = y ;
10. ANALYSIS:
11. ESTIMATOR = ML ; LINK = LOGIT ;
12. MODEL POPULATION:      !specify population model
13. [t*0] ;                !set mean when generating original continuous t
14. t*1 ;                  !set var when generating original continuous t
15. [y$1*1.38629]  ;       !set threshold for y
16. y ON t*0.538992  ;     !set slope coefficient for y
17. MODEL:                 !specify analysis model
18. [y$1*1.38629] (thresh) ;    !assign a label to the y threshold
19. y ON t*0.538992 (b) ;       !assign a label to the y coefficient
20. MODEL CONSTRAINT:
```

```
21. NEW(PCTRL*.2000 PTREAT*.3000 DIFF*.1000 ) ;
22. PCTRL = exp(-thresh)/(1+exp(-thresh)) ;
23. PTREAT = exp(-thresh+1*b)/(1+ exp(-thresh+1*b)) ;
24. DIFF = PTREAT-PCTRL ;
25. OUTPUT: TECH9 ;
```

Most of the syntax should be familiar based on my initial explanation of Mplus based simulations in the main text. Line 8 generates a binary endogenous variable in the simulation. In this case the target variable is named `y` and the first entry in parentheses is the number 1 to signify there is one break point (or threshold), making `y` binary. If the number of thresholds was 2, then y would be a trichotomy. And so on. The second entry is a lower case L to tell Mplus I will later define the threshold value using a logistic function. Line 9 declares `y` as categorical, per binary regression in Mplus.

Line 15 defines the threshold for the logistic regression and, as noted, it is the opposite signed log odds intercept value for the control group. The intercept essentially reflects a proportion/probability of 0.20. Line 16 tells Mplus to regress `y` onto `t` and to set the coefficient for `t` to 0.538992. Note that if I calculate the predicted `y` for the intervention group, based on Y = a + b T, I obtain

Predicted log odds for intervention group = -1.38629 + (0.538992)(1.0) = -0.847298

which maps onto a probability/proportion of 0.30.

On Lines 18 and 19, I add labels to the two parameters in the analysis phase as opposed to data generation phase of the simulation. On Line 20, I invoke the `MODEL CONSTRAINT` option. Line 21 specifies three `NEW` parameter names, `PCTRL`, `PTREAT`, and `DIFF`, each followed by a population value it is to take on. Line 22 uses the parameter labels to define `PCTRL` to be the proportion of high passes for the control group, Line 23 defines `PTREAT` as the proportion of high passes for the intervention group, and Line 24 defines `DIFF` as the difference between the two proportions.

The logistic model is just identified so the global fit statistics are not germane. Here is the simulation output for the coefficients of interest:

```
                          ESTIMATES              S. E.      M. S. E.   95%   % Sig
                Population   Average   Std. Dev.    Average             Cover  Coeff

Y          ON
 T                 0.539     0.5511     0.4029     0.3914     0.1624   0.949  0.289
Thresholds
 Y$1               1.386     1.4129     0.3042     0.2952     0.0932   0.952  1.000


                          ESTIMATES              S. E.      M. S. E.   95%   % Sig
```

```
                Population   Average   Std. Dev.    Average             Cover  Coeff

New/Additional Parameters
  PCTRL              0.200     0.2001     0.0466     0.0458     0.0022   0.935  1.000
  PTREAT             0.300     0.2998     0.0534     0.0526     0.0029   0.939  1.000
  DIFF               0.100     0.0997     0.0713     0.0699     0.0051   0.943  0.305
```

Two lines of output are of particular interest. First is the logistic coefficient for `Y` on `T`, which is the log odds outcome difference between the intervention and control groups. The parameter estimate appears to be relatively unbiased and the confidence interval coverage for it (0.949) is reasonable. The statistical power for the test is low, 0.289. If you examine the last line of the `New/Additional Parameters` section that focuses on the proportion differences between the two groups, the effect estimate when translated into proportions also is relatively unbiased and the confidence interval coverage is reasonable (0.943). The statistical power is 0.305, which is quite close to the result for the logistic coefficient. The two estimates are slightly different because different statistical tests are being applied. The `Y` on `T` logistic coefficient uses a Wald-like test but the approach in the `New/Additional Parameters` section uses a delta method to define the standard error.

Both methods also yielded reasonable results when I evaluated Type I errors by setting the coefficient for T to zero and re-running the simulation with appropriate syntax adjustments for the zero effect. Aside from the low power and the rather large margin of error (a MOE for the proportion difference is (2)(0.07) = ±0.14), this analysis showed that the Type I error rate was reasonable.

For a sample size of 150, what is the effect size sensitivity for the effect of the intervention on obtaining a high pass? I need to engage in a trial and error process in which I systematically alter the proportion difference between the two groups and iterate through different values until I find statistical power of 0.80 for the effect size in question and N = 150. As a first step, I leave the control group proportion at 0.20 but first try the case where instead of 0.30, the intervention proportion is 0.40. To do so, I calculate the log odds that maps onto a proportion of 0.40, which is ln(0.40/(1-.40)) = -0.405465. I then subtract from this the log odds for a proportion of 0.20, which as noted above is -1.38629. The difference is 0.980825 and this becomes the value of the coefficient associated with T in the new syntax. I then rewrite Lines 15 to 25 read as follows:

```
15. [y$1*1.38629]  ;
16. y ON t*0.980825  ;
17. MODEL:
18. [y$1*1.38629] (thresh) ;
19. y ON t*0.980825 (b) ;
20. MODEL CONSTRAINT:
21. NEW(PCTRL*.2000 PTREAT*.4000 DIFF*.2000 ) ;
```

```
22. PCTRL = exp(-thresh)/(1+exp(-thresh)) ;
23. PTREAT = exp(-thresh+1*b)/(1+ exp(-thresh+1*b)) ;
24. DIFF = PTREAT-PCTRL ;
25. OUTPUT: TECH9 ;
```

Here is the output for this proportion difference that provides me the statistical power of the contrasts when the population proportion difference is 0.40 – 0.20 = 0.20:

```
                          ESTIMATES              S. E.  M. S. E.   95%   % Sig
                Population   Average   Std. Dev.    Average            Cover  Coeff
New/Additional Parameters
  PCTRL              0.200    0.2001    0.0466     0.0458    0.0022   0.935  1.000
  PTREAT             0.400    0.3999    0.0568     0.0563    0.0032   0.944  1.000
  DIFF               0.200    0.1998    0.0738     0.0727    0.0055   0.944  0.780
```

Power for `DIFF` is just under 0.80. I decide to rerun the program one more time to get even closer to power of 0.80 by setting the proportion for intervention group to 0.410 and then repeating the above process. The result was an estimated power of 0.802. The effect size sensitivity of a sample size of 150 is thus a proportion difference of approximately 0.21 or greater, which, to me, is poor sensitivity.

What sample size do I need to achieve power of 0.80 in the original analysis? Here are the results if I increase N to 600 but leave the target proportions at 0.20 and 0.30:

```
                          ESTIMATES              S. E.    M. S. E.  95%   % Sig
                Population   Average   Std. Dev.    Average            Cover  Coeff

 Y          ON
  T                 0.539    0.5411    0.1929     0.1925    0.0372   0.951  0.808

 Thresholds
  Y$1               1.386    1.3924    0.1449     0.1450    0.0210   0.953  1.000

New/Additional Parameters
  PCTRL             0.200    0.2000    0.0230     0.0230    0.0005   0.948  1.000
  PTREAT            0.300    0.2998    0.0265     0.0264    0.0007   0.948  1.000
  DIFF              0.100    0.0998    0.0351     0.0351    0.0012   0.950  0.811
```

I got lucky and stumbled onto power very close to 0.80 on the first try. Based on this analysis, if I want to detect a proportion treatment-control difference of 0.10 (or 0.30 minus 0.20), I need a sample size of about 300 per group. With this sample size, the likely margin of error for the difference will be approximately (2)(0.35) = ±0.07. Notice how much more sample size demanding this analysis is than if my outcome had been continuous. Dichotomous outcomes often are less informative and sample size demanding than continuous outcomes, so I tend not to use them unless circumstances so dictate.

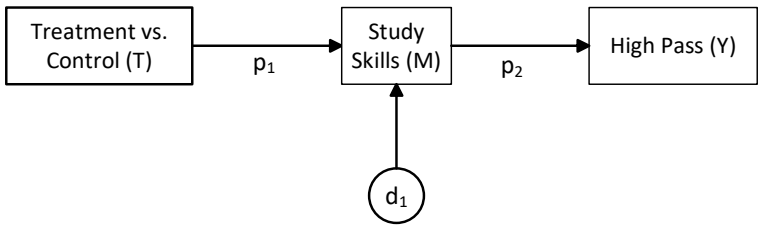
Note also that the power analysis will produce different results if the 0.10 proportion

difference is 0.20 versus 0.10, or 0.30 versus 0.20, or 0.40 versus 0.30. It turns out the most sample size demanding analysis will be one in which one of the target proportions is 0.50. Given this, many researchers conduct power analysis where they transform what they think is the true proportion difference to one where one of the target proportions is 0.50. In the current case, I would use 0.50 versus 0.40 in the simulation (or 0.60 versus 0.50, which will produce the same results).

There are many analytic strategies that statisticians have developed for comparing proportions in independent groups. The analytic method used here tends to do well except for smaller sample sizes and when percentages are near their extremes. An adjusted Bayes method described by Agresti and Caffo (2000) fares well more generally as do methods by Storer and Kim, by Beal, and by Kulinskaya, Morgenthaler, and Staudte, all of which are described in Wilcox (2021). The local simulation here suggests that the logistic regression strategy as implemented in Mplus with maximum likelihood estimation works reasonably well in terms of bias, asymptotic theory, and confidence interval coverage for the specific conditions of my study, but these other methods might provide yet more statistical power.

Mediation Analysis with a Binary Outcome Using Logistic Analysis

I now consider the case of introducing a mediator into the model, study skills. For purposes of power analysis, I place the mediator on a standardized metric that has a mean of zero and a standard deviation of 1.0. This simplifies the reporting of the power analyses, as elaborated below. The model I use is:



There is no Y disturbance term because Y is binary endogenous (see Chapter 5).

To execute the simulation, here are the population parameters I need to specify:

var(T), var(M), var(d_1), p_1, p_2, a_1, and a_2

where a_1 and a_2 are intercepts for the endogenous variables M and Y, respectively, and all other terms are as previously defined. In the main text, I noted that for a two group intervention with equal n per group vis-à-vis random assignment, the variance of T is 0.25.

Because I treat M as having a standardized metric, its population variance equals 1.0. To specify the population value of var(d_1), I must decide how much of the variance in M that I want T to account for. Suppose I decide to evaluate an effect size or value of p_1 that reflects 5% explained variance in M or an eta squared of 0.05. This means that the disturbance variance for M should equal 1 (the variance of M) minus 0.05 = 0.95. From the above information, I can calculate the value of p_1. Recall the decomposition equation

$$var(M) = p_1^2\, var(T) + var(d_1)$$

Substituting the above known values, I obtain

$$1.0 = p_1^2\, 0.25 + 0.95$$

and with algebraic manipulation I find that

$$p_1^2 = (1-.95) / 0.25 = 0.200$$

and $p_1 = 0.447$

I next specify the value of p_2. My presentation of logistic regression in Chapter 5 showed that the exponent of p_2 is the odds ratio or multiplicative factor associated with the effect of M on Y. Suppose for this link I want to explore the statistical power associated with an odds ratio of 1.75. This means that for every 1 unit that M increases (i.e., for every one standard deviation that M increases because M has a standardized metric), the odds of Y increases by a multiplicative factor of 1.75. The natural log of 1.75 is 0.559616, so I set $p_2 = 0.559616$.

For M, a_1 is the mean of M when all the predictors of it equal zero. In the present case, a_1 defines the mean on M for the control group because T is the only predictor of M and a score of 0 on T indicates the control group. If the intervention and control groups are of equal size and the overall mean of M is zero, the mean for the control group should be negative half the size of p_1 (or -.447/2 = -0.2235) and for the intervention group it should be positive half the size of p_1 (or +.447/2 = 0.2235). I set a_1 to -0.2235.

For Y, a_2 is the log odds of Y when all predictors equal zero. In the present model, because the only predictor of Y is M, a_2 is the log odds of Y when M = 0 (which is the mean of M). I set the a_2 log odds to map onto a value that reflects the probability that Y = 1 for 0.25, the midway value between the intervention and control groups. The log odds value for a_2 is thus ln[0.25/(1-0.25)] = -1.09861. The threshold equivalent is 1.0986.[2]

Given the above parameterizations, I can write the Mplus syntax. It appears in Table

[2] The parameter values yield a total effect of T on Y that is different from the previous example. I chose the values to make some points later. They key is that you understand the logic of the current example.

9. All of the syntax should be self-explanatory.

Table 9: Binary Outcome with Mediation

```
1. TITLE: PROPORTION SIMULATION WITH MEDIATION ;
2. MONTECARLO:
3. NAMES ARE t m y ;
4. CUTPOINTS = t(0);
5. NOBS = 150 ;           !sample size
6. NREPS = 20000 ;        !number of replicates
7. SEED = 2222 ;          !random seed
8. GENERATE = y(1 l) ;    !binary DVs 1 threshold using logit
9. CATEGORICAL = y ;      !declare y as categorical
10. ANALYSIS:
11. ESTIMATOR=ML ; LINK=LOGIT ;
12. MODEL POPULATION:     !specify population model
13. [t*0] ;               !set mean when generating original continuous t
14. t*1 ;                 !set var when generating original continuous t
15. [m*-0.2235];
16. m*0.95 ;
17. m ON t*0.447  ;
18. [y$1*1.0986]  ;
19. y ON m*0.559616 ;    !ln of odds ratio of 1.75
20. MODEL:                !specify analysis model
21. [m*-0.2235];
22. m*0.95 ;
23. m ON t*0.447 (p1) ;
24. [y$1*1.0986] (thresh) ;
25. y ON m*0.559616 (p2) ;  !ln of odds ratio of 1.75
26. MODEL INDIRECT:
27. y IND t ;           !evaluate omnibus mediation effect
28. OUTPUT: TECH9 ;
```

Here are the results for the key parameter estimates:

			ESTIMATES		S. E.	M. S. E.	95%	% Sig
		Population	Average	Std. Dev.	Average		Cover	Coeff
M	ON							
T		0.447	0.4487	0.1603	0.1584	0.0257	0.947	0.803
Y	ON							
M		0.560	0.5751	0.2118	0.2076	0.0451	0.953	0.816

All seems to be in order in terms of bias and confidence interval coverage. The statistical power for the effect of T on M is 0.803 and for the effect of M on Y it is 0.816. Here are the results for the overall omnibus test of mediation:

```
TOTAL, TOTAL INDIRECT, SPECIFIC INDIRECT, AND DIRECT EFFECTS

                          ESTIMATES             S. E.      M. S. E.   95%   % Sig
                Population     Average   Std. Dev. Average               Cover  Coeff

Effects from T to Y

  Indirect           0.250      0.2575     0.1349     0.1337     0.0182   0.927  0.408
  Direct effect      0.000      0.0000     0.0000     0.0000     0.0000   0.000  1.000
```

In the current case, the indirect effect is the same as the total effect because there is only one mediator and there is no direct effect of T on Y over and above the mediator. The product of the two logistic coefficients, p_1 times p_2, is (0.447)(0.560) = 0.250. The statistical power for it is 0.408, which is low. Many researchers prefer the use of bootstrapping when evaluating omnibus indirect effects and total effects of this type. When I did so, the results for T→M and M→Y were comparable to the non-bootstrap analyses but this was not the case for the indirect effect of T→M→Y. With bootstrapping, the statistical power for the indirect effect was 0.61 and the confidence interval coverage was 0.948, yielding a non-trivial improvement in statistical power relative to the non-bootstrap analysis.[3] Parenthetically, when I used the joint significance test, the statistical power for the omnibus test was 0.63.

I do not pursue the analyses further, but I could conduct other simulation analyses for the current model to explore the implications of increased sample sizes, effect size sensitivity, ensuring viable uses of bootstrapping, and reducing margins of errors.

Comparing Two Groups on a Binary Outcome Using Probit Analysis

I work with the same scenario as above for a logistic analysis but now I use a probit analysis. The underlying probit model, stated using sample notation, takes the form:

$$\text{Probit}(HP=1) = a + b\,T \qquad [13]$$

where HP=1 indicates a high pass, T is a dummy variable for the treatment condition, *a* is the probit-based intercept, *b* is the probit-based coefficient, and the term 'Probit" refers to a translating the probability into a Z score based on the cumulative distribution function (CDF) of the standard normal distribution. In this equation *a* is probit value for the control group. The coefficient *b* is the difference between the probit value for a high pass for the intervention group minus the probit value for a high pass for the control group.

[3] Mplus sometimes reports the message ERRORS DURING ESTIMATION WITH BOOTSTRAP DRAW NUMBER __, with the draw number varying to reflect the replicate where the error occurred. As long as this message is not accompanied by an elaboration of the error, all is well.

As discussed in Chapter 5, probit regression focuses on Z scores in a cumulative normal distribution. Each such Z score can be transformed into a probability, so one can think of probit regression as analyzing transformed probabilities. In R, one can convert a probability to a probit value or Z score in a cumulative standard normal distribution using the qnorm function. For example, if the probability of a high pass for the control group is 0.20, the probit value or Z score equivalent for it is

qnorm(.20) = -0.8416212

If the probability of a high pass for the intervention group is 0.30, the probit value or Z score for it is

qnorm(.30) = -.5244005

I also can convert a probit value or Z score to a probability/proportion using the pnorm function in R, like this:

pnorm(-0.8416212) = 0.20

pnorm(-.5244005) = 0.30

For this power simulation demonstration, I again conduct the power analysis simulation using a sample size of N=150 (75 per group for the treatment versus control conditions) where the population effect size is a proportion of 0.30 with a high pass in the intervention group minus a corresponding proportion of 0.20 in the control group, per the example in Chapter 28. The intercept is the probit value for the control group, -0.8416212, and for the slope it is the difference between the probit values for the two groups, (-.5244005) - (-0.8416212) = 0.317221. As with the logit analysis, Mplus works with thresholds, which is the opposite signed intercept, in this case 0.8416212.

Table 10 presents the relevant Mplus syntax for the simulation.

Table 10: Probit Simulation for Test of Proportions

```
1.  TITLE: PROPORTION SIMULATION ;
2.  MONTECARLO:
3.  NAMES ARE t y ;
4.  CUTPOINTS = t(0);
5.  NOBS = 800 ;          !sample size
6.  NREPS = 20000 ;       !number of replicates
7.  SEED = 2222 ;         !random seed
8.  GENERATE = y(1 p) ;   !binary DV, 1 threshold with probit link
9.  CATEGORICAL = y ;
```

```
10. ANALYSIS:
11. ESTIMATOR = ML ; LINK = PROBIT ;
12. MODEL POPULATION:      !specify population model
13. [t*0] ;              !set mean of treatment to 0 for cutoff
14. [t*0] ;              !set mean when generating original continuous t
15. t*1 ;                !set var when generating original continuous t
16. y ON t*0.317221 ;    !set slope coefficient for y
17. MODEL:                 !specify analysis model
18. [y$1*0.8416212] (thresh) ;    !assign a label to the y threshold
19. y ON t*0.317221 (b) ;         !assign a label to the y coefficient
20. MODEL CONSTRAINT:
21. NEW(PCTRL*.2000 PTREAT*.3000 DIFF*.1000 ) ;
22. PCTRL = exp(-thresh)/(1+exp(-thresh)) ;
23. PTREAT = exp(-thresh+1*b)/(1+ exp(-thresh+1*b)) ;
24. DIFF = PTREAT-PCTRL ;
25. OUTPUT: TECH9 ;
```

All of the syntax should be familiar. Line 8 generates the binary endogenous variable, `y`, but now the second entry in the parentheses has a *p* to indicate the use of a probit function rather than a lower case L.

The probit model is just identified so the global fit statistics are not germane. Here is the output for the coefficients of interest:

```
                        ESTIMATES              S. E.     M. S. E.   95%   % Sig
               Population   Average   Std. Dev.    Average             Cover  Coeff

Y          ON
  T                 0.317     0.3222     0.2336     0.2274     0.0546   0.947  0.296

 Thresholds
  Y$1               0.842     0.8536     0.1717     0.1673     0.0296   0.950  1.000

New/Additional Parameters
  PCTRL             0.200     0.2001     0.0466     0.0458     0.0022   0.935  1.000
  PTREAT            0.300     0.2998     0.0534     0.0526     0.0029   0.939  1.000
  DIFF              0.100     0.0997     0.0713     0.0699     0.0051   0.943  0.305
```

The results parallel closely those for the logistic analysis.

Mediation Analysis with a Binary Outcome Using Probit Analysis

This section shows you how to analyze the mediation model for the effect of a treatment versus control condition, T, on study skills, M, (measured on a 0 to 100 metric with a standard deviation of 15) which, in turn, affects the binary outcome of obtaining a high pass, Y. Given two endogenous variables in the causal model, two causal equations are implied:

$M = a_1 + p_1 T + d_1$

$Y = a_2 + p_2 M$

As with the logistic analysis, the population parameters I need to specify are:

var(T), var(M), var(d_1), p_1, p_2, a_1, and a_2

Because T is a two group dummy variable with random assignment, the variance of T is 0.25. The standard deviation of M is 15 and the variance is SD^2, so $var(M) = 15^2 = 225$. To specify the population value of var(d_1), I must first decide the effect size I want to use as my standard for estimating power for the effect of T on M. Suppose I decide 7.0 is the minimally important effect size I want to use, which is roughly half the standard deviation of M. Recall the variance decomposition equation

$var(M) = p_1^2 var(T) + var(d_1)$

If $p_1 = 7.0$, I obtain

$225 = (7^2)(0.25) + var(d_1)$

and with algebraic manipulation of these numbers, I find that

$var(d_1) = 225 - (49)(0.25) = 212.75$

I next specify the values of a_1, a_2 and p_2. The intercept a_1 is the mean M score for the control group, i.e., when T = 0. I decide based on past experience with the scale that a score of 60 is a reasonable value for it. In practice, I can rescale M by mean centering it, i.e., by subtracting 60 from the raw M scores. This transformation keeps the variance of M at 225 but it recenters M so that a score of 60 on the original metric corresponds to a score of 0 on the transformed M. Using the transformed M in the simulation, the intercept a_2 becomes the probit value or Z score associated with the proportion of individuals who have a high pass when M is at its average score on the original metric. I decide to set this equal to a probit value mapping onto a proportion of 0.20, which is -0.8416212. The threshold counterpart of this value is 0.8416212.

My presentation of probit regression in Chapter 5 noted that the probit coefficient for a predictor is the number of Z scores in a cumulative normal distribution that the outcome changes for every one unit increase in the predictor, in this case M. If I set p_2 equal to 0.05, then for every one unit increase in M, the Z score for Y increases by 0.05 units. For example, if when the transformed M equals 0 the Z score for Y is -0.8416212, then

increasing M by one unit will increase the Z score to -0.8416212 + 0.05 = -0.791621. The latter value corresponds to a change from a probability of 0.200 to a probability of 0.214 or about 0.014 probability units. A 10 unit change in M will lead to a Z score change from -0.8416212 to -0.8416212 + (10)(0.05) = -0.341621 or a change from 0.200 to 0.366 probability units or about 0.166 probability units. Suppose I decide to set $p_2 = 0.05$ for the minimally important effect size for the M→Y link.

Given the above parameterizations, I write the Mplus syntax in Table 11 and where `m` has already been transformed via mean centering. The syntax should be self-explanatory.

Table 11: Binary Outcome with Continuous Mediator and Probit Analysis

```
1.  TITLE: LOCAL SIMULATION PROBIT MEDIATION;
2.  MONTECARLO:
3.  NAMES ARE t m y ;
4.  CUTPOINTS = t(0);
5.  NOBS = 150 ;          !sample size
6.  NREPS = 20000 ;       !number of replicates
7.  SEED = 2222 ;         !random seed
8.  GENERATE = y(1 p) ;   !binary DVs 1 threshold with probit link
9.  CATEGORICAL = y ;
10. ANALYSIS:
11. ESTIMATOR = ML ; LINK = PROBIT ;
12. MODEL POPULATION:      !specify population model
13. [t*0] ;                !set mean when generating original continuous t
14. t*1 ;                  !set var when generating original continuous t
15. [m*0];                 !set intercepts to 0
16.  m ON t*7.0 ;          !set effect of t on m
17.  m*212.75 ;            !set disturbance variance for M
18.  [y$1*0.8416212] ;     !set threshold for y
19.  y ON m*.05 ;          !set effect of m on y
20. MODEL:                 !specify analysis model
21. [m*0];
22.  m ON t*7.0 ;
23.  m*212.75 ;
24.  [y$1*0.8416212] ;
25.  y ON m*.05 ;
26. MODEL INDIRECT:
27. y IND t ;            !evaluate omnibus mediation effect
28,  OUTPUT: TECH9 ;;
```

Here are the results for the key parameter estimates:

```
                          ESTIMATES                S. E.    M. S. E.    95%   % Sig
                Population   Average   Std. Dev.    Average             Cover Coeff

M          ON
  T                7.000     7.0255     2.3984     2.3701     5.7528   0.947  0.838

 Y          ON
  M                0.050     0.0514     0.0101     0.0098     0.0001   0.953  1.000
```

All seems to be in order in terms of bias and confidence interval coverage. The statistical power for the effect of T on M is 0.838 and for the effect of M on Y it is >0.999. Here are the results for the overall omnibus test of mediation:

```
TOTAL, TOTAL INDIRECT, SPECIFIC INDIRECT, AND DIRECT EFFECTS

                          ESTIMATES                S. E.    M. S. E.    95%   % Sig
                Population   Average   Std. Dev.    Average             Cover  Coeff

Effects from T to Y

  Indirect           0.350     0.3609     0.1437     0.1414     0.0208   0.947  0.793
  Direct effect      0.000     0.0000     0.0000     0.0000     0.0000   0.000  1.000
```

In the current case, the indirect effect is the same as the total effect because there is only one mediator and there is no direct effect of T on Y over and above the mediator. The product of the two coefficients, p_1 times p_2, is (7.0)(0.05) = 0.350. The statistical power for it is 0.793. Many researchers prefer to use bootstrapping when evaluating omnibus indirect effects and total effects of this type. When I did so, the results for T→M and M→Y were comparable to the non-bootstrap analyses but this was only approximately so for the indirect effect of T→M→Y. With bootstrapping, the statistical power was 0.84 and the confidence interval coverage was 0.949. When I used the joint significance test to assess the statistical significance of the indirect effect, the statistical power of it for the omnibus indirect effect test was 0.86.

I do not pursue the analyses further, but I could explore other analyses for the current model to explore the implications of increased sample sizes, effect size sensitivity, exploring viable uses of bootstrapping, Type I error rates and reducing margins of errors. I leave that as an exercise for you if you are so inclined.

Mediation Analysis with a Binary Mediator and Binary Outcome

In this simulation, I again work with the model in Figure 1 but study skills is defined as a dichotomous variable (1 = achieved high mastery of the targeted study skills, 0 did not achieve high mastery of the targeted study skills) as is exam performance (1 = obtained a high pass, 0 = did not obtain a high pass). I have two equations, each of which I will analyze

using logistic regression, although I could use probit regression instead:

$M = a_1 + p_1 T$

$\ln(\text{Odds}(Y=1)) = a_2 + p_2 M$

The population parameters I specify are:

var(T), a_1, a_2, p_1, and p_2,

The variance of T, as before, is 0.25, a_1 is the intercept for M and is a Z score that maps onto the probability of M=1 for the control group (which I will set to 0.15) or -1.7346. I set the probability of M=1 equal to 0.30 for the intervention group, which maps onto a Z score of -0.847298. The coefficient p_1 is the difference between these two Z scores which is (-0.847298) – (-1.7346) = 0.887302. I set the coefficients for a_2 and p_2 to the same values, a_2 = -1.7346 and p_2 = 0.887302. The thresholds for the two equations are each 1.7346. Table 12 has the syntax for the program. I use an N=300 in this example.

Table 12: Binary Outcome with Binary Mediator and Logit Analysis

```
1.  TITLE: BINARY OUTCOME AND BINARY MEDIATOR;
2.  MONTECARLO:
3.  NAMES ARE t m y ;
4.  CUTPOINTS = t(0);
5.  NOBS = 300 ;                !sample size
6.  NREPS = 20000 ;             !number of replicates
7.  SEED = 2222 ;               !random seed
8.  GENERATE = m(1,l) y(1 l) ;  !binary m and y 1 threshold using logit
9.  CATEGORICAL = m y ;         !declare m and y as categorical
10. ANALYSIS:
11. ESTIMATOR=ML ; LINK=LOGIT ;
12. MODEL POPULATION:           !specify population model
13. [t*0] ;                  !set mean when generating original continuous t
14. t*1 ;                    !set var when generating original continuous t
15.  [m$1*1.7346] ;
16.   m ON t*.887302  ;
17.  [y$1*1.7346]  ;
18.  y ON m*.887302    ;
19. MODEL:                       !specify analysis model
20.  [m$1*1.7346] (thresh1) ;
21.  m ON t*.887302 (p1) ;
22.  [y$1*1.7346]  (thresh2) ;
23.  y ON m*.887302 (p2)   ;
24. MODEL INDIRECT:
   25. y IND m t ;               !evaluate omnibus mediation effect
   26. OUTPUT: TECH9 ;
```

Here are the key results for the core parameters in the model:

```
                                ESTIMATES                S. E.    M. S. E.    95%    % Sig
                    Population  Average   Std. Dev.      Average               Cover  Coeff

 M          ON
  T                    0.887     0.8929     0.2936     0.2934     0.0862    0.953  0.878

 Y          ON
  M                    0.887     0.8877     0.3305     0.3282     0.1092    0.955  0.767
```

All looks reasonable in terms of bias and confidence interval coverage. The statistical power for the T→M link is 0.878 and for M→Y it is 0.767. Mplus does not print the output for the traditional indirect and total effects because the results are mathematically intractable for this scenario. However, the counterfactual effects are tractable:

```
TOTAL, INDIRECT, AND DIRECT EFFECTS BASED ON COUNTERFACTUALS (CAUSALLY-DEFINED EFFECTS,
CONDITIONAL ON ALL OTHER COVARIATES BEING ZERO)

                                ESTIMATES                S. E.    M. S. E.    95%    % Sig
                    Population  Average   Std. Dev.      Average               Cover  Coeff

Effects from T to Y

  Tot natural IE         0.022     0.0224     0.0117     0.0118     0.0001    0.920  0.401
  Pure natural DE        0.000     0.0000     0.0000     0.0000     0.0000    0.000  1.000
  Total effect           0.022     0.0224     0.0117     0.0118     0.0001    0.920  0.401
```

The statistical power for the total and indirect effects is 0.401. Here are the results for a comparable bootstrap analysis:

```
                                ESTIMATES                S. E.    M. S. E.    95%    % Sig
                    Population  Average   Std. Dev.      Average               Cover  Coeff

Effects from T to Y

  Tot natural IE         0.022     0.0225     0.0118     0.0121     0.0001    0.948  0.622
  Pure natural DE        0.000     0.0000     0.0000     0.0000     0.0000    0.000  0.000
  Total effect           0.022     0.0225     0.0118     0.0121     0.0001    0.948  0.622
```

The statistical power for the indirect and total effects improves considerably.

LATENT VARIABLE LOCALIZED SIMULATIONS

In this section, I conduct a localized simulation for a model that contains a latent variable. I use the model shown in Figure 3, which is the same as Figure 1 but where the study skills mediator has three interchangeable indicators.

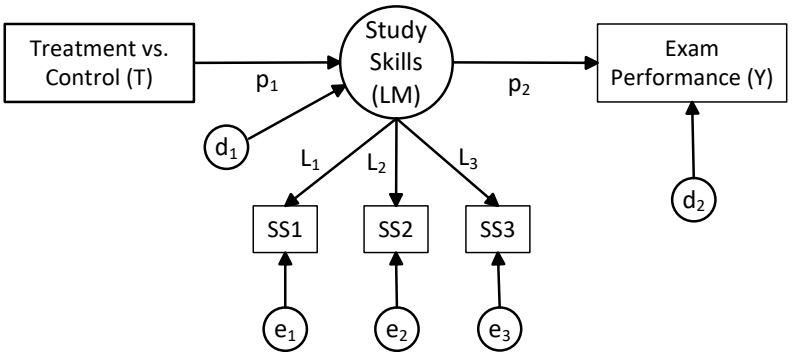


FIGURE 3. Simulation Example with Latent Variables

I signify the paths from the latent variable to the indicators by Ls because the paths are analogous to (unstandardized) factor loadings. Measurement errors are indicated by *e* and are distinct from disturbance variables which usually are designated as *d*. With interchangeable indicators, it is common to conceptualize the error variances of indicators as reflecting measure unreliability. The study skill indicators all are measured on a 0 to 100 scale with higher scores indicating better skills. The scales usually have standard deviations of about 15. The outcome is performance on the final math exam with scores typically ranging from 0 to 100 with a standard deviation usually near 15.

When I conduct power analyses with latent variables, I often use a standardized metric approach because it tends to be more intuitive with factor loadings. Given this, the variances of the observed variables SS1, SS2, SS3, and Y are set to 1.0 as is the variance of the latent mediator, LM. I make one exception to this parameterization for reasons I explain shortly. The population parameters I need to specify for the model in Figure 3 are:

var(T), L_1, L_2, L_3, var(e_1), var(e_2), var(e_3), var(d_1), var(d_2), p_1, p_2

but doing so in ways that respect var(SS1), var(SS2), var(SS3), var(LM) and var(Y) all being equal to 1.0.

As in previous programs, var(T) is set to 0.25 to reflect the variance of a dummy variable with dummy coding for two same-sized groups vis-à-vis random assignment. I want the variance of LM to equal 1.0 because it is continuous and such an assignation honors the spirit of a standardized metric strategy. However, a complication results because traditional practice is to fix one of the loadings of the latent variable indicators, in this case L_1, to 1.0 to define the metric of LM. If LM has a variance of 1.0 and L1 is fixed at 1.0, then it follows that the variance of SS1 must be larger than 1 unless there is no measurement error in SS1, which usually is unrealistic. This is because the variance of SS1 is an additive

function of the LM variance plus the error variance for SS1, as follows:

$var(SS1) = L_1^2\ var(LM) + var(e_1)$

Substituting the values of L_1 and var(LM) into this equation, I obtain

$var(SS1) = (1^2)\ (1) + var(e_1)$

Suppose I want to model a case where SS1 has a reliability of 0.80. Given this, I need to determine the value for which $(1^2)\ (1)$ represents 80% of it, which is $[(1^2)(1)]/.80 = 1.25$. This yields

$1.25 = (1^2)\ (1) + var(e_1)$

and with algebraic manipulation, I find

$1.25 - (1^2)\ (1) = var(e_1) = 0.25.$

Thus, to mimic a situation where I fix L_1 to 1.0 and where I want SS1 to have reliability of 0.80, I need to set $var(e_1)$ to 0.25 and let var(SS1) = 1.25, not 1.0. This results in the following population parameterizations so far:

var(T)=0.25, L_1=1.0, $var(e_1)$=0.25, and, by implication, var(LM)=1.0 and var(SS1)=1.25

Continuing with the measurement model, I define L_2, L_3, $var(e_2)$ and $var(e_3)$ such that both var(SS2) and var(SS3) equal 1.0 per my standardized metric strategy and so that the reliabilities of SS1 and SS2 also equal 0.80. Note that I do not have to set all indicators to have the same reliabilities; I just do so here because that is what I think is the case.

It turns out that if the variance of the latent variable is 1.0, and the variance of its indicator also is 1.0, the value of the loading, L, for the indicator, will equal the square root of the indicator reliability assuming (a) there are no correlated errors among the indicators (which there are not) and (b) there are no cross loadings with other latent variables (which there are not). This means that

$L_2 = L_3 = \sqrt{0.80} = 0.894427$

Yet another regularity for the above measurement scenario is that the measure error variance will equal the unreliability of the measure, or one minus its reliability. This yields

$var(e_2) = var(e_3) = 1 - 0.80 = 0.20$

Updating our population parameterizations thus far, we have

var(T)=0.25, L_1=1.0, L_2=0.894427, L_3=0.894427, var(e_1)=0.25, var(e_2)=0.20, var(e_3)=0.20, and, by implication, var(SS1)=1.25, var(SS2)=1.0, var(LM)=1.0, and var(SS3)=1.0

Continuing the parameter designation process, LM is a function of T. I must decide how much of the variation in LM that I want T to account for. Suppose I choose a value near Cohen's medium effect size, namely an eta squared or proportion of explained variance of 0.069. Given that var(LM) = 1, the disturbance variance, var(d_1), must be 1 - 0.069 = 0.931. I use this information to calculate the value of p_1. Using Equation 28.6:

$$var(LM) = p_1^2 \, var(T) + var(d_1)$$

I substitute into the equation the known values:

$$1.0 = p_1^2 \, (0.25) + 0.931$$

and with algebraic manipulation, I obtain

$$(1.0 - 0.931)/0.25 = 0.276 = p_1^2$$

Taking the square root of 0.276, I obtain

$$p_1 = 0.525357$$

To set the values of p_2 and var(d_2), I must decide what proportion of the variance I want LM to account for in Y. Suppose I decide on an eta squared of 0.09 or 9% explained variance. Because LM is the only predictor of Y, this means var(d_2) = 1 – 0.09 = 0.91. To determine the value of p_2, I use the equation

$$var(Y) = p_2^2 \, var(LM) + var(d_2)$$

I substitute into the equation the known values:

$$1.0 = p_2^2 \, (1.00) + 0.910$$

and with algebraic manipulation, I obtain

$$(1.0 - 0.910)/1.00 = 0.090 = p_2^2$$

Taking the square root of 0.090, I obtain

$p_2 = 0.30$

With these parameter values in hand, I can write the Mplus syntax for the power simulation, which is shown in Table 13.

Table 13: Simulation with Latent Variable

```
1. TITLE: LATENT VARIABLE POWER ANALYSIS;
2. MONTECARLO:
3. NAMES ARE t ss1 ss2 ss3 y ;
4. CUTPOINTS = t(0);
5. NOBS = 150 ;                  !sample size
6. NREPS = 20000 ;               !number of replicates
7. SEED = 2222 ;                 !random seed
8. ANALYSIS:
9. ESTIMATOR = MLR ;
10. MODEL POPULATION:   !specify population model
11. [t*0] ;             !set mean when generating original continuous t
12. t*1 ;               !set var when generating original continuous t
13. [ss1*0]; [ss2*0] ; [ss3*0] ;      !indicator intercepts
14. ss1*0.25 ; ss2*0.29 ; ss3*0.29 ;  !indicator error variances
15. LM BY ss1@1 ss2*0.89442 ss3*0.89442 ;
16. [LM@0] ; LM*0.931 ;               !intercept of LM and LM disturbance var
17. [y*0]; y*0.91 ;                   !y intercept and y disturbance var
18. y ON LM*0.30 ;                    !effect of LM on Y
19. LM ON t*0.525357 ;                !effect of t on M
20. MODEL:                            !specify analysis model
21. [ss1*0]; [ss2*0] ; [ss3*0] ;
22. ss1*0.25 ; ss2*0.2 ; ss3*0.2 ;
23. LM BY ss1@1 ss2*0.89442 ss3*0.89442 ;
24. [LM@0] ; LM*.931 ;
25. [y*0]; y*0.91 ;
26. y ON LM*0.30 ;
27. LM ON t*0.525357 ;
28. MODEL INDIRECT:
29. y IND t ;                   !evaluate omnibus mediation effect
30. OUTPUT: TECH9 ;
```

Most of the syntax should be familiar and at this point is self-explanatory. I set the measurement intercepts to zero, which has the effect of treating SS1, SS2, and SS3 as having means of zeros, which is consistent with the standardized metric strategy. On Line 15, I specify the indicators for LM and their loadings. On Line 16, I fix the intercept for LM to zero because it is not a formal part of the model that has any implications for model fit or for meaningful parameter tests.

I do not delve into all of the output details which should already be familiar to you.

Here is the output for the core parameters of interest:

		ESTIMATES		S. E.	M. S. E.	95%	% Sig
	Population	Average	Std. Dev.	Average		Cover	Coeff
LM BY							
SS1	1.000	1.0000	0.0000	0.0000	0.0000	1.000	0.000
SS2	0.894	0.8961	0.0587	0.0576	0.0035	0.946	1.000
SS3	0.894	0.8964	0.0586	0.0576	0.0034	0.944	1.000
LM ON							
T	0.525	0.5261	0.1679	0.1652	0.0282	0.944	0.887
Y ON							
LM	0.300	0.3012	0.0837	0.0822	0.0070	0.941	0.953
Residual Variances							
SS1	0.250	0.2465	0.0449	0.0443	0.0042	0.848	1.000
SS2	0.200	0.1975	0.0359	0.0355	0.0013	0.937	1.000
SS3	0.200	0.1970	0.0359	0.0354	0.0013	0.938	1.000
Y	0.910	0.8970	0.1050	0.1030	0.0112	0.927	1.000
LM	0.931	0.9180	0.1350	0.1325	0.0184	0.928	1.000

All appears to be in order in terms of bias and confidence interval coverage. The statistical power for the effect of T on LM is 0.887 and for the effect of LM on Y, it is 0.953. The margins of error for the loadings are about ±0.12. Here is the output for the indirect effect of T→LM→Y:

TOTAL, TOTAL INDIRECT, SPECIFIC INDIRECT, AND DIRECT EFFECTS

		ESTIMATES		S. E.	M. S. E.	95%	% Sig
	Population	Average	Std. Dev.	Average		Cover	Coeff
Effects from T to Y							
Total	0.158	0.1581	0.0671	0.0662	0.0045	0.925	0.713
Tot indirect	0.158	0.1581	0.0671	0.0662	0.0045	0.925	0.713

The statistical power for the effect is 0.71. For bootstrapping, I found the statistical power to be 0.77.

MULTIPLE GROUP LOCALIZED SIMULATIONS

In this section, I demonstrate how to conduct a power analysis simulation for a multi-group SEM. I use the model in Figure 1 as applied to two groups defined by biological sex, males and females. For females, I use the same population parameterizations as those derived in the main chapter text:

var(T)=0.25, var(M)=225, var(Y)=225, var(d_1)=212.75, var(d_2)=204.75, p_1=7.0, p_2=0.30

For males, I use the same values but with the modification that the treatment T has no effect on M. This means var(d_1) = 225 and p_1 = 0, yielding

var(T)=0.25, var(M)=225, var(Y)=225, var(d_1)=225, var(d_2)= 204.75, p_1=0, and p_2=0.30

I will focus on the statistical power associated with the contrast comparing p_1 for females versus males and the difference in the total effect of T on Y for the two groups. Table 14 presents the Mplus syntax for the simulation. The programming is based on multigroup analyses described in Chapter XX. Recall that in multiple group programming, there are two variants of the population `MODEL` command, `MODEL` and `MODEL` followed by a label. The unqualified `MODEL` command describes the overall model to be estimated for each group. `MODEL` followed by a label describes differences between the overall model and the model for the group designated by the label, i.e., specialized group parameter qualifications in the model that I want to introduce. Although it is not necessary, I make it a habit of describing the full model in the individual groups so I can see explicitly what I am up to when changing group parameters and to make sure I override certain unwanted defaults that Mplus sometimes invokes. In addition to these population specifications, the group specific `MODEL` variants comprise the model analyses section of the simulation syntax, per lines 41 to 52 below.

Table 14: Multigroup Simulation

```
1. TITLE: MULTIGROUP SIMULATION ;
2. MONTECARLO:
3. NAMES ARE t m y ;
4. NGROUPS=2 ;
5. CUTPOINTS = t(0) | t(0) ;
6. NOBS = 125 125 ;        !sample size
7. NREPS = 20000 ;         !number of replicates
8. SEED = 2222 ;           !random seed
9. ANALYSIS:
10. ESTIMATOR = MLR ;
11. MODEL POPULATION:    !specify population model
12. [t*0] ;              !set mean when generating original continuous t
13. t*1 ;                !set var when generating original continuous t
14. [y*0]; [m*0];        !intercepts to 0
15. y ON m*.30 ;         !effect of m on y
16. m ON t*7.0 ;         !effect of t on m
17. y*204.75 ;           !disturbance variance for y
18. m*212.75 ;           !disturbance variance for m
19. MODEL POPULATION-G1:
```

```
20. [t*0] ;              !mean of treatment to 0 for cutoff
21. t*0.25 ;             !define var of treatment variable
22. [y*0]; [m*0];        !intercepts to 0
23. y ON m*.30 ;         !effect of m on y
24. m ON t*7.0 ;         !effect of t on m
25. y*204.75 ;           !disturbance variance for y
26. m*212.75 ;           !disturbance variance for m
27. MODEL POPULATION-G2:
28. [t*0] ;              !mean of treatment to 0 for cutoff
29. t*0.25 ;             !define var of treatment variable
30. [y*0]; [m*0];        !intercepts to 0
31. y ON m*.30 ;         !effect of m on y
32. m ON t*0  ;          !effect of t on m
33. y*204.75 ;           !disturbance variance for y
34. m*225.0 ;            !disturbance variance for m
35. MODEL :
36. [y*0]; [m*0] ;
37. y ON m*.30   ;
38. m ON t*7.0   ;
39. y*204.75 ;
40. m*212.75 ;
41. MODEL G1:
42. [y*0]; [m*0];
43. y ON m*.30 (p2g1) ;
44. m ON t*7.0 (p1g1) ;
45. y*204.75 ;
46. m*212.75 ;
47. MODEL G2:
48. [y*0]; [m*0];
49. y ON m*.30 (p2g2) ;
50. m ON t*0  (p1g2) ;
51. y*204.75 ;
52. m*225.00 ;
53. MODEL INDIRECT:
54. y IND t ;        !evaluate omnibus mediation effect
55. MODEL CONSTRAINT :
56. NEW (DIFF1*7 DIFF2*2.1) ;
57. DIFF1 = p1g1 - p1g2 ;             !p1 group difference
58. DIFF2 = p1g1*p2g1 - p1g2*p2g2 ;   !total effect group difference
59. OUTPUT: TECH9 ;
```

Most of the syntax should be familiar. Line 5 specifies the cutpoints for the treatment dummy variable for each group per standard simulation practice discussed in Chapter 28, with the group-specific specifications separated by a "|". Line 6 specifies the sample size for each group. In this case, they are equal (n = 125 per group). On Lines 55 to 58, I use `MODEL CONSTRAINT` for the two contrasts I am interested in for the analysis.

Here is the output for the overall chi square test of fit, which is the combined group

chi square that is applicable to multi-group solutions:

```
Chi-Square Test of Model Fit

        Degrees of freedom                        2

        Mean                                  2.056
        Std Dev                               2.056
        Number of successful computations     20000

             Proportions                  Percentiles
        Expected    Observed         Expected       Observed
           0.990       0.991            0.020          0.022
           0.980       0.980            0.040          0.041
           0.950       0.950            0.103          0.103
           0.900       0.902            0.211          0.215
           0.800       0.804            0.446          0.460
           0.700       0.709            0.713          0.739
           0.500       0.511            1.386          1.423
           0.300       0.311            2.408          2.486
           0.200       0.206            3.219          3.283
           0.100       0.106            4.605          4.732
           0.050       0.055            5.991          6.215
           0.020       0.023            7.824          8.130
           0.010       0.011            9.210          9.466
```

All seems to be in order. The mean chi square value (2.056) is close to the value of its degrees of freedom (2) and the square root of double the degrees of freedom is close to the value of the standard deviation of the chi square statistic. The expected and observed values in the respective distributions are close, all of which suggests a well behaved solution.

Here are the results for the core parameter values of interest to me:

```
                          ESTIMATES                 S. E.   M. S. E.  95%   % Sig
                Population     Average   Std. Dev.  Average            Cover  Coeff

Group G1

 Y        ON
  M              0.300      0.3002     0.0859     0.0845     0.0074   0.941  0.935

 M        ON
  T              7.000      7.0302     2.6212     2.5926     6.8715   0.946  0.769

Group G2

 Y        ON
  M              0.300      0.2995     0.0867     0.0843     0.0075   0.938  0.932
```

```
                              ESTIMATES                S. E.   M. S. E.  95%    % Sig
                  Population     Average   Std. Dev.  Average            Cover  Coeff

 M        ON
  T                  0.000      0.0082     2.6639     2.6659     7.0961   0.949  0.051

New/Additional Parameters
  DIFF1              7.000      7.0220     3.7328     3.7226    13.9332   0.948  0.472
  DIFF2              2.100      2.1090     1.3059     1.3205     1.7053   0.951  0.338
```

The statistical power for the group differences in p_1 is in the `New/Additional Parameters` section and is 0.472. For group differences in the total effects, the statistical power is 0.338. I also can take note of the magnitude of the margins of errors and other facets of localized power analysis described earlier and in Chapter 28..

MISSING DATA LOCALIZED SIMULATIONS

The simulations described thus far have used complete data. It is possible to incorporate missing data into your simulations. I illustrate doing so using the study skills example in Figure 1 with both M and Y as continuous variables. Suppose I think a likely missing data scenario is where my data are missing completely at random (MCAR) and I have about 15% missing data on M and 5% missing data on Y. I will first evaluate statistical power when using FIML to adjust for it and then I will evaluate addressing missing data using listwise deletion (see Chapter 26).

To conduct the simulation based on FIML, I will use the same parameter values and sample size I used in the main text in section on *Choosing Parameter Values* in the context of the model in Figure 1. Here are the parameter values I used in that section:

var(T)=0.25, var(M)=225, var(Y)=225, var(d_1)=212.75, var(d_2) = 204.75, p_1=7.0, p_2 = 0.30

The Mplus syntax I used is shown in Table 15 but with the addition of two statements, Lines 4a and 4b, that introduce the missing data.

Table 15: Local Simulation with Missing Data

```
1. TITLE: LOCAL SIMULATION 1;
2. MONTECARLO:
3. NAMES ARE t m y ;
4. CUTPOINTS = t(0);
4a. PATMISS = m(.15) y(.05) ;
4b. PATPROBS = 1.0 ;
5. NOBS = 150 ;          !sample size
6. NREPS = 20000 ;       !number of replicates
7. SEED = 2222 ;         !random seed
```

```
8. !SAVE = temp.dat;
9. ANALYSIS:
10. ESTIMATOR = MLR ;
11. MODEL POPULATION:   !specify population model
12. [t*0] ;             !set mean when generating original continuous t
13. t*1 ;               !set var when generating original continuous t
14. [y*0]; [m*0];       !set intercepts to 0
15. y ON m*.30 ;        !set effect of m on y
16. m ON t*7.0 ;        !set effect of t on M
17. y*204.75 ;          !disturbance variance for y
18. m*212.75 ;          !disturbance variance for m
19. MODEL:              !specify analysis model
20. y ON m*.30 ;        !outcome equation
21. m ON t*7.0 ;        !mediation equation
22. y*204.75 ;          !disturbance variance for y
23. m*212.75 ;          !disturbance variance for m
24. MODEL INDIRECT:
25. y IND t ;           !evaluate omnibus mediation effect
26. OUTPUT: TECH9 ;
```

All lines other than 4a and 4b should be familiar. These latter two lines define the nature and amount of missing data. Missing data in the Mplus simulation package are by default MCAR but this can be overridden as desired (see Muthén, Muthén & Asparouhov, 2016, for examples). For each endogenous variable, the `PATMISS` command specifies the proportion of scores on the variable that I want to be randomly missing. For `m` the value of 0.15 on Line 4a means I want 15% of the cases on it to have missing values. For `y` I request 5% missing data. If a variable is not listed, it is assumed to have no missing data. As I explain in the missing data simulation document on my web page, you can have multiple patterns of missing data but describing how to do so is beyond the scope of what I want to show you here. The `PATPROBS` command on Line 4b specifies the proportion of the total number of cases in the study that you want to have the missing data pattern(s) applied to. In this case, there is only one pattern and it will be applied across the total N, so I enter a value of 1.00. For a more detailed explanation of the use of these two commands, see the document on my web page for missing data simulations. The bottom line is that for the current RET, I will have data that are MCAR with 15% missing data on `m` and 5% missing on `y` by virtue of including lines 4a and 4b.

The results for the chi square statistic of global fit were favorable in terms of asymptotic theory and the applicability of the chi square statistic. Here is the output for the core model parameters, first with no missing data from my analysis at the outset of the chapter and then with the missing data from the current section:

		ESTIMATES		S. E.	M. S. E.	95%	% Sig
	Population	Average	Std. Dev.	Average		Cover	Coeff
Y ON							
M	0.300	0.3001	0.0789	0.0772	0.0062	0.939	0.966
M ON							
T	7.000	7.0181	2.3648	2.3693	5.5924	0.948	0.842

and here are the results with missing data:

		ESTIMATES		S. E.	M. S. E.	95%	% Sig
	Population	Average	Std. Dev.	Average		Cover	Coeff
Y ON							
M	0.300	0.3006	0.0861	0.0844	0.0074	0.940	0.931
M ON							
T	7.000	7.0175	2.5485	2.5435	6.4949	0.946	0.785

The missing data slightly lowered statistical power, but not by much. In this case, the effects of the missing data on statistical power are modest.

What would the simulation results be if I used listwise deletion instead of FIML? Evaluating listwise deletion in the Mplus Monte Carlo package requires two steps because you must use the Mplus external simulation option. In the first step, you generate separate data files for each of the Monte Carlo replications and store those files on your computer. For the second step, you ask Mplus to analyze each data file separately and then combine the results in a format similar to other Monte Carlo simulations. Table 16 shows the syntax I used to generate the data files (step 1).

Table 16: Local Simulation with Missing Data

```
1. TITLE: LOCAL SIMULATION 1;
2. MONTECARLO:
3. NAMES ARE t m y ;
4. CUTPOINTS = t(0);
4a. PATMISS = m(.15) y(.05) ;
4b. PATPROBS = 1.0 ;
5. NOBS = 150 ;           !sample size
6. NREPS = 5000 ;        !number of replicates
7. SEED = 2222 ;         !random seed
8a. REPSAVE = ALL ;      ! Save the files
8b. SAVE = rep*.dat ;    ! Name of files to save
9. ANALYSIS:
10. ESTIMATOR = MLR ;
11. MODEL POPULATION:   !specify population model
12. [t*0] ;             !set mean when generating original continuous t
```

```
13. t*1 ;               !set var when generating original continuous t
14. [y*0]; [m*0];       !set intercepts to 0
15. y ON m*.30 ;        !set effect of m on y
16. m ON t*7.0 ;        !set effect of t on M
17. y*204.75 ;          !disturbance variance for y
18. m*212.75 ;          !disturbance variance for m
19. MODEL:              !specify analysis model
20. y ON m*.30 ;        !outcome equation
21. m ON t*7.0 ;        !mediation equation
22. y*204.75 ;          !disturbance variance for y
23. m*212.75 ;          !disturbance variance for m
24. MODEL INDIRECT:
25. y IND t ;           !evaluate omnibus mediation effect
26. OUTPUT: TECH9 ;
```

The syntax is identical to that of Table 16 with some exceptions. First, I changed the number of replications in Line 6 to 5000 so that the number of files littering my computer is reduced (the lower number does not affect results in any notable way in this case). Second, I added Lines 8a and 8b to tell Mplus to save the generated data files (Line 8a) and what names to give to the files (Line 8b). The naming convention is the same as for generating imputed data sets that I described in Chapter 26. Each data set will be named "rep" (you can use any name you want per Line 8b) followed by a number from 1 to 5000 (the number of requested data sets) and the tag will be dat (you can use any tag designation you want per Line 8b). Mplus will also generate a file called replist.dat, which is named using the name you gave to each data set (in this case "rep") followed by the word "list.dat" instead of a number with the tag "dat." This file contains the list of the names of all the generated data sets in a single column for input into the step 2 program. The files will be in the same folder that the input syntax is stored in because I do not specify a folder path.

The data are written to the data files in an order that Mplus tells you at the end of the step 1 output, like this:

```
SAVEDATA INFORMATION

  Order of variables

    M
    Y
    T
    PATTERN

  Save file
    rep*.dat
```

An additional variable is added by Mplus called `PATTERN` that I ignore at step 2.

Table 17 presents the syntax for the step 2 analysis.

Table 17: Local External Simulation: Step 2

```
1. TITLE: LOCAL SIMULATION DATA ANALYSIS ;
2. DATA:
3. FILE = replist.dat ;
4. TYPE = MONTECARLO ;
5. LISTWISE = ON ;
6. VARIABLE:
7. NAMES ARE m y t PATTERN ;
8. USEVARIABLES ARE m y t  ;
9. MISSING = ALL(999);
10. ANALYSIS:
11. ESTIMATOR = MLR;
12. MODEL:        ! specify analysis model
13. y ON m*.30 ;        !outcome equation
14. m ON t*7.0 ;        !mediation equation
15. y*204.75 ;          !disturbance variance for y
16. m*212.75 ;          !disturbance variance for m
17. MODEL INDIRECT:
18. y IND t ;      ! analyze indirect effects
19. OUTPUT:
```

Most of the syntax is self-explanatory. Line 4 tells Mplus to do a Monte Carlo simulation and Line 5 tells Mplus to use listwise deletion. Note the input file in Line 3 is the replist.dat file. This tells Mplus where to find all the separate externally generated files.

The fit indices and parameter estimates across the 5,000 replications were well behaved which is to be expected given MCAR. Here are the results for the key model parameters:

		ESTIMATES		S. E.	M. S. E.	95%	% Sig
	Population	Average	Std. Dev.	Average		Cover	Coeff
Y ON							
M	0.300	0.2999	0.0868	0.0855	0.0075	0.941	0.931
M ON							
T	7.000	7.0066	2.6297	2.6217	6.9140	0.942	0.760
Residual Variances							
M	212.750	208.7959	26.9334	26.2277	740.8978	0.917	1.000
Y	204.750	202.1090	26.3337	25.4252	700.3012	0.921	1.000

The results are similar to those for FIML, suggesting the choice of FIML versus listwise deletion in this case matters little. This will not always be the case.

LOCALIZED SIMULATIONS WITH NON-NORMALITY

The primary approach Mplus uses to evaluate the effects of non-normality on statistical power vis-à-vis local simulations is by mixing together two or more normal distributions that then results in a non-normal distribution in the mixed distribution. The mixing is typically (but not always) applied to the disturbance term of the outcome or endogenous variable.

In this next section, I show you how to explore the implications of the distribution that results when you mix different normal distributions using R. The code I provide allows you to make more informed decisions about how you want to mix normal distributions when you conduct a formal simulation in Mplus that uses mixed normal distributions to create non-normality.

For symmetric unimodal distributions, excessive positive kurtosis reflects the presence of heavy tails and peakedness relative to the normal distribution whereas excessive negative kurtosis suggests light tails and flatness. Many statistical tests face challenges when confronted with heavy tailed distributions (Wilcox, 2021), so you might want to be sure to explore cases where positive kurtosis is likely to be present. Here is R syntax that combines two normal distributions, computes the skewness and kurtosis of the mixed distribution, and plots the densities of the distribution and superimposes a normal distribution onto the plot:

```
library(distr)
library(moments)
# Construct the mixed distribution
myMix <- UnivarMixingDistribution(Norm(mean=0, sd=1),
                                  Norm(mean=0, sd=3),
                                  mixCoeff=c(0.9,0.1))
# create function to sample cases
rmyMix <- r(myMix)
# Sample a million cases
x <- rmyMix(1e6)
# Generate a normal distribution to overlay
xnorm<-rnorm(1000000,mean=mean(x),sd=sd(x))
#create plot
plot(density(x),col='blue',lwd=2)
lines(density(xnorm),col='red',lwd=2)  #omit to not superimpose normal
#report descriptive statistics
skewness(x)
kurtosis(x)-3
mean(x)
var(x)
sd(x)
```

In this code, I use the R package called distr to accomplish the mixing. On Lines 4 and 5, I specify the two normal distributions I want to mix. The term `Norm` indicates a normal distribution and it is followed by specification of the mean and standard deviation of the distribution. Line 4 specifies the first distribution and Line 5 specifies the second distribution. In this case, the means of each distribution are zero but the standard deviation of the second distribution is three times larger than that of the first distribution. Line 6 specifies the proportion of cases from the first normal distribution (0.90) that should comprise the mixed distribution and the corresponding proportion for the second normal distribution (0.10). You change these values to match the respective proportions you want to use, but they must sum to 1.0. In the current example, 90% of the cases in the mixed distribution will have been sampled from the first normal distribution and 10% from the second normal distribution. If I wanted the split to be 50-50, I would change the values to `mixCoeff=c(0.5,0.5)`. The Line `x <- rmyMix(1e6)` says to sample one million cases from the mixture population (using scientific notation; if you want to sample two million, change change `1e6` to `2e6`). I use a large N so as to minimize sampling error, allowing me to better appreciate what is happening at the population level when normal distributions are mixed. The remaining lines plot the scores for the mixed normal distribution, overlays a normal distribution on the plot, and calculate descriptive statistics using the R package *moments*. The skewness and kurtosis indices are defined so that the standard normal distribution has a skewness and kurtosis value of zero.

Figure 4 shows the density plot that results when I execute the above R code with the mixed distribution in blue and a normal distribution with the same mean and variance as the mixed distribution in red. The mixed distribution in this case has skewness of -0.02 and kurtosis of 5.39. Note that it is more peaked and the tails are heavier than the corresponding normal distribution. Figure 5 shows a 75% and 25% mix from a normal distribution with a mean of 0 and a SD of 2.0 mixed with a normal distribution with a mean of 1 and a SD of 2.0. The skewness and kurtosis of the mixed distribution are 1.04535 and 1.84266, respectively. I omit the normal distribution from this plot to accentuate the shape of the mixed distribution. I eliminate the normal plot by adding the # symbol to code line that generates the normal overlay, which then comments out the entire line, like this

```
#lines(density(xnorm),col='red',lwd=2)
```

The strategy you use for creating skew is to create a “base” distribution and then shift the second distribution to the right or left by altering its specified mean depending on the desired direction of skewness. If you shift it considerably the result will be a bimodal distribution. With practice, the amount and types of non-normality you can create using a mixture distribution strategy is considerable (see Ray & Lindsay,2005).

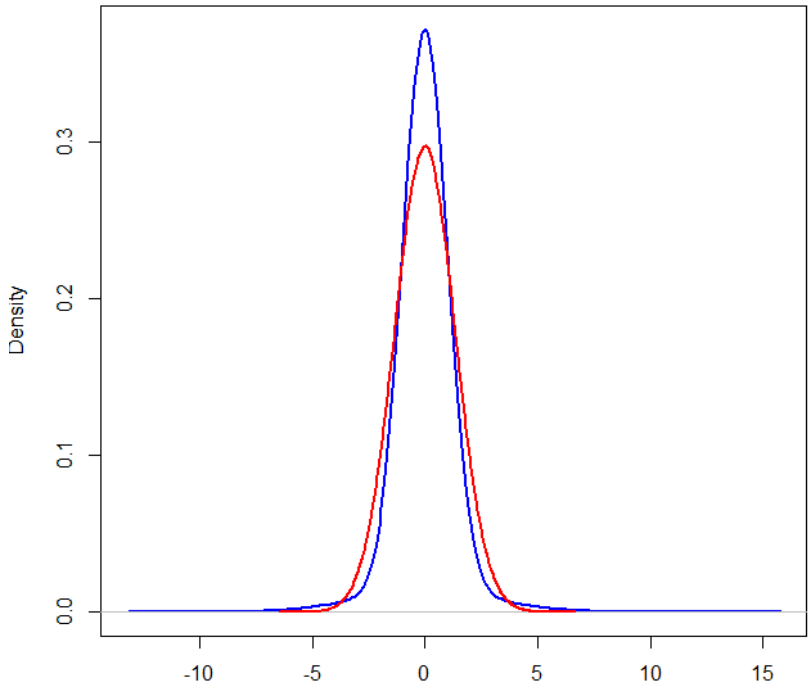


FIGURE 4. Mixed normal distribution, skewness = 0.0196, kurtosis = 5.253

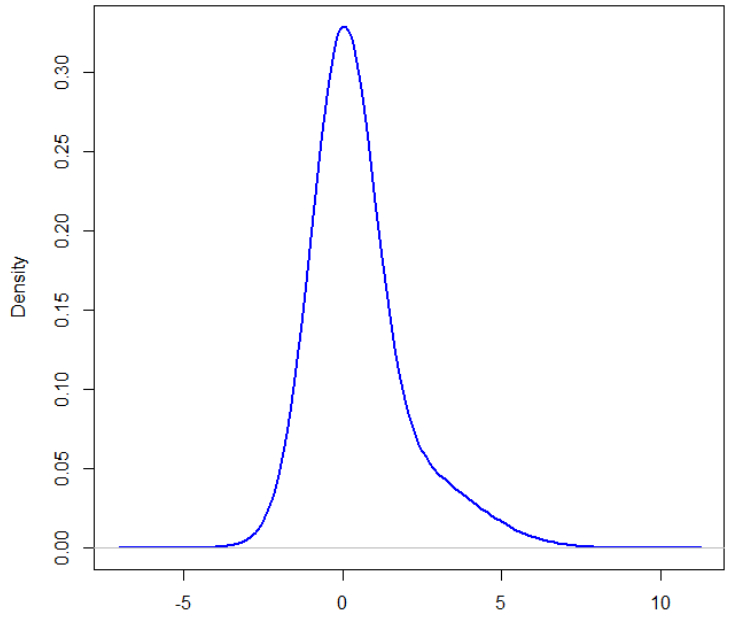


FIGURE 5. Mixed normal distribution, skewness = 1.04535, kurtosis = 1.84266

Muthén and Asparouhov (2015) present an example that shows a mixture of three distributions that when combined produce a skewed distribution, per Figure 6 that shifts the means of the Y variable to the right. Note that even though the combined distribution looks "normal," it is not. There is right skew.

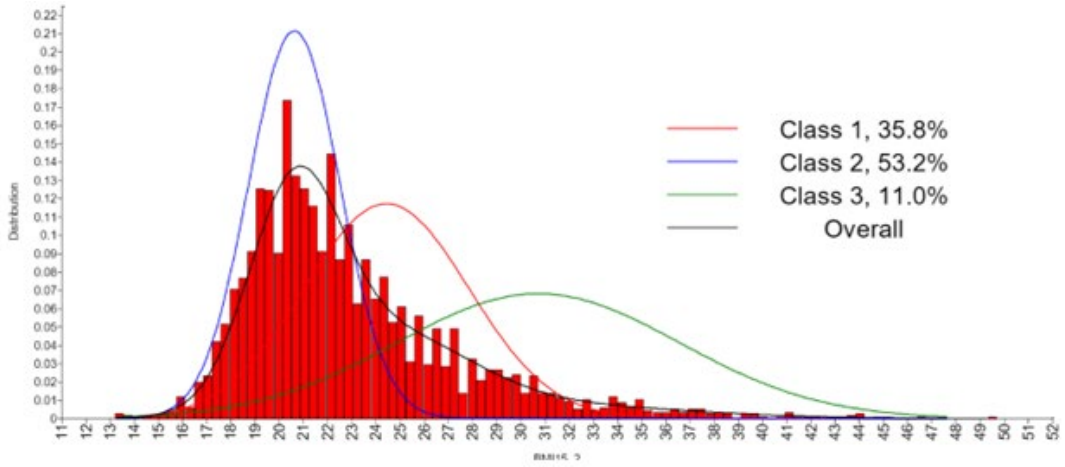


FIGURE 6. Mixture of three normal distributions to create skewness

Here is R code for mixing and plotting three normal distributions that follows the same structure as the previous R code focused on mixing two distributions (I highlight in yellow the changes in code I made to accommodate three versus two mixtures):

```
library(distr)
library(moments)
# Construct the distribution
myMix <- UnivarMixingDistribution(Norm(mean=0, sd=1),
                                  Norm(mean=0, sd=3),
                                  Norm(mean=0, sd=3),
                                  mixCoeff=c(0.8,0.1,0.1))
# create function to sample cases
rmyMix <- r(myMix)
# Sample a million cases
x <- rmyMix(1e6)
# Generate a normal distribution to overlay
xnorm<-rnorm(1000000,mean=mean(x),sd=sd(x))
#create plot
plot(density(x),col='blue',lwd=2)
lines(density(xnorm),col='red',lwd=2)  #omit to not superimpose normal
#report descritive statistics
```

```
skewness(x)
kurtosis(x)-3
mean(x)
var(x)
sd(x)
```

The program is flexible and allows you to mix many types of distributions other than normal distributions (see the library *distr* for details). However, Mplus only allows us to work with mixed normal distributions for its internal simulations.[4]

With this as background, I now show you how to conduct a simulation in Mplus that introduces non-normality. The strategy uses mixture modeling, which I described in Chapter XX.[5] I consider first the simplest case of comparing the mean outcome values of a treatment versus control condition to familiarize you with the basic structure of such simulations. I then describe programming a simulation for a more complex RET using non-normal data.

Localized Simulation for a Total Effect: Non-normal Data

In this example, I program a simulation to compare the mean outcome, Y, for a treatment and control condition, T, in a limited information estimation context using mixture modeling in which I apply mixing to the disturbance term for Y. I set the population mean outcome difference on Y between the treatment and control conditions to 0.50, so the path coefficient when I regress Y onto T in the population is 0.50. I explore the statistical power I obtain for a total sample size of 125 but I introduce non-normality into the analysis by creating two different "classes" or subgroups of cases in the population with each class having normally distributed disturbance scores when I regress Y onto T for the people in a given class. In one of the classes, the disturbance variability for Y after regressing Y onto T maps onto a standard deviation of 1.0. For the other class, the disturbance standard deviation after regressing Y onto T is three times larger than this, namely 3.0. I decide to create a scenario where 90% of the population is in the first class and only 10% is in the second class. When I combine these two classes into a single group to define my population, the disturbance term when I regress Y onto T will now be non-normal with heavy tails. The question becomes what is the statistical power for the traditional test of the mean difference on Y as a function of the treatment condition when this type of non-normality characterizes the disturbances? Table 18 presents the syntax for the simulation.

[4] You can use other mixtures and forms of nonnormality in Mplus using its external simulation feature, which I illustrate below.

[5] Another approach is to use the Mplus skewt feature to generate nonnormal data for use in the Mplus external simulation. Consideration of such an approach is beyond the scope of the current document.

Table 18: Simulation for Non-normality for Total Effect

```
1.  TITLE: MIXTURE SIMULATION ;
2.  MONTECARLO: NAMES ARE y t;
3.  CUTPOINTS = t(0);
4.  NOBSERVATIONS = 125;
5.  NREPS = 20000;
6.  SEED = 2222;
7.  GENCLASSES = c(2);  !generate two classes named c
8.  CLASSES = c(1);     !number of classes called c to analyze
9.  ANALYSIS: TYPE = MIXTURE;
10. ESTIMATOR = MLR;
11. MODEL MONTECARLO:
12. %OVERALL%
13. [t*0] ;            !set mean when generating original continuous t
14. t*1 ;              !set var when generating original continuous t
15. [y*0];             !set y intercepts to 0
16. y ON t*.50 ;       !set effect of t on y
17. y*1 ;              !set disturbance variance for y but will override
18. [c#1*2.19722];     !logit to define proportion of cases in c1 90%
19. %c#1%
20. [y*0];             !set intercept in class 1 to shift mean y if desired
21. y*1 ;              !set disturbance variance for y in class 1
22. %c#2%
23. [y*0];             !set intercept in class 2 to shift mean y if desired
24. y*9 ;              !set disturbance variance for y in class 2
25. MODEL:             !analysis model
26. %OVERALL%
27. [y*0];             !population intercept for y on t ignoring classes
28. y ON t*0.50 ;      !population effect of m on y ignoring classes
29. y*1.8058 ;          !population disturbance variance for y ignoring classes
30. OUTPUT: TECH9;
```

Most of the syntax should be familiar. Line 7 is new as is the `GENCLASSES` subcommand. It tells Mplus to generate two mixture classes and provides the labels to refer to the two classes. The number of classes is contained in parentheses and I use the letter `c` as the label for each class. You can use a different label if you want. The label occurs just before `(2)` in Line 7. Mplus will add a `#` followed by a sequential integer to the label to give a unique name to each class. In this case, the classes will be labeled `c#1` and `c#2`.

Line 8 tells Mplus the number of classes to actually analyze. In this case, I combine the two classes into a single group and analyze the data as if there is only one class.

Line 9 tells Mplus to use a mixture model. In the `MODEL MONTECARLO` section starting on Line 11, I specify an overall model that applies to each class/group (in the section labeled `%OVERALL%`). Later I will specify deviations from the overall model within each class that I want the classes to differ on, first for the `c#1` class and then for the `c#2` class.

On Lines 13 and 14 in the %OVERALL% section, I define the treatment condition as a continuous variable with a mean of 0 and a standard deviation of 1 but this variable is converted to a 0-1 dummy variable using the CUTPOINTS command on Line 3 per my discussion of the command in the main text. Line 15 arbitrarily sets the intercept of Y to zero because it does not factor into the model. Line 16 regresses Y onto T in both classes with a path coefficient of 0.50 to reflect the population coefficient for the effect of T on Y in each class. Line 17 sets the disturbance variance for Y at 1.0 in both classes, but I will override this later when I specify the specific parameter values for c#1 and c#2. Line 18 tells Mplus the proportion of the overall sample size to assign to the first class, with the remaining proportion of cases assigned to the second class. I assign 90% of the sample to c#1 and 10% of the sample to c#2. The value in the brackets is 2.19722, which is the proportion 0.90 translated into a logit scale, the latter of which Mplus makes use of. I calculated this value by converting 0.90 to an odds (0.90/(1-0.90)) = 9.0 and then taking the natural log of this result, which yielded 2.19722.

Lines 20 and 21 reassert the parameters in the %OVERALL% model as applied to the first class or group, c#1. The same is true of Lines 23 and 24 for c#2 but note that I now set the disturbance variance of Y to 9 (which is a standard deviation of 3 squared). This will override the disturbance variance I specified in the %OVERALL% model specification when data are generated for c#2. Now the standard deviation for c#2 will be 3 times larger than that for the standard deviation of c#1.

Line 25 specifies the analysis model to be applied. In this case, only the analytics of %OVERALL% are specified because I am analyzing only one class, namely the combined c#1 and c#2 classes. On Line 29, I specify the population value of the disturbance variance for the mixed normal distribution that has 90% of the cases representing random draws from a normal distribution with a mean of 0 and a standard deviation of 1.0 and 10% of the cases from a normal distribution with a mean of 0 and a standard deviation of 3.0. I obtained the value 1.8058 from the R syntax I provided above for mixing normal distributions. Note also I make no reference to mean or variance of the exogenous dummy variable T, which is standard practice in Mplus simulations.

Here are the results of the simulation as focused on the main parameters of interest:

		ESTIMATES		S. E.	M. S. E.	95%	% Sig
	Population	Average	Std. Dev.	Average		Cover	Coeff
Y ON							
T	0.500	0.5027	0.2414	0.2372	0.0583	0.947	0.570
Residual Variances							
Y	1.806	1.7698	0.4274	0.3870	0.1840	0.847	1.000

The statistical power for the effect of T on Y is 0.570. The confidence interval coverage seems good as does the lack of bias for the path coefficient per se. The estimated margin of error using the standard error (2 times 0.24 = 0.48) is fairly large.

How does this compare to the case where the assumption of normally distributed disturbances is met? I can determine this by changing Line 24 from a disturbance variance of 9 to a disturbance variance of 1.0 (so that it is the same as the disturbance variance of `c#1`) and on Line 29 from 1.8058 to 1.0. I am mixing two identical normal distributions represented by the two classes, the result of which is the same normal distribution. Here are the results:

```
                        ESTIMATES              S. E.    M. S. E.   95%  % Sig
               Population   Average   Std. Dev.    Average             Cover  Coeff

Latent Class 1

 Y          ON
  T                 0.500     0.5032     0.1794     0.1778     0.0322   0.947  0.805

 Residual Variances
  Y                 1.000     0.9847     0.1261     0.1224     0.0161   0.920  1.000
```

The statistical power under normality is 0.80, which is substantially higher than the power of 0.57 with the heavy tailed non-normal distribution from the prior analysis (note how much lower the margin of error is as well). This decrease in power occurs despite the fact that only 10% of the sample (class 2) was "acting up" to disrupt normality. Note also that most canned power analysis software assumes normally distributed disturbances and would inform you your statistical power is 0.80 when it is, in fact, considerably lower. For those who advocate examining plots to detect heavy tailed mixed normal distributions as a post data collection warning of low power, keep in mind how difficult it is for the naked eye to detect such non-normality in many cases.

In this example, I applied robust maximum likelihood (MLR) but the results seem anything but robust. It turns out the results *are* robust to the nonnormality with respect to Type I errors (the rate maintains near 0.05 even in the face of the non-normality when I set the path coefficient from T to Y to zero in a new analysis) but statistical power and MOEs *are* affected by the non-normality. This has been a theme of Rand Wilcox's work as emphasized in his 1998 *American Psychologist* article "How many discoveries have been lost by ignoring modern statistical methods?" Wilcox suggests the use of specialized robust analytic methods that do not show such power loss, usually in the context of a LISEM framework. See Wilcox (2021) for details. The methods include the analysis of M estimators or trimmed means, among others.

Localized Simulations for Complex Mediation Model: Non-normal Data

In this section, I apply the same logic as above but to the model from the section *Working with Complex Models: Part II* (per Figure 2). The model of interest has a treatment versus control group (0 = control, 1 = treatment), three posttest mediators (M1, M2, M3) each with a baseline measure (M1B, M2B, M3B), and a follow-up outcome, Y, coupled with a baseline Y (YB). All mediator and outcome measures have a metric from 0 to 100 with a standard deviation of 10, as do their baseline counterparts.

I show a mixture scenario for the disturbance variance (var(d_4) in Figure 2) in which 85% of the cases in the population have a disturbance normal distribution with a mean of zero and variance equal to 48.923 (SD = 6.9945) and 15% of the cases have a different disturbance distribution in which the mean is zero and the standard deviation of the disturbances is twice the size of the first normal distribution (SD = 13.989, variance = 195.692). Figure 7 shows the densities of this mixed normal distribution with a normal distribution superimposed on it generated from my R code.

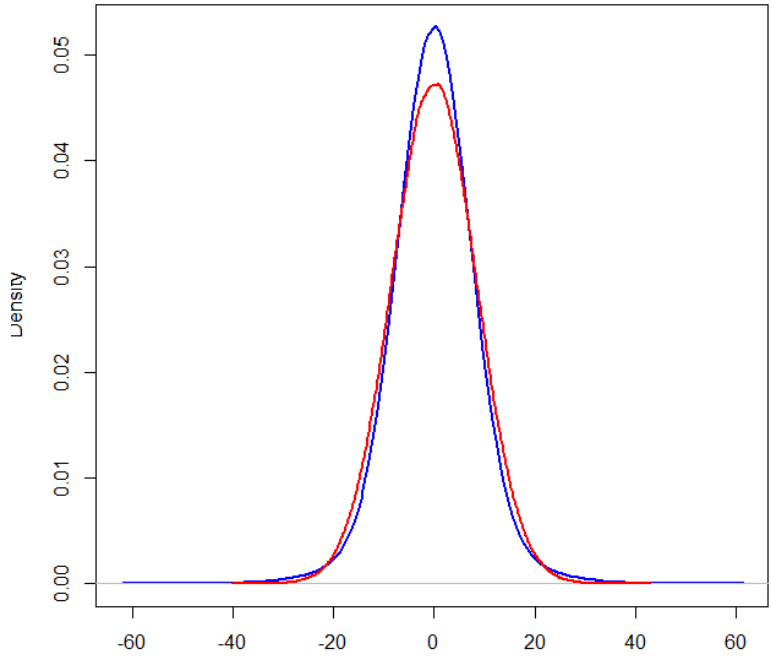


FIGURE 7. Mixed normal distribution, skewness = 0.0040, kurtosis = 1.6096

Table 19 presents the Mplus syntax to evaluate the statistical power for the model. All syntax should be self-explanatory at this point. The population value for the variance

of the mixed normal distribution in Line 44 was taken from execution of the previous R code where I set the first distribution to have a mean of 0 and a standard deviation of 6.99 and the second distribution to have a mean of 0 and a standard deviation of 13.99 coupled with the a priori proportion of cases to sample from each distribution.

Table 19: Simulation for Non-Normality for Mediator Equation

```
1.  TITLE: NON-NORMALITY FOR MEDIATOR EQUATION;
2.  MONTECARLO:
3.  NAMES ARE t m1 m2 m3 m1b m2b m3b y yb ;
4.  CUTPOINTS = t(0);
5.  NOBS = 100 ;        !sample size
6.  NREPS = 20000 ;     !number of replicates
7.  SEED = 2222 ;       !random seed
8.  GENCLASSES = c(2);  !generate two classes named c
9.  CLASSES = c(1);     !number of classes to analyze
10. ANALYSIS: TYPE = MIXTURE;
11. MODEL MONTECARLO:
12. %OVERALL%
13. [t*0] ;             !set mean when generating original continuous t
14. t*1 ;               !set var when generating original continuous t
15.  [m1b*0]; [m2b*0]; [m3b*0]; [yb*0];    !set means
16.  m1b*100; m2b*100; m3b*100; yb*100 ;   !set variances
17.  [y*0]; [m1*0]; [m2*0]; [m3*0];        !set intercepts
18.  m1b WITH m2b*30 m3b*30 yb*30 ;        !set covariances
19.  m2b WITH m3b*30 yb*30 ;
20.  m3b WITH yb*30 ;
21.  t WITH m1b*0 m2b*0 m3b*0 yb*0 ;
22.  m1 ON m1b*0.30 t*0 ;   !define equations
23.  m2 ON m2b*0.30 t*5.0 ;
24.  m3 ON m3b*0.30 t*8.0 ;
25.  y ON yb*.30 m1*0 m2*0.20 m3*0.40 t*5 ;
26.  m1*91.0 ;     !define disturbance variances
27.  m2*84.75 ;
28.  m3*75.0 ;
29.  y*48.923 ;
30.  [C#1*1.7346];
31. %C#1%
32. [y*0];         !set intercept to shift mean y if desired
33.  y*48.923  ;   !disturbance variance for y
34. %C#2%
35. [y*0];         !set intercept to shift mean y if desired
36.  y*195.692  ;  !disturbance variance for y
37. MODEL:         !specify analysis model; don't mention exogenous
38. %OVERALL%
39.  m1 ON m1b*0.30 t*0 ;
40.  m2 ON m2b*0.30 t*5.0 ;
```

```
41.  m3 ON m3b*0.30 t*8.0 ;
42.  y ON yb*.30 m1*0 m2*0.20 m3*0.40 t*5 ;
43.  m1*91.0 ;
44.  m2*84.75 ;
45.  m3*75.0 ;
46.  [y*0];
47.  y*71.085
48. MODEL INDIRECT:
49.  y IND t ;        !evaluate omnibus mediation effect
50. OUTPUT: TECH9 ;
```

Here are the results for the key parameters in the analysis:

		Population	ESTIMATES Average	Std. Dev.	S. E. Average	M. S. E.	95% Cover	% Sig Coeff
Latent Class 1								
M1	ON							
M1B		0.300	0.2988	0.0978	0.0941	0.0096	0.933	0.871
T		0.000	-0.0050	1.9334	1.8937	3.7379	0.943	0.056
M2	ON							
M2B		0.300	0.3006	0.0940	0.0907	0.0088	0.936	0.898
T		5.000	5.0105	1.8709	1.8270	3.5004	0.941	0.774
M3	ON							
M3B		0.300	0.3011	0.0882	0.0853	0.0078	0.938	0.927
T		8.000	8.0075	1.7243	1.7182	2.9731	0.946	0.996
Y	ON							
YB		0.300	0.3007	0.0882	0.0837	0.0078	0.933	0.928
M1		0.000	-0.0005	0.0872	0.0831	0.0076	0.934	0.066
M2		0.200	0.2000	0.0908	0.0857	0.0082	0.933	0.640
M3		0.400	0.3995	0.0958	0.0908	0.0092	0.933	0.981
T		5.000	5.0110	1.9377	1.8640	3.7546	0.938	0.758

Here are the results when I used normally distributed disturbances with var(d_4) = 48.923 in both classes so that the assumption of normality of disturbances is met:

		Population	ESTIMATES Average	Std. Dev.	S. E. Average	M. S. E.	95% Cover	% Sig Coeff
Latent Class 1								
M1	ON							
M1B		0.300	0.2988	0.0978	0.0941	0.0096	0.933	0.871
T		0.000	-0.0050	1.9334	1.8937	3.7379	0.943	0.056

		Population	ESTIMATES Average	Std. Dev.	S. E. Average	M. S. E.	95% Cover	% Sig Coeff
M2	ON							
M2B		0.300	0.3006	0.0940	0.0907	0.0088	0.936	0.898
T		5.000	5.0105	1.8709	1.8270	3.5004	0.941	0.774
M3	ON							
M3B		0.300	0.3011	0.0882	0.0853	0.0078	0.938	0.927
T		8.000	8.0075	1.7243	1.7182	2.9731	0.946	0.996
Y	ON							
YB		0.300	0.3006	0.0731	0.0698	0.0053	0.934	0.982
M1		0.000	-0.0002	0.0726	0.0694	0.0053	0.933	0.067
M2		0.200	0.1999	0.0753	0.0715	0.0057	0.931	0.782
M3		0.400	0.3999	0.0795	0.0757	0.0063	0.933	0.998
T		5.000	5.0087	1.6112	1.5519	2.5959	0.936	0.884

I expect to see adverse effects of non-normality, if any, for the equation predicting Y from the mediators and the treatment (plus covariates) because it is this equation where I introduced the disturbance non-normality. In this equation, for M1, the Type I error rate was relatively unaffected by the nonnormality, which is not surprising given my use of a robust estimator. The statistical power for M2 was non-trivially affected by the non-normality, with power decreasing from 0.782 to 0.640. This also tended to be true for the statistical power for the direct effect of T on Y over and above the mediators and covariates, where the power decreased from 0.884 to 0.758. The statistical power for M3 was relatively unaffected by the non-normality because it was quite high to begin with (0.998 that decreased to 0.981).

Non-Normality for External Simulations in Mplus

Local simulations with non-normality also can be conducted using the Mplus external simulation feature. This strategy has the advantage of being able to introduce non-normality other than via mixed normal distributions. I provide two programs on my website for generating such data (*Generate Data I* and *Generate Data II*).

In the first program, *Generate Data I*, you input a covariance matrix and the program then generates data that are randomly selected from a population with that covariance structure. For each variable in the covariance matrix, you specify the amount of population skewness and excess kurtosis that you want the variable to have. The program uses the R package *covsim* and outputs the generated data based on this information in a format that is compatible with Mplus external simulation syntax. If you specify 0 skewness and 0 kurtosis, the variable will have a population normal distribution. Positive numbers for skewness indicate positive skew and negative numbers indicate negative skew. Positive

numbers for kurtosis indicate excess leptokurtosis and negative numbers indicate excess platykurtosis. The program on my website can only be used with continuous variables, although the *covsim* package is more flexible than this and can be used with ordinal variables as well.

The generation of non-normal data from a population characterized by an a priori specified covariance matrix has received considerable attention from statisticians and is somewhat controversial. Multiple methods have been proposed for doing so and each has strengths and weaknesses. One popular approach uses polynomial transformations on data sampled from a population based on normally distributed variables so as to create the nonnormal data. Vale and Maurelli (1983) proposed such a method as an extension of an earlier technique by Fleishman (1978). A flexible method that uses fifth-order polynomial transformations was proposed by Headrick (2002, 2004). The disadvantages of these power methods are that they are limited in the types of non-normal distributions they can generate and they tend to yield somewhat biased skewness and excessive kurtosis relative to that requested by the user (Astivia & Zumbo, 2015). Cario and Nelson (1997) developed an alternative algorithm called the NORTA (NORmal To Anything) method. Like power methods, NORTA tends to rely on an underlying normal distribution to generate the nonnormal data. Foldnes and Gronneberg (2015) argue that by doing so, it may not capture the types of nonnormality observed in the real world.

The *covsim* package has several state of the art data generation methods. In my program, I use an approach by Foldnes and Olsson (2016) called the **independent generator (IG)** algorithm. *Covsim* offers a powerful method called *vita* (VIne-To-Anything) that is more flexible than the IG algorithm but it also is complicated and computer intensive. The IG approach with a non-normal vector ξ is defined as

$$\xi = A\,X$$

where A is a square matrix and X a vector consisting of mutually independent generator variables with unit variance. The user specifies desired skewness and kurtosis values in ξ, and the IG algorithm numerically determines the skewness and kurtosis in each generator variable to match these values. The A matrix is a square root of the specified covariance matrix, which can be calculated using either a triangular square or a symmetrical square root, yielding two different types of distributions. The marginal distributions for X can be freely chosen.

The IG approach in *covsim* uses the Pearson family of distributions. For example, a chi-square distribution with one degree of freedom has skewness equal to 2.83 and excess kurtosis equal to 12. To generate scores for a variable that are roughly distributed as such, one would specify these values for skewness and kurtosis in the IG algorithm for that

variable. The resulting distribution will not exactly be a 1single degree of freedom chi square because the program makes adjustments to obtain the mean you requested and the desired covariance structure. But it usually will be in the ballpark. Among the distributions in the Pearson family are the beta distribution, the beta prime distribution, the Cauchy distribution, the chi-squared distribution, the continuous uniform distribution, the exponential distribution, the gamma distribution, the F distribution, the inverse-chi-squared distribution, the inverse-gamma distribution, the normal distribution, and Student's t-distribution. For details, see Grønneberg, Foldnes and Marcoulides (2022).

Users of the IG method sometimes specify a desired covariance matrix between variables with values of skewness and kurtosis that are incompatible because they are mathematically intractable or statistically impossible to co-exist. The IG algorithm may produce an error message if the inconsistencies are too great. My program provides diagnostics to help determine how close the approximation is. I review these shortly.

Another important point to keep in mind is that a univariate distribution is determined by more than just its skewness and kurtosis. Given this, by specifying the desired covariance matrix and the respective skewness and kurtosis values of the separate variables, there are different data-generating processes that can reproduce the target covariance matrix, skewnesses and kurtosis. For this reason, some methodologists suggest using multiple simulation algorithms for data generation and then replicating results across those algorithms (e.g., Fairchild et al., 2024).

A core concept in generating simulation data is that of a copula. Copulas are mathematical specifications that implement the dependency structure between variables after taking the marginal distributions of the variables into account. There are many classes of copulas. Each class includes one or more parameters that control the strength of dependence between the variables. One copula class is called the **normal or Gaussian copula**. It is popular in many simulation packages in the social and health sciences. Shortcomings associated with this copula have been described by Astivia & Zumbo (2015) and Foldnes and Gronneberg (2015). The IG method uses a non-normal copula that produces different distributions than those of Gaussian copulas. When applied to SEM, it is possible for the data generated by Gaussian copulas to produce different results for robustness to nonnormality despite sharing the exact same covariance matrix and univariate skewness and kurtosis values for the variables comprising the covariance matrix (Foldnes & Gronneberg, 2015). One also can obtain differences as a function of the two distribution forms generated with the IG algorithm using the triangular square versus symmetrical square of the covariance matrix. Mplus uses a Gaussian copula.

I consider here an example that examines the relationship between an outcome, Y, and three mediators in a linear regression. The first mediator is a latent variable and has

three normally distributed indicators, the second mediator has a single indicator that is positively skewed and leptokurtic, and the third mediator is normally distributed. The latent variable has a variance of 1.0 as does M2, M3 and Y. In the study I am planning, I evaluate the regression of Y onto the three mediators using robust maximum likelihood in Mplus with a sample size of 120. I expect the outcome variable Y also to have a degree of non-normality. Figure 8 presents the relevant path diagram with the population values of the core parameters that I want to simulate. Of concern to me in my study is M2 and the fact that it almost certainly will be positively skewed and highly leptokurtic and the non-normality in Y. Granted, I will be using a robust estimation method, but I am uncertain with the smaller sample size if it will perform satisfactorily. I expect LM1, M1a, M1b, M1c and M3 all to be normally distributed.

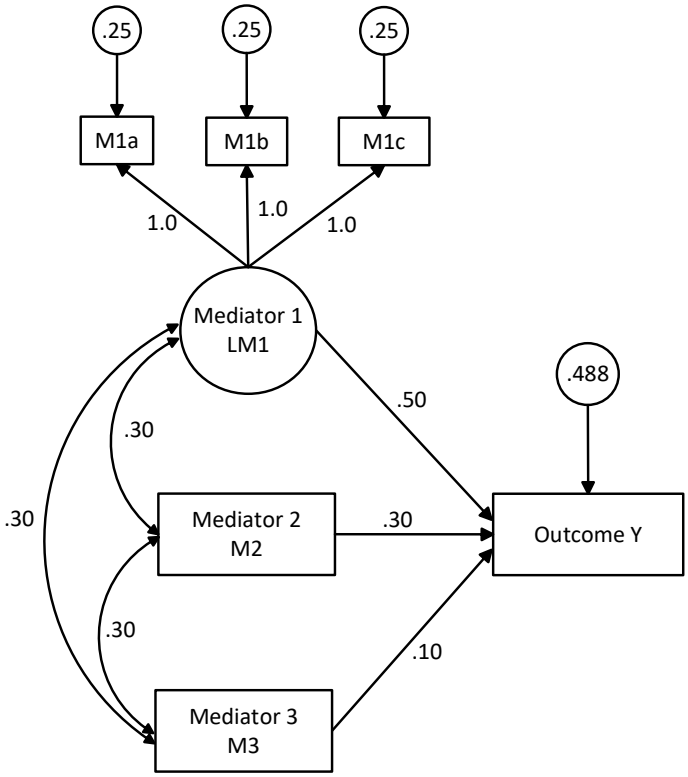


FIGURE 8. Model to be evaluated using *covsim*

In many SEM simulations, the researcher first specifies a referent SEM model together with its population parameter values, as I have done above. The simulation study is designed to select random samples from a multivariate distribution whose covariance

matrix equals the model-implied covariance matrix. The scores are "selected" from the population (i.e., generated) in this case using the IG method but I first need to determine the model-implied covariance matrix for input into my program. I can derive this covariance matrix for the model in Figure 8 either by hand (which can be quite cumbersome) or using Mplus as a computational aid. Here is the Mplus program I use to derive it:

```
 1. TITLE: GENERATE MODEL IMPLIED COVARIANCE ;
 2. DATA:
 3. FILE IS c:\ret\covsimcovdat.txt ;
 4. TYPE IS CORRELATION ;
 5. NOBSERVATIONS = 1000;
 6. VARIABLE:
 7. NAMES ARE  m1a m1b m1c y m2 m3 ;
 8. ANALYSIS: ESTIMATOR = ML ;
 9. MODEL:
10.  lm1 BY m1a@1 m1b@1 m1c@1 ;
11.  lm1@1; m2@1 ; m3@1 ; y@.512 ;
12.  m1a@.25 ; m1b@.25 ; m1c@.25;
13.  lm1 WITH m2@.3  ;
14.  lm1 WITH m3@.3 ;
15.  m2 WITH m3@.3 ;
16.  y ON lm1@.5 m2@.3 m3@.1 ;
17. OUTPUT: SAMP STDYX TECH4 RESIDUAL;
```

Most but not all of the syntax should be familiar. As with all my examples, you do not number the lines in Mplus per se; I do so here just so I can refer to them. The program uses summary data as input, in this case a correlation matrix (see Lines 3 to 5). The correlation matrix I input is not real data. It is a placeholder that I use as a device to allow me to calculate the model-implied covariance matrix. As such, the values in the correlation matrix can be any values as long as they are not degenerate. Line 3 identifies the free format ASCII file where the correlation matrix is located, Line 4 tells Mplus the matrix is a correlation matrix, and Line 5 tells Mplus the sample size for the correlation matrix, which can be any positive integer. It also is arbitrary and irrelevant to the real task at hand. I usually set it to 1000. The input correlation matrix stored in the data file is a lower triangular matrix (including the diagonals) with the same number of variables that your desired model-implied covariance matrix for *covsim* will have, in this case 6 (because there are six observed variables). Here is the correlation matrix I used exactly as it appears in the referenced data file (`c:\ret\covsimcovdat.txt`):

```
1.00
0.30    1.00
0.30    0.30    1.00
0.30    0.30    0.30    1.00
0.30    0.30    0.30    0.30    1.00
0.30    0.30    0.30    0.30    0.30    1.00
```

The entries are either space or tab delimited and the values are such that I am certain the matrix is positive definite, i.e., that all of its eigenvalues are positive. Creating a matrix with small but equal positive correlations typically is a safe bet.

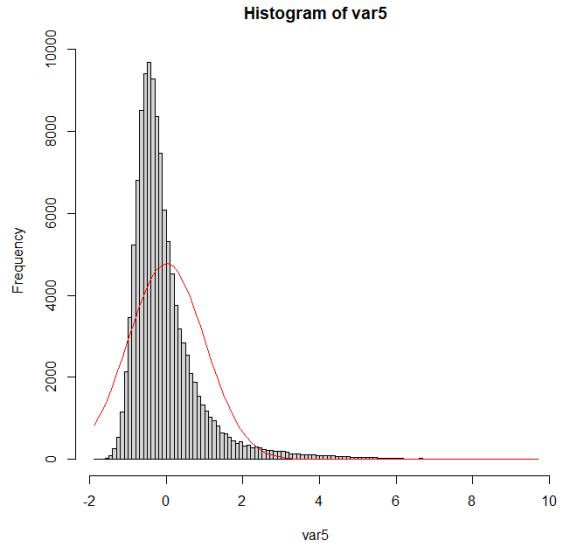
Line 7 of the syntax provides the names of the model variables in the correlation matrix from first to last. Lines 10 to 16 specify the model but note that I have fixed the values of every parameter to its corresponding population value using the @ character. There is no formal estimation of these parameters when I run the program. The syntax will apply these values and then produce output that contains the model-implied covariance matrix. It occurs in the output section called `RESIDUAL OUTPUT` as follows:

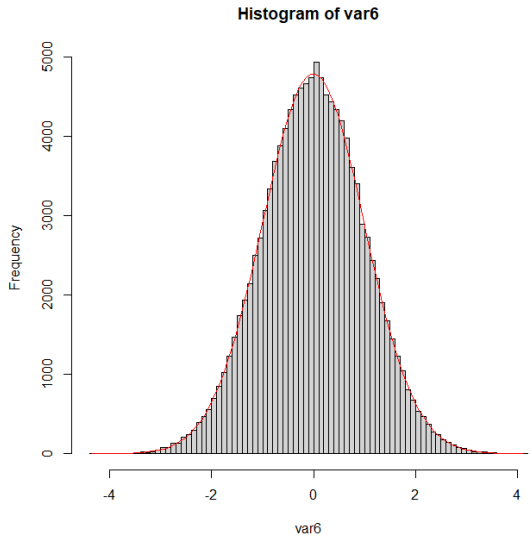
```
      Model Estimated Covariances/Correlations/Residual Correlations

               M1A           M1B           M1C           Y             M2
               ________      ________      ________      ________      ________
M1A              1.250
M1B              1.000         1.250
M1C              1.000         1.000         1.250
Y                0.620         0.620         0.620         1.000
M2               0.300         0.300         0.300         0.480         1.000
M3               0.300         0.300         0.300         0.340         0.300

           Model Estimated Covariances/Correlations/Residual Correlations
              M3
              ________
M3               1.000
```

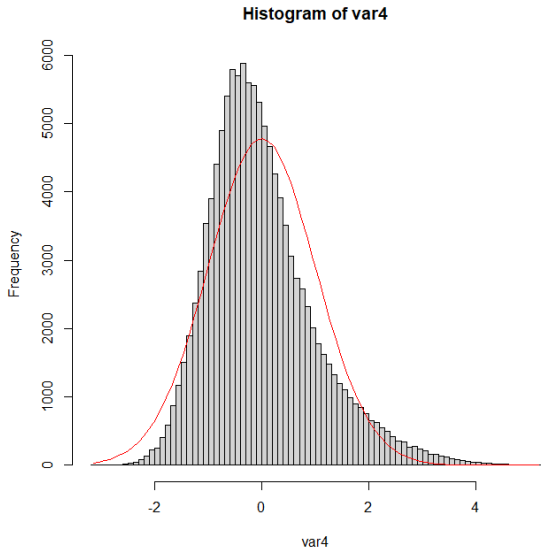
This is the covariance matrix I input into the program on my website including my choices for skewness and excess kurtosis for each variable. For the latter, I decide to set all variables except M2 and Y to have zero skewness and zero excess kurtosis, i.e., to be normally distributed. I set M2 to have skewness of 2.82 and kurtosis of 12 (roughly mapping onto a chi square distribution with 1 df) and Y to have skewness of .9 and excess kurtosis of 1.2, a milder form of non-normality (roughly mapping onto a chi square distribution with 10 df). The video associated with the program shows you the steps I used to execute it. I created data sets for 1,000 simulation replications with a sample size of 120 per replication. Across all simulation replicates, the sample size was (1,000)(120) =

120,000 but the simulation itself was conducted assuming a per replicate N of 120.

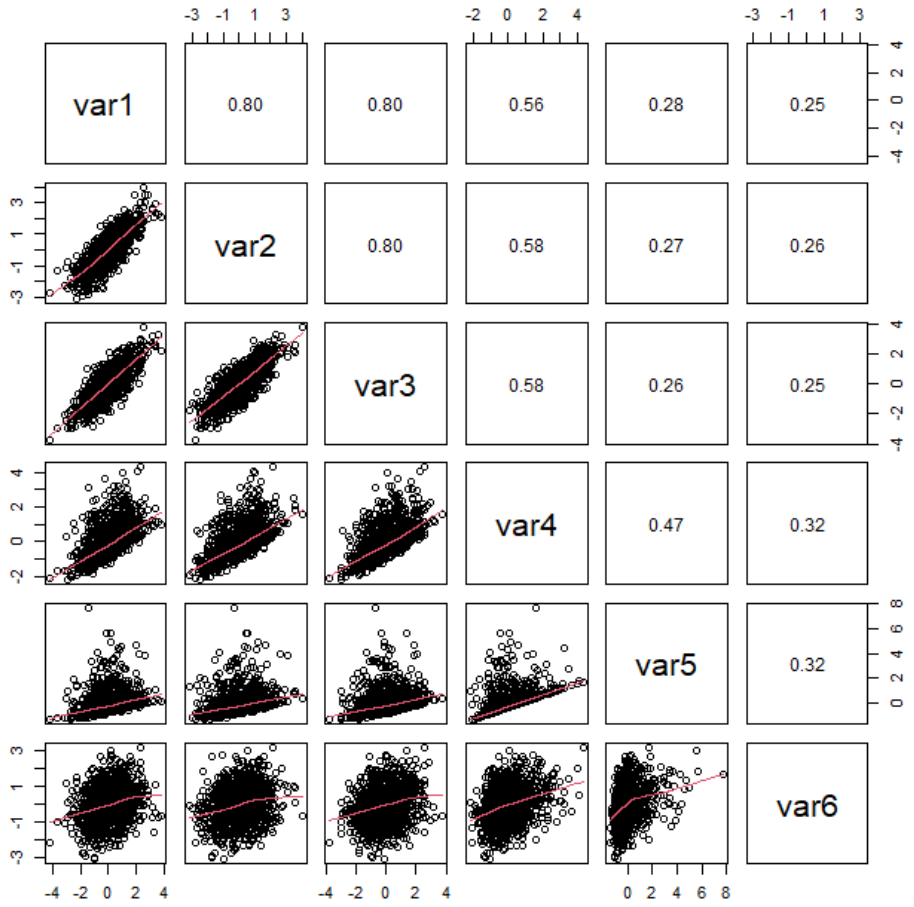
To help gain a sense of the IG based population distributions of the generated variables, the program plots histograms for the generated scores of each of the variables collapsing across all simulation replicates (N = 120,000). Here are the histograms for M2 (which should be decidedly non-normal with positive skew and positive excess kurtosis), M3 (which should be normally distributed), and Y (which should have some positive skew and positive excess kurtosis). The variables are labeled by the program as var5, var6, and var4 to reflect M2, M3, and Y, respectively. The densities for a normal distribution are superimposed on the histograms in red :







The program also produces a scatter matrix of all possible pairs of variables for 1,000 cases randomly selected from the 120,000 cases. In the scatter matrix, var1 = M1a, var2 = M1b, var3 = M1c, var4 = Y, var5 = M2, and var6 = M3. Here is the scatter matrix that has a scatterplot with smoothers in the lower triangle and the correlations in the upper triangle.



The scatterplots appear as expected given the mixture of variables with normal and non-normal distributions.

Here is the requested covariance matrix and the computed covariance matrix for the 120,000 cases of generated data, for which I expect close correspondence:

```
Input covariance matrix
       var1   var2   var3   var4   var5   var6
var1   1.25   1.00   1.00   0.62   0.30   0.30
var2   1.00   1.25   1.00   0.62   0.30   0.30
var3   1.00   1.00   1.25   0.62   0.30   0.30
var4   0.62   0.62   0.62   1.00   0.48   0.34
var5   0.30   0.30   0.30   0.48   1.00   0.30
var6   0.30   0.30   0.30   0.34   0.30   1.00
```

```
Covariance matrix from generated data
           var1       var2       var3       var4       var5       var6
var1  1.2519659  0.9986506  1.0009941  0.6232038  0.2989172  0.3044287
var2  0.9986506  1.2461415  1.0000638  0.6220878  0.2991803  0.3035087
var3  1.0009941  1.0000638  1.2504359  0.6248098  0.3006416  0.3047523
var4  0.6232038  0.6220878  0.6248098  1.0033469  0.4782759  0.3432240
var5  0.2989172  0.2991803  0.3006416  0.4782759  1.0012994  0.3027376
var6  0.3044287  0.3035087  0.3047523  0.3432240  0.3027376  0.9996258
```

The requested values are reasonably close to the generated values. The statistics based on the generated data will not be exact reproductions because there is sampling error in them, but with an N of 120,000, sampling error should be small. Here are the descriptive statistics for the generated data

```
Descriptive statistics for generated data
           n  mean   sd median trimmed  mad   min   max range skew kurtosis
var1  120000 -0.01 1.12  -0.01   -0.01 1.12 -4.91  4.81  9.72 0.01     0.00
var2  120000 -0.01 1.12  -0.01   -0.01 1.11 -5.05  4.85  9.90 0.00     0.01
var3  120000 -0.01 1.12  -0.01   -0.01 1.12 -4.68  5.29  9.96 0.01     0.00
var4  120000  0.00 1.00  -0.16   -0.09 0.86 -3.15  5.17  8.32 0.91     1.21
var5  120000  0.00 1.00  -0.26   -0.17 0.57 -1.87 10.31 12.18 2.80    11.91
var6  120000 -0.01 1.00   0.00   -0.01 1.01 -4.23  4.43  8.66 0.00    -0.01
```

Of interest are the last two columns that indicate the amount of skew and excess kurtosis in the generated data. The values are reasonably close to those values that were requested.

Mplus uses an estimation strategy where non-normality in the exogenous variables is not typically problematic because they are treated as fixed predictors. However, because of the exogenous latent variable, all the exogenous variables are treated as random predictors rather than fixed predictors. In such cases, distributional assumptions are made about the exogenous as well as the endogenous variables. The simulation reported below evaluates if the specified levels of non-normality create inferential problems.

I next wrote and executed the external simulation code in Mplus to conduct the simulation analysis on the generated data. Here is the code (as usual, ignore the numbering of lines, which are for referring in the text to the lines):

```
 1. TITLE: LOCAL EXTERNAL SIMULATION ;
 2. DATA: FILE = c:/ret/covsimdatlist.txt ;
 3. TYPE = MONTECARLO ;
 4. VARIABLE:
 5. NAMES ARE trial id m1a m1b m1c y m2 m3 ;
 6. USEVARIABLES ARE m1a m1b m1c y m2 m3 ;
 7. ANALYSIS:
 8. ESTIMATOR = MLR ;
```

```
 9. MODEL:
10.  lm1 BY m1a@1 m1b*1 m1c*1 ; d
11.  lm1*1; m2*1 ; m3*1 ; y*.512
12.  m1a*.25 ; m1b*.25 ; m1c*.25
13.  lm1 WITH m2*.3 ;
14.  lm1 WITH m3*.3 ;
15.  m2 WITH m3*.3 ;
16.  y ON lm1*.5 m2*.3 m3*.1 ;
17. OUTPUT: TECH9 ;
```

Most of the syntax is self-explanatory. Line 4 tells Mplus to do a Monte Carlo simulation. Note the input file in Line 3 is the replist.dat file. This is the file that tells Mplus where to find all of the separate externally generated files. It was generated by the *Data Generation 1* program. The syntax assumes the separate files are located in the same folder that the above Mplus program is in given the absence of a folder path specification. Lines 10 to 16 describe the model to be fit to the data but they also include the values of the population parameters preceded by a * or a @.

The output is the same as that for any Mplus simulation which you should now be familiar with. Here are the results for the chi square test of global fit:

```
Chi-Square Test of Model Fit

        Degrees of freedom                     6

        Mean                                6.734
        Std Dev                             4.302
        Number of successful computations    1000

             Proportions                  Percentiles
        Expected    Observed         Expected       Observed
           0.990       0.989            0.872          0.771
           0.980       0.984            1.134          1.288
           0.950       0.961            1.635          1.729
           0.900       0.909            2.204          2.278
           0.800       0.808            3.070          3.121
           0.700       0.721            3.828          4.043
           0.500       0.559            5.348          5.769
           0.300       0.381            7.231          8.272
           0.200       0.273            8.558          9.585
           0.100       0.150           10.645         12.001
           0.050       0.089           12.592         14.697
           0.020       0.042           15.033         17.480
           0.010       0.026           16.812         20.013
```

The average chi square value (6.734) should approximate the degrees of freedom, 6.

The two values seem reasonably close. The standard deviation of the chi square values should equal the square root of double the degrees of freedom. The square root of 12 is 3.46, which is reasonably close to the reported standard deviation value of 4.302 on the output. In the column `Proportions Expected` at the row for a theoretical p value of 0.05, the entry in the `Proportions Observed` column is 0.089. This indicates that the chi square test is rejecting about 9% of the models when it should be rejecting only 5% of the models, the alpha level expressed in percent from. As such, the test shows a tendency to over-reject correctly specified models. I also note that in a chi square distribution with 6 degrees of freedom, one expects the critical value of chi square that rejects 5% of the models to equal 12.592. The critical value in the observed data was 14.697. Overall, in my opinion, the chi square test is in the ball park of adequacy but I wish it had done better.

Here are the results for the parameter estimates:

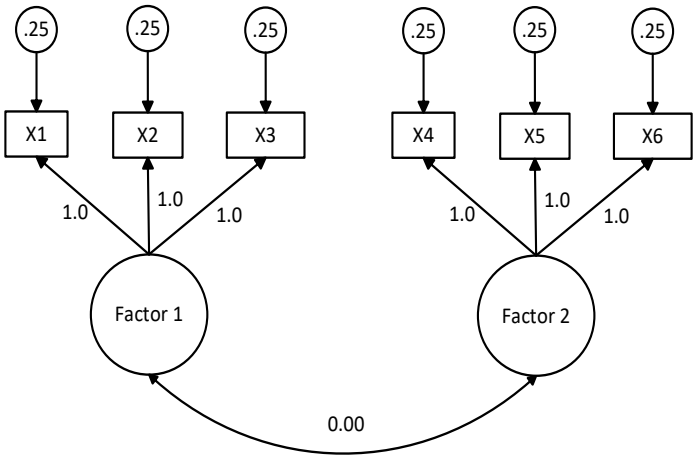
```
MODEL RESULTS

                                ESTIMATES              S. E.     M. S. E.  95%  % Sig
                  Population    Average   Std. Dev.    Average             Cover Coeff

 LM1       BY
  M1A                1.000     1.0000     0.0000     0.0000     0.0000 1.000 0.000
  M1B                1.000     1.0010     0.0726     0.0713     0.0053 0.939 1.000
  M1C                1.000     1.0037     0.0741     0.0713     0.0055 0.942 1.000

 Y         ON
  LM1                0.500     0.4946     0.0912     0.0835     0.0083 0.916 1.000

 Y         ON
  M2                 0.300     0.3370     0.1636     0.1283     0.0281 0.878 0.729
  M3                 0.100     0.0960     0.0738     0.0714     0.0055 0.941 0.276

 LM1       WITH
  M2                 0.300     0.2976     0.0974     0.0963     0.0095 0.947 0.883
  M3                 0.300     0.3010     0.1021     0.0979     0.0104 0.940 0.889

 M2        WITH
  M3                 0.300     0.3000     0.1101     0.1034     0.0121 0.913 0.888
```

Scanning down the `Estimates` column, there is no disconcerting parameter bias, although the parameter estimate for M2 is flirting with unacceptable positive bias (population value = 0.300 as compared with the average estimate across the 1,000 trials being 0.3370). However, there is some notable downward bias in the standard errors (compare the `Std. Dev.` column with the `S.E. Average` column). For example, the standard error for the coefficient regressing Y onto M2 is 0.1636 and the average standard error for this coefficient across the 1,000 trials was 0.1283. The 95% confidence interval coverage

should be near 0.95 for each coefficient. There are several exceptions to this were the coverage is too low. For example, the coverage for the coefficient regressing Y onto M2 is 0.878 and the coverage for the coefficient regressing Y onto LM1 it is 0.916.

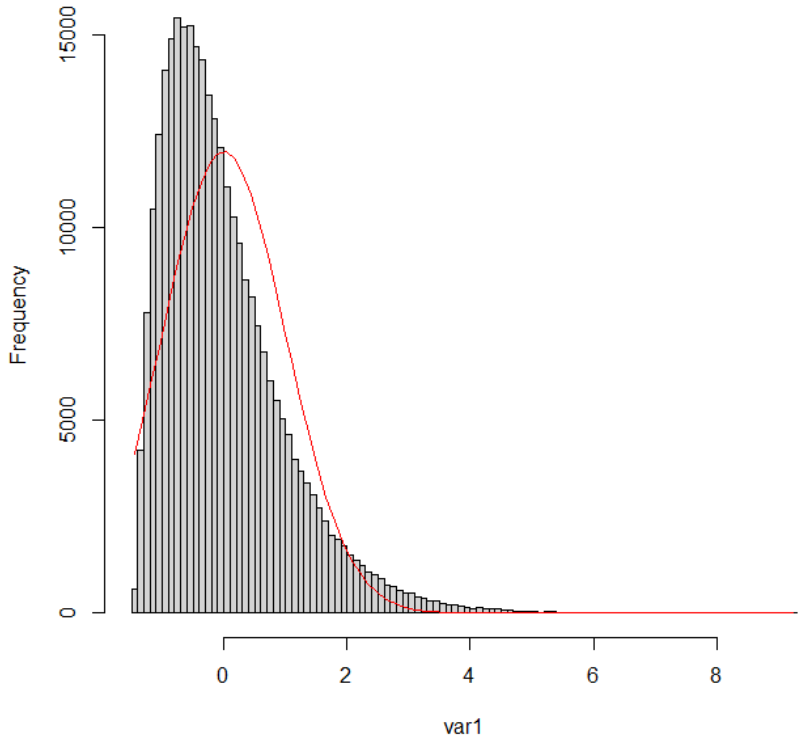
My overall conclusion is that there are sufficient deviations in the above that I can't move forward with my study as planned. Perhaps a larger N than 120 would help because larger N sometimes improves the robustness qualities of estimators. When I repeated the simulation but used a sample size of 200 per trial, the performance of the global chi square test of fit improved to acceptable levels as did estimation of most of the separate parameter estimates with the one exception being the Y on M2 coefficient 95% confidence interval coverage, which improved to 0.92. This point is important more generally because it indicates that the performance of the MLR estimation strategy in Mplus can be influenced by sample size and that for small N, it may not perform as well as we might assume. Also, the results are dependent on my presumed levels of skewness and kurtosis in the population. Perhaps when I conduct my actual study, the non-normality will not be as extreme as what I modeled here. I can get a handle on the extent to which this matters by re-doing the simulation with less extreme values for skewness and kurtosis. Another corrective strategy might be to introduce covariates that when included in the model may reduce some of the consequential skewness or kurtosis in the endogenous variable. The important point here is that localized simulations can provide useful perspectives on sample size selection and robustness and that the *covsim* package in R in conjunction with Mplus allows us to explore the effects of different types of non-normality other than through mixed normal distributions in Mplus and using a non-Gaussian copula (which is used by Mplus).

Another way of using the *Data Generation I* program is to use it to generate scores for parts of a population model and then add the additional data using the R random number generators to finish out score generation. I illustrate this approach for the design of a simulation in Mplus that uses the Mplus external simulation. The population model for my simulation is a simple two factor model where I want to estimate the covariance/correlation between the two latent factors but where one of the factors has a non-normal distribution (typically in SEM, latent variables are assumed to be normally distributed). Each factor has three indicators. Here is the population model and the parameter values I will use (note: each latent factor has a variance of 1.0 and hence, a standard deviation of 1.0):



All indicators have unstandardized factor loadings of 1.0. The error variance for each indicator is 0.25 which maps onto each indicator having a reliability of 0.80, that is 80% of the variation in each X indicator is systematic and 20% is random error.

I begin by creating a partial data set in R using the *Data Generation I* program. I specify 1,000 simulations each with my target sample size per simulation replicate, which in this example will be 300. If I collapse across the simulation trials, I have 300,000 cases from the larger population. I specify a 2X2 covariance matrix for input into the *Data Generation I* program that sets the variances of the two variables to be generated to 1.0 and the covariance between them to 0.00. These scores represent scores on Factor 1 and Factor 2, respectively. I set the covariance between the factors to zero because I am interested in whether the non-normality I introduce affects the Type I error rate for the test of the covariance between the factors. Suppose I decide to explore the case where the non-normality for one of the factors has a skewness value of 1.4 and an excess kurtosis value of 3.0. (I show you the resulting histogram for it shortly so you can see the distribution shape). I therefore specify 1.4 and 3.0 as my skewness and excess kurtosis values when generating the scores in *Data Generation I*. For the second factor, I set the skewness and excess kurtosis values to 0 thereby creating a normal distribution for that factor. I set the data generation option to 2 in the *Data Generation I* program and the file format option to 2. After creating the relevant R syntax in *Data Generation I* and pasting it into R and executing it, there will be two generated scores in an R data file called var1 and var2. The histogram generated by the program for var2 will look like a normal distribution. Here is the histogram generated by the program for var1with a normal curve superimposed on it:



I keep R open with access to var1 representing factor scores for Factor 1 and var2 representing factor scores for Factor 2. I next use R to generate the error scores for the X variables. I use the R command rvnorm(300000,mean=0,sd=.50) which I execute six times, one for each X indicator. The command generates a random variable for 300,000 cases where scores on the variable are normally distributed with a mean of zero and a standard deviation of 0.50 (which corresponds to a variance of 0.25):

```
e1<-rnorm(300000,mean=0,sd=.50)
e2<-rnorm(300000,mean=0,sd=.50)
e3<-rnorm(300000,mean=0,sd=.50)
e4<-rnorm(300000,mean=0,sd=.50)
e5<-rnorm(300000,mean=0,sd=.50)
e6<-rnorm(300000,mean=0,sd=.50)
```

I next create the scores for the six X variables by executing the following commands (which capitalizes on the already executed R code in the main run, which created a data file called dat2):

```
attach(dat2)
X1<-(1.0)*var1+e1
X2<-(1.0)*var1+e2
X3<-(1.0)*var1+e3
X4<-(1.0)*var2+e4
X5<-(1.0)*var2+e5
```

```
X6<-(1.0)*var2+e6
```

The next two R commands combine the generated variables into a single data file called dat3 followed by the creation of a new variable that contains the name of the folder and data file (with no tag) where you want to store the Mplus external simulation files:

```
dat3<-as.data.frame(cbind(trial,id, X1,X2,X3,X4,X5,X6))
namefile<-'c:/ret/simdat'
```

Finally, you need to save the 1,000 data files in a way that is compatible with the external simulation structure of Mplus. The following code will accomplish this task, so you copy and paste it into R after typing the above two lines (which assume your saved data files will be named simdat):

```
simdat<-split(dat3,trial)
jtemp<-list()
filename2<-paste(namefile,'list.txt',sep='')
for (i in names(simdat)) {
filename1<-paste(namefile,i,'.txt',sep='')
jtemp[[i]]<-filename1
write.table(simdat[[i]],filename1,row.names=F,col.names=F,quote=F,sep='  ') }
jtemp2<-as.matrix(jtemp)
write.table(jtemp2,filename2,row.names=F,col.names=F,quote=F)
```

This code writes onto your computer separate files called `simdat#.txt` for each simulation trial but substitutes a number for the # to correspond to the trial number. In addition, a file called `simdatlist.txt` is written to your computer that lists all of the replication files for input into the primary Mplus syntax that is shown below. Watch the video associated with the *Data Generation I* program to see how I execute the above.

After executing the above R syntax, I execute the following Mplus syntax to conduct the simulation:

```
TITLE: LOCAL EXTERNAL SIMULATION ;
DATA: FILE is c:/temp/simdatlist.txt ;
TYPE = MONTECARLO ;
VARIABLE:
 NAMES ARE trial id x1 x2 x3 x4 x5 x6  ;
 USEVARIABLES ARE x1 x2 x3 x4 x5 x6 ;
ANALYSIS:
 ESTIMATOR = MLR ;
MODEL:
 f1 BY x1@1 x2*1 x3*1 ;
 f2 BY x4@1 x5*1 x6*1 ;
```

```
 x1*.25 ; x2*.25 ; x3*.25 ;
 x4*.25 ; x5*.25 ; x6*.25 ;
 f1 WITH f2*0 ;
OUTPUT: TECH9 ;
```

All of the syntax should be familiar. Note that I specify the population values for each parameter after a @ or * in the `MODEL` section. This tells Mplus the population values to use when calculating confidence interval coverage. Here is the simulation output for the chi square statistic:

```
Chi-Square Test of Model Fit

        Degrees of freedom                        8

        Mean                                  8.268
        Std Dev                               4.387
        Number of successful computations      1000

             Proportions                  Percentiles
        Expected    Observed         Expected       Observed
           0.990       0.984            1.646          1.508
           0.980       0.971            2.032          1.693
           0.950       0.939            2.733          2.487
           0.900       0.888            3.490          3.378
           0.800       0.790            4.594          4.493
           0.700       0.706            5.527          5.554
           0.500       0.524            7.344          7.591
           0.300       0.336            9.524         10.057
           0.200       0.230           11.030         11.428
           0.100       0.120           13.362         13.967
           0.050       0.061           15.507         16.052
           0.020       0.028           18.168         19.922
           0.010       0.019           20.090         21.743
```

The average chi square value (8.268) should approximate the degrees of freedom, 8. The two values are reasonably close. The standard deviation of the chi square values should equal the square root of double the degrees of freedom. The square root of 16 is 4.00, which roughly equals the reported standard deviation value of 4.387 on the output. In the column `Proportions Expected` at the row for a theoretical p value of 0.05, the entry in the `Proportions Observed` column is 0.06. This indicates that the chi square test is rejecting about 6% of the models when it should be rejecting only 5% of the models, the alpha level expressed in percent from. However, these values seem reasonably close. I also note that in a chi square distribution with 8 degrees of freedom, one expects the critical value of chi square that rejects 5% of the models to equal 15.507. The critical value in the observed

data that did so was 16.052.

Here are the results for the parameter estimates:

```
MODEL RESULTS

                    ESTIMATES             S. E.     M. S. E.  95%  % Sig
               Population   Average   Std. Dev.   Average            Cover Coeff

 F1       BY
  X1                1.000    1.0000     0.0000    0.0000     0.0000 1.000 0.000
  X2                1.000    0.9998     0.0463    0.0456     0.0021 0.953 1.000
  X3                1.000    0.9986     0.0451    0.0455     0.0020 0.954 1.000
                    ESTIMATES             S. E.     M. S. E.  95%  % Sig
               Population   Average   Std. Dev.   Average            Cover Coeff

 F2       BY
  X4                1.000    1.0000     0.0000    0.0000     0.0000 1.000 0.000
  X5                1.000    1.0013     0.0464    0.0456     0.0021 0.944 1.000
  X6                1.000    1.0003     0.0464    0.0455     0.0022 0.943 1.000

 F1       WITH
  F2                0.000   -0.0005     0.0598    0.0620     0.0036 0.951 0.049

Residual Variances
  X1                0.250    0.2473     0.0330    0.0318     0.0011 0.932 1.000
  X2                0.250    0.2468     0.0326    0.0320     0.0011 0.934 1.000
  X3                0.250    0.2498     0.0322    0.0320     0.0010 0.937 1.000
  X4                0.250    0.2479     0.0327    0.0319     0.0011 0.939 1.000
  X5                0.250    0.2477     0.0332    0.0319     0.0011 0.930 1.000
  X6                0.250    0.2482     0.0322    0.0320     0.0010 0.946 1.000
```

None of the estimates show consequential bias in either the parameter estimate per se or their standard errors. The confidence interval coverage is generally reasonable although a bit on the low side for the residual variances. The Type I error rate for the factor covariance was 0.049 which is close to the theoretical alpha level of 0.05. Overall, the non-normal factor does not seem to be creating problems, at least for the levels of skewness and excess kurtosis I explored.

The second program on my website relevant to simulations is *Data Generation II*. This program shows you histograms and descriptive statistics for a wide range of non-normal distributions that helps you make choices about levels of skewness and excess kurtosis to use for distributions from the Pearson distribution family in *Data Generation I*. For example, my choice of skewness and excess kurtosis for the non-normal distribution in the prior example was based on my examination of different chi squared distributions in *Data Generation I*. Watch the video for the *Data Generation I* program to see strategies you can use to help you select distribution values. The *Data Generation II* program also generates randomly selected scores from non-normal distributions of your choice that you

can then use in conjunction with R code and Mplus to conduct simulations. For example, in the above simulation on factor analysis, instead of using a non-normal distribution with a priori specified levels of skewness and excess kurtosis, I might use scores from an exponential distribution generated by the *Data Generation II* program.

LOCALIZED SIMULATIONS FOR MODERATION

In this section, I show you how to conduct power analysis simulations for moderated effects in which the variables are measured with single indicators. Ironically, this scenario can be more challenging to simulate than moderation between two latent variables or between an observed variable and a latent variable. The latter cases use Mplus programming with the `XWITH` command and can be implemented straightforwardly using methods discussed in Chapter 28 and in this document. In this section, I focus on the case where Y is an outcome that ranges from, say, -5 to +5, Z is a continuous moderator that ranges from -5 to +5, and T is a binary treatment condition, 1 = intervention, 0 = control group. The model focuses on how the effect of T on Y varies as a function of Z and relies on the use of product terms. The programming logic readily generalizes to cases where Z is binary or where both the predictors are continuous, such as the case of two continuous mediators. I also have discussed in another section of this document how you would evaluate power in a simulation study for moderation with multigroup analysis, so I do not consider that here.

To make the logic explicit, I briefly review the core equations from product term analysis of moderation as discussed in Chapter 19. The equation has the form

$$Y = a_1 + b_1 T + b_2 Z + b_3 TZ \qquad [14]$$

The coefficient b_1 reflects the estimated effect of T on Y when Z = 0; a_1 reflects the estimated mean of Y when both T and Z = 0, so it is the mean for the control group when the moderator equals 0, in this case the middle score of the Z metric. For traditional product term analysis, the coefficient b_1 is conceptualized as a linear function of Z, as follows:

$$b_{1j} = a_2 + b_4 Z_j \qquad [15]$$

where a_2 is the value of b_1 when $Z_j = 0$ and b_4 tells us for every one unit that Z increases, how much b_1 is predicted to change. If a_2 is 0.3, it means that when Z = 0, the effect of T on Y (or the mean Y difference between the intervention and control groups) is 0.30. As such, a_2 often is of interest when a value of Z = 0 is substantively meaningful. It turns out that the value of a_2 will equal the value of b_1 in Equation 11. It also turns out that the value of b_4 will equal the value of b_3 in Equation 11. Given this, when we estimate Equation 11 with sample data, the coefficients for b_1 and b_3 provide useful information about

moderation (see Chapter 19 for more details). Note that for later programming purposes, Equation 12, does not have a disturbance term, i.e., its variance is fixed at zero. I discussed the reasons for this property in Chapter 19.

The programming strategy I show you makes use of the Mplus external simulation features, which I illustrated in Chapter 28 when I considered listwise deletion of missing data. I use this strategy because of its flexibility, as illustrated later. I assume you have read the section on listwise deletion of missing data in Chapter 28 to provide an appropriate framework for my exposition here. The analysis requires two steps. The first step is to generate the data on your computer for purposes of simulation analysis. The second step is to analyze the generated data. Table 20 presents the relevant syntax for Step 1 in which I evaluate power of the moderation effect for a sample size of 250 and where the population product term coefficient equals 0.30 and the population b_1 equals 0.30.

Table 20: Simulation for Product Terms: Step 1

```
1.  TITLE: STEP 1 FOR PRODUCT POWER ANALYSIS
2.  MONTECARLO:
3.  NAMES = y z t;
4.  NOBS = 250;
5.  NREPS = 500;
6.  SEED = 2222 ;
7.  REPSAVE = all;
8.  SAVE = rep*.dat;
9.  CUTPOINTS = t(0);
10. MODEL POPULATION:
11.  t*1;
12.  [t*0] ;
13.  t WITH z *0 ;
14.  [z*0];          !set mean of moderator z to 0
15.  z*1;           !set moderator variance to 1.0
16.  b1 | y on t;   !define b1 as the effect of t on y
17.  y ON z*.2 ;    !this is b2 in Eq 11
18.  b1 ON z*.2;    !this is b3 in Eq 11 and b4 in Eq 12, which are equal
19.  [b1*.3];       !this is b1 in Eq 11 and a2 in Eq 12, which are equal
20.  b1@0;          !set the disturbance variance of Eq 12 to zero
21.  [y*0];         !mean y when t = 0 and z = 0
22.  y*1;           !set disturbance variance to 1.0
23. ANALYSIS:
24.  TYPE=RANDOM ; !need to invoke to use Line 16
25. MODEL:
26.  b1 | y on t;   !define b1 as the effect of t on y
27.  y ON z*.2 ;    !this is b2 in Eq 11
28.  b1  ON z*.2;   !this is b3 in Eq 11 and b4 in Eq 12, which are equal
29.  [b1*.3];       !this is b1 in Eq 11 and a2 in Eq 12, which are equal
30.  b1@0;          !set the disturbance variance of Eq 12 to zero
```

```
31.  [y*0];         !mean y when t = 0 and z = 0
32.  y*1;           !set disturbance variance to 1.0
33. OUTPUT: TECH9 ;
```

The syntax coupled with the comment lines should be self-explanatory given material already covered in Chapter 28 and this supplement. The primary function of this program in to generate the data sets for analysis and the file that lists those data sets vis-à-vis Lines 7 and 8 (the format of these lines is discussed in the listwise missing data example in Chapter 28). I make careful note at the end of the Step 1 output the order in which variables are written to the data sets as this impacts the NAMES statement in the Step 2 program:

```
SAVEDATA INFORMATION

  Order of variables

    Y
    Z
    T

  Save file
    rep*.dat
```

Table 21 presents the syntax for Step 2 that performs the simulation analysis.

Table 21: Simulation for Product Terms: Step 2

```
1.  TITLE: STEP 2 PRODUCT POWER ANALYSIS ;
2.  DATA:
3.  FILE = replist.dat ;
4.  TYPE = MONTECARLO ;
5.  VARIABLE:
6.  NAMES ARE y z t  ;
7.    USEVARIABLES ARE y z t prod ;
8.  DEFINE:
9.   !z = z-1 ;
10.  prod=z*t ;
11. ANALYSIS:
12. ESTIMATOR = MLR;
13. MODEL:
14. y ON z*.20 t*.30 prod*.20  ;
15. y*1 ;
16. OUTPUT: TECH9 ;
```

All of the syntax should be familiar. Line 3 points to the file that was generated by the Step 1 program that lists the different data set names. Note that I can use the DEFINE command

to calculate the product term for analysis per Line 10. However, it was essential in the Step 1 program that I made the product term impact the outcome and that I coordinate the values between the programs for everything to work properly. Although I commented it out, Line 9 can be used to redefine the zero point of Z per chapter 19 to explore power for the simple effect of T on Y at different levels or values of Z.

Here is the core output:

```
                        ESTIMATES              S. E.   M. S. E.   95%  % Sig
                Population   Average   Std. Dev.    Average             Cover  Coeff

 Y        ON
  Z                 0.200    0.2042     0.0875     0.0877     0.0077   0.936  0.658
  T                 0.300    0.3120     0.1286     0.1262     0.0166   0.944  0.698
  PROD              0.200    0.1988     0.1293     0.1253     0.0167   0.938  0.370
```

The statistical power for the coefficient associated with the product term is 0.37.

As another example, I show the Step 1 syntax in Table 22 for the case of an equation with a continuous outcome and two continuous mediators, m1 and m2, and where m2 moderates the effect of m1 on y. The sample size I use is again 250 and the population product term coefficient equals 0.15 and the population b_1 equals 0.30.

Table 22: Simulation for Continuous Variable Product Term: Step 1

```
1.  TITLE: STEP 1 FOR PRODUCT POWER ANALYSIS
2.  MONTECARLO:
3.  NAMES = y m2 m1;
4.  NOBS = 250;
5.  NREPS = 500;
6.  SEED = 2222 ;
7.  REPSAVE = all;
8.  SAVE = rep*.dat;
9.  MODEL POPULATION:
10.  m1*1;              !set variance of m1 to 1
11.  [m1*0] ;           !set mean of m1 to 0
12.  m2*1 ;             !set variance of m2 to 1
13.  [m2*0] ;           !set mean of m2 to 0
14.  m1 WITH m2 *.25 ;  ! covariance between m1 and m2
15.  b1 | y on m1;      !define b1 as the effect of m1 on y
16.  y ON m2*.2 ;       !this is b2 in Eq 11
17.  b1 ON m2*.15;      !this is b3 in Eq 11 and b4 in Eq 12, which are equal
18.  [b1*.30];          !this is b1 in Eq 11 and a2 in Eq 12, which are equal
19.  b1@0;              !set the disturbance variance of Eq 12 to zero
20.  [y*0];             !mean y when t = 0 and z = 0
21.  y*1;               !set disturbance variance to 1.0
22. ANALYSIS:
```

```
23. TYPE = RANDOM ;
24. MODEL:
25.  b1 | y on m1;       !define b1 as the effect of m1 on y
26.  y ON m2*.2 ;         !this is b2 in Eq 11
27.  b1 ON m2*.15;        !this is b3 in Eq 11 and b4 in Eq 12, which are equal
28.  [b1*.30];            !this is b1 in Eq 11 and a2 in Eq 12, which are equal
29.  b1@0;               !set the disturbance variance of Eq 12 to zero
30.  [y*0];              !mean y when t = 0 and z = 0
31.  y*1;                !set disturbance variance to 1.0
32. OUTPUT: TECH9 ;
```

Table 23 shows the Step 2 syntax, which is self-explanatory.

Table 23: Simulation for Continuous Variable Product Term: Step 2

```
1.  TITLE: STEP 1 FOR PRODUCT POWER ANALYSIS
2.  DATA:
3.  FILE = replist.dat ;
4.  TYPE = MONTECARLO ;
5.  VARIABLE:
6.  NAMES ARE y m2 m1  ;
7.  USEVARIABLES ARE y m2 m1 prod ;
8.  DEFINE:
9.  !m2 = m2-1 ;
10. prod=m1*m2 ;
11. ANALYSIS:
12.   ESTIMATOR = MLR;
13. MODEL:
14.  y ON m1*.30 m2*.20 prod*.15  ;
15.  y*1 ;
16. OUTPUT: TECH9 ;
```

Here is the output table with the relevant power values:

		ESTIMATES		S. E.	M. S. E.	95%	% Sig
	Population	Average	Std. Dev.	Average		Cover	Coeff
Y ON							
M1	0.300	0.3018	0.0680	0.0653	0.0046	0.942	0.998
M2	0.200	0.2032	0.0635	0.0646	0.0040	0.954	0.894
PROD	0.150	0.1518	0.0642	0.0608	0.0041	0.930	0.698

The statistical power for the coefficient associated with the product term is 0.70.

CREATING UNEQUAL N AND THREE OR MORE TREATMENT GROUPS

Throughout this document, I have defined two groups of equal size for the treatment versus control conditions for an RET by use of the Mplus CUTPOINTS option. If need be, you can create unequal sizes between the groups in the referent populations. Mplus initially defines the specified variable in the CUTPOINTS command, in this case t, as continuous and normally distributed with a mean of 0 and a variance of 1. When I specify a cutpoint of 0 on the command, I essentially create equal sample sizes in the resulting two groups for t. because half the scores are above zero in the distribution and half are below zero. If I instead specify the cutpoint as 1.0, t is cut at the value of 1.0, which is one standard deviation above the mean (because the mean and variance used for data generation are zero and one). This means that after the cut, t is a 0/1 binary variable where 16 percent of the population have the value of 1. The default cutpoint values in Mplus essentially take the form of a z-score, so you can use a z-score table to select the value that corresponds to the proportional split you desire. For example, a cutpoint of -1.65 creates a variable with 5% zeros and 95% 1s.

In multiple group analysis, the CUTPOINT option is specified as follows where the cutpoints for the groups are separated using the | symbol:

```
CUTPOINTS = x1 (0) x2 ( 1) | x1 (1) x2 (0);
```

where the cutpoints before the | symbol are the cutpoints for group 1 and the cutpoints after the | symbol are the cutpoints for group 2.

If you want to conduct a simulation with more than two groups (e.g., a three group treatment condition variable), then you will need to either use a multigroup strategy or use a dummy variable strategy with the two step external Mont Carlo approach, much like I illustrated for listwise deletion of missing data in Chapter 28. In the dummy variable approach, the Step 1 program generates the continuous variable and the Step 2 program uses the DEFINE command to cut it up into categories using the CUT option (not the CUTPOINT option) in the DEFINE command at Step 2. See the Mplus manual for how the CUT option works.

LOCALIZED SIMULATIONS FOR MULTILEVEL SEM

In this section, I show you how to conduct power analysis simulations for multilevel SEM models. I use an example with an outcome variable (y) and two mediators (m1 and m2) and a treatment variable (treat) in which 50 clusters of size 20 each are randomly assigned to a treatment or control condition. The Mplus code uses principles already developed in

this document, so I do not repeat those principles here. I used Bayes estimation and constructed the simulation such that the across cluster effect of the treatment condition on the outcome accounted for 1% of the variation in m1 (a Cohen's d of approximately 0.10) and 6% of the variation in m2 (a Cohen's d of approximately 0.50). The two mediators accounted for approximately 30% of the variance in the across cluster outcomes, representing standardized regression coefficients of approximately 0.20 and 0.50, respectively. To make it easier to specify parameter values, I structured the simulation so that the within and between clusters variances of the variables were approximately equal to 1.0. Table 24 presents the Mplus syntax.

Table 24: Syntax for Multilevel SEM for Clustered RET

```
1.  TITLE: Multilevel SEM  ;
2.  MONTECARLO:
3.  NAMES ARE
4.    y t m1 m2 ;
5.  NREPS = 1000 ;                ! number of simulation reps
6.  !NREPS = 1 ;                  ! used to test choice of parameter values
7.  NCSIZES = 1  ;                ! just one cluster size
8.  NOBSERVATIONS = 1000 ;        ! sample size per simulation replicate
9.  !NOBSERVATIONS = 1000000 ;    ! used to test choice of parameter values
10. CSIZES 50(20) ;               ! number of clusters and size of clusters
11. !CSIZES 50000(20) ;           ! used to test choice of parameter values
12. SEED = 89939 ;
13. !SAVE = simpow.dat;             ! used to test choice of parameter values
14. CUTPOINTS = t(0);
15. BETWEEN IS t ;
16. ANALYSIS:
17.   TYPE = TWOLEVEL  ;
18.   ESTIMATOR = BAYES ;
19. MODEL POPULATION:
20.   %WITHIN%
21.    y*.71
22.    m1*1 ; m2*1 ;
23.    y ON m1*.20 m2*.50  ;
24.    m1 WITH m2*.1 ;             ! covariance between m1 and m2
25.   %BETWEEN%
26.    [t*0]; t*1 ;
27.    [y*0] ;  y*.71 ;
28.    [m1*0] ; m1*.99;
29.    [m2*0] ; m2*.9375 ;
30.    y ON m1*.20 m2*.50 ;
31.    m1 on t*.2;
32.    m2 ON t*.5 ;
33.    m1 WITH m2*.1;              ! covariance between m1 and m2
34.  MODEL :
```

```
35.    %WITHIN%
36.     y*.71
37.     m1*1 ; m2*1 ;
38.     y ON m1*.20 m2*.50  ;
39.     m1 WITH m2*.1 ;
40.    %BETWEEN%
41.     [y*0] ;  y*.71 ;
42.     [m1*0] ; m1*.99;
43.     [m2*0] ; m2*.9375 ;
44.     y ON m1*.20 m2*.50 ;
45.     m1 on t*.2;
46.     m2 ON t*.5 ;
47.     m1 WITH m2*.1;
48. OUTPUT:
```

Lines 6, 9 11 and 13 are used to double check the parameter values. I will explain their use shortly. Line 7 specifies the number of different cluster sizes you want to use. In this case, there is only one cluster size. Line 10 tells Mplus how many clusters you want (in this case 50) and the size of each cluster (in this case, 20 observations per cluster). Suppose instead I specified 3 cluster sizes. The format of `CSIZES` might then be as follows:

```
CSIZES = 40 (5) 50 (10) 20 (15);
```

This tells Mplus to create 40 clusters of size 5, 50 clusters of size 10 and 20 clusters of size 15. Line 8 tells Mplus the total number of observations to generate. The remainder of the syntax is self-explanatory.

I like to double check the parameter values I chose and examine the squared correlations for the endogenous variables to gain additional perspectives on the simulated effect sizes. By uncommenting Lines 6, 9, 11 and 13 and commenting out the duplicate lines just above Lines 6, 9 and 11. This has the effect of creating one very large data set of 1,000,000 cases that I then analyze using a standard Bayesian multilevel SEM (note: I need to be careful to reorder the input variables because Mplus saves them to the data file in a different order than how they were generated; Mplus also generates a cluster number variable). Table 25 presents the "test" syntax code I execute:

Table 25: Syntax for Test Run

```
TITLE: Test analysis ;
DATA: FILE IS simpow.dat ;
VARIABLE:
NAMES ARE
   y  m1 m2 t clus ;
USEVARIABLES ARE
   y  m1 m2 t ;
```

```
  CLUSTER is clus ;
  BETWEEN IS t ;
  ANALYSIS:
    TYPE = TWOLEVEL  ;
    ESTIMATOR = bayes ;
  MODEL:
   %WITHIN%
    y*.71
    m1*1 ; m2*1 ;
    y ON m1*.20 m2*.50  ; !100+100+2*10*10*.3 = 260
    m1 WITH m2*.1 ;
   %BETWEEN%
    [y*0] ;  y*.71 ;
    [m1*0] ; m1*.99; [m2*0] ; m2*.9375 ;
    y ON m1*.20 m2*.50 ;
    m1 on t*.2;
    m2 ON t*.5 ;
    m1 WITH m2*.1;
  OUTPUT: Samp STAND(STDYX) Cinterval ;
```

Here is the output for the squared correlations, which is in accord with what I expected:

```
R-SQUARE

Within Level

                                 Posterior  One-Tailed         95% C.I.
    Variable        Estimate       S.D.      P-Value   Lower 2.5%  Upper 2.5%

    Y                  0.305       0.001      0.000       0.304       0.306

Between Level

                                 Posterior  One-Tailed         95% C.I.
    Variable        Estimate       S.D.      P-Value   Lower 2.5%  Upper 2.5%

    Y                  0.303       0.004      0.000       0.296       0.311
    M1                 0.010       0.001      0.000       0.009       0.012
    M2                 0.063       0.002      0.000       0.058       0.066
```

Returning to the simulation in Table 24, here is the core output for the power analysis:

```
                          ESTIMATES              S. E.     M. S. E.  95%  % Sig
                Population   Average   Std. Dev.   Average             Cover Coeff
Within Level

 Y          ON
  M1               0.200     0.1999     0.0274     0.0275     0.0008 0.944 1.000
  M2               0.500     0.5009     0.0281     0.0275     0.0008 0.926 1.000
```

```
Between Level

 Y          ON
  M1                0.200      0.1990      0.1343      0.1362      0.0180 0.944 0.334
  M2                0.500      0.4986      0.1303      0.1366      0.0170 0.947 0.948

 M1         ON
  T                 0.200      0.2010      0.3004      0.3068      0.0901 0.947 0.117

 M2         ON
  T                 0.500      0.4981      0.2848      0.3006      0.0810 0.946 0.373
```

The parameter estimates generally were unbiased as were the "standard errors." The power estimates are in the last column. Information about the width and coverage of the credible intervals also is provided.

LOCALIZED SIMULATIONS FOR ROBUST CLUSTERED SEM

Mplus does not have the option to directly conduct a Monte Carlo simulation using COMPLEX analysis. Rather, you must do so in two steps by invoking the Mplus external simulation approach. The first step generates the clustered data to be analyzed. In this example, I use the multilevel SEM program from Table 24, which I reproduce here but with two changes shown in red that save the data for the external simulation. The name of the data file in Line 11 must end with a * because each separate saved data file is differentiated with a number in place of the *.

Table 26: Step 1 for External Simulation

```
1.  TITLE: Multilevel SEM  ;
2.  MONTECARLO:
3.  NAMES ARE
4.     y t m1 m2 ;
5.  NREPS = 1000 ;                ! number of simulation reps
6.  NCSIZES = 1  ;                ! just one cluster size
7.  NOBSERVATIONS = 1000 ;        ! sample size per simulation replicate
8.  CSIZES 50(20) ;               ! number of clusters and size of clusters
9.  SEED = 89939 ;
10. REPSAVE = ALL ;               ! save generated data for Step 2 input
11. SAVE = extern*.dat;           ! name of file to save data in
12. CUTPOINTS = t(0);
13. BETWEEN IS t ;
14. ANALYSIS:
15.    TYPE = TWOLEVEL  ;
16.    ESTIMATOR = BAYES ;
17. MODEL POPULATION:
18.    %WITHIN%
```

```
    y*.71
    m1*1 ; m2*1 ;
    y ON m1*.20 m2*.50  ;
    m1 WITH m2*.1 ;              ! covariance between m1 and m2
   %BETWEEN%
    [t*0]; t*1 ;
    [y*0] ;  y*.71 ;
    [m1*0] ; m1*.99;
    [m2*0] ; m2*.9375 ;
    y ON m1*.20 m2*.50 ;
    m1 on t*.2;
    m2 ON t*.5 ;
    m1 WITH m2*.1;               ! covariance between m1 and m2
  MODEL :
   %WITHIN%
    y*.71
    m1*1 ; m2*1 ;
    y ON m1*.20 m2*.50  ;
    m1 WITH m2*.1 ;
   %BETWEEN%
    [y*0] ;  y*.71 ;
    [m1*0] ; m1*.99;
    [m2*0] ; m2*.9375 ;
    y ON m1*.20 m2*.50 ;
    m1 on t*.2;
    m2 ON t*.5 ;
    m1 WITH m2*.1;
OUTPUT:
```

In Step 2, I execute the COMPLEX samples program of interest and direct it to the saved data file from Step 1, per Table 27. Note that the name of the data file corresponds to the name of the data file in Step 1 but now has “list” at the end of the file name.

Table 27: Sep 2 for External Simulation

```
TITLE: Step 2 of external simulation ;
DATA: FILE = externlist.dat ;
TYPE = MONTECARLO;
VARIABLE: NAMES = y m1 m2 t clus ;
USEVARIABLES = y m1 m2 t ;
CLUSTER = clus;
ANALYSIS: TYPE = COMPLEX;
MODEL:
Y ON m1*.2 m2*.5 ;
m1 ON t*.2 ;
m2 ON t*.5 ;
[y*0] ; [m1*0] ; [m2*0] ;
```

```
OUTPUT: TECH9;
```

Here are the core results for the power analysis:

```
                                ESTIMATES              S. E.     M. S. E.  95%  % Sig
                   Population   Average   Std. Dev.   Average             Cover Coeff

 Y          ON
  M1                    0.200     0.1997     0.0638     0.0616     0.0041 0.934 0.878
  M2                    0.500     0.4994     0.0626     0.0612     0.0039 0.941 1.000

 M1         ON
  T                     0.200     0.2002     0.2980     0.2854     0.0887 0.943 0.124

 M2         ON
  T                     0.500     0.4977     0.2835     0.2783     0.0803 0.939 0.435
```

LOCALIZED SIMULATIONS WITH BOOTSTRAPPING

W conduct many analyses in SEM using bootstrapping. There are conflicting recommendations on the what sample size is appropriate for bootstrapping. This is not surprising because the performance of the bootstrap can be both model and data dependent. You can use localized simulations to determine how bootstrapping fares in terms of parameter bias, standard error bias, and confidence interval coverage for your model and for different sample sizes. You structure the simulation exactly as you would for a standard simulation with MLR, but on the `ANALYSIS` command, you indicate use of bootstrapping in the usual fashion. For example, here are the two relevant syntax lines from the simulation that uses MLR in Table 1:

```
ANALYSIS:
ESTIMATOR = MLR ;
```

These would be changed to

```
ANALYSIS:
ESTIMATOR = ML ; BOOTSTRAPS = 1000 ;
```

The simulation study will then use bootstrapping with 1,000 bootstrap replicates per simulation replicate. These simulations can take considerable processing time to execute, so be patient. You can speed things up by setting the number of bootstrap and simulation replicates to a smaller number, but this can lead to a sacrifice in precision.

REFERENCES

Agresti, A. & Caffo, B. (2000). Simple and effective confidence intervals for proportions and differences of proportions result from adding two successes and two failures. *American Statistician*, 54, 280-288.

Astivia, O. L. O., & Zumbo, B. D. (2015). A cautionary note on the use of the Vale and Maurelli method to generate multivariate, nonnormal data for simulation purposes. *Educational and Psychological Measurement, 75*, 541–67.

Auerswald, M. (2017). Generating non-normal distributions: Methods and effects. University of Mannheim.

Cario, M. C., & Nelson, B. L. (1997). Modeling and generating random vectors with arbitrary marginal distributions and correlation matrix. *Technical Report, Department of Industrial Engineering and Management Sciences* (pp. 1–19). Northwestern University

Fleishman, A. I. (1978). A method for simulating non-normal distributions. *Psychometrika*, 43, 521–532.

Foldnes, N., & Gronneberg, S. (2015). How general is the Vale-Maurelli simulation approach? *Psychometrika, 80*, 1066–83.

Foldnes, N. & Olson, U. (2016). A simple simulation technique for nonnormal data with prespecified skewness, kurtosis, and covariance matrix. *Multivariate Behavioral Research*, 51, 207-219.

Grønneberg, S., Foldnes, N. & Marcoulides, K. (2022). covsim: An R Package for simulating non-normal data for structural equation models using copulas. *Journal of Statistical Software*, 102.

Journal of Statistical Software, 102.

Headrick, T. C. (2002). Fast fifth-order polynomial transforms for generating univariate and multivariate nonnormal distributions. *Computational Statistics and Data Analysis*, 40, 685–711

Headrick, T. C. (2004). On polynomial transformations for simulating multivariate non-normal distributions. *Journal of Modern Applied Statistical Methods*, 3, 65–71.

Muthén, B. & Asparouhov, T. (2002). Using Mplus Monte Carlo simulations in practice: A note on non-normal missing data in latent variable models https://www.statmodel.com/download/webnotes/mc2.pdf

Muthén, B. & Asparouhov T. (2015). Growth mixture modeling with non-normal distributions. *Statistics in Medicine*, 34, 1041–1058.

Muthén, L. & Muthén, B. (2002). How to use a Monte Carlo study to decide on sample size and determine power. *Structural Equation Modeling*, 4, 599-620.

Muthén, B. & Curran, P. (1997). General longitudinal modeling of individual differences in experimental designs: A latent variable framework for analysis and power estimation. *Psychological Methods*, 2, 371–402.

Ray, R. & Lindsay, B. (2005). The topography of multivariate normal mixtures. *Annals of Statistics*, 33, 2042–2065.

Vale, C. D., & Maurelli, V. A. (1983). Simulating multivariate nonnormal distributions. *Psychometrika*, 48, 465–471.

Wilcox, R. (2021). *Introduction to robust estimation and hypothesis testing*. Academic Press (Fifth edition).

APPENDIX: LOCATING VALUES IN RESULTS.TXT FILES

This Appendix describes an alternative method for identifying the location of parameters in the `results.txt` file for the joint significance test power analysis other than using the technical matrices associated with `TECH1`. When you run the Step 1 syntax, Line 8 of Table 6 saves the data for the first simulated replicate in the file called `temp.dat`. We can analyze that data for the first replicate by applying the sample model to it using the following Mplus syntax:

```
TITLE: FILE CHECK ;
DATA:
FILE IS temp.dat ;
VARIABLE:
NAMES ARE m y t ;
ANALYSIS:
ESTIMATOR = MLR ;
MODEL:            !specify analysis model
y ON m  ;         !outcome equation
m ON t ;          !mediation equation
y* ;              !disturbance variance for y
m* ;              !disturbance variance for m
MODEL INDIRECT:
y IND t ;
OUTPUT: SAMP STDYX TECH1 ;
```

After the syntax is executed, we examine the output for the values of p_1 and p_2 and their estimated standard errors. Here is the relevant output:

```
MODEL RESULTS

                                                    Two-Tailed
                    Estimate       S.E. Est./S.E. P-Value

 Y        ON
    M                  0.168      0.079      2.115      0.034

 M        ON
    T                  2.710      2.093      1.295      0.195
```

Using a text editor, I next open the `results.txt` file that lists the results for all 20,000 replicates and I examine the results shown just for the first replicate. Here are the contents of the file for the first three replicates (note that most of the numbers are represented using scientific notation):

```
1
0.17702861E+001 0.27500146E+001 0.27109617E+001 0.16757173E+000 0.15808479E+003 0.20312839E+003
0.16206239E+001 0.12099228E+001 0.20933450E+001 0.79233057E-001 0.17054334E+002 0.21856593E+002
0.60000000E+001  -.12039552E+004  -.12039234E+004  0.24199105E+004  0.24379743E+004  0.24189854E+004
0.59860959E-001 0.10000000E+001 0.80671584E+000 0.10000000E+001
0.10000000E+001 0.00000000E+000 0.76612416E-002
2
```

```
0.41656261E+000 0.13437156E+001 0.67715328E+001 0.14607882E+000 0.23088046E+003 0.22836936E+003
0.18083595E+001 0.13039331E+001 0.24851695E+001 0.84489915E-001 0.24050656E+002 0.24166802E+002
0.60000000E+001  -.12411463E+004  -.12404379E+004  0.24942926E+004  0.25123564E+004  0.24933676E+004
0.14261109E+001 0.10000000E+001 0.23240019E+000 0.95196738E+000
0.85590214E+000 0.53298586E-001 0.35373158E-001
3
-.19770407E+001 0.21149300E+000 0.96978656E+001 0.31756576E+000 0.21319533E+003 0.16896376E+003
0.16622254E+001 0.11095465E+001 0.23673605E+001 0.76849539E-001 0.24555815E+002 0.19786739E+002
0.60000000E+001  -.12125743E+004  -.12116511E+004  0.24371487E+004  0.24552125E+004  0.24362237E+004
0.19576898E+001 0.10000000E+001 0.16176061E+000 0.97188661E+000
0.91565984E+000 0.79903683E-001 0.37165227E-001
```

The first line has the replicate number followed by the results for that replicate. I scan the numbers for the first replicate and find the numbers that match the p_1 and p_2 values and their estimated standard errors from the prior output. I have highlighted them in yellow here. I then count the entries sequentially starting from the beginning until I reach the highlighted numbers, and note that they are the third, fourth, ninth and tenth entries, meaning they will be in `f3`, `f4`, `f9` and `f10` in the Step 2 program.