Supplementary Material for Bauer & Hussong (2009), Psychometric Approaches for Developing Commensurate Measures Across Independent Studies: Traditional and New Models

Here we provide details on conducting integrative data analysis using the moderated nonlinear factor analysis model presented in Bauer & Hussong (2009). The topics we cover are, in this order: data preparation, model fitting, plotting of results, and scoring.

Preliminary Data Management

Before model fitting begins, the data must be properly prepared. The following steps must be completed:

(1) Measures in each of the studies to be combined must be named identically and scored identically. Identical scoring may require harmonization of the measures (see Bauer & Hussong, 2009).

(2) A study indicator variable must be created within each independent study data file. For instance, one might create the variable *study*, and score it 0 for the first study, 1 for the second study, etc.

(3) An ID variable must exist or be created that provides a unique designation for each individual in the two studies. No two people can have the same ID number or label within or across studies.

(4) The data files from the two studies must be stacked together in a single data file. In SAS, this can be accomplished within a DATA step using the SET statement. For instance, for the example in Bauer & Hussong (2009), the following code was used to create a stacked data file, *combine*, from the data files from the AFDP study conducted by Laurie Chassin, *chassin*, and the AHBP study of Kenneth Sher, *sher.*

```
data combine; set chassin sher;
run;
```

(5) If the data are longitudinal, as in the example from Bauer & Hussong (2009), then the selection of a “calibration sample” is necessary to ensure that the assumption of local independence is met for the nonlinear factor analysis. The code below randomly selects one observation per individual in the *combine* data set:

```
proc surveyselect data=combine method=srs n=1 seed=5 out=calib noprint;
  strata id;
run;
```

(The seed value is arbitrary and simply starts the random number generator used to select observations.)

Important: the example code above assumes that the stacked data file is in long format, that is, there are multiple lines of data for each person, with each line reflecting a particular assessment point or age*.* For instance, for the first four individuals, the layout of the *combine* data set looks like this:

```
study          id    ageyr    disorder    conseq      hvyuse     usefreq       expect

  0           102      14         0          0           0           1           2.00
  0           102      15         0          0           1           1           2.00
  0           102      16         0          0           1           1           1.50
  0           102      21         1          0           1           2           1.50
  0           109      12         0          0           0           0           0.00
  0           109      13         0          0           0           2           0.25
  0           109      14         0          0           0           0           0.00
  0           109      21         0          0           0           1           1.25
  0           110      14         0          0           1           2           1.00
  0           110      16         0          2           2           2           1.75
  0           110      17         1          2           2           2           2.00
  0           110      21         1          1           2           2           1.75
  0           111      15         0          0           2           2           2.50
  0           111      16         0          2           2           2           1.75
  0           111      17         0          3           2           2           2.00
  0           111      22         1          1           2           2           2.25
```

The SURVEYSELECT procedure randomly selects one observation per person, as indicated by the variable *id*, to put in the *calib* data set, e.g.,

```
study          id    ageyr    disorder    conseq      hvyuse     usefreq       expect

  0           102      15         0          0           1           1           2.00
  0           109      21         0          0           0           1           1.25
  0           110      21         1          1           2           2           1.75
  0           111      22         1          1           2           2           2.25
```

(6) To fit the moderated nonlinear factor analysis models in the NLMIXED procedure, this data must now be rearranged. The NLMIXED procedure requires the definition of a single outcome variable to contain all of the item responses. We will thus restructure the data so that each individual has five rows of data, and each row reflects the response to one of the five items (*disorder, conseq, hvyuse, usefreq, expect*). We will also create a new variable *item* that will be numbered sequentially to index item so that responses to different items can be differentiated. The code used to accomplish this is shown below. (Note that this code also creates a new age variable, *age17*, which is used to model age-related changes in alcohol involvement in some of the models described in Bauer & Hussong, 2009).

```
data nlfa; set calib;
 array dv [5] disorder conseq hvyuse usefreq expect;
 age17 = ageyr-17; *17 will be join point in later models;
 if ageyr > 17 then age17 = 0;
 do i = 1 to 5;
    item = i;
    resp = dv[i];
    output;
 end;
 keep id study ageyr age17 item resp;
run;
```

For the first four cases, the restructured data looks like this:

```
study          id    ageyr    age17    item    resp

  0           102      15       -2       1     0.00
  0           102      15       -2       2     0.00
  0           102      15       -2       3     1.00
  0           102      15       -2       4     1.00
  0           102      15       -2       5     2.00
  0           109      21        0       1     0.00
  0           109      21        0       2     0.00
  0           109      21        0       3     0.00
  0           109      21        0       4     1.00
  0           109      21        0       5     1.25
  0           110      21        0       1     1.00
  0           110      21        0       2     1.00
  0           110      21        0       3     2.00
  0           110      21        0       4     2.00
  0           110      21        0       5     1.75
  0           111      22        0       1     1.00
  0           111      22        0       2     1.00
  0           111      22        0       3     2.00
  0           111      22        0       4     2.00
  0           111      22        0       5     2.25
```

Notice that there is now a single variable, *resp*, that gives the values of the item responses. The variable *item* indicates which item each value of *resp* refers to.

Fitting Moderated Nonlinear Factor Analysis Models in the NLMIXED Procedure

In this section, we provide example code for fitting the models described in Bauer & Hussong (2009).

Model 1. Basic Nonlinear Factor Analysis

With reference to Bauer & Hussong (2009), this model is shown in Figure 2, and details regarding item distributions and link functions given in Table 2. The SAS code for this model is shown below, with explanation following.

```
proc nlmixed data=nlfa gconv=.0000000000001 qpoints=15;
parms /*intercepts*/  u01=-2.15 u02=-1.29 u03=1.01 u04=2.64 u05=1.19
      /*loadings  */  l01=2.46  l02=1.33  l03=8.23 l04=4.00 l05=.45
      /*thresholds*/  t023=8.1  t024=2.4
      /*resid var */  s2_05=.93
;

if (item=1) then do; *Disorder: Bernoulli w/ logit link;
   gmu = u01 + l01*eta;
   if (resp=1) then mu = 1/(1+exp(-gmu));
   else mu = 1 - ( 1/(1+exp(-gmu)) );
   if (mu > 1e-8) then ll = log(mu);
   else ll = -1e100;
end;
```

```
if (item=2) then do; *Conseq: Poisson w / log link;
   gmu = u02 + l02*eta;
   mu = exp(gmu);
   ll = -mu - lgamma(1 + resp) + resp*log(mu);
end;
if (item=3) then do; *HvyUse: ordinal w/ cumulative logit;
  gmu = u03 + l03*eta;
  *inverse link to obtain cumulative probabilities;
  mu0 = 1/(1+exp(-(0-gmu))); *first threshold set to 0;
  mu1 = 1/(1+exp(-(t023-gmu)));
  *probabilities for categories;
  if (resp=0) then p = mu0;
  else if (resp=1) then p = mu1 - mu0;
  else if (resp=2) then p = 1 - mu1;
  *writing log-likelihood;
  if (p > 1e-8) then ll = log(p);
  else ll = -1e100;
end;
if (item=4) then do; *UseFreq: ordinal w/ cumulative logit;
  gmu = u04 + l04*eta;
  *inverse link to obtain cumulative probabilities;
  mu0 = 1/(1+exp(-(0-gmu))); *first threshold set to 0;
  mu1 = 1/(1+exp(-(t024-gmu)));
  *probabilities for categories;
  if (resp=0) then p = mu0;
  else if (resp=1) then p = mu1 - mu0;
  else if (resp=2) then p = 1 - mu1;
  *writing log-likelihood;
  if (p > 1e-8) then ll = log(p);
  else ll = -1e100;
end;
if (item=5) then do; *Expect: censored normal w/ identity link;
   mu = u05 + l05*eta; *mean in absence of censoring;
   if (resp eq 0) then ll = log(probnorm((0-mu)/sqrt(s2_05))); *censoring;
   else ll =  -.5*(2*(22/7))-.5*log(s2_05)-((resp-mu)**2)/(2*s2_05);
end;
model resp~general(ll);
random eta~normal(0,1) subject=id;
ods output ParameterEstimates=Estimates;
run;
```

The NLMIXED statement invokes the procedure, indicates the data file, and specifies options concerning estimation. The options used here, *gconv=.0000000000001 qpoints=15,* make the convergence criterion more stringent and tell NLMIXED to use 15 adaptive quadrature points in the estimation.

The PARMS statement gives the labels for the parameters of the model and provides starting values for the corresponding estimates. For instance, *u01=-2.15* indicates that there is a parameter labeled u01 (the intercept for *disorder*) and it is to be given an initial value of -2.15. Likewise, *l01=2.46* specifies that there is a parameter labeled l01 (the factor loading for *disorder*) and it is to be given an initial value of 2.46. In the example code, the labeling of the parameters and their numbers correspond to the notational system used in Bauer & Hussong (2009). Note that good starting values can greatly speed model convergence. Fitting simpler models first is usually a good way to obtain starting values.

The sequence of IF statements specifies how the log-likelihood of the model is to be computed for different items. For instance, the section of code between *if (item=1) then do;* and the first *end;* indicates how the log-likelihood is to be computed for the first indicator, the binary *disorder* variable. The line *gmu = u01 + l01*eta;* corresponds to Equation 3 of Bauer & Hussong (2009) and defines the relationship between the item and the factor for item=1, *disorder*. The probability of scoring a 1 on this indicator is obtained from the inverse link function in the line *if (resp=1) then mu = 1/(1+exp(-gmu));* and the probablility of scoring a zero is the complement given in the line *else mu = 1 - (1/(1+exp(-gmu));*. The log-likelihood is then the log of the obtained probability (or *mu*), and is computed with the statements *if (mu > 1e-8) then ll = log(mu); else ll = -1e100;*. Note that these statements include an error trap: to avoid convergence problems, the log-likelihood is computed from the usual formula only if *mu* is not too close to zero, otherwise it is set to an arbitrarily small value (since log(0) is undefined). Likewise, the remaining *if(item=)* blocks of code specify the log-likelihoods for the remaining indicators, *conseq* (item=2, Poisson likelihood)*, hvyuse* (item=3, cumulative logit multinomial)*, usefreq* (item=4, cumulative logit multinomial)*,* and *expect* (item=5, censored normal likelihood).

The MODEL statement, *model resp ~general(ll);* tells the NLMIXED procedure that the outcome variable is *resp* and that we are providing a user-specified log-likelihood *ll* for *resp*.

The RANDOM statement is where we indicate that *eta* is the latent factor in the model. The code *random eta~normal(0,1) subject=id;* scales eta to a standard normal distribution (mean=0,variance=1) and indicates that the item responses are grouped by the subject ID variable *id*.

The ODS OUTPUT statement, *ods output ParameterEstimates=Estimates;*, is optional but tells SAS to take the table of model estimates, internally referenced by SAS as *ParameterEstimates* and place them into a SAS data file, *Estimates*, for later use. Having these estimates in a data file is helpful for producing graphs of item characteristic curves, such as Figure 3 of Bauer & Hussong (2009), as demonstrated later.

Model 2. Including Age Trends and Study Differences in the Factor Mean and Variance

To allow age trends and study differences in the factor mean and variance (see Figure 5, Bauer & Hussong, 2009), we modify the preceding code to be as follows:

```
proc nlmixed data=nlfa fconv=.0000000000001 gconv=.0000000000001
qpoints=15;
parms /*intercepts*/  u01=-1.93 u02=-1.12 u03=1.39 u04=4.02 u05=1.27
      /*loadings  */  l01=2.00  l02=1.04  l03=3.78 l04=3.50 l05=.31
      /*thresholds*/  t023=5.1  t024=4.37
      /*resid var */  s2_05=.94
      /*Eta       */  a1=-1.12 a2=1.27 omega1=.06 omega2=-.05
;
if (item=1) then do; *Disorder: Bernoulli w/ logit link;
   gmu = u01 + l01*eta;
   if (resp=1) then mu = 1/(1+exp(-gmu));
   else mu = 1 - ( 1/(1+exp(-gmu)) );
   if (mu > 1e-8) then ll = log(mu);
   else ll = -1e100;
end;
```

```
<snip>
if (item=5) then do; *Expect: censored normal w/ identity link;
   mu = u05 + l05*eta;
   if (resp eq 0) then ll = log(probnorm((0-mu)/sqrt(s2_05))); *censoring;
   else ll =  -.5*(2*(22/7))-.5*log(s2_05)-((resp-mu)**2)/(2*s2_05);
end;
alpha = 0 + a1*age17 + a2*study;
psi = 1*exp(omega1*age17 + omega2*study);
model resp~general(ll);
random eta~normal(alpha,psi) subject=id;
ods output ParameterEstimates=Estimates;
run;
```

The only changes to the prior model code are…

-- The new parameters are added to the PARMS statement (i.e., *a1, a2, omega1, omega2*) and start values are updated.

-- The lines *alpha = 0 + a1*age17 + a2*study;* and *psi = 1*exp(omega1*age17 + omega2*study);* specify that the factor mean and variance are linear and loglinear functions, respectively, of the *age17* and *study* variables.

-- The line *random eta~normal(alpha,psi) subject=id;* has been changed to reflect that the factor mean is now given by alpha and the factor variance is psi (both of which are conditional on age and study).

Model 3. Incorporating Differential Item Functioning (DIF)

The final model presented in Bauer & Hussong (2009), represented in Figure 7, includes DIF for several items. The code is

```
proc nlmixed data=nlfa fconv=.0000000000001 gconv=.0000000000001
qpoints=15;
parms /*intercepts*/  u01=-1.93 u02=-1.12 u03=1.39 u04=4.02 u05=1.27
      /*loadings  */  l01=2.00  l02=1.04  l03=3.78 l04=3.50 l05=.31
      /*thresholds*/  t023=5.1  t024=4.37
      /*resid var */  s2_05=.94
      /*Eta       */  a1=-1.12 a2=1.27 omega1=.06 omega2=-.05
      /*DIF       */  u11=1.15 u14=.38 u24=3.32 u15=-.62
                      l14=.33 l24=4.52
                      s2_15=-.20
;
if (item=1) then do; *Disorder: Bernoulli w/ logit link, intercept DIF;
   gmu = (u01 + u11*age17) + l01*eta;
   if (resp=1) then mu = 1/(1+exp(-gmu));
   else mu = 1 - ( 1/(1+exp(-gmu)) );
   if (mu > 1e-8) then ll = log(mu);
   else ll = -1e100;
end;
if (item=2) then do; *Conseq: Poisson w / log link, no DIF;
   gmu = u02 + l02*eta;
   mu = exp(gmu);
   ll = -mu - lgamma(1 + resp) + resp*log(mu);
end;
```

```
if (item=3) then do; *HvyUse: ordinal w/ cumulative logit, no DIF;
  gmu = u03 + l03*eta;
  *inverse link to obtain cumulative probabilities;
  mu0 = 1/(1+exp(-(0-gmu))); *first threshold set to 0;
  mu1 = 1/(1+exp(-(t023-gmu)));
  *probabilities for categories;
  if (resp=0) then p = mu0;
  else if (resp=1) then p = mu1 - mu0;
  else if (resp=2) then p = 1 - mu1;
  *writing log-likelihood;
  if (p > 1e-8) then ll = log(p);
  else ll = -1e100;
end;
if (item=4) then do; *UseFreq: ordinal w/ cumulative logit, int & load DIF;
  gmu = (u04 + u14*age17 + u24*study) + (l04 + l14*age17 + l24*study)*eta;
  *inverse link to obtain cumulative probabilities;
  mu0 = 1/(1+exp(-(0-gmu))); *first threshold set to 0;
  mu1 = 1/(1+exp(-(t024-gmu)));
  *probabilities for categories;
  if (resp=0) then p = mu0;
  else if (resp=1) then p = mu1 - mu0;
  else if (resp=2) then p = 1 - mu1;
  *writing log-likelihood;
  if (p > 1e-8) then ll = log(p);
  else ll = -1e100;
end;
if (item=5) then do; *Expect: censored normal w/ ident link, int & var DIF;
   mu = (u05 + u15*study) + l05*eta;
   s2 = s2_05*exp(s2_15*age17);
   if (resp eq 0) then ll = log(probnorm((0-mu)/sqrt(s2))); *censoring;
   else ll =  -.5*(2*(22/7))-.5*log(s2)-((resp-mu)**2)/(2*s2);
end;
alpha = 0 + a1*age17 + a2*study;
psi = 1*exp(omega1*age17 + omega2*study);
model resp~general(ll);
random eta~normal(alpha,psi) subject=id;
ods output ParameterEstimates=Estimates;
run;
```

Note the addition of new parameters to the PARMS statement. These parameters appear in the *if(item=)* blocks to allow intercepts, loadings, or residual variances to vary as a function of the *age17* and/or *study* variables. For instance, under *if(item=4)*, the line

*gmu = (u04 + u14*age17 + u24*study) + (l04 + l14*age17 + l24*study)*eta;*

allows for DIF in both the intercept and factor loading of the *usefreq* item, where *u14* and *u24* capture age and study DIF in the intercept, and *l14* and *l24* capture age and study DIF in the factor loading, respectively. The parameters *u14*, *u24*, *l14* and *l24* must therefore appear in the PARMS statement.

Plotting Item Characteristic Curves (ICCs)

Here we show how to generate plots of ICCs. In general, these plots are produced in three steps.

(1) When fitting the model in the NLMIXED procedure, include the statement *ods output ParameterEstimates=estimates;*, as shown in the example code above. This will produce an output dataset called *estimates* that contains the item parameters needed to generate the plots.

(2) Transpose the data file *estimates.* The original *estimates* file contains a row for each parameter estimate, with 1 column for the parameter labels and 1 column for the estimates (among other columns). The transposed data file will contain only <u>one</u> row, but with a column for <u>each</u> estimate, and these columns will be designated with the parameter labels. Example code for accomplishing the transposition is shown here:

```
proc transpose data=estimates out=estT;
 var Estimate;
 id Parameter;
run;
```

(3) Generate points to plot the ICCs by computing the expected values of the items across the range of the latent variable *eta*. For instance, to generate ICCs for Model 1, as shown in Figure 3 of Bauer & Hussong (2009), the following code increments from eta = -3 to eta = +3, by units of .1 (e.g., -3.0, -2.9, …, 2.9, 3.0). Each value of eta is used, in combination with the item parameters in the data set *estT*, to generate an expected value for each item. These expected values will subsequently be plotted.

```
data graph; set estT;
 do eta = -3 to 3 by .1;
   *probability of disorder, given eta;
   ICC1 = 1/(1+exp(-(u01 + l01*eta)));
   *expected conseq, given eta;
   ICC2 = exp(u02 + l02*eta);
   *probability of hvyuse category endorsements, given eta;
   gmu = u03 + l03*eta;
   mu0 = 1/(1+exp(-(0-gmu))); *first threshold set to 0;
   mu1 = 1/(1+exp(-(t023-gmu)));
   ICC30 = mu0;
   ICC31 = mu1 - mu0;
   ICC32 = 1 - mu1;
   *probability of usefreq category endorsements, given eta;
   gmu = u04 + l04*eta;
   mu0 = 1/(1+exp(-(0-gmu))); *first threshold set to 0;
   mu1 = 1/(1+exp(-(t024-gmu)));
   ICC40 = mu0;
   ICC41 = mu1 - mu0;
   ICC42 = 1 - mu1;
   *mean expect, given eta;
   Eystar = u05 + l05*eta; *mean of underlying normal variate;
   s = sqrt(s2_05); *standard deviation of underlying normal variate;
   ICC5 = CDF('NORMAL',Eystar/s)*Eystar + s*PDF('NORMAL',Eystar/s);
   output;
 end;
run;
```

Note that items with different conditional distributions require the use of different functions to generate the expected value of the item response. Note also that for the ordinal items, *hvyuse* and *usefreq*, there are three values generated that correspond to the probabilities of the three categories.

(4) Generate the ICCs by plotting the expected values against *eta*. For example, this is the code used to generate the panels in Figure 3 of Bauer & Hussong (2009):

```
goptions reset=all hsize=5 vsize=4 ftext=simplex htext=1;
symbol1 value=none interpol=spline w=2 color=black;
symbol2 value=none interpol=spline w=2 l=2 color=black;
symbol3 value=none interpol=spline w=2 l=5 color=black;
axis1 order=0 to 1 by .1 minor=none label=("P(y)");
axis2 minor=none order=(-2.5 to 2.5 by .5) label=(font=greek "h");
axis3 minor=none label=("E(y)") order=(0 to 3 by .5);
axis4 minor=none order=(0 to 9 by 1) label=("E(y)");
legend1 value=("P(Y=0)" "P(Y=1)" "P(Y=2)")
        position=(inside right middle) down=3 label=none;
legend2 value=("P(Y=0)" "P(Y=1)" "P(Y=2)")
        position=(inside left middle) down=3 label=none;
proc gplot data=graph;
 title 'Probability of Disorder';
 plot ICC1*eta / vaxis=axis1 haxis=axis2;
run;
proc gplot data=graph;
 title 'Expectation for # Consequences';
 plot ICC2*eta / vaxis=axis4 haxis=axis2;
run;
proc gplot data=graph;
 title 'Probabilities for Use Freq';
 plot (ICC30 ICC31 ICC32)*eta / overlay vaxis=axis1 haxis=axis2
                                legend=legend1;
run;
proc gplot data=graph;
 title 'Probabilities for Heavy Use';
 plot (ICC40 ICC41 ICC42)*eta / overlay vaxis=axis1 haxis=axis2
                                legend=legend2;
run;
proc gplot data=graph;
 symbol value=none interpol=join w=2;
 title 'Expectation for Pos Alc Exp';
 plot ICC5*eta / vaxis=axis3 haxis=axis2;
run;
quit;
```

Scoring

It is quite easy to generate scores for the sample to which the nonlinear factor analysis is fit. One merely adds the OUT option to the RANDOM statement of the NLMIXED procedure. For instance, if we wished to generate Modal a Posteriori (MAP) estimates from Model 1, we would simply modify the RANDOM statement shown previously to be

random eta~normal(0,1) subject=id out=MAPs;

Including the OUT option tells NLMIXED to generate a data set called *MAPs* that includes a factor score estimate for each subject (i.e., unique value of *id*). The MAPs can be quite helpful for aiding in model specification. For instance, plotting the MAPs obtained from Model 1 as a function of age and study can help to show the importance of adding age and study effects in Model 2.

For our case, however, scoring in this manner would be of limited use for subsequent analyses because we would only obtain scores for assessment points in the calibration sample. Our goal was to use the parameter estimates obtained from the calibration sample to generate scores for the full sample, including all repeated assessments.

Scoring for “new” observations not in the calibration sample is a bit tricky. The catch is that NLMIXED only generates MAPs as a by-product of model estimation. However, we don’t want to estimate a new model for the full sample. We wish to use the estimates already obtained from the calibration sample to generate the MAPs. So we must get NLMIXED to “estimate” a model so that it will produce MAPs, while not actually updating any of the item parameter estimates obtained from the calibration sample. The steps for doing this are as follows:

(1) The data for the full sample (all studies, all assessment points) must be arranged similarly to the calibration sample data. That is, there should be one variable indicating the item response, another variable indicating the item, and multiple lines of data for the multiple items for each person at each year. Relative to how the calibration data was prepared, however, there are two key differences (shown in example code below):

-- For the full sample, a unique ID is required for each person-year. That is, person 001 at age 18 must have a different ID label than person 001 at age 20. We accomplished this by concatenating the original *id* variable with *ageyr* to produce a new variable *idyr.* For instance, for person 001 at ages 18 and 20, *idyr* would equal 00118 and 00120, respectively. (In the calibration sample this step was unnecessary given the presence of only one year of data per person).

-- For the full sample, a “dummy item” must be created. In our case, we had 5 real items, so we created a 6th dummy item. The “responses” for this item were generated as random values from a standard normal distribution. The purpose of the dummy item will be explained in Step 3, below.

```
data score; set combine;
 array dv [5] disorder conseq hvyuse usefreq expect;
 idyr = COMPRESS(id||ageyr);
 age17 = ageyr-17;
 if ageyr > 17 then age17 = 0;
 do i = 1 to 5;
    item = i;
    resp = dv[i];
    output;
 end;
 item = 6; resp = rannor(-1); output; *dummy item;
 keep id idyr study ageyr age17 item resp;
run;
```

(2) Use the estimates obtained from the calibration sample as the parameter values for the same model for the full sample. This is best accomplished by first fitting the model to the calibration sample, then saving the estimates from the calibration sample to a data set using ODS (e.g., *ods output ParameterEstimates=Estimates;*), then converting the values in the output data file into macro variables using the CALL SYMPUT statement. The macro variables can subsequently be referenced in the NLMIXED code for the full sample. Here is example code using CALL SYMPUT :

```
data _null_; set estimates;
 call symput(parameter,estimate);
run;
```

CALL SYMPUT must be used within a DATA step, but no data file is being created here, hence we reference an empty (_NULL_) data file. Within the *estimates* data file are two columns (variables) labeled *parameter* and *estimate*. *Parameter* is a column of text entries that correspond to the parameter labels declared in the PARMS statement of the NLMIXED program run with the calibration sample (e.g., *u01*, *u02*, etc.). *Estimate* is a column of numerical estimates for those parameters (e.g., -2.15, -1.29, etc.). The statement *call symput(parameter, estimate);* creates a sequence of macro variables named *u01*, *u02*, etc. with values -2.15, -1.29, etc.

(3) “Fit” the model to the full sample within NLMIXED, but using the macro variables from Step 2 to set each parameter to the exact value obtained from fitting the model to the calibration sample. For instance, to score based on Model 3, we used the code

```
proc nlmixed data=score qpoints=15;
/*intercepts*/  u01=&u01.; u02=&u02.; u03=&u03.; u04=&u04.; u05=&u05.;
/*loadings  */  l01=&l01.; l02=&l02.; l03=&l03.; l04=&u04.; l05=&l05.;
/*thresholds*/  t023=&t023.; t024=&t024.;
/*resid var */  s2_05=&s2_05.;
/*Eta       */  a1==&a1.; a2=&a2.; omega1=&omega1.; omega2=&omega2.;
/*DIF       */  u11=&u11.; u14=&u14.; u24=&u24.; u15=&u15.;
                l14=&l14.; l24=&l24.;
                s2_15=&s2_15.;
parms a=0 b=0;
if (item=1) then do; *Disorder: Bernoulli w/ logit link, intercept DIF;
   gmu = (u01 + u11*age17) + l01*eta;
   if (resp=1) then mu = 1/(1+exp(-gmu));
   else mu = 1 - ( 1/(1+exp(-gmu)) );
   if (mu > 1e-8) then ll = log(mu);
   else ll = -1e100;
end;

<snip>

if (item=5) then do; *Expect: censored normal w/ identity link
   mu = (u05 + u15*study) + l05*eta;
   s2 = s2_05*exp(s2_15*age17);
   if (resp eq 0) then ll = log(probnorm((0-mu)/sqrt(s2))); *censoring;
   else ll =  -.5*(2*(22/7))-.5*log(s2)-((resp-mu)**2)/(2*s2);
end;
```

```
if (item=6) then do;
   mu = a + b*eta;
   ll = .001*(-.5*log(2*(22/7)) -.5*log(1)-((resp-mu)**2)/(2));
end;
alpha = 0 + a1*age17 + a2*study;
psi = 1*exp(omega1*age17 + omega2*study);
model resp~general(ll);
random eta~normal(alpha,psi) subject=idyr out=MAPs;
ods output ParameterEstimates=Estimates;
run;
```

Notice that, unlike previous examples, the statements at the start of the code, beginning with */*intercepts*/* are actually programming statements, not part of the PARMS statement. These programming statements assign each parameter the value of the macro variable that holds the corresponding estimate from the calibration sample (e.g., the statement *u01=&u01.;* sets the parameter *u01* equal to the value held in the macro variable *u01* referenced by *&u01.*, which was created in the previous CALL SYMPUT statement). Thus all of the parameters of the model are held fixed at the values obtained by fitting the same model to the calibration sample. These values will not be updates when NLMIXED is run.

The problem is that NLMIXED will only generate MAPs in the course of model estimation, and to this point we've discussed no parameters (all are being held fixed). Here enters the dummy item. In the PARMS statement, we indicate two "dummy parameters" for this item that NLMIXED will try to estimate, arbitrarily labeled *a* and *b*, but which will have no real effect on scoring. Likewise, we added

if (item=6) then do;
  *mu = a + b*eta;*
  ll = .001(-.5*log(2*(22/7)) -.5*log(1)-((resp-mu)**2)/(2));*
end;

where *a* and *b* are "intercept" and "loading" parameters (started at zero in the PARMS statement since item 6 was generated randomly from a standard normal distribution and therefore has no relationship to the latent factor). The log-likelihood is computed from the usual log-likelihood of a normal distribution with variance of one, but with the caveat that the log-likelihood value is made arbitrarily small by multiplying by .001 so that the contribution of the dummy item to the overall model log-likelihood is insignificant. The model will therefore "converge" immediately. Further, since the dummy item has no relationship to the latent factor, it also has no meaningful impact on the MAPs obtained for the observations. To verify that the dummy item has no impact on scoring, one can compare the MAPs obtained from the model fit to the calibration sample to the MAPs for the same observations within the full sample to confirm that they are identical (at least to the desired level of precision). If the scores differ, one can simply make the log-likelihood multiplier for the dummy item smaller (e.g., .0001).

The MAPs are generated via the OUTPUT option of the RANDOM STATEMENT with the code, *random eta~normal(alpha,psi) subject=idyr out=MAPs;* Note that *idyr* is indicated in the SUBJECT option so that a distinct MAP will be generated for each person at each year within the *MAPs* data set.