1 Overview

This document describes how to use SAS and Mplus to fit the models discussed in Anderson, Kim & Keller (2013). In sum, we show here how estimation of discrete multilevel models can be achieved through the use of SAS and Mplus. This information provided here is designed to provide an explanation of some of the code used to fit model reported in the chapter and
does not replace documentation for either of these programs. This document is organized according to programs and subsections to models.

2 SAS

2.1 Data

2.2 Data in SAS

The data set required for this is names “mi_wscaled”.

Get output from PROC CONTENTS and put here

2.3 Binary Logistic Regression

When weights are not used, models can be fix using either PROC GLIMMIX or NLMIXED; however, when weights are used, then you must use NLMIXED. In general, SAS takes a generalized linear mixed model approach to model specification; that is, a fixed/structural/systematic part of the model is specified in one command and the random part is specified in another command.

2.3.1 SAS PROC GLIMMIX

First we present the code for fitting the binary logistic regression model on the left side of Table 5 using PROC GLIMMIX and then present the code for combining the results over imputations. The code for fitting the model to each imputed data set:

```
proc GLIMMIX data=mi_wscaled method=quad noclprint empirical;
    by _imputation_;
    model bschnet = girl screent ctimeRdg mtimeRdg short Alllun
            /link=logit dist=binomial solution;
    random intercept / subject=idschool;
    ods output ParameterEstimates=gparms FitStatistics=gfitstats;
run;
```

Below is an explanation of the commands in the above coded

* proc GLIMMIX data=mi_wscaled method=quad noclprint empirical;
  This command line indicates
    - data=mi_wscaled Indicates which data set to be used.
- **method=quad** Specifies adaptive Gaussian quadrature as that estimation algorithm. If you have trouble with convergence, the algorithm can start using RSPL for some number of iterations and then switch over to QUAD. The code for this is `method=quad(initpl=4)` or some number smaller or larger than 4.

- **noclpirnt** Requests that the class level ids not be printed (i.e., the school id numbers).

- **empirical** Requests sandwich standard errors be computed and used.

- **by _imputation_;** Indicates that the model should be fit to each imputed data set (these are indexed by the variable named _imputation_. When first writing and running the code for all imputed data sets, I used the command `where _imputation_ = 1`; that is, I ran it for only the first data set. This was much faster to debug code and check to make sure that it was running correctly.

- **model bschnet = girl screent ctimeRdg mtimeRdg short Alllun /link=logit dist=binomial solution;** This command gives the structural part of the model including both within and between level variables. The variable to the left of = is the response variable and those listed to the right are predictors. Options are given to the right of `/`. The link is a logit and the distribution for the response is assumed to be Binomial (i.e., Bernoulli for binary coding). Solution is needed to have SAS print out fixed effects.

- **random intercept / subject=idschool;** This command requests that a random intercept be fit. If other (Level 1) variables have random coefficients, they would be listed here. The option `subject=idschool` tells SAS what the id variable is. If there are multiple random coefficients, you may want to add in the “type=” to specify the structure of the covariance matrix of the random effects.

- **ods output ParameterEstimates=gparms FitStatistics=gfitstats;** “ods” is short for “Output Delivery System” and this line requests that parameter estimates (the fixed effects) be output to a SAS file that I names “gparms” and the fit statistics be sent to the file named “gfitstats”. Any table in the output can be sent to a SAS file. The names of these tables (in this case “ParameterEstimates” and “FitStatistics”) can be found in the GLIMMIX documents under details > ODS Table Names.

The next step is to combine the parameter estimates over imputations. The file “gparms” contains the parameter estimates for each imputed data set and the file “fitstatistics” contains the fit statistics for each data set.

```sas
/* using PROC MIANALYZE for fitstatistics */
proc sort data=gcovparms;
    by covparm _imputation_;
run;
```
title 'Fit statistic means';
proc mianalyze data=gcovparms;
   by covparm;
   modeleffects estimate;
   stderr stderr;
run;

/*Now fixed effects*/
title 'Parameter Estimates';
proc mianalyze parms=gparms;
   modeleffects intercept girl screent ctimeRdg mtimeRdg short Alllun ;
run;

2.3.2 SAS PROC NLMIXED

If you can write down the model, you can use NLMIXED. The code that fits the model to each imputated data set is

proc nlmixed data= mi_wscaled method=gauss qpoints=15 gconv=0 cov EMPIRICAL ;
   /* Fit the model for each imputated data set */
   by _imputation_;

   /* indicate what are model parameters 
   (I put in some good ones to speed things up) */
   parms g00= -3.5426701 g10= 0.1350379 g20= 0.1067511
      g30= 0.2874239 g40= 1.2065955 g50= -0.1643729
      g60= .40
      var = 0.2774892;

   /* Linear predictor */
   etal = g00 + g10*girl + g20*screent + g30*ctimeRdg +g40*mtimeRdg + g50*short
         +g60*AllLun +u0;

   /* Define likelihood */
   if bschnet=1 then prob = exp(eta1)/(1+exp(eta1));
      else prob = 1/(1+exp(eta1));

   /* To make sure that probabilities are valid ones */
   p = (prob>0 and prob<=1)*prob + (prob<=0)*1e-8 + (prob>1);
   pprob = log(p);
/* Specify distribution for response variable and random effect */
model bschnet ~ general(pprob);
random u0 ~ normal(0,var) subject=idschool out=wRand_Effects;

/* Output results that will be combined over imputations */
ods output ParameterEstimates=parms FitStatistics=fitstats CovMatParmEst=cov;
run;

Explanations of each line are given below.

- **proc nlmixed data= mi wscaled method=gauss qpoints=15 gconv=0 cov EMPIRICAL ;** The estimation algorithm is Gaussian quadrature. This line requests that 15 quadrature points be used. This was chosen because it was the same number as the default in Mplus and we desired to verify that both programs gave the same results. The other important options is the “EMPIRICAL” options (doesn’t have to be capitalized) that indicates that standard errors should be sandwich or robust ones.

- **by .imputation ;** Same as in GLIMMIX: fit the model to each imputed data set.

- **parms g00= -3.5426701 g10= 0.1350379 g20= 0.1067511 g30= 0.2874239 g40= 1.2065955 g50= -0.1643729 g60= .40 var = 0.2774892 ;** This command indicates what the parameters of the model are. You do not need to give starting values for them; however, I put in reasonable ones that I got from fitting the model previously. Using reasonable starting values can speed up processing time (i.e., for one binary model in our example took about 12-15 seconds).

- **eta1 = g00 + g10*girl + g20*screent + g30*ctimeRdg +g40*mtimeRdg + g50*short +g60*AllLun +u0 ;** This is where I write the structural part of the model; it is the cluster-specific part in that is include the random intercept, u0. This is analogous to the “model” statement in GLIMMIX.

- **if bschnet=1 then prob = exp(eta1)/(1+exp(eta1));
else prob = 1/(1+exp(eta1));** These lines apply the link to the structural part of the model; namely, they define how to compute the probabilities.

- **p = (prob>0 and prob<=1)*prob + (prob<=0)*1e-8 + (prob>1);** If the equation yields a probability < 0, it is set equal to 0 and it yield a probability> 1, it is set equal to 1. With our data, we never ran into this problem and the code runs fine without it. We give it here because this may not always be the case.

- **pprob = log(p)** This completes the definition of the (log) of the likelihood.

- **model bschnet ~ general(pprob) ;** This command specifies the distribution of the random binary variable “bschnet”. The term general indicates that we have defined the distribution above and the pprob gives the equation of it.
random u0 ~ normal(0, var) subject=idschool out=wRandEffects; This command tells SAS that the variable u0 is random and gives the distribution of it as normal with mean 0 and parameter var. The variable that indicates the clusters is specified by subject=idschool. In this version, I requested estimates of the u0s and this is what the out option provides (I looked at their distribution). These are BLUPs.

ods output ParameterEstimates=parms FitStatistics=fitstats CovMatParmEst=cov; This is similar to what was done in GLIMMIX; namely I wanted variable tables that were in the output sent to SAS files so that I can combined estimates over the imputations. In addition to the parameters and fit statistics, I also requested the covariance matrix of the parameter estimates (these are empirical or sandwich).

2.4 Baseline Multinomial Logistic Regression

Both procedures NLMIXED and GLIMMIX can be used to fit these models; however, we have had better luck using NLMIXED. Only the code for NLMIXED is given here. It is

title 'Model: Multinomial model via nlmixed withn no weights';
proc nlmixed data= mi_wscaled method=gauss qpoints=15 gconv=0 cov EMPIRICAL ;
/* Fit the model for each imputated data set */
by _imputation_;
/* indicate what are model parameters (will use lowest level as reference) */
parms g01= -4.0 g02= -3.0 g03= -2.0
g11= .38 g12= .31 g13= .42
g21= .16 g22= .03 g23= .00
g31= .44 g32= .27 g33= .13
g41= 1.72 g42= 1.50 g43= 0.94
g51= -.18 g52= -.38 g53= -.28
var = 0.2;
/* Linear predictors */
eta1 = g01 + g11*girl + g21*screent + g31*ctimeRdg +g41*mtimeRdg + g51*short + u0;
eta2 = g02 + g12*girl + g22*screent + g32*ctimeRdg +g42*mtimeRdg + g52*short + u0;
eta3 = g03 + g13*girl + g23*screent + g33*ctimeRdg +g43*mtimeRdg + g53*short + u0;
/* Define likelihood */
if schnet=1 then prob = exp(eta1)/(1 + exp(eta1) + exp(eta2) + exp(eta3));
if schnet=2 then prob = exp(eta2)/(1 + exp(eta1) + exp(eta2) + exp(eta3));
if schnet=3 then prob = exp(eta3)/(1 + exp(eta1) + exp(eta2) + exp(eta3));
if schnet=4 then prob = 1/(1 + exp(eta1) + exp(eta2) + exp(eta3));
/* To make sure that probabilities are valid ones */
p = (prob>0 and prob<=1)*prob + (prob<=0)*1e-8 + (prob>1);
llike = log(p);

/* Specify distribution for response variable and random effect */
model schnet ~ general(llike);
random u0 ~ normal(0, var) subject=idschool out=RandEffects1;

/* Output results that will be combined over imputations */
ods output ParameterEstimates=pcomplex FitStatistics=fcomplex CovMatParmEst=parmcov
    AdditionalEstimates=extra_est CovMatAddEst=extra_cov;
run;

/* Computing average fit statistics */
title2 'Fit Statistics';
proc sort data=fcomplex;
   by Descr;
run;
proc means data=fcomplex;
   class Descr;
   var value;
run;

/* Using mianalyze */
data fixup;
set pcomplex;
StdErr = StandardError;
run;
proc sort data=fixup;
   by _imputation_
run;
proc mianalyze parms=fixup;
   modeleffects g01 g02 g03 g11 g12 g13 g21 g22 g23 g31 g32 g33
       g41 g42 g43 g51 g52 g53 var;
run;

The code is very similar to that for binary logistic regression. The differences are

More parameters  We now have g (for gamma) for each variable and 3 (= K – 1) response
options.

**More linear predictors** We now have a different linear predictor, $\eta_1$, $\eta_2$ and $\eta_3$ for each of the 3 ($= K-1 = 4-1$) response options. In this case, the fourth response option is the baseline.

**prob** When defining the likelihood, the probability of response depends on the response option. This is done by the if/then statements.

If some parameters are set equal to each other or parameters are set to 0, changes are made to the listed parameters (i.e., `parms` and the linear predictor). For example, if the response due to girl have the same gamma, the parameters $g_{12}$ and $g_{13}$ would be omitted from the `parms` statements. In the linear predictor, `girl` would have the same parameter for each response:

$$
\eta_1 = g_{01} + g_{11}\text{girl} + g_{21}\text{screent} + g_{31}\text{ctimeRdg} + g_{41}\text{mtimeRdg} + g_{51}\text{short} + u_0; \\
\eta_2 = g_{02} + g_{11}\text{girl} + g_{22}\text{screent} + g_{32}\text{ctimeRdg} + g_{42}\text{mtimeRdg} + g_{52}\text{short} + u_0; \\
\eta_3 = g_{03} + g_{11}\text{girl} + g_{23}\text{screent} + g_{33}\text{ctimeRdg} + g_{43}\text{mtimeRdg} + g_{53}\text{short} + u_0;
$$

To set a parameter equal to 0, the gamma is dropped from the `parm` statement and the predictor dropped from the linear predictor. For example, to set the parameters of `screent` for responses 2 and 3 equal to 0, the linear predictor is

$$
\eta_1 = g_{01} + g_{11}\text{girl} + g_{21}\text{screent} + g_{31}\text{ctimeRdg} + g_{41}\text{mtimeRdg} + g_{51}\text{short} + u_0; \\
\eta_2 = g_{02} + g_{11}\text{girl} + g_{32}\text{ctimeRdg} + g_{41}\text{mtimeRdg} + g_{51}\text{short} + u_0; \\
\eta_3 = g_{03} + g_{11}\text{girl} + g_{33}\text{ctimeRdg} + g_{41}\text{mtimeRdg} + g_{51}\text{short} + u_0;
$$

### 2.5 Proportional Odds Model

The code for the proportional odds models is similar to that for the binary and nominal responses, except that we replace the model with the proportional odds mode when defining what the probability for each response options equals. It is

```plaintext
/**********************************************************************
 Ordinal Model 1
*********************************************************************/
title1 'Proportional Odds Models, Ordinal Model 1';
proc nlmixed data=mi_wscaled method=gauss qpoints=15 gconv=0 EMPIRICAL cov;
*where _imputation_ = 1;
by _imputation_;

parms g01=-4.0   g02=-3.0   g03=-2.0
    g11=.38  g21=.16  g31=.44
    g41=1.72  g51=-.38
```
```
var11 = 0.5682966;

/* Linear predictor */
eta = g11*girl + g21*screent + g31*ctimeRdg + g41*mtimeRdg + g51*short + u1;

/* Cumulative probabilities */
cp1 = exp(g01 + eta)/(1 + exp(g01 + eta));
cp2 = exp(g02 + eta)/(1 + exp(g02 + eta));
cp3 = exp(g03 + eta)/(1 + exp(g03 + eta));

/* Define probabilities */
if schnet=1 then prob = cp1;
if schnet=2 then prob = cp2-cp1;
if schnet=3 then prob = cp3-cp2;
if schnet=4 then prob = 1-cp3;

/* To make sure that probabilities are valid ones */
p = (prob>0 and prob<=1)*prob + (prob<=0)*1e-8 + (prob>1);

/* Define likelihood*/
like = log(p);

/* Specify distribution for response variable and random effect */
model schnet ~ general(like);
random u1 ~ normal(0, var11) subject=idschool;

/* Output results that will be combined over imputations */
ods output ParameterEstimates=oparm1 FitStatistics=ofit1 CovMatParmEst=oparmcov1;
run;

/* Computing average fit statistics */
title2 'Fit Statistics';
proc sort data=ofit1;
   by Descr;
run;
proc means data=ofit1;
   class Descr;
   var value;
run;

/* Using mianalyze */
```
data fixup;
set oparm1;
StdErr = StandardError;
run;
proc sort data=fixup;
  by _imputation_; 
run;
proc mianalyze parms=fixup;
  modeleffects g01 g02 g03 g11 g21 g31 g41 g51 var11;
run;

Notes on this...

parms included fewer parameters.

**Linear predictor** Only a single one if defined that does not include an intercept. This is named *eta*.

**Cumulative probabilities** are *cp1*, *cp2* and *cp3*. They defined as using the linear predictors *eta* but at this point I’ve added in the intercepts that differ over the response options. With these data we had no problem in terms of the ordering of the intercepts—they came out the order that we needed. If the ordering is not correct or conform to the ordering of the response options, they can be restricted to the desired order using a *lincon* statement (short for “linear constraints”). For example, if the ordering should be $g_1 \geq g_2 \geq g_3$, then the following line could be added:

```
lincon 0<= g01-g02, 0<= g02-g03;
```

**prob** The probabilities equal the difference between the cumulative probabilities.

### 2.6 Partial Proportional Odds Model

The SAS code for this model is kind of mix between that for the multinomial and proportional odds model.

```
title1 "Partial Proportional Odds Model, Model 4 (with weights)";
title2 "---dropping non-significant effects ";
proc nlmixed data=mi_wscaled method=gauss qpoints=15 gconv=0 EMPIRICAL cov;
  by _imputation_; 
  parms g01= -4.0   g02= -3.0   g03= -2.0
  g11= .38
  g21= .16
```
g31 = .44
g41 = 1.72
g52 = .1
var11 = 0.5682966;

/* Linear predictor */
eta1 = g11*girl + g21*screent + g31*ctimeRdg + g41*mtimeRdg + u1;
eta2 = g11*girl + g21*screent + g31*ctimeRdg + g41*mtimeRdg + g52*short + u1;
eta3 = g11*girl + g31*ctimeRdg + g41*mtimeRdg + g52*short + u1;

/* Cumulative probabilities */
cp1 = exp(g01 + eta1)/(1 + exp(g01 + eta1));
cp2 = exp(g02 + eta2)/(1 + exp(g02 + eta2));
cp3 = exp(g03 + eta3)/(1 + exp(g03 + eta3));

/* Define probabilities */
if schnet=1 then prob = cp1;
if schnet=2 then prob = cp2-cp1;
if schnet=3 then prob = cp3-cp2;
if schnet=4 then prob = 1-cp3;

/* To make sure that probabilities are valid ones */
p = (prob>0 and prob<=1)*prob + (prob<=0)*1e-8 + (prob>1);

/* Define pseudo-likelihood*/
maxlike = log(p);

/* Specify distribution for response variable and random effect */
model schnet ~ general(maxlike);
random u1 ~ normal(0, var11) subject=idschool;

/* Output results that will be combined over imputations */
ods output ParameterEstimates=oparm4 FitStatistics=ofit4 CovMatParmEst=oparmcov4;
run;

Notes: The difference between this model and the proportional model is that the linear predictors are not all the same. In particular, there is not fixed effect for `screent` for response 3rd and no `short` for the 1st response option.
If may also be the case that some fixed effects are the same but others differ over response options. For example

/* Linear predictor */
eta1 = g11*girl + g21*screent + g31*ctimeRdg + g41*mtimeRdg + g51*short + u1;
eta2 = g11*girl + g22*screent + g32*ctimeRdg + g41*mtimeRdg + g52*short + u1;
eta3 = g11*girl + g23*screent + g33*ctimeRdg + g41*mtimeRdg + g53*short + u1;

Here, the predictor all have the same coefficients for girl, mtimeRdg and the random effect (u1) but different ones for screent, ctimeRdg and short. To test the proportional odds assumption for these latter, Wald statistics can be computed. We did this using SAS PROC IML and this is in the program code.

2.7 All Models in SAS with Design/Sampling Weights

These models can only be fit in NLMIXED. Suppose that the level 1 weights are \( w_1 \) and the level 2 ones are \( w_2 \). Further assume that \( w_1 \) and \( w_2 \) have the desired scaling (in the paper, they sum to equal the sample size). Only two modifications are needed.

1. **Level 1 weights**: The line `pprob = log(p);` should be replaced by `pprob= w1*log(p).`

2. **Level 2 weights**: To add these, an additional line of code is required. After the `RANDOM` statement the line `REPLICATE w2;` should be added.

The NLMIXED documentation at the time of this writing states that the variable in the `REPLICATE` statement should be a positive integer (i.e., frequency weighting); however, this is incorrect (Personal communication, June 18, 2012). The documentation will be corrected in the near future. Since the `REPLICATE` variable need not be an integer, SAS does exactly what we need it to do without any modifications such as that done by Grilli & Pratesi (2004). The conditional/cluster specific likelihoods are weighted (multiplied) by their \( w_2 \).

If only Level 1 weights are desired, then only change (1) is required and if only Level 2 weights are desired, then only do change (2).

As an example, below is the modification for the binary logistic regression:

```sas
proc nlmixed data= mi_wscaled method=gauss qpoints=15 gconv=0 cov EMPIRICAL ;
/* Fit the model for each imputed data set */
by _imputation_;
/* indicate what are model parameters
(I put in some good ones to speed things up) */
parms g00= -3.5426701  g10= 0.1350379  g20= 0.1067511
g30= 0.2874239  g40= 1.2065955  g50= -0.1643729
```

12
The code for binary logistic regression and proportional odds model is given in this Section. Mplus at this time cannot be used to fit multinomial or partial proportional odds models with random coefficients.

### 3.1 Data in Mplus

There is a separate text file for each imputed data set (i.e., cMIdata1.dat – cMIdata15.dat) and a file cMIdatalist.dat that simply lists the data files; that is, cMIdatalist.dat is

```
cMIdata1.dat
cMIdata2.dat
cMIdata3.dat
cMIdata4.dat
cMIdata5.dat
cMIdata6.dat
cMIdata7.dat
```
3.2 Binary Logistic Regression

The input code is that is in the file named “Binary_final_no_weights.inp” is

Title: chose a description one;

Data: FILE = cMIdataList.dat; 
TYPE = Imputation;

Variable: Names = ImpNum idschool schcomp num4th affluent econdis sclimate short urban suburban AllLun SomLun npercomp schnet girl screenT RdgEasy timerdg bschnet wLevel2 wLevel1 AlunComp SlunComp AlunScr SlunScr UrbanScr SubScr CompScr mscreenT mRdgEasy mtimeRdg cscreenT ctimeRdg cRdgEasy;

USEVARIABLES idschool bschnet girl screenT ctimeRdg 
mTimeRdg short AllLun 

CATEGORICAL = bschnet;
CLUSTER = idschool;
WITHIN = girl screenT ctimeRdg;
BETWEEN = mTimeRdg short AllLun;

ANALYSIS: TYPE = twolevel random;
ESTIMATOR = mlr;
ALGORITHM=INTEGRATION;

MODEL: %WITHIN%
   bschnet on girl screenT ctimeRdg 
%BETWEEN%
   bschnet on short mTimeRdg AllLun ;

Unlike SAS where there is more of a generalized linear mixed effects approach to specifying models, MPlus put more focus on within versus between. Mplus has kind of a para-
graph/sentence structure to the code. The end of a command (i.e., “sentence”) is indicated by a semi-colon, “;”. The explanation of the above code follows.

- **Title:** chose a description one; You can put any title that you want here. It will be in the output. I gave a short one.

- **Data:** This “paragraph” provides information about the data.
  - The command FILE specifies the file with the data, which in this case is a list of data files.
  - The command TYPE indicates the type of data, which in this case is imputed.

- **Variable** This paragraph provide information about the variables in the data sets.
  - The command NAMES = gives the names of variables in the order that they are in the data file.
  - Not all model will use all the variables, so the command USEVARIABLES is used to indicate which variable will be used in the current model.
  - The command CATEGORICAL tells the program that variable bschnet is a categorical or discrete variable.
  - The command CLUSTER indicates that the data are multilevel and the variable idschool that defines the clusters or groups.
  - The command WITHIN specifies which variable are Level 1 or within predictors.
  - The command BETWEEN specific which variables are between or Level 2 predictors.

- **ANALYSIS:** is the paragraph that deals with estimation (method and algorithm) to be used. The commands here indicate a two level random effects model where the estimation method is maximum likelihood with empirical predictors (MLR) and numerical integration (quadrature) is to be used.

- **Model:** paragraph gets the models for within groups or clusters (i.e., models for students) through the %WITHIN% statement and the model for the between groups or clusters (i.e., models for the random coefficients) through the %BETWEEN% statement.

Unlike SAS, there is no addition steps required to combined the results of the imputations. These are done by Mplus.

### 3.3 Proportional Odds Model

The only change is to change “bschnet” to “schnet” in all places. In our data set “schnet” is the multi-category response variable. We should also note that the command CATEGORICAL assumes that the response variable is ordinal.
3.4 Models in Mplus with Weights

The only change that needed to be made is the addition of the following two lines to the Variables paragraph:

\[
\begin{align*}
\text{WEIGHT} &= \text{wLevel1}; \\
\text{BWEIGHT} &= \text{wLevel2}; \\
\end{align*}
\]

Mplus will (by default) scale the weights such that they sum to the observed sample sizes (number of school and number of students within schools).