Introduction to Weighted Mixed-Effects Models With WeMix

Developed by Paul Bailey, Claire Kelley, and Trang Nguyen*†

June 17, 2019

Introduction

The `WeMix` package fits **We**ighted **Mix**ed models, also known as a multilevel, mixed, or hierarchical linear model (HLM). The weights could be inverse selection probabilities, such as those developed for an education survey where schools are sampled probabilistically, and then students inside those schools are sampled probabilistically. Although mixed-effects models are already available in `R`, `WeMix` is unique in implementing methods for mixed models using weights at multiple levels and computing cluster-robust standard errors.

The package `lme4` fits mixed models when there are no weights or weights only for first-level units (Bates, Maechler, Bolker, & Walker, 2015) and is recommended when both of two conditions hold: no weights are above the first level, and cluster-robust standard errors are not required. `WeMix` can fit models with weights at every level of the model and also calculates cluster-robust standard errors that account for covariance between units in the same groups.

Linear models are fitted with an analytic solution to the integral; the method is similar to the one of Bates et al. (2015) but adapted for weighted hierarchical models—details of which can be found in the vignette "Weighted Linear Mixed-Effects Models." Nonlinear models are fit with numerical quadrature using a method similar to GLLAMM (Rabe-Hesketh, Skrondal, & Pickles, 2002, 2005; Rabe-Hesketh & Skrondal, 2006)—see Appendix A, "Binomial Model Fitting." Because the model applies weights at every level, the units must be nested in "levels," so `WeMix` only fits those models where the units have a nested structure.

Installing and Loading `WeMix`

Inside `R`, run the following command to install `WeMix`:

```
install.packages("WeMix")
```

Once the package is successfully installed, `WeMix` can be loaded with the following command:

```
library(WeMix)
```

Specifying a Mixed-Effects Model

To illustrate the functionality of `WeMix`, we will use an example based on publicly available data from the Programme for International Student Assessment (PISA) 2012 data for the United States (Organisation for Economic Co-operation and Development, 2013). PISA is a repeated multinational assessment of skills and attitudes of 15-year-olds, with students (the unit of observation) probabilistically sampled within schools (the second-level unit) that also are probabilistically sampled within the country. In the United States, there were 3,136 student observations in 157 schools. We provide examples of a model with a random intercept and a model with both a random slope and intercept.

*This publication was prepared for NCES under Contract No. ED-IES-12-D-0002 with the American Institutes for Research. Mention of trade names, commercial products, or organizations does not imply endorsement by the U.S. government.

†The authors would like to thank Mike Cohen and Dan Sherman for reviewing this document, and Yuqi Liao, Bitnara Park, and Jiayi Li for their help using mixed models in other software packages.

The first model can be specified as the mathematics assessment predicted by a few variables chosen from the PISA survey:

- Dependent variable: `pv1math`, a mathematics assessment score
- Independent variables:
  - `escs` continuous socio-economic status index
  - `sc14q02` school questionnaire item: "Is your school's capacity to provide instruction hindered by any of the following... A lack of qualified mathematics teachers" (levels: "A lot"; "To some extent"; "Very little"; "Not at all", the reference group)
  - `st04q02` student gender (levels: "male" and "female," the reference group)
  - `st29q03` student questionnaire item: "I look forward to mathematics lessons" (levels: "Strongly agree"; "Agree"; "Disagree"; "Strongly disagree", the reference group)
- School-level random effect: school intercept
- Student-level weights: `pwt1`
- School-level weights: `pwt2`

In the second model, a second random effect is added at the school level for the `escs` variable, but there is no covariance between the two random effects.

An appendix describes the transformation used to prepare the data for analysis.

`WeMix` calls for this model, using a `data.frame` containing the PISA 2012 data for the United States and named "data," would be as follows:

```
# model with one random effect
m1 <-  mix(pv1math ~ st29q03 + sc14q02 +st04q01+escs+ (1|schoolid), data=data,
    weights=c("w_fstuwt", "w_fschwt"))

# model with two random effects assuming zero correlation between the two
m2 <- mix(pv1math ~ st29q03 + sc14q02 +st04q01+escs+ (1|schoolid)+ (0+escs|schoolid),
    data=data, weights=c("w_fstuwt", "w_fschwt"))

#Wald tests for beta and Lambda parameters
waldTest(m2,type="beta")
waldTest(m2,type="Lambda")
```

Note that the syntax for the model formula is the same syntax one would use to specify a model using `lme4`, except the weights argument. Thus, in the random slope and intercept model, the slope and intercept are in separate terms to constrain their covariance to be zero. For the weights argument, the weights are specified as a vector with the first element giving the name of the weights for level 1 units, the second element giving the name of the weights for the level 2 groups, and—when fitting a model with three levels—the third element giving the name of the weights for the level 3 groups.

The `WeMix` package assumes that the weights are unconditional—as is typical for sample weights found on international surveys such as PISA—but can accept conditional weights by setting the `cWeights` argument to `TRUE`.

Comparison to Alternate Software

For readers familiar with the specifications of other software, this section shows results compared with those from Stata `mixed`, Stata GLLAMM, SAS `PROC GLIMMIX`, HLM, and M+. When possible, the code specifies a random slope model and then a random slope and intercept model with the covariance of slope and intercept fixed to be zero. The models are fitted by maximum likelihood estimation and example code for Stata GLLAMM, Stata `mixed`, SAS `PROC GLIMMIX`, HLM, and M+ are shown in Appendix B.

Table 1 shows the results of the model with a single random effect, where `WeMix`, GLLAMM, Stata `mixed`, M+, and SAS `PROC GLIMMIX` show agreement on the likelihood, variance estimates of random effects, and fixed effects. HLM normalizes the weights (for both students and schools), so it estimates a related but different model. All the programs differ in the standard errors estimated for the fixed effects; `WeMix` most closely matches the results from Stata `mixed`.

Table 1. Results for Random Intercept Model

	WeMix	Stata: mixed	Stata: GLLAMM	SAS	HLM[a]	M+
Run Time	00:29	00:30	02:00	00:03	00:10	00:05
Log-Likelihood	−12789992.0	−12789991.9	−12789991.9	−12789992.0	−17965.44	−12789992.0
Variance Estimates						
Intercept	1413.81	1413.85	1413.83	1413.81	1020.05	1413.79
Residual	5264.80	5264.80	5264.80	5264.80	5061.13	5264.81
Standard Error (SE) of Variance Estimates						
SE Intercept	327.50	340.75	365.91	364.73	139.26	364.725
SE Residual	152.49	152.58	152.42	151.93	131.07	151.93
Fixed Effects						
(Intercept)	486.80	486.80	486.80	486.8	494.47	486.804
st29q03: A[b]	−11.108	−11.108	−11.108	−11.108	−24.119	−11.109
st29q03: SD	−41.542	−41.542	−41.542	−41.5422	−54.788	−41.543
st29q03: D	−19.255	−19.255	−19.255	−19.255	−26.110	−19.256
sc14q02: A lot[c]	−26.913	−26.913	−26.913	−26.9126	−24.504	−26.912
sc14q02: TSE	−11.782	−11.782	−11.782	−11.7824	−10.362	−11.782
sc14q02: VL	−21.340	−21.341	−21.340	−21.340	−19.506	−21.341
st04q01: M	9.5077	9.5077	9.5077	9.5077	12.164	9.508
escs	25.568	25.568	25.568	25.568	28.332	25.568
Standard Error of Fixed Effects						
SE (Intercept)	7.7780	7.7780	7.9545	7.929	7.2026	7.929
SE st29q03: A[b]	5.6998	5.6998	5.6998	5.6816	7.1401	5.682
SE st29q03: SD	6.8643	6.8643	6.8629	6.84	12.108	6.841
SE st29q03: D	5.4556	5.4556	5.4574	5.4400	6.2607	5.440
SE sc14q02: A lot[c]	7.6573	7.6573	7.7959	7.7714	8.2081	7.772
SE sc14q02: TSE	12.821	12.821	13.000	12.958	13.587	12.958
SE sc14q02: VL	17.060	17.061	17.294	17.239	15.286	17.239
SE st04q01: M	2.9860	2.9860	2.9850	2.9754	3.7008	2.975
SE escs	2.1175	2.1175	2.1373	2.131	2.5250	2.130

[a] The HLM software requires that the weights are normalized before they are fit, so the results do not match exactly. Also, HLM cannot set slope and intercept covariance to 0 (Raudenbush, Bryk, Cheong, Congdon, & Toit, 2016).
[b] levels for the variable st29q03 are "Strongly Agree" (the reference level), "Agree" (A), "Disagree" (D), and "Strongly Disagree" (SD).
[c] levels for the variable sc14q02 are "Not at all" (the reference group), "A lot", "To some extent" (TSE), "Very little" (VL).

Table 2 shows the results of the model with two random effects. Here the results are similar. `WeMix`, Stata `mixed`, M+, and SAS `PROC GLIMMIX` show agreement on the likelihood, variance term estimates of random effects, and fixed effects. However, in this case, Stata GLLAMM is unable to converge. After 150 iterations of the optimization phase, GLLAMM fails with warning "convergence not achieved." In HLM, we fit the most similar model we could get the software to fit, but it is not the same, so the results are different. Similar to the one random effect model, Stata `mixed` reports a lower standard error for the random intercept than most other models. In terms of the standard errors of the fixed effects, differences are evident between the estimates of all the programs; `WeMix` most closely matches Stata `mixed`.

Table 2. Results for Random Intercept and Slope Model

	WeMix	Stata: mixed	Stata: GLLAMM[a]	SAS	HLM[b]	M+
Run Time	0:36	00:30	—	00:10	00:10	00:05
Log-Likelihood	−12741522.7	−12741522.7	—	−12741522.5	−17964.0	−12741522.7
Variance Estimates						
Intercept	1354.71	1354.74	—	1354.71	984.87	1354.71
escs slope	370.33	370.33	—	370.33	42.957	370.33
Residual	4967.36	4967.36	—	4967.36	5010.3	4967.36
Standard Error of Variance Estimates						
Intercept	287.89	296.97	—	311.72	137.65	311.73
escs slope	67.99	68.566	—	66.325	48.378	66.324
Residual	137.70	137.74	—	137.36	132.48	137.36
Fixed Effects						
(Intercept)	483.89	483.89	—	483.89	494.03	483.89
st29q03: A[c]	−10.628	−10.628	—	−10.628	−24.0123	−10.628
st29q03: SD	−38.708	−38.708		−38.7081	−54.471	−38.708
st29q03: D	−17.303	−17.303	—	−17.3031	−25.909	−17.303
sc14q02: A lot[d]	−39.941	−39.941	—	−39.941	−27.782	−39.941
sc14q02: TSE	−12.592	−12.591		−12.592	−11.048	−12.592
sc14q02: VL	−20.064	−20.064	—	−20.0644	−19.513	−20.064
st04q01: M	10.294	10.294	—	10.294	12.252	10.294
escs	28.016	28.016	—	28.016	28.625	29.067
Standard Error of Fixed Effects						
(Intercept)	7.4055	7.4054	—	7.4985	7.1488	7.499
st29q03: A[c]	5.4905	5.4905	—	5.4737	7.1236	5.474
st29q03: SD	6.7682	6.7681	—	6.7593	12.1794	6.759
st29q03: D	5.3723	5.3722	—	5.3572	6.2107	5.357
sc14q02: A lot[d]	7.2393	7.2393	—	7.4109	7.3716	7.411
sc14q02: TSE	12.669	12.668	—	12.831	13.497	12.831
sc14q02: VL	15.583	15.583	—	15.5911	15.008	15.591
st04q01: M	3.0285	3.0285	—	3.0188	3.7084	3.709
escs	2.5631	2.5631	—	2.5341	2.6105	3.019

[a] Stata GLLAMM does not converge for this model.
[b] HLM requires that the weights are normalized before they are fit, so the results do not match exactly. Also, HLM cannot set slope and intercept covariance to 0 (Raudenbush, Bryk, Cheong, Congdon, & Toit, 2016).
[c] levels for the variable st29q03 are "Strongly Agree" (the reference level), "Agree" (A), "Disagree" (D), and "Strongly Disagree" (SD).
[d] levels for the variable sc14q02 are "Not at all" (the reference group), "A lot", "To some extent" (TSE), "Very little" (VL).

Mathematical Specification

`WeMix` maximizes similar likelihood functions for both linear and nonlinear models.

The simplest version of this model has two levels and has the form:

$$\boldsymbol{y} = \boldsymbol{X\beta} + \boldsymbol{Zu} + \boldsymbol{\epsilon} \ , \tag{1}$$

where $\boldsymbol{y}$ is the vector of outcomes, $\boldsymbol{X}$ is a matrix of covariates associated with regressors that are assumed to be fixed, $\boldsymbol{\beta}$ is the vector of fixed-effect regression coefficients, $\boldsymbol{Z}$ is a matrix of covariates associated with regressors that are assumed to be random, and $\boldsymbol{u}$ is the vector of random effects. The meaning of $\boldsymbol{u}$ being random is simply that a level is shared within a group and across groups:

$$\boldsymbol{u} \sim \text{MVN}(\vec{\boldsymbol{0}}, \boldsymbol{\Sigma}) \tag{2}$$

where $\text{MVN}(\vec{\boldsymbol{0}}, \boldsymbol{\Sigma})$ is the multivariate-normal distribution with mean $\vec{\boldsymbol{0}}$ (a vector of all zeros), and covariance $\boldsymbol{\Sigma}$.

Hierarchical Linear Models Notation

The models `WeMix` fits also can be called HLMs (Raudenbush & Bryk, 2002) where the first level has the following form:

$$y_{ij} = \beta_{0j} + X\beta_{1j} + \epsilon_{ij} \tag{3}$$

So, the second level could then be, for a random slope and intercept model, as follows:

$$\beta_{0j} = \gamma_{00} + \delta_{0j} \tag{4}$$

$$\beta_{1j} = \gamma_{01} + \delta_{1j} \tag{5}$$

where δ_{0j} and δ_{1j} are the error terms for the intercept and slope, respectively, and have variances of τ_{00} and τ_{11}, respectively, and δ_{0j} and δ_{1j} have covariance τ_{01}.

This is just one example; many other models also can be fit in `WeMix`. `WeMix` can fit models that are stated as an HLM or as a mixed model. For notational convenience for the rest of this document, we will use the non-HLM mixed model notation.

Multiple Levels

When there are more than two levels, eq. 1 can be rewritten as follows:

$$\boldsymbol{y} = \boldsymbol{X\beta} + \sum_{l=2}^{L} \boldsymbol{Z}^{(l)}\boldsymbol{u}^{(l)} + \boldsymbol{\epsilon} \tag{6}$$

where a superscript (l) is added to $\boldsymbol{Z}$ and $\boldsymbol{u}$ to indicate that they are at the lth level. Note: In the summation, l starts at $l = 2$ because random effects cannot be at the lowest level of observation ($l = 1$).

As of `WeMix 3.0`, models with up to three levels are supported.

Cluster-Robust Variance Estimation

The variance estimator was calculated following the method of Rabe-Hesketh and Skrondal (2006). Thus, the variance is expressed as follows:

$$\text{var}(\boldsymbol{u}) = \boldsymbol{I^{-1}JI^{-1}} \tag{7}$$

where $\boldsymbol{I}$ is the observed Fisher information matrix, which is calculated by observing the Fisher information at the maximum likelihood estimates. Given that the likelihood is twice differentiable, we estimate the Fisher information as the second derivative (Hessian) of the log-likelihood evaluated at the maximum likelihood estimate:

$$\boldsymbol{I} = \frac{\partial^2 \ell}{\partial \theta^2} \tag{8}$$

and $\boldsymbol{J}$ is estimated as follows:

$$\boldsymbol{J} = \sum_{L} \frac{n_L}{n_L - 1} \sum_{j'} \frac{\partial \ell_{j'}^{L-1}}{\partial \theta} \cdot \left[\frac{\partial \ell_{j'}^{L-1}}{\partial \theta} \right]^T \tag{9}$$

where n_L represents the number of observations in the $(L)^{\text{th}}$ level (i.e., the top level), and the subscript L indicates that the outermost sum is over the groups or units j' in the $(L-1)^{\text{th}}$ level.

Hierarchical Generalized Linear Models

If data come in a nested structure and the assumptions of linearity and normality cannot be met, even after applying transformations, hierarchical generalized linear models (HGLMs) offer a possible solution. An HGLM model has three components: a sampling model, a link function, and a structural model (Raudenbush & Bryk, 2002). Since `WeMix 2.0` has supported two models: linear models and binomial sampling model with logit link function (for binary outcomes). The binomial models can be specified using the `family` argument in the `mix` function. Note that the first model (linear models) is the default, so there is no need to specify the `family` argument for linear models.

Binomial With Logit Link Function

This model is used when the data have binary outcomes (i.e., the number of successes in m trials). Here is an example of code using the **sleepstudy** data set in the `lme4` package:

```
library(lme4)
library(WeMix)
ss1 <- sleepstudy
doubles <- c(308, 309, 310) # subject with double obs

# Create weights
ss1$W1 <- ifelse(ss1$Subject %in% doubles, 2, 1)
ss1$W2 <- 1

# Create binary outcome variable called "over300"
ss1$over300 <- ifelse(sleepstudy$Reaction<300,0,1)

# Run mixed model with random intercept and fixed slope
bi_1 <- mix(over300~ Days + (1|Subject),data=ss1,
            family=binomial(link="logit"),verbose=FALSE,
            weights=c("W1", "W2"),nQuad=13)
```

Table 3 shows the output comparison between `WeMix` and Stata GLLAMM.

Table 3. Results for Binomial Model With Logit Link Function

	WeMix	Stata: GLLAMM
Log-Likelihood	-93.75179	93.75168
Random Effects		
Var Random Intercept	4.127208	4.13350
Standard Error (SE) of Random Effects		
SE Random Intercept	2.966179	2.98157
Fixed Effects		
(Intercept)	−3.34411	−3.34499
Days	0.59271	0.59285
Standard Error of Fixed Effects		
SE (Intercept)	0.92390	0.9250737
SE Days	0.12429	0.1244216

Weight Adjustments

`WeMix` assumes that the weights are already scaled according to the user's needs. This section describes two common methods that a user may wish to implement.

The desire to reweight comes from evidence shown by Pfeffermann, Skinner, Holmes, Goldstein, & Rasbash (1998) that reweighting (adjusting the inverse sampling probabilities) can improve the random effect variance estimator in a linear model, especially when the sample of units is small in groups, and from Rabe-Hesketh & Skrondal (2006) wherein even the coefficients can be highly biased in the binomial case when the inverse sampling probabilities are used. Carle (2009) also generally recommends rescaling. All the papers show indications that all methods lead to biased estimates in some cases.

Method A:

$$w_{ij}^* = w_{ij}\left(\frac{n_j}{\sum_i w_{ij}}\right) \tag{10}$$

where w_{ij} are the full sample weights, i indexes the individuals, j indexes the groups, and n_j represents the number of observations in group j.

Method B:

$$w_{ij}^* = w_{ij}\left(\frac{\sum_i w_{ij}}{\sum_i w_{ij}^2}\right) \tag{11}$$

These w^* are then used to scale the likelihood function.

Centering

Since version 2, `WeMix` has allowed users to incorporate grand- or group-mean centering when fitting mixed-effects models. In the group-mean centered model, the predictors are centered on the group-level mean. Compared with eq. 8, the model can be expressed as follows:

$$y_{ij} = \beta_{0j} + \beta_{1j}(X_{ij} - \overline{X_{.j}}) + \epsilon_{ij} \tag{12}$$

and the intercept can be interpreted as the unadjusted mean for group j:

$$\beta_{0j} = \mu_{Y_j} \tag{13}$$

In the grand-mean centered model, the predictors are centered on the overall mean, so the model can be written as follows:

$$y_{ij} = \beta_{0j} + \beta_{1j}(X_{ij} - \overline{X_{..}}) + \epsilon_{ij} \tag{14}$$

and the intercept can be interpreted as the adjusted mean for group *j:*

$$\beta_{0j} = \mu_{Y_j} - \beta_{1j}(X_{ij} - \overline{X_{..}}) \tag{15}$$

There are two main advantages of centering the predictors. First of all, centering makes estimates of level 1 coefficients (β_{0j}) and other effects easier to interpret because it decomposes the relationship between predictors and outcome into within-group and between-group components. Second, it removes high correlations between the random intercept and slopes, as well as high correlations between first- and second-level predictors and cross-level interactions (Kreft & de Leeuw, 1998).

The following example shows how to implement group- or grand-mean centering:

```
library(lme4)  #to use example sleep study data

#create dummy weights
sleepstudy$weight1L1 <- 1
sleepstudy$weight1L2 <- 1

# Group mean centering of the variable Days within group Subject
```

```
group_center <- mix(Reaction ~ Days + (1|Subject), data=sleepstudy,
                    center_group=list("Subject"= ~Days),
                    weights=c("weight1L1", "weight1L2"))

# Grand mean centering of the variable Days
grand_center <- mix(Reaction ~ Days + (1|Subject), data=sleepstudy,
                    center_grand=~Days,weights=c("weight1L1", "weight1L2"))
```

References

Bates, D., Maechler, M., Bolker, B., & Walker, S. (2015). Fitting linear mixed-effects models using lme4. *Journal of Statistical Software*, *67*(1), 1–48.

Carle, A. C. (2009). Fitting multilevel models in complex survey data with design weights: Recommendations. *BMC Medical Research Methodology*, *9*(1), 1–13.

Hartzel, J., Agresti, A., & Caffo, B. (2001). Multinomial logit random effects models. *Statistical Modelling: An International Journal*, *1*(2), 81–102.

Kreft, I. G., & de Leeuw, J. (1998). *Introducing statistical methods: Introducing multilevel modeling.* London, UK: SAGE

Liu, Q., & Pierce, D. A. (1994). A note on Gauss-Hermite quadrature. *Biometrika*, *81*(3), 624.

Organisation for Economic Co-operation and Development. (2013). *PISA 2012 assessment and analytical framework: Mathematics, reading, science, problem solving and financial literacy.* Paris, France: OECD Publishing. Retrieved from http://www.oecd.org/pisa/pisaproducts/PISA%202012%20framework%20e-book_final.pdf

Pfeffermann, D., Skinner, C., Holmes, D., Goldstein, H., & Rasbash, J. (1998). Weighting for unequal selection probabilities in multilevel models. *Journal of the Royal Statistical Society.* Series B (Statistical Methodology), *60*(1), 23–40.

Rabe-Hesketh, S., & Skrondal, A. (2006). Multilevel modelling of complex survey data. *Journal of the Royal Statistical Society.* Series A (Statistics in Society), *169*(4), 805–827.

Rabe-Hesketh, S., Skrondal, A., & Pickles, A. (2002). Reliable estimation of generalized linear mixed models using adaptive quadrature. *Stata Journal*, 2, 1–21.

Rabe-Hesketh, S., Skrondal, A., & Pickles, A. (2004). *GLLAMM manual* (Working Paper No. 160). Berkeley, CA: University of California–Berkeley, Division of Biostatistics.

Rabe-Hesketh, S., Skrondal, A., & Pickles, A. (2005). Maximum likelihood estimation of limited and discrete dependent variable models with nested random effects. *Journal of Econometrics. 128*(2), 301–323.

Raudenbush, S. W., & Bryk, A. S. (2002). *Hierarchical linear models: Applications and data analysis methods* (2nd ed.). Thousand Oaks, CA: SAGE.

Raudenbush, S. W., Bryk, A. S., Cheong, Y. F., Congdon, R. T., & Toit, M. (2016). *HLM7 hierarchical linear and nonlinear modeling user manual: User guide for Scientific Software International's (S.S.I.) Program.* Lincolnwood, IL: Scientific Software International.

SAS Institute Inc. (2013). *SAS/ACCESS 9.4 interface to ADABAS: Reference.* Cary, NC: Author.

Smyth, G. K. (1998). Numerical integration. In P. Armitage & T. Colton (Eds.), *Encyclopedia of biostatistics* (pp. 3088–3095). London, UK: Wiley.

StataCorp. (2015). *Stata statistical software: Release 14.* College Station, TX: Author.

Appendix A. Binomial Model Fitting

Table A1. Notation

$\boldsymbol{\theta}$	vector of all the model parameters fit, including $\boldsymbol{\beta}$ and the nonduplicated elements of the covariate matrices $\boldsymbol{\Sigma}^{(l)}$. Because they are integrated out, $\boldsymbol{u}$ is included in $\boldsymbol{\theta}$.
$\boldsymbol{\Sigma}^{(l)}$	covariance matrix for level l
$\boldsymbol{y}$	response vector
$\boldsymbol{X}$	covariate matrix for fixed effects
$\boldsymbol{\beta}$	coefficients of the fixed effects
$\boldsymbol{Z}^{(l)}$	covariate matrix for random effects at level l
$\boldsymbol{Z}$	covariate matrix for all random effects
$\boldsymbol{u}^{(l)}$	vector of the random effects at level l
$\boldsymbol{U}^{(l)}$	vector of the random effects at all levels l and higher $(u^l, ..., u^L)'$
$\boldsymbol{w}^{(l)}$	vector of the weights at level l
$\boldsymbol{W}$	matrix of the weights at all levels
k_l	number of random effects at level l
L	number of levels in the model
i	subscript denoting the individual
j	subscript denoting group
j'	subscript denoting a group within j
$\mathcal{L}^{(l)}$	likelihood function at level l
$\ell^{(l)}$	log-likelihood function at level l
$\boldsymbol{\epsilon}$	regression residuals, net of fixed and random effects

The central concern in `WeMix` is properly incorporating sampling weights into the mixed model. Because each individual or group may have an unequal probability of selection into the sample, the estimate of the distribution of the multivariate normal must include those weights to correctly estimate the parameters of the distribution. Also, except for trivial cases, the nested nature of the multilevel model creates a likelihood function that is not analytically calculable.

We consider the likelihood as a summation at each level, starting with the lowest (unit) level. The likelihood is conditional on the random effects at each higher level and is scaled by the weights at the lowest level.

In the case of the normal distribution, the likelihood ($\mathcal{L}^{(1)}$) at the individual unit level is given by the equations that follow. Note that here the subscript i is used to indicate that this is the likelihood of the i^{th} individual.

$$\mathcal{L}_i^{(1)}(\boldsymbol{\theta}; \boldsymbol{y}, \boldsymbol{U}^{(l)}, \boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W}) = \frac{1}{\Sigma^{(1)}} \phi\left(\frac{\hat{e_i}}{\Sigma^{(1)}}\right) \tag{16}$$

where $\phi(\cdot)$ is the standard normal density function, and $\Sigma^{(1)}$ is the residual variance (a scalar). The residuals vector $\hat{\boldsymbol{e}}$ represents the residuals $\hat{e}_i$ for each individual, which is calculated as follows:

$$\hat{\boldsymbol{e}} = \boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}} - \sum_{l=2}^{L} \boldsymbol{Z}^{(l)} \hat{\boldsymbol{u}}^{(l)} \tag{17}$$

The conditional likelihood given the vector $\boldsymbol{U}^{(l+1)}$ of random effects at the $l+1$ level and higher is then recursively defined, for the jth unit, at level l ($\mathcal{L}_j^{(l)}$; Rabe-Hesketh et al. 2002, eq. 3):

$$\mathcal{L}_j^{(l)}(\boldsymbol{\theta}; \boldsymbol{y}, \boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W} | \boldsymbol{U}^{(l+1)}) = \int_{-\infty}^{\infty} ... \int_{-\infty}^{\infty} g^{(l)}(\boldsymbol{u}^{(l)}) \prod_{j'} \left[\mathcal{L}_{j'}^{(l-1)}(\boldsymbol{\theta}; \boldsymbol{y}, \boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W} | \boldsymbol{U}^{(l)})\right]^{\boldsymbol{w}_{j'}^{(l-1)}} du_1^{(l)} ... du_{k_l}^{(l)} \tag{18}$$

where the subscript j' that the product is indexed over indicates that the likelihood $\mathcal{L}_{j'}^{(l-1)}$ is for the units of level $l-1$ nested in unit j. In addition, $g(\boldsymbol{u}^{(l)})$ is the empirical Bayes "prior" probability density of the random effects (at level l) having a value of $u^{(l)}$, so $g(\boldsymbol{0}, \boldsymbol{\Sigma})$ is a multivariate normal distribution parameterized by a mean of $\boldsymbol{0}$ and variance $\boldsymbol{\Sigma}$. The integrals are over the elements of $\boldsymbol{u} = (u_1, ..., u_{k_l})$, where k_l is the number of random effects at

level l. It is important to note that each $\mathcal{L}^{(l)}$ is independent for each j'. This allows us to integrate out the values of u for all groups simultaneously, which leaves us with a k_l dimensional integral and is essential to making this problem computationally feasible. At the highest level, the result is not conditional on any $\boldsymbol{u}$ values but is otherwise the same.

For ease of computation and to avoid problems with accurate storage of extremely small numbers, we first take the log of the function before maximizing. Following Rabe-Hesketh & Skrondal (2006, eq. 1), the log-likelihood function is as follows:

$$\ell_j^{(l)}(\boldsymbol{\theta}; \boldsymbol{y}, \boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W}|\boldsymbol{U}^{(l+1)}) = \ln\left\{\int ... \int g^{(l-1)}(u^{(l-1)}) \cdot \exp\left[\sum_{j'} \boldsymbol{w}_{j'}^{(l-1)} \ell_{j'}^{(l-l)}(\boldsymbol{\theta}; \boldsymbol{y}, \boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W}|U^{(l)})\right] du_1^{(l)} ... du_{k_l}^{(l)}\right\} \tag{19}$$

where $\ell_j^{(l)}$ is the log-likelihood function for the jth unit at level l.

Adaptive Gauss-Hermite Quadrature

Unfortunately, in the nonlinear case, there is no closed-form expression for the integral in eq. 18, and so we must evaluate the likelihood numerically. This is possible with adaptive Gauss-Hermite quadrature, which is a modification of Gauss-Hermite quadrature.

First, to evaluate the integral at level l, we must evaluate the integral over k_l random effects variables that have a covariance matrix $\boldsymbol{\Sigma}^{(l)}$. To avoid dependence, we use a change of variables to create independent and standard normal distributed vectors $v^{(l)}$. Using the Cholesky decomposition $\boldsymbol{\Sigma}^{(l)} = \boldsymbol{C}^{(l)}\left(\boldsymbol{C}^{(l)}\right)^T$, the value of $u^{(l)}$ can then be calculated as $u^{(l)} = \boldsymbol{C}^{(l)}\boldsymbol{v}^{(l)}$.

For nonlinear models, we use the transformation of variables previously described above:

$$\ell_j^{(l)} = \int ... \int g^{(l)}(u^{(l)}) \cdot \exp\left[\sum_{j'} \boldsymbol{w}_{j'}^{(l-1)} \ell_{j'}^{(l-l)}(\boldsymbol{\theta}; \boldsymbol{y}, \boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W}|U^{(l)})\right] du^{(l)} \tag{20}$$

$$= \int \phi(v_{k_l}^{(l)}) ... \int \phi(v_1^{(l)}) \cdot \exp\left[\sum_{j'} \boldsymbol{w}_{j'}^{(l-1)} \ell_{j'}^{(l-l)}(\boldsymbol{\theta}; \boldsymbol{y}, \boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W}|U^{(l)})\right] du^{(l)} \tag{21}$$

where $u^{(l)}$ is now a function of the v values and ϕ is the standard normal density.

Quadrature replaces integration with a summation over a finite set of points (known as quadrature points) and weights so that the sum approximates the integral. We annotate the quadrature points with a tilde so that, for example, u becomes $\tilde{u}$.

These equations come from Rabe-Hesketh et al. (2002, p. 5, eq. 4) and follow from eq. 21.

$$\ell_j^{(l)}(\boldsymbol{\theta}; \boldsymbol{y}, \boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W}|\boldsymbol{U}^{(l+1)})) \approx \sum_{r_1=1}^{R} p_{r_1}^{(l)} ... \sum_{r_{k_l}=1}^{R} p_{r_{k_l}}^{(l)} \cdot \exp\left[\sum_{j'} \boldsymbol{w}_{j'}^{(l-1)} \ell_{j'}^{(l-l)}(\boldsymbol{\theta}; \boldsymbol{y}, \boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W}|\tilde{\boldsymbol{u}}^{(l)}, \boldsymbol{U}^{(l+1)})\right] \tag{22}$$

where R is the number of quadrature points, $\boldsymbol{p}$ are the quadrature weights, and $\tilde{\boldsymbol{u}}^{(l)} = C^{(l)}\tilde{\boldsymbol{v}}^{(l)}$ are the quadrature point locations for each random effect vector. This results in a grid with R quadrature points per element in u (and v), for a total of R^{k_l} quadrature points, and the summation is over every point in that grid. The quadrature points and weights come from the `statsmod` implementation of Gaussian quadrature in `R` (Smyth 1998).

Although Gauss-Hermite quadrature centers the quadrature points on the prior distribution, adaptive Gauss-Hermite quadrature (AGHQ) centers the quadrature points on either the likelihood surface or the posterior distribution. We use the posterior maximum, the likelihood function of which, including $g(\cdot)$, is detailed next, but this section is general to AGHQ as detailed in Lui and Pierce (1994) and Hartzel, Agresti, and Caffo (2001); we use notation similar to Hartzel et al. (2001, p. 87). The goal of adaptive quadrature is to reduce the number of quadrature points needed for an accurate evaluation of the integral by centering the points closer to the bulk of the integral. The location of the points is scaled based on the values of the likelihood function. To do this, the mode of the likelihood

is used to center the points, and the dispersion is the estimated standard deviation of the likelihood at that point (using the inverse second derivative):

$$\tilde{\boldsymbol{u}} = \hat{\boldsymbol{\omega}}_j + \sqrt{2}\hat{\boldsymbol{Q}}_j^{1/2}\tilde{\boldsymbol{v}} \tag{23}$$

where $\tilde{\boldsymbol{u}}_i^{(l)}$ are the quadrature points, $\hat{\boldsymbol{\omega}}_i$ is a vector of locations for group j, and $\hat{\boldsymbol{Q}}_j$ is the matrix that is the inverse numerical second derivative of the likelihood function evaluated at the unit normal iid points $\boldsymbol{v}$.

We can then use the adapted quadrature points to calculate the log likelihood as follows:

$$\ell_j^{(l)}(\boldsymbol{\theta}; \boldsymbol{y}, \boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W}|\boldsymbol{U}^{(l+1)}) \approx$$
$$\sum_{r_1=1}^{R} p_{r_1}^{(l)} ... \sum_{r_{k_l}=1}^{R} p_{r_{k_l}}^{(l)} \cdot \exp\left[\sum_{j'} \boldsymbol{w}_{j'}^{(l-1)} \ell_{j'}^{(l-1)}(\boldsymbol{\theta}; \boldsymbol{y}, \boldsymbol{X}\boldsymbol{Z}, \boldsymbol{W}|\tilde{u}_{r_1}...\tilde{u}_{r_{k_l}}, \boldsymbol{U}^{(l+1)}) + g(\tilde{\boldsymbol{u}}^{(l)}; \boldsymbol{\Sigma}^{(l)}) + \boldsymbol{v}^T\boldsymbol{v}\right] \tag{24}$$

This approximation of the likelihood can then be evaluated and minimized numerically. In this package, we minimize the function using Newton's method.

Calculation of Conditional Mode

AGHQ requires an estimated location ($\hat{\boldsymbol{\omega}}_j$) and variance ($\boldsymbol{Q}_j$) for each group. These are calculated by iteratively finding the conditional mode of the random effects (to find $\hat{\boldsymbol{\omega}}_j$) and then using the inverse of the second derivative of the likelihood surface as an estimate of $\boldsymbol{Q}_j$.

The conditional modes are identified by sequentially increasing the levels of l; the software first identifies the conditional mode at level 2 and then, using those estimates, uses AGHQ at level 2 to estimate conditional modes at level 3, and so on.

For each group, the conditional mode is identified using Newton's method on the likelihood for that unit ($\ell_j^{(l)}$) at a particular level of $u^{(l)}$, called $\hat{u}$, and conditional on an existing estimate of θ,

$$\ell_j^{(l)}(\hat{\boldsymbol{u}}^{(l)}; \boldsymbol{y}, \boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W}|\boldsymbol{\theta}) = \ln\left[g^{(l)}(\hat{\boldsymbol{u}}^{(l)}) \sum_{j'} \boldsymbol{w}_{j'}^{(l-1)} \ell_{j'}^{(l-1)}(\boldsymbol{\theta}; \boldsymbol{y}, \boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W}|\hat{\boldsymbol{u}}^{(l)})\right] \tag{25}$$

where this formulation implicitly sets all values of $\boldsymbol{U}^{(l)}$ to zero. Note that the values of $\ell_{j'}^{(l-l)}$ still integrate out the values of $\boldsymbol{u}$ for all levels below l.

Newton's method requires a first and second derivative, and these are calculated with numerical derivatives of the likelihood calculated using numerical quadrature.

Estimate of the Conditional Means

The conditional mean is estimated by simply taking the expected value of each parameter using the following equation:

$$E\left(\hat{\boldsymbol{u}}|\boldsymbol{y}, \boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W}, \boldsymbol{\theta}\right) = \frac{\int_{-\infty}^{\infty} \tilde{\boldsymbol{u}}^{(l-1)} \cdot \ell_j^{(l)}(\hat{\boldsymbol{u}}; \boldsymbol{y}, \boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W}|\boldsymbol{\theta}) d\hat{u}}{\int_{-\infty}^{\infty} \ell_j^{(l)}(\tilde{\boldsymbol{u}}^{(l-1)}; \boldsymbol{y}, \boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{W}|\boldsymbol{\theta}) d\hat{u}} \tag{26}$$

Appendix B. Alternative Software Specifications

For reference, these sections s how the specification of the models in Stata GLLAMM, Stata `mixed`, SAS `PROC GLIMMIX`, HLM, and M+.

Stata: GLLAMM

In Stata prior to Version 14, weighted mixed-effects models could be estimated only with GLLAMM (Rabe-Hesketh, Skrondal, & Pickles, 2004). The work of these GLLAMM authors provided the methods that we used in our implementation of nonlinear weighted mixed models in `WeMix`.

```
import delimited "PISA2012_USA.csv"

generate intercept = 1
eq intercept: intercept
eq slope: escs
tabulate st29q03, generate (st29q03d)
tabulate sc14q02, generate (sc14q02d)
tabulate st04q01, generate (st04q01d)

//Random intercept model
gllamm pv1math st29q03d1 st29q03d2 st29q03d4 sc14q02d1 sc14q02d3 sc14q02d4 st04q01d2
escs, i(schoolid) pweight(pwt) l(identity) f(gau) nip(27) nrf(1) eqs(intercept)
adapt nocorrel


//Random slope and intercept model
gllamm pv1math st29q03d1 st29q03d2 st29q03d4 sc14q02d1 sc14q02d3 sc14q02d4 st04q01d2
escs, i(schoolid) pweight(pwt) l(identity) f(gau) nip(13) nrf(2) eqs(intercept slope)
adapt nocorrel
```

Stata: mixed

In Stata Version 14, the `mixed` command includes the ability to fit models with survey weights (StataCorp, 2015).

```
import delimited "PISA2012_USA.csv"
tabulate st29q03, generate (st29q03d)
tabulate sc14q02, generate (sc14q02d)
tabulate st04q01, generate (st04q01d)

//Random intercept model
mixed pv1math st29q03d1 st29q03d2 st29q03d4 sc14q02d1 sc14q02d3 sc14q02d4 st04q01d2
escs [pw = pwt1] || schoolid: , pweight (pwt2)

//Random slope and intercept model
mixed pv1math st29q03d1 st29q03d2 st29q03d4 sc14q02d1 sc14q02d3 sc14q02d4 st04
q01d2 escs [pw = pwt1] || schoolid: escs, pweight (pwt2)
```

SAS `PROC GLIMMIX`

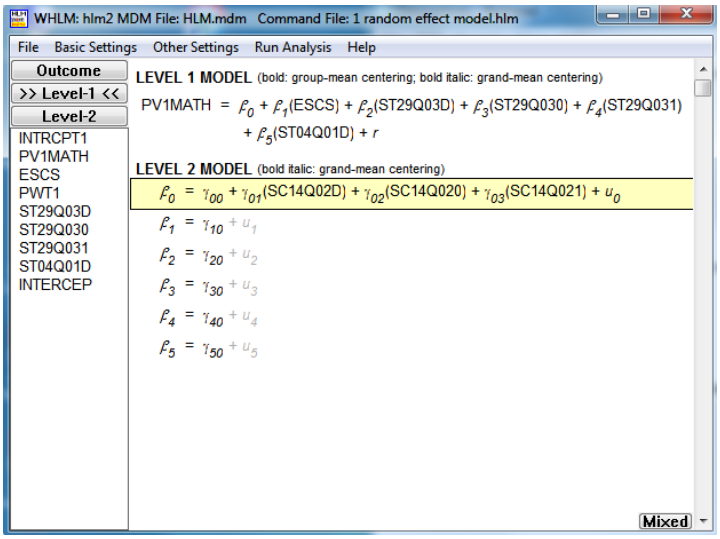
Model specification in SAS uses the `PROC GLIMMIX` procedure (SAS Institute Inc., 2013). It is notable here that when fit with the default optimization parameters, the model converged to a likelihood lower than the maximum likelihood estimate found by other software. Decreasing the convergence parameter GCONV to E-10 was necessary to find the same maximum likelihood as other software.

```
PROC IMIPORT DATAFILE="PISA2012_USA.csv"
     OUT=pisa_data DBMS=csv REPLACE;
RUN;
```

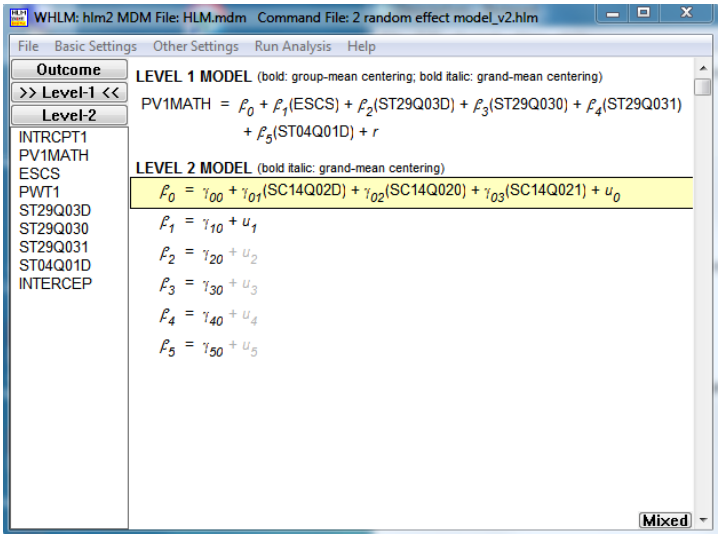
```
PROC GLIMMIX DATA=pisa_data METHOD=quadrature(qpoints=27) EMPIRICAL=classical NOREML;
  NLOPTIONS GCONV=1E-10 TECHNIQUE=TRUREG;
  CLASS  sc14q02(REF='Not at all') st04q01(REF='Female') st29q03(REF='Strongly agree');
  MODEL pv1math = escs sc14q02 st04q01 st29q03/ OBSWEIGHT=pwt1 SOLUTION;
  RANDOM INT/subject=schoolid WEIGHT=pwt2;
RUN;


PROC GLIMMIX DATA=pisa_data METHOD=quadrature(qpoints=13) EMPIRICAL=classical NOREML;
  NLOPTIONS GCONV=1E-10 TECHNIQUE=TRUREG;
  CLASS  sc14q02(REF='Not at all') st04q01(REF='Female') st29q03(REF='Strongly agree');
  MODEL pv1math = escs sc14q02 st04q01 st29q03/ OBSWEIGHT=pwt1 SOLUTION;
  RANDOM intercept escs/subject=schoolid WEIGHT=pwt2;
RUN;
```

HLM

HLM is another software package for estimated mixed-effects models (Raudenbush et al., 2016). It is important to note that HLM has two differences from the methods specified in other software. HLM normalizes all weights (which other programs do not) and also does not allow the correlation between slope and random effect to be fixed at 0. Using the "Diagonalize Tau" option reduces covariance but does not fix it at 0 (Raudenbush et al., 2016). In addition, HLM is entirely based on a graphical user interface. Specification of the HLM model for comparison here was done through the interface. The random intercept model was specified as follows:



The random slope and intercept model was specified as follows:



Note: The specifications for the random intercept model and the random slope model are extremely similar; in the second image for β_1, the u_1 is highlighted. This is how users in HLM add random effects.

M+

M+ is another common software used to fix mixed effects models. You should plan to make all categorical variables into dummy variables prior to fitting the model. To fit the random intercept model, use the following:

```
DATA: FILE= pisa_2012_with_dummies.csv;

VARIABLE: NAMES= id schoolid pv1math escs pwt1 pwt2 s29q03d1 s29q03d2
s29q03d4 int c14q02d1 c14q02d2 c14q02d4 s04q01d2;
CLUSTER = schoolid;
WITHIN = escs s29q03d1 s29q03d2 s29q03d4 c14q02d1 c14q02d2 c14q02d4 s04q01d2;

USEVARIABLES= schoolid pv1math escs  s29q03d1 s29q03d2
s29q03d4 c14q02d1 c14q02d2 c14q02d4 s04q01d2 pwt1 pwt2;

WEIGHT IS pwt1;
BWEIGHT = pwt2;
wtscale=unscaled;
bwtscale=unscaled;

ANALYSIS: TYPE = TWOLEVEL;

MODEL:
%WITHIN%
pv1math ON escs s29q03d1 s29q03d2 s29q03d4 c14q02d1 c14q02d2 c14q02d4 s04q01d2;
%BETWEEN%
pv1math;
```

To fit the random slope model, use the following:

```
DATA: FILE=pisa_2012_with_dummies.csv;

VARIABLE: NAMES= id schoolid pv1math escs pwt1 pwt2 s29q03d1 s29q03d2
s29q03d4 int c14q02d1 c14q02d2 c14q02d4 s04q01d2 escs2;
CLUSTER = schoolid;
WITHIN = escs s29q03d1 s29q03d2 s29q03d4 c14q02d1 c14q02d2 c14q02d4 s04q01d2;

USEVARIABLES= schoolid pv1math escs  s29q03d1 s29q03d2
```

```
s29q03d4 c14q02d1 c14q02d2 c14q02d4 s04q01d2 pwt1 pwt2;

WEIGHT IS pwt1;
BWEIGHT = pwt2;
wtscale=unscaled;
bwtscale=unscaled;

ANALYSIS: TYPE = TWOLEVEL RANDOM;

MODEL:
%WITHIN%
pv1math on s29q03d1 s29q03d2 s29q03d4 c14q02d1 c14q02d2 c14q02d4 s04q01d2;
slope | pv1math ON escs ;
%BETWEEN%
pv1math with slope @0
```

Appendix C. Example Data Preparation

Data are read using the `EdSurvey` package to access the PISA data efficiently.

```
library(EdSurvey)

#read in data
downloadPISA("~/", year=2012)
cntl <- readPISA("~/PISA/2012", countries="USA")
data <- getData(cntl,c("schoolid", "pv1math", "st29q03", "sc14q02", "st04q01",
                       "escs", "w_fschwt", "w_fstuwt"),
                omittedLevels=FALSE, addAttributes=FALSE)

# conditional weights
data$pwt2 <- data$w_fschwt
data$pwt1 <- data$w_fstuwt / data$w_fschwt

# Remove NA and omitted Levels
om <- c("Invalid","N/A","Missing","Miss",NA,"(Missing)")
for (i in 1:ncol(data)) {
  data <- data[!data[,i] %in% om,]
}

# relevel factors for model
data$st29q03 <- relevel(data$st29q03, ref="Strongly agree")
data$sc14q02 <- relevel(data$sc14q02, ref="Not at all")

write.csv(data, file="PISA2012_USA.csv")
```