Weighting in MLwiN

Rebecca Pillinger

8th September 2011

Centre for Multilevel Modelling
University of Bristol

The author wishes to gratefully acknowledge the contributions of Paul Clarke and Fiona Steele, who were very insistent about not being credited as authors, but who assisted greatly with the design of the simulation study, and aided the author's understanding of weighting through a series of in-depth discussions, and who also provided valuable feedback on a series of drafts.

Contents

1 Introduction to weights

1.1 Informative designs

The use of weights in an analysis is a strategy for dealing with an informative design. The informativeness or non-informativeness of a design is not a property of the sampling design by itself, but of the combination of sampling design and analysis to be performed. Thus for a given dataset the design may be informative or non-informative depending on what analysis is carried out on it; while given a particular method of analysis (e.g. specific regression model to be fitted), the design may be informative or non-informative depending on how the data it is applied to were sampled.

When data are collected from a sample of some population, some sampling design must be employed to determine which individuals from that population will comprise the sample. Such sampling designs may of course be non-random (e.g. convenience sampling or snowball sampling); however this lack of randomness makes it difficult to say anything mathematically about the design and for this reason we will be dealing in this document entirely with sampling designs which use random selection. A design which selects individuals into the sample randomly must assign each individual in the population a probability of being selected. This selection probability may be equal for all individuals (in which case the sampling design is known as simple random sampling or SRS). Alternatively, the selection probabilities may depend on the characteristics of the individuals. For example, the proportion of Black respondents in the sample used by the British Crime Survey[1] is greater than the proportion of Black individuals in the British population, i.e. Black indviduals are *oversampled*. (The survey is designed like this because if the proportion of Black respondents was the same as the proportion of British people who are Black there would be too few Black people sampled to provide accurate estimates). Thus each Black person in the British population has a greater probability of being selected into the sample than each non-Black person.

We can think of the probability of being selected into the sample for each individual in the population as being determined by a variable (or set of variables). In the case of SRS, the variable is simply a constant: taking on the value 1 for every individual in the dataset. This constant variable is then multiplied by the desired selection probability to give each individual's selection probability (which will of course be the same for every individual in the population). For a sample designed to include a greater proportion of individuals from each ethnic minority than the corresponding proportions in the population, the probability is determined by a variable indicating the ethnicity. The selection probabilities for the individuals in the population can be generated by creating dummy (indicator) variables from this ethnicity variable, multiplying the dummy for

[1] http://rds.homeoffice.gov.uk/rds/bcs1.html

White ethnicity by the selection probability for White individuals, the dummy for Black ethnicity by the selection probability for Black individuals, the dummy for Asian ethnicity by the selection probability for Asian individuals and so on, then adding all these together to give a single variable of selection probabilities. Selection probabilities could also be based on some continuous measure such as income (perhaps in such a way that the greater an individual's income the more likely they are to be selected into the sample, since the distribution of income in the population is often positively skewed, and we may wish to ensure that there are not too few individuals with higher incomes in our sample). It is also possible for several variables to be involved in calculating the selection probabilities. We will refer to the set of variables on which the selection probabilities depend as Z.

To explain what is meant by an informative design, now let us imagine a researcher who wishes to carry out an analysis involving an outcome variable y and explanatory variables x. y measures whether the individual is suffering from a particular health condition or not. x is the sex of the individual (coded 1 for women and 0 for men). Whether or not this researcher's design is informative depends on the relation between y and the z used to determine the selection probabilities for the data they are working with. For simplicity, let us imagine that Z consists simply of a binary White/non-White indicator (which we will refer to as z), with non-White individuals being oversampled.

Let us first imagine that the analysis the researcher wishes to perform is very simple. They simply want to estimate the proportion of individuals in the population who are suffering from the health condition of interest. (Thus the explanatory variable x does not enter into this analysis). They will use the natural estimator: the proportion of individuals in the sample who are suffering from the health condition. The researcher will calculate this proportion directly using y, by taking the mean of y, which is equivalent to this formula:

$$p = \frac{\sum_{i=1}^{n} y_i}{n} \tag{1}$$

where p is the sample proportion with the health condition, y_i is the value of y for the ith individual (equal to 1 if the individual has the health condition and 0 otherwise), and N is the number of individuals in the sample. However in order to understand how the proportion calculated using this formula may differ from the proportion of individuals in the population with the health condition, it is helpful for us to break the formula down into an equivalent but more complicated form, which calculates proportions separately for Whites and non-Whites and then adds these together, weighting for the sample proportions of Whites and non-Whites:

$$p = \frac{\sum_{i=1}^{n} (y_i \times \mathbf{white}_i)}{\sum_{i=1}^{n} \mathbf{white}_i} \frac{\sum_{i=1}^{n} \mathbf{white}_i}{n} + \frac{\sum_{i=1}^{n} (y_i \times \mathbf{nonwhite}_i)}{\sum_{i=1}^{n} \mathbf{nonwhite}_i} \frac{\sum_{i=1}^{n} \mathbf{nonwhite}_i}{n} \tag{2}$$

where p is the sample proportion with the health condition, **white** is a dummy variable taking the value 1 if the individual is White and 0 otherwise, and **nonwhite** is a dummy variable taking the value 1 if the individual is non-White and 0 otherwise. (It can readily be seen how this simplifies to the formula given in Equation (1).)

There are two possibilties. Either the proportion of individuals in the population suffering from the health condition is the same for White and non-White individuals (i.e. the probability of suffering from the health condition does not depend on ethnicity: y is unrelated to z); or the proportion of White individuals in the population suffering from the condition is different from the proportion of non-White individuals in the population suffering from the condition (i.e. the probability of suffering from the health condition does depend on ethnicity: y is related to x). In the first case, it does not matter that the proportion of non-White individuals in the sample is different to the proportion of non-White individuals in the population. The proportion of individuals in the sample suffering from the condition will still be an unbiased estimator of the proportion of individuals in the population suffering from the condition. Mathematically, looking at the formula for the sample proportion given in Equation (2), we will find that $\frac{\sum_{i=1}^{n}(y_i \times \mathbf{white}_i)}{\sum_{i=1}^{n} \mathbf{white}_i}$ is very close to $\frac{\sum_{i=1}^{n}(y_i \times \mathbf{nonwhite}_i)}{\sum_{i=1}^{n} \mathbf{nonwhite}_i}$). (They will differ only due to sampling variation). Thus the formula will be close to[2]

$$p = \frac{\sum_{i=1}^{n}(y_i \times \mathbf{white}_i)}{\sum_{i=1}^{n} \mathbf{white}_i} \frac{\sum_{i=1}^{n} \mathbf{white}_i}{n} + \frac{\sum_{i=1}^{n}(y_i \times \mathbf{white}_i)}{\sum_{i=1}^{n} \mathbf{white}_i} \frac{\sum_{i=1}^{n} \mathbf{nonwhite}_i}{n} = \frac{\sum_{i=1}^{n}(y_i \times \mathbf{white}_i)}{\sum_{i=1}^{n} \mathbf{white}_i} \frac{n}{n}$$

Thus the sample proportion of individuals with the health condition does not depend on the proportions of White and non-White individuals sampled; and so the researcher should expect to get a very similar estimate for the population proportion with the health condition using their data which oversamples non-White individuals compared to if they had used data collected using simple random sampling (the two estimates would differ only due to sampling variation, not due to the difference in the designs). This case, where y is unrelated to z, is called an *non-informative design*.

In the second case, however, where the proportions of White and of non-White individuals in the population with the condition are different, the proportion of sampled individuals with the condition will depend on the ethnic composition of the sample. For example, if the sample consists entirely of White individuals the proportion will equal $\frac{\sum_{i=1}^{n}(y_i \times \mathbf{white}_i)}{n}$ while if the sample consists entirely of non-White individuals it will equal $\frac{\sum_{i=1}^{n}(y_i \times \mathbf{nonwhite}_i)}{n}$. If half the sample is White and the other half non-White it will equal

[2] Equivalently, of course, we could replace $\frac{\sum_{i=1}^{n}(y_i \times \mathbf{white}_i)}{\sum_{i=1}^{n} \mathbf{white}_i}$ with $\frac{\sum_{i=1}^{n}(y_i \times \mathbf{nonwhite}_i)}{\sum_{i=1}^{n} \mathbf{nonwhite}_i}$

$0.5(\frac{(\sum_{i=1}^{n}(y_i \times \mathbf{white}_i))+(\sum_{i=1}^{n}(y_i \times \mathbf{nonwhite}_i))}{n})$. Given that

$$p = p_{\text{white}}\frac{W}{N} + p_{\text{nonwhite}}\frac{N-W}{N}$$

where p is the proportion of the population with the condition, p_{white} is the proportion of white people in the population with the condition, W is the number of white people in the population, N is the total number of people in the population, and p_{nonwhite} is the proportion of non-White people in the population with the condition, and that

$$\frac{\sum_{i=1}^{n}(y_i \times \mathbf{white}_i)}{\sum_{i=1}^{n}\mathbf{white}_i}$$

should be a good estimator of the proportion of White people in the population with the condition and

$$\frac{\sum_{i=1}^{n}(y_i \times \mathbf{nonwhite}_i)}{\sum_{i=1}^{n}\mathbf{nonwhite}_i}$$

a good estimator of the proportion of non-White people in the population with the condition, the sample proportion calculated using a sample with the same proportions of White and non-White people as the population (such as would be obtained with SRS for example) should be a good estimator of the proportion population with the condition, and sample proportions calculated using samples with any other ethnic composition will be biased estimators, being more biased the further their composition deviates from the population composition. This case, where z is related to y, is called an *informative design.*

Now let us imagine that the researcher wants to perform a slightly more complicated analysis: they are interested in how sex is related to an individual's probability of having the health condition. They thus fit a (logistic) regression with sex as an explanatory variable. (Thus x does now enter the analysis).

Once again, there are various possibilities. It could be that both x and y are unrelated to z. That is, the proportion of White individuals who are male is equal to the proportion of non-White individuals who are male (which is likely to be true), and the proportion of White individuals with the health condition is the same as the proportion of non-White individuals with the health condition (which may or may not be true). In this case the intercept and the coefficient of x will both be unbiased estimators of the respective population parameters. This is because the expected proportion of males in the sample will be the same as the proportion of males in the population (because ethnicity is unrelated to sex) and it makes no difference what the ethnic composition of those males, and of the sampled females, is (because ethnicity is unrelated to the health condition: thus the same proportions of males and of females with the health condition are expected no matter what their ethnicity). In this case, again the design is non-informative.

Another possibility is that x is again unrelated to z - once again the proportion of White individuals who are male is the same as the proportion of non-White individuals who are male - but that z is related to y (perhaps White individuals are more likely to have the condition than non-White individuals, for example). In this case the intercept (which in this analysis estimates the proportion of men in the population who have the condition) will be biased, since the men in the sample have a different ethnic composition to the men in the population, and ethnicity is related to the probability of having the condition. The coefficient of x (which in this analysis estimates the log odds of having the condition for women versus men) may or may not be unbiased. If, despite the different prevalences of the condition among Whites and non-Whites, the log odds of having the condition for women versus men is the same for these ethnic groups, then the coefficient will be unbiased (because the ethnic composition of the sample cannot make a difference to the value of the estimate). If however there is an interaction between ethnicity and gender, so that for example women are more likely to have the condition than men and Whites are more likely to have the condition than non-Whites, but the probabilities of having the condition for White males and for White females are more similar than the probabilities of having the condition for non-White males and non-White females, then the coefficient will be biased. This is because the value of the coefficient will be affected by the sample ethnic composition: if non-Whites are oversampled then the coefficient will be closer to the value that would be obtained were the model fitted to the population of non-White individuals, and further from the value that would be obtained were the model fitted to the population of White individuals, than would be the value obtained were the model fitted to whole population. (Thus in this example, the coefficient will be biased upwards as an estimator for the population value). This case, where x is unrelated to z but z is related to y, is again an informative design (whether or not there is an interaction between x and z in the effect on y).

Let us now imagine cases where x is related to z. Since it is implausible to imagine that the proportion of White individuals who are male is not the same as the proportion of non-White individuals who are male, let us now suppose that x is income instead of sex. It is reasonable to suppose that non-White individuals might have lower incomes than White individuals (through discrimination, for example, or because some of them may be immigrants from countries whose professional qualifications are not recognised in the country they have emigrated to, obliging them to take lower paid less skilled jobs).

If x is related to z, but z is unrelated to y then both the intercept and the coefficient of x will again be unbiased estimators for the respective population quantities. The intercept is an estimate of the population proportion of individuals with $x = 0$ who have the condition. (Let us suppose that income has been centred around its mean). Since x is related to z (income is related to ethnicity), the proportion of individuals in the sample who have the mean income ($x = 0$) will be different to the proportion of individuals in the population who have this income; and the proportion of individuals in the sample with mean income who are White will differ between the sample and the population. However, within this group of sampled individuals with this income, the proportion with the condition should be the same (to within sampling variation) as the proportion of individuals with this income in the population with the condition, because ethnicity is unrelated to an individual's probability of having the condition. Similarly, although the distribution of income in the sample is different to that in the population (due to its different ethnic composition), given any particular income the proportion of individuals with that income in the sample who have the condition should be the same as the proportion of individuals with that income in the population who have the condition (to within sampling variation). Thus the coefficient of income - the increase in the log odds of having the condition for a 1 unit increase in income - should be an unbiased estimator of the population value. We again have an non-informative design.

If, on the other hand, x is related to z and z is related to y, then both the intercept and the coefficient of x will be biased estimators for the respective population quantitities. The intercept will be equal to the proportion of individuals in the sample with the mean income who have the health condition. Since the proportion of individuals in the sample with the mean income who are White is different to the population proportion of individuals with the mean income who are White, and since the probability of having the health condition depends on ethnicity, the proportion of individuals in the sample with mean income who have the health condition will be different to the proportion of individuals in the population with mean income who have the health condition, and thus the intercept will be a biased estimator of this population proportion. Similarly, at each value of income the ethnic composition in the sample will be different to the ethnic composition at that value of income in the population. The ethnic composition will also differ for different income values within the sample (and, in a different way, for the population), in such a way that (since y is related to ethnicity) the coefficient of income estimated from the sample will be a biased estimator of the equivalent population value. We again have an informative design.

Let us now imagine that we have a more complicated analysis, with a set of multiple explanatory variables X, and multiple variables on which the selection probabilities depend Z. We fit a regression model including all the variables in X as explanatory variables (and not including any of the variables in Z). Imagine that at least one of the variables in X is related to at least one of the variables in Z, and that this variable or variables in Z are related to the response y. Any variables related to the response but not included as explanatory variables in the regression (including those just mentioned in Z) will be correlated with the residuals. Thus the variable(s) in X which are related to variable(s) in Z that are related to Y will also be correlated with the residuals. This violates one of the assumptions of the regression model, and will result in biased regression coefficients for these variable(s) in X. (The estimate of the intercept will also be biased). Again, we have an informative design.

Thus we have an informative design whenever one or more of the variables in Z are related to y, and an non-informative design whenever all the variables in Z are unrelated to y. Whenever we have an informative design, the intercept (or sample mean, if we are not fitting a model with explanatory variables) will be a biased estimator for the equivalent population parameter. The coefficients of the explanatory variables in X may or may not be unbiased estimators. For any of the variables in X which are related to variable(s) in Z that are related to y, and any variables in X for which there is an interaction with Z in their effect on y, the coefficients will be biased; for any variables in X which do not meet either of these conditions the coefficients will be unbiased.

1.2 Stratified sampling and sampling with probability proportional to size

The examples we have considered so far have all used *stratified sampling.* In stratified sampling, individuals are grouped into *strata* according to their characteristics. An individual's probability of being selected into the sample then depends on which stratum they belong to. In our example, individuals were grouped into a White stratum and a non-White stratum, and each individual's probability of selection depended on which of these strata they belonged to. Stratified sampling need not be based on a single binary characteristic as in this case; it can be based on multiple variables which can have multiple categories, potentially leading to a large number of strata. Stratified sampling will lead to an informative design when (some or all of) the characteristics which are used to group individuals into strata are related to the outcome and these characteristics are not included in the fitted model.

Another common design is sampling with *probability proportional to size.* This can be a special case of stratified sampling, where an individual's probability of selection depends on which stratum they belong to, and these probabilities are related to the number of individuals in the stratum. This will occur whenever we wish to select an equal number of individuals from each stratum, since in this case the selection probability for individual i will be

$$\frac{m}{N_{h(i)}}$$

where m is the number we desire to sample from each stratum, N_h is the number of individuals in the population who belong to stratum h, and $h(i)$ is the stratum that individual i belongs to. Thus sampling with probability (inversely) proportional to stratum size is a fairly common design.

1.3 Strategies for dealing with informative designs

As we mentioned in Section 1.1, one strategy for dealing with informative designs, so that unbiased estimates of population parameters can be obtained, is to use weights in the analysis. This is not, however, the only strategy.

Another possibility is to include as explanatory variables the variables in Z (or at least those which are related to Y). Doing this makes the design non-informative, so that the parameters estimated in this model are unbiased estimates of the corresponding population parameters. This will produce estimates of the coefficients of the variables in X controlling for the variables in Z. Often this will be desirable, but in other cases the interest will lie in the values of the coefficients unadjusted for the variables in Z.

An example of such a situation might be if we were interested in differences in annual income between males and females in employment, and had oversampled part-time workers. Clearly whether an individual works part-time or full-time will affect their annual income, so we have an informative design. We might also expect that working part-time or full-time is related to gender (perhaps women with children are more likely to work part-time than men with children). If we run an (unweighted) analysis not including part-time versus full-time employment status, we will get biased estimates of the population difference between men and women in annual income. (Our estimates will be biased upwards compared to the population value because the oversampling of part-time workers will increase the proportion of women in the sample who work part-time more than it increases the proportion of men in the sample who work part-time). We could run a regression analysis including a dummy variable for working part-time as well as gender as an explanatory variable. The coefficient of the gender variable would be an unbiased estimator of the population difference in annual income between men and women controlling for working part-time. In other words, it would tell us the magnitude of the difference in annual income between men and women that cannot be explained by the differences between the genders in the proportion who work part-time, and so must be due to other factors. Sometimes this might be what we are interested in. For example, if we were interested in differences in income between men and women due to discrimination we would want to control for factors arising from individuals' own decisions that might affect their income such as the decision to work part-time (we would of course also want to control for many other things, such as the sector of employment, the individual's qualifications, and their length in the position, so that the present analysis would be just a starting point). However, in other cases we might be interested in the total effect of gender on annual income: we might want to know not just about the effect of discrimination but instead might want to know what difference in income a woman might expect compared to a man due to the combined effect of discrimination, all her choices through life that are influenced by her gender, and anything else which is different for her because she is female.

If we include the variables in Z as explanatory variables, it is still possible to get estimates for the coefficients of the variables in X not controlling for the variables in Z. We would need to use our estimated coefficients for the variables in Z and the variables in X, and put these together with information on what proportion of people in the population have each combination of values of the variables in Z. It is not too difficult to perform this calculation when the variables in X and Z are all categorical, but becomes more difficult when they are continuous; and more cumbersome when there are many variables in Z and/or these variables can take on many different values (since then there are many possible combinations of values of the variables in Z). For this reason, it is often easier to perform a weighted analysis (which will give directly coefficients for the variables in X not controlling for the variables in Z) rather than to include the variables in Z as explanatory variables and then perform this calculation.

When taking this approach, if there are any interactions between variables in X and variables in Z, then it is important that they be included in the model. When there are many variables in X and/or Z, there may be many such interactions, which may make the model undesirably complicated in terms of estimation or interpretation: the researcher may only be interested in the main effect of the variables in X (controlling for the variables in Z) and not in these interactions. Interactions will also make the calculation of the coefficients of the variables in X not controlling for the variables in Z more complicated.

A modified form of this approach, which simplifies both the model and the calculation of the coefficients of the variables in X not controlling for the variables in Z, is, instead of including all the variables in Z that are related to Y as explanatory variables, to include a single weights variable as an explanatory variable. Since, although unbiased, the coefficients of the variables in X in this model will be the coefficients controlling for the weights, they are not very easily interpretable, and thus it is usually desirable to use the results of the model to calculate the coefficients of the variables in X not controlling for the weights. When taking this approach, there is only one extra variable (in addition to the variables in X) to be included in the model, possibly along with the interactions between this single variable and some or all of the variables in X, and in our calculation we do not have to work out values for combinations of values of many variables since again we just have one variable. Thus this method may not be too complicated or time consuming and researchers may well opt to use it. However, it is still generally simpler and easier to use weights.

1.4 Multistage designs

In multistage designs groups of individuals (called *clusters*) are selected in the first stage, and then within those groups, subgroups are selected in the second stage, and so on until the individuals are selected from within the sampled smallest clusters. Cluster sampling is thus a *multistage design*. For simplicity, we will mostly consider a two stage design here, in which clusters are sampled first, and then within the selected clusters, individuals are sampled.

The clusters may be already clearly defined in the population as level 2 units in a multilevel data structure (e.g. schools) or they may be groups of individuals defined for the purposes of the sampling procedure. In the latter case, if the responses of individuals from the same cluster are correlated, then there is still a multilevel data structure with clusters at level 2 and individuals at level 1. This might happen for example if the clusters were geographical areas with boundaries defined for the purpose of the sampling procedure.

1.5 Multistage designs and weighting

With a 2 stage sampling design, there is the possibility both for the sample to differ in composition from the population in terms of the level 2 units included, and for the level 1 units included in the sample from a given level 2 unit to differ in composition from the totality of level 1 units in that level 2 unit. In other words, unequal selection probabilities may be used at either or both levels, and there is thus the potential for the residual at each level in a multilevel model to be related to the selection probabilites. So we could have a design which was informative at both levels, and accordingly there may be two sets of weights.

For example, imagine that our researcher is studying a country which is divided into districts and each district is classified as Rural or Urban. Imagine that whether an individual lives in an Urban or Rural district affects their health outcome. The researcher uses multistage sampling to reduce costs, and oversamples Urban districts since there are not very many of these in the country. They also oversample non-White individuals for the same reasons as before. (Let us suppose for the moment that the ethnic composition is the same across all districts). Now the analysis must incorporate one set of weights to reflect the fact that the proportion of sampled districts that are Urban is greater than the proportion of Urban districts in the population, and another set to reflect the fact that within each district the proportion of sampled individuals who are non-White is greater than the proportion of individuals who are non-White in the population.

In this scenario, the value of the first weight (to adjust for different Urban-Rural composition of the sample districts compared to the population districts) depends only on whether the district is Urban or Rural.

The value of the second weight (to adjust for the different ethnic composition of the sample compared to the population) depends only on the ethnicity of the individual. However, it is common for the design to be more complicated than this. Imagine that instead of all districts having the same ethnic composition, Urban districts have 70% White and 30% non-White people, while Rural districts have 95% White and 5% non-White people. It is still desired to keep the numbers of White and of non-White individuals sampled constant across the selected districts. In this case, although the weights adjusting for the different Urban-Rural composition of the sample would still only depend on whether the district is Urban or Rural, the weights adjusting for the different ethnic composition of the sample compared to the population would now depend on both whether the district was Urban or Rural and the individual's ethnicity. If we complicate the situation still further by imagining that instead of all Urban districts having exactly 30% non-White people and all Rural districts having exactly 5% non-White people, the Urban districts have non-White proportions distributed around 0.3 and the Rural districts have Black proportions distributed around 0.05, then instead of four different individual level weights (White person in Urban district, White person in Rural district, non-White person in Urban district and non-White person in Rural district) each district will have its own pair of weights, consisting of one weight for White and one for non-White individuals.

1.6 Probability dependent on continuous variables

In the scenario just described, stratified sampling is once again being used (this time first at level 2 and then at level 1). We saw in section 1.2 when equal numbers of units are picked from each stratum, this is equivalent to selecting with probability inversely proportional to the size of the stratum. Thus in this modified version of our example, we will be sampling with probability proportional to size if we sample the same number of Urban and Rural districts and/or if within each district we sample the same number of White and of non-White individuals.

In the multistage case, an informative design involving selection probabilities inversely proportional to size can also arise when we do not use stratified sampling, if the clusters are selected with probability proportional to cluster size and cluster size is related to the outcome variable. An example is if cluster size is a proxy for population density and y is a health outcome; there might be poorer (or better) availability of health services in high-density areas, or the greater population density could make it easier for disease to spread (for example through poor sanitation). Another case would be with a three stage design, sampling schools, then classes within selected schools, and then students within selected classes, with classes selected with probability related to class size, and the outcome being attainment (which is known to be related to class size). Selecting with probability proportional to cluster size is common because, as Rabe-Hesketh & Skrondal

(2006) point out, if individuals are then selected in such a way that all individuals in the same cluster have an equal probability of selection, a so-called *self-weighting* design results, in which the probabilities at each stage combine in such a way as to result in equal overall selection probabilities (an individual's overall selection probability being their probability of selection not conditional on the cluster to which they belong being selected, or in other words their probability of selection before the clusters are selected in the first stage). Note that despite the name, which refers to single level models, in the multilevel case such designs are not self-weighting but require weights to be used, as shown by Pfeffermann et al. (1998).

Another continuous variable which may be used in generating selection probabilities is the cost involved in collecting data from a cluster (so as to try to minimise the amount that must be spent on data collection). This might be related to the distance from the centre where the researchers are based, which in turn would be related to the geographical location of the clusters, which could be related to the outcome; it might also or alternatively be related to the distance between individuals within the cluster which would be related to population density, and thus again perhaps to the outcome.

1.7 Calculating weights

For single level models, there is only one set of weights which we will call w_i. For 2 level models two sets of weights are necessary. One set is the level 2 weights w_j which adjust for the unequal probability of selection of the cluster. The other may be conditional level 1 weights $w_{i|j}$ which adjust for the unequal probability of selection of the individual from within the cluster; or may be the unconditional level 1 weights w_{ij} which adjust for the individual's unequal probability of selection (before selection of the clusters) from among all individuals in the population. w_{ij} thus adjust for both an individual's unequal probability of selection from among all individuals within their cluster, and for the unequal probability of selection of their cluster from among all clusters in the population. The conditional and unconditional level 1 weights are related by

$$w_{ij} = w_{i|j} w_j$$

Thus whichever kind of level 1 weight is provided in the dataset, the other can be calculated (provided that the level 2 weights are also provided). Whether the conditional or unconditional level 1 weights should be specified in the analysis differs from software package to software package. (MLwiN requires conditional level 1 weights).

There are two possible methods of calculating the weights, which will give very similar values. One method uses the selection probabilities while the other uses the number of units sampled.

Method 1

Method 1 is based directly on the selection probabilities. Analagously to the weights, with a single level model we have just one selection probability, p_i, which is an individual's probability of selection from among all the individuals in the population; while with a 2 level model we have p_j, the probability of selection of the cluster that the individual belongs to from among all clusters in the population, $p_{i|j}$, the individual's probability of selection from among all individuals in the same cluster (given that the individual's cluster has been selected), and $p_{ij} = p_{i|j}p_j$, the individual's probability of selection from among all individuals in the population (i.e. the individual's probability of selection before the clusters have been selected).

Single level case

$$w_i = \frac{1}{p_i}$$

Multilevel case

$$\begin{aligned} w_j &= \frac{1}{p_j} \\ w_{i|j} &= \frac{1}{p_{i|j}} \\ w_{ij} &= w_{i|j}w_j = \frac{1}{p_{i|j}p_j} = \frac{1}{p_{ij}} \end{aligned}$$

Method 2

Method 2 can only be used with stratified sampling, when all the variables used to generate the selection probabilities are categorical. It may be preferred by some researchers to Method 1 when the sampling procedure is such that the actual numbers of units selected (from each stratum) can vary around the expected values predicted by the probabilities.

For the purposes of these definitions, let 'characteristics' refer to the characteristics that we are basing unequal selection probabilities on.

Single level case Let $N_{h(i)}$ be the number of units in the population which have the same characteristics as unit i, and $n_{h(i)}$ be the number of units in our sample which have the same characteristics as unit i, then

$$w_i = \frac{N_{h(i)}}{n_{h(i)}}$$

Multilevel case Let $M_{k(j)}$ be the number of level 2 units in the population with the same characteristics as level 2 unit j and let $m_{k(j)}$ be the number of level 2 units in the sample with the same characteristics as level 2 unit j. Let $N_{h(i,j)}$ be the number of units in the population in level 2 unit j which have the same level 1 characteristics as unit i in level 2 unit j, and let $n_{h(i,j)}$ be the number of units in the sample in level 2 unit j which have the same level 1 characteristics as unit i in level 2 unit j. Then

$$\begin{aligned} w_j &= \frac{M_{k(j)}}{m_{k(j)}} \\ w_{i|j} &= \frac{N_{h(i,j)}}{n_{h(i,j)}} \\ w_{ij} &= w_{i|j} w_j = \frac{N_{h(i,j)}}{n_{h(i,j)}} \frac{M_{k(j)}}{m_{k(j)}} \end{aligned}$$

Standardisation of weights

In the multilevel case, frequently instead of $w_{i|j}$ as given here a standardised version $w^*_{i|j}$ is used. We define

$$w^*_{i|j} = w_{i|j} \frac{n_j}{\sum_{i=1}^{n_j} w_{i|j}}$$

where n_j is the number of level 1 units in level 2 unit j in the sample. $w^*_{i|j}$ are therefore $w_{i|j}$ adjusted so that for each j

$$\sum_{i=1}^{n_j} w^*_{i|j} = n_j$$

Divide both sides by n_j and it can be seen that an equivalent way of saying this is that $w^*_{i|j}$ are adjusted so that the mean weight for each level 2 unit j is 1.

This is the standardisation method that we consider in this report (and it is the method that is used by MLwiN if standardisation of the weights is requested). However, it is not the only logical way to standardise the weights, and there has been much discussion of this topic in the literature. See for example Pfeffermann et al. (1998); Carle (2009)

2 Weights in MLwiN

2.1 Estimation of multilevel models with weights

2.1.1 PWIGLS

Pfeffermann et al. (1998) propose an estimation procedure for multilevel models with weights which they call Probability-Weighted Iterative Generalised Least Squares (PWIGLS) estimation[3]. This procedure adapts the Iterative Generalised Least Squares (IGLS) estimation procedure which is MLwiN's default method of estimation to incorporate weights. The authors show mathematically that the method produces consistent estimators under certain assumptions, and note that the estimation of the fixed part parameters is unbiased. Estimation of the random part parameters may however suffer from bias when the number of level 1 units within each level 2 unit is small. To reduce this bias, they suggest using the standardised weights $w^*_{i|j}$ described above[4] at level 1, instead of the raw $w_{i|j}$. They carry out a simulation study which confirms the unbiased estimation of the fixed part parameters.

The simulation study also compares the estimates produced when using the standardised level 1 weights $w^*_{i|j}$ with those produced when using the raw weights $w_{i|j}$. They find that using the standardised weights reduces the bias in the estimator of the level 2 variance so that it becomes negligible when the sampling procedure is such that the level 1 units within each sampled level 2 unit are selected with equal probability (i.e., when after selection of the level 2 units simple random sampling is used to select level 1 units from these). When the sampling procedure is such that unequal selection probabilities are used to sample level 1 units from within the selected level 2 units, as well as to select the level 2 units themselves, they find that the standardisation overcorrects for the bias in the estimator of the level 2 variance, so that it becomes biased upwards instead of downwards. The resulting bias is still negligible for samples where the number of level 1 units within each level 2 unit is large, but is more substantial where the number of level 1 units within each level 2 unit is smaller. However, the bias is still less than when using the raw weights $w_{i|j}$ at level 1.

2.1.2 Estimation procedure used in MLwiN

Pfeffermann et al. (1998) note that implementation of PWIGLS in a software package which already offers

[3] We will not present the details of this estimation method here; the reader is referred to the paper itself, which describes it fully.

[4] They actually propose two methods of standardisation. Their method 2 uses the $w^*_{i|j}$ as described above. We do not discuss here their method 1, for which they use the notation $w^*_{i|j}$ - note this difference from the present report in what is referred to by $w^*_{i|j}$

IGLS estimation (such as MLwiN) would involve considerable reprogramming of the IGLS estimation engine if using the algorithm as they initially present it. They therefore propose a different but exactly equivalent algorithm for PWIGLS which involves two main steps. The first step, which they call Step A, involves multiplying the explanatory variables associated with the random effects at each level of the data by a function of the weights at that level, and thus does not involve any alterations to the IGLS algorithm. The second step, Step B, involves changing the way that one of the matrices involved in producing estimates for the random parameters is calculated, and thus does involve some reprogramming of the IGLS estimation procedure.

Since it is desirable to be able to use weights in multilevel models estimated via IGLS without reprogramming the estimation engine, the authors also investigated in their simulation study whether unbiased parameter estimates could be achieved by performing Step A without Step B. They found that using only Step A gave almost identical results to the full PWIGLS algorithm, except that the estimate of the standard error of the level 1 variance was found to be biased upwards, and the point estimate of the level 1 variance was biased downwards in sampling schemes where the size of the level 2 unit was related to its probability of selection and to the number of level 1 units sampled from it.

Due to this near identicality in most cases of the results produced by the full PWIGLS algorithm and Step A only, and the resources that would be involved in reprogramming MLwiN to implement PWIGLS, MLwiN's weighting facility performs Step A only. Because this does not involve any changes to the IGLS algorithm itself, it is actually possible to program Step A in an MLwiN macro, and Appendix presents a replication of the weights facility in MLwiN commands which may be of interest as an alternative to reading about Step A in Pfeffermann et al. (1998) to the reader who wants to know the details of how weights are implemented in MLwiN.

2.2 Applying weights in MLwiN

In the multilevel case, MLwiN requires the user to input w_j as the 'level 2 weights' and $w_{i|j}$ as the 'level 1 weights'. Note that w_{ij} cannot be input directly; if the dataset supplies w_{ij} the user must calculate $w_{i|j}$ and w_j and give those to MLwiN.

2.2.1 Standardisation

The weights do not need to be already standardised, as the user can as part of the weights facility instruct MLwiN to standardise the weights before using them (the **Use standardised weights** option in the GUI[5] or `WEIGhts 2` and `WEIGhts` commands). In this case, MLwiN will calculate $w^*_{i|j}$ as described above and use this as the level 1 weight. However, there is no problem with providing pre-standardised weights if the user so desires. MLwiN will still consider these to be 'raw weights' and if using the GUI, the drop-down boxes where the weights should be specified will still be headed **Raw weights in**; but this is just a question of labelling and does not affect what MLwiN does with the weights.

When providing pre-standardised weights, the user can choose to **Use raw weights** which means MLwiN will not perform any further standardisation before applying the weights to the model. Thus the user can use other methods of standardisation than that which MLwiN would use, by performing their desired standardisation before using the weights facility and selecting **Use raw weights** within the weights facility. If the user instead chooses **Use standardised weights**, then MLwiN's standardisation procedure will be carried out on the weights. If the standardisation already performed by the user is the same as MLwiN's standardisation, this should be equivalent to selecting **Use raw weights** since standardising the already standardised weights will leave them unchanged. If the user has used a different method of standardisation then choosing **Use standardised weights** can be expected to produce different results to **Use raw weights** since the standardisation will in this case probably change the values of the weights, and thus this is probably not the recommended option.

Both the **Use raw weights** and the **Use standardised weights** options are of course still available if the user has not pre-standardised their weights. Though it is thus technically possible to use unstandardised weights by selecting **Use raw weights**, as discussed above we do not recommend this. The recommended options are thus **Use standardised weights** where the user has not previously standardised the weights themself, and **Use raw weights** where the user wants to use some form of standardisation different to that performed by MLwiN and has already carried out this standardisation on the weights.

2.2.2 Sandwich estimators

As will be seen from the results of the simulation study in Section 3.3, we recommend using both fixed and random part sandwich estimators for standard errors when using weights. Since v2.17, fixed and random

[5] Graphical User Interface, i.e. the point-and-click menu and button way of working with MLwiN as opposed to the use of commands

part sandwich estimators have automatically been applied when weights are specified via the GUI. However, sandwich estimators are not automatically used when weights are specified using commands and thus the user must include extra commands to explicitly specify the use of sandwich estimators if these are desired.

2.2.3 Specifying weights using the GUI

To run a model using weights via the Graphical User Interface (i.e. using the point and click method as opposed to using the **Command Interface** window or a macro), first set up the model you wish to fit in the **Equations** window, then

- Select **Weights** from the **Model** menu
- In the **Level 2** drop-down box under **Raw weights in**, select the column into which you have put w_j
- In the **Level 1** drop-down box under **Raw weights in**, select the column into which you have put $w_{i|j}$
- You can change the columns to which MLwiN will output standardised weights using the drop-down boxes underneath **Standardised weight to**, but usually it's fine to leave these as they are
- Select **Use raw weights** or **Use standardised weights** using the radio buttons under **Weighting mode** (or leave **Off** selected if you don't want to use weights)
- Click the **Done** button
- Press the **Start** button

2.2.4 Specifying weights using commands

To run a model with weights using commands, first set up the model (using commands or the **Equations** window) and then use the following commands, substituting for 'wtsL2' the name of the column where you have put w_j and for 'wtsL1' the name of the column where you have put $w_{i|j}$[6]:

► `WEIGhts 2 1 'wtsL2'` *The level 2 weights are in column 'wtsL2'*

[6] Note that (as with all MLwiN commands) only the first 4 letters of the command are required; however we provide the full command name here as this is often easier to remember

- ▶ `WEIGhts 1 1 'wtsL1'` *The level 1 weights are in column 'wtsL1'*
- ▶ `WEIGhts 2 2 c111` *Put the standardised level 2 weights in c111*
- ▶ `WEIGhts 1 2 c112` *Put the standardised level 1 weights in c112*
- ▶ `WEIGhts 2` *Use the standardised weights*
- ▶ `WEIGhts` *Create the standardised weights*
- ▶ `FSDE 2` *Specify sandwich estimators for fixed part standard errors*
- ▶ `RSDE 2` *Specify sandwich estimators for random part standard errors*

Then run the model as normal using a command or by pressing the **Start** button

To use raw weights instead of standardised weights, change the fifth command from `WEIGhts 2` to `WEIGhts 1` (the sixth command, `WEIGhts`, is not needed in this case). To turn weights off again, use the command `WEIGhts 0`.

3 Testing MLwiN's weighting facility

3.1 Background to the investigation

We received some reports from users which suggested that the weights facility in MLwiN might not be working as intended. We therefore decided to carry out a simulation study to check its performance.

The reports seemed to indicate that if there was a problem it was more likely to apply to models with discrete responses than those with continuous responses. Despite this, we decided only to test continuous response models in our investigation. This was because even with non-informative designs, the quasi-likelihood methods used in MLwiN are known to give biased estimates, particularly with small cluster sizes (see Rodríguez & Goldman (2001)). We therefore do not recommend the use of these methods for discrete response models (apart from in initial model exploration), but rather urge users to use Markov Chain Monte Carlo (MCMC) estimation for final results for publication. However, at present the weights facility is not available when using MCMC estimation (since the method of implementing weights for MCMC would be radically different from the method for likelihood or quasi-likelihood estimation and require further methodological work and programming which has not yet been undertaken). We thus recommend that users with discrete response data and an informative design either find some way to make their design non-informative (such as including relevant covariates in the model fitted), or use a different software package to carry out their analysis. In any case, the bias presents an obstacle to testing the weights facility, since it is more pronounced for smaller cluster sizes: thus even if the weights facility were working perfectly the results from the model fitted to the sample would be expected to be different to the results from a model fitted to the population (and both would differ from the values used to generate the data).

We have also omitted to test every possible model that we would recommend fitting with weights in MLwiN, since there is an infinite number of such models. We have confined the investigation to continuous response random intercept models and models with a single random slope, for one particular set of population parameter values, and with a limited selection of combinations of numbers of clusters in the population and cluster sizes. We do, however, provide the macros used in the Appendix to this report, and users can adapt these to test the functioning of MLwiN's weights facility for any model that they desire to fit. We would welcome any results that users might wish to share of such an investigation (even into the discrete response models that we do not recommend fitting) and would be glad to be able to display them on our website[7].

[7] Users can send results to Rebecca Pillinger at `r.j.pillinger@bristol.ac.uk`

3.2 Simulation design

In the example we described in Section 1 (in all its various versions), the selection probabilities depended on a covariate or covariates which featured in the data generating model but not in the model that we fit to the data. (For example, Black at the individual level and Urban at the cluster level). In the case of real data, it makes most sense to think of the situation like this. However, we could equivalently think of the data generating model (for the simplest, original version of our example) as being

$$y_i = \beta_0^{(\mathrm{DG2})} + e_i^{(\mathrm{DG2})}$$

instead of

$$y_i = \beta_0^{(\mathrm{DG})} + \beta_1^{(\mathrm{DG})}\mathbf{Black}_i + e_i^{(\mathrm{DG})}$$

where **Black** is now an unmeasured characteristic and therefore part of the error term $e_i^{(\mathrm{DG2})}$. If we use this conceptualisation in our simulation, we can generate data with an informative design without needing to generate the characteristics which enter the data generating model and determine the selection probabilities: we simply use the residual to determine the selection probabilities[8]. We can do this because when we simulate data (unlike, of course, when we have real data) we have access to the value of the residual term. Our data generating model will then be identical in appearance to the model that we fit, since both will have just an intercept and a random effect. However, due to the informative nature of the design, the parameter estimates from the model we fit will not match the data generating values (unless we use weights).

We follow this approach in the simulation study, partly for convenience but mostly because we want our study to be comparable to that of Rabe-Hesketh & Skrondal (2006), who use a design like this. Their and our design is of course a multilevel version. We create two explanatory variables, x_1 and x_2, which are included in both the data generating model and the model we fit. x_1 is a continuous, Normally distributed variable, made to vary only within (and not between) level 2 units by subtracting the average for the level 2 unit from the value for each case. x_2 is a level 2 binary variable with mean 0.5. The data generating equation for the random intercept model we consider is

$$y_{ij} = 1 + x_{1ij} + x_{2j} + u_j + e_{ij}$$

where the variance σ_u^2 of the (Normally distributed) level 2 random effects u_j is 0.25 and the variance σ_e^2 of the (Normally distributed) level 1 random effects e_{ij} is 1. The selection probability for each level 2 unit is

[8]This is not quite the same as generating characteristics which determine the selection probabilities and enter the data generating model, since using the residual to determine the selection probabilities means that there are no unmeasured factors which affect the value of the response but not the selection probabilities; however this is not an important difference

0.25 if the random effect for that level 2 unit is more than 1 standard deviation above or below the mean, and 0.75 otherwise. The individual selection probabilities (given that the level 2 unit the individual belongs to has been selected) are 0.25 if the random effect for that individual is greater than 0 and 0.75 if the random effect is less than 0. The weights are calculated using the actual numbers selected, i.e. using Method 2 as described above in Section 1.7.

The data generating equation for the random slopes model we consider is

$$
\begin{aligned}
y_{ij} &= 1 + x_{1ij} + x_{2j} + u_{0j} + u_{1j}x_{1ij} + e_{0ij} \\
\begin{bmatrix} u_{0j} \\ u_{1j} \end{bmatrix} &\sim \mathrm{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \frac{1}{4} & \\ \frac{\sqrt{3}}{8} & \frac{3}{4} \end{bmatrix} \right) \\
e_{0ij} &\sim \mathrm{N}(0, 1)
\end{aligned}
$$

(where the covariance between the random intercepts and random slopes is chosen to give a correlation of 0.5). x_1 and x_2 are as for the random intercept model. The selection probability for each level 2 unit is related to the value of both the intercept and slope residuals: it is 0.225 when the intercept residual is more than 1 standard deviation above or below the mean and the slope residual is greater than 1 in absolute value; 0.425 when the intercept residual is within 1 standard deviation of the mean and the slope residual is greater than 1 in absolute value; 0.525 when the intercept residual is more than 1 standard deviation above or below the mean and the slope residual is less than or equal to 1 in absolute value; and 0.725 when the intercept residual is within 1 standard deviation of the mean and the slope residual is less than or equal to 1 in absolute value. The conditional level 1 selection probabilities are the same as for the random intercept model. Once again the weights are calculated using the actual numbers selected.

For each parameter, we judge the adequacy of the estimation both by comparison of the average estimate across replications with the true (data generating) parameter value (to assess bias), and by calculation of the percentage of replications for which the true parameter value lies inside 95% confidence intervals constructed from the parameter estimate and estimated standard error for that replication (to assess coverage).

We generate a new population for each replication. This is because (assuming the weights are working correctly) the parameter estimates we get from fitting the model to the sample should be close to the parameter estimates we would get if we were to fit the (unweighted) model to the whole population. If we generate a large number of populations, the average value of the parameter estimates from a model fitted to each population in turn should be close to the data generating values; but for any one population that we generate, the value of the parameter estimates from a model fitted to that population may be quite far from the data generating values by chance. Thus were we to generate a single population that was the same across replications, with each replication simply taking a different sample from that population, the average parameter estimates from fitting the weighted model to the samples might be reasonably different to the data generating values.

In order to decide how many replications to perform for each investigation, we tried performing the simulation for a population with 500 level 2 units each of size 50 for differing numbers of replications. For each number of replications, we ran the simulation for that number of replications 5 times, and compared the results across the 5 repeats to examine their consistency. We wanted to find the number of replications which would give the same results to 1 decimal place across all 5 repeats[9]. Although fewer than 500 replications were needed to produce this consistency in the average parameter estimate, 60,000 were needed for the coverage. We therefore ran the simulations for all investigations for 60,000 replications.

3.3 Results: random intercept model

First, we generated data from the random intercept model, creating a population of 500 level 2 units each containing 50 level 1 units, and compared the estimates from weighted models to the true (data generating) parameter values and to the estimates from an unweighted model. For the weighted model, we explored different options for the standard errors: no sandwich estimators, sandwich estimators for the random part only, sandwich estimators for the fixed part only, and sandwich estimators for the fixed and random parts.

Mean estimates

	True value	Unweighted	Weighted, with sandwich estimators:			
			None	Fixed only	Random only	Fixed and random
β_0	1	0.6014	0.9982	0.9982	0.9982	0.9982
β_1	1	1.0000	1.0001	1.0001	1.0001	1.0001
β_2	1	0.9996	0.9993	0.9993	0.9993	0.9993
σ_u^2	0.25	0.1464	0.2404	0.2404	0.2404	0.2404
σ_e^2	1	0.8407	1.0054	1.0054	1.0054	1.0054

[9]We did this for the weighted random intercept model with sandwich errors in the fixed and random part.

Coverage

	Unweighted	Weighted, with sandwich estimators:			
		None	Fixed only	Random only	Fixed and random
β_0	0.0000	0.8324	0.9418	0.8324	0.9418
β_1	0.9500	0.8436	0.9457	0.8436	0.9457
β_2	0.9465	0.8345	0.9447	0.8345	0.9447
σ_u^2	0.0004	0.8427	0.8427	0.9591	0.9591
σ_e^2	0.0000	0.8465	0.8465	0.9990	0.9990

It can be seen that the weighted model produces mean estimates that are much better than those of the unweighted model, coming close to the true values. As might be expected, the mean estimates are unaffected by what options are chosen in terms of sandwich estimators for the standard errors.

Again as expected, the coverage for the unweighted model is very poor (with the exception of β_2). For the weighted model, coverage for the fixed part parameters is close to the desired 95% when sandwich estimators are used for the fixed part, and coverage for the level 2 variance is close to the desired 95% when sandwich estimators are used for the random part. Wherever sandwich estimators are not used, the coverage is less than it should be, i.e. the true value falls inside the 95% confidence interval for fewer than 95% of the replications, implying that the estimated standard errors are too small. The coverage for the level 1 variance is somewhat high when sandwich estimators are used for the random part, at over 99%, but the coverage is again too low when random part sandwich estimators are not used, and it is usually better to err on the side of conservativeness in confidence intervals. Thus these results indicate that both fixed and random part sandwich estimators should always be used with weighted models in MLwiN.

3.4 Results: varying cluster sizes and numbers

We next investigated the behaviour of the same random intercept model as the structure of the population is altered in terms of the number of level 2 units (which we will refer to as M) and the number of level 1 units within each level 2 unit (which we will refer to as N). The sampling probabilities used are the same as those in the previous section; so the actual numbers of units sampled at each level will change as the structure of the population changes. Given the findings of Section 3.3, for each combination of number and size of clusters we fitted only a weighted model with sandwich estimators for both the fixed and random part standard errors.

Mean estimates

	True values	M	100	500			1000		
		N	100	25	50	100	25	50	100
β_0	1		0.9968	0.9725	0.9982	0.9999	0.9725	0.9978	0.9999
β_1	1		0.9999	1.0001	1.0001	1.0000	1.0001	1.0000	1.0001
β_2	1		1.0012	0.9999	0.9993	0.9999	1.0000	1.0003	1.0000
σ_u^2	0.25		0.2274	0.2527	0.2404	0.2427	0.2552	0.2424	0.2452
σ_e^2	1		1.0036	0.9924	1.0054	1.0035	0.9926	1.0056	0.9815

Coverage

	M	100	500			1000		
	N	100	25	50	100	25	50	100
β_0		0.9215	0.9235	0.9418	0.9414	0.9076	0.9460	0.9037
β_1		0.9385	0.9462	0.9457	0.9459	0.9483	0.9477	0.9476
β_2		0.9215	0.9439	0.9447	0.9441	0.9462	0.9483	0.9473
σ_u^2		0.8885	0.9716	0.9591	0.9701	0.9813	0.9684	0.9800
σ_e^2		0.9995	0.9837	0.9990	1.0000	0.9849	0.9989	0.9547

For all these cluster numbers and sizes, the mean estimates of the fixed part slope coefficients are extremely close to their true values. The intercept is slightly underestimated for populations with clusters of size 25 (although its estimate is still close to the true value), but its estimate is also extremely close to the true value for larger cluster sizes. The estimate of the level 2 variance is close to its true value.

Coverage is also reasonable. The coverage for the fixed part slope coefficients is very close to the desired 95% except for the population of 100 clusters of size 100, where it is still reasonably close. The coverage for the intercept is somewhat low. The coverage for the level 2 variance is also low for the population of 100 clusters of size 100, but a bit high for populations with more clusters. The coverage for the level 1 variance is too high for all numbers and sizes of clusters, but again this should not be such a big problem as conservative confidence intervals that are too narrow.

3.5 Results: random slope model

Finally, we investigated the behaviour of the weights with the random slope model, looking again at a range of different population cluster sizes and population numbers of clusters. We again used sandwich estimators for both the fixed and random part standard errors.

Mean estimates

	True values	M	100	500			1000		
		N	100	25	50	100	25	50	100
β_0	1		0.9906	0.9449	0.9812	0.9913	0.9447	0.9809	0.9904
β_1	1		0.9996	0.9922	0.9968	0.9994	0.9923	0.9972	0.9990
β_2	1		0.9997	0.9996	0.9996	0.9999	1.0000	0.9999	1.0002
$\sigma_u^2 0$	0.25		0.2358	0.2909	0.2608	0.2521	0.2935	0.2630	0.2533
$\sigma_u 01$	0.22		0.1905	0.1751	0.1953	0.2064	0.1761	0.1964	0.2074
$\sigma_u^2 1$	0.75		0.7198	0.7873	0.7645	0.7541	0.7897	0.7680	0.7561
σ_e^2	1		0.9956	0.9548	0.9865	0.9878	0.9546	0.9802	0.9957

Coverage

	M	100	500			1000		
	N	100	25	50	100	25	50	100
β_0		0.9204	0.8536	0.9287	0.9251	0.7701	0.9011	0.9397
β_1		0.9311	0.9441	0.9470	0.9463	0.9448	0.9476	0.9475
β_2		0.9225	0.9435	0.9454	0.9468	0.9465	0.9474	0.9472
$\sigma_u^2 0$		0.8764	0.9356	0.9741	0.9732	0.8383	0.9692	0.9831
$\sigma_u 01$		0.8639	0.8647	0.9305	0.9626	0.7969	0.8447	0.9644
$\sigma_u^2 1$		0.9126	0.9885	0.9880	0.9877	0.9873	0.9935	0.9928
σ_e^2		0.9945	0.7960	0.9750	0.9815	0.6247	0.9479	0.9980

Again, while the estimates of the fixed part slope coefficients are close to their true values, the intercept is somewhat underestimated for smaller population cluster sizes. The level 2 intercept and slope variances are slightly underestimated for populations with 100 clusters of size 100, but overestimated for populations with more clusters, although this overestimation becomes negligable as the cluster size increases. The covariance between the intercepts and slopes is underestimated for all population cluster sizes and numbers, although the extent of this underestimation decreases as the population cluster size increases. The level 1 variance is also underestimated for populations with smaller clusters.

As for the random intercept model, the coverage for the fixed part slope coefficients is very close to 95% except for the population of 100 clusters of size 100, where it is still fairly close. The coverage for the intercept is again too low, perhaps problematically so for the populations with smaller clusters. Coverage is also very low in these populations for the level 2 intercept variance, the covariance between the intercepts and slopes, and the level 1 variance, and again increases with cluster size, becoming somewhat too high for larger cluster sizes. Coverage for the variance of the slopes is too low for the population with 100 clusters of size 100, but too high for the populations with more clusters.

3.6 Conclusions

It appears that, despite reports to the contrary, MLwiN's weights facility is working as intended, at least for the limited set of models and data structures investigated here. The investigation has also confirmed that sandwich estimators should be used for both the fixed and random part standard errors.

The results do suggest that coverage of certain parameters might be poor for datasets sampled from populations with few clusters (less than 500) and/or small clusters (less than 25 for a random intercept model or 50 for a random slope model). It is advised that researchers bear this in mind before embarking on a weighted analysis, and where their sample is drawn from a population with fewer or smaller clusters, that they only proceed if their parameters of interest are among those for which the coverage is high.

Appendix A: Manually replicating MLwiN's weighting

Two worksheets accompany this document. **weights-demonstration-start.wsz** contains the data needed to follow the instructions below, and **weights-demonstration-completed.wsz** is the same worksheet saved after all the instructions have been carried out (so you can check you have the same results). There is also a macro **weights-demonstration.txt** which carries out all the instructions contained in this document.

The data come from a simulation.

We will be fitting 4 models: an unweighted model ('Unweighted'), a model using the MLwiN weights facility ('MLwiN'), a model where we standardise the weights ourselves and then pass these to MLwiN as 'raw weights' ('Manually standardised'), and a model where we replicate MLwiN's weighting but without usinig its weights facility at all, using Pfeffermann et al. (1998)'s Step A ('Manually weighted').

Naming columns

- Give these names to columns 8 to 13:
  - **stand_L2**
  - **stand_L1**
  - **stand_comp**
  - **stand_L2wt**
  - **stand_L1wt**
  - **stand_compwt**

Constructing variables

For the model using MLwiN's weights facility, the weights are already provided in the worksheet (**L1weight** and **L2weight**). We need to construct the standardised weights **stand_L2wt** and **stand_compwt** for the 'Manually standardised' model, and also the variables **stand_L2** and **stand_comp** we will need for the 'Manually weighted' model. (Note that the commands provided in this section do the same thing as the instructions written in words)

stand_L2wt

- Take first entry of **L2weight** in blocks defined by **L2ID_ss** (to **c20**)
- Sum resulting column (to **b7**)
- Calculate (number of L2 units/**b7**) × **L2weight** (this is the standardised level 2 weight)

```
► TAKE 'L2ID_ss' 'L2weight' c150 c20
► SUM c20 b7
► CALC 'stand_L2wt' = (108/b7)*'L2weight'
```

stand_L1wt

- Take average of **L1weight** for each block defined by **L2ID_ss** (to **c21**)
- Divide **L1weight** by this averages column (this is the standardised level 1 weight)

```
► MLAV 'L2ID_ss' 'L1weight' c21
► CALC 'stand_L1wt' = 'L1weight'/c21
```

stand_compwt

- Sum **stand_L2wt** (to **b8**)
- Form product of **stand_L2wt** and **stand_L1wt**, multiply by the number of level 1 units, divide by **b8** (this is the standardised composite weight)

```
► SUM 'stand_L2wt' b8
► CALC 'stand_compwt' = 'stand_L1wt'*'stand_L2wt'*(10870/b8)
```

stand_L2

- Take square root of **stand_L2wt**
- Divide 1 by result

```
► CALC 'stand_L2' = sqrt('stand_L2wt')
► CALC 'stand_L2' = 1/'stand_L2'
```

stand_L1

- Take square root of **stand_L1wt**
- Divide 1 by result

```
► CALC 'stand_L1' = sqrt('stand_L1wt')
► CALC 'stand_L1' = 1/'stand_L1'
```

stand_comp

- Take square root of **stand_compwt**
- Divide 1 by result

```
► CALC 'stand_comp' = sqrt('stand_compwt')
► CALC 'stand_comp' = 1/'stand_comp'
```

Fitting models

- Set up the following model

$$\text{y\_ss}_{ij} \sim \mathrm{N}(XB, \Omega)$$
$$\text{y\_ss}_{ij} = \beta_{0ij}\text{cons\_ss} + 11.000(3.317)\text{x\_ss}_{ij}$$
$$\beta_{0ij} = 11.000(3.317) + u_{0j} + e_{0ij}$$

$$\begin{bmatrix} u_{0j} \end{bmatrix} \sim \mathrm{N}(0,\ \Omega_u) \ : \ \Omega_u = \begin{bmatrix} 11.000(2.362) \end{bmatrix}$$

$$\begin{bmatrix} e_{0ij} \end{bmatrix} \sim \mathrm{N}(0,\ \Omega_e) \ : \ \Omega_e = \begin{bmatrix} 8.000(3.317) \end{bmatrix}$$

- Press **Start** and store the model under the name **Unweighted**

The Equations window should look like this:

$$\text{y\_ss}_{ij} \sim \mathrm{N}(XB, \Omega)$$
$$\text{y\_ss}_{ij} = \beta_{0ij}\text{cons\_ss} + 10.044(0.119)\text{x\_ss}_{ij}$$
$$\beta_{0ij} = 6.767(0.749) + u_{0j} + e_{0ij}$$

$$\begin{bmatrix} u_{0j} \end{bmatrix} \sim \mathrm{N}(0,\ \Omega_u) \ : \ \Omega_u = \begin{bmatrix} 57.537(8.047) \end{bmatrix}$$

$$\begin{bmatrix} e_{0ij} \end{bmatrix} \sim \mathrm{N}(0,\ \Omega_e) \ : \ \Omega_e = \begin{bmatrix} 152.475(2.078) \end{bmatrix}$$

*-2*loglikelihood(IGLS Deviance)* = 85886.455(10870 of 10870 cases in use)

- Select **Weights** from the **Model** menu
- Under **Raw weights in**, specify **L2weight** for level 2 and **L1weight** for level 1
- Under **Standardised weights to** select **c14** for level 2 and **c15** for level 1
- Under **Weighting mode** select **Use standardised weights** and click **Done**
- Press **Start** and store the model under the name **MLwiN**

$$\text{y\_ss}_{ij} \sim \text{N}(XB, \Omega)$$
$$\text{y\_ss}_{ij} = \beta_{0ij}\text{cons\_ss} + 5.581(0.087)\text{x\_ss}_{ij}$$
$$\beta_{0ij} = 3.608(0.807) + u_{0j} + e_{0ij}$$

$$\begin{bmatrix} u_{0j} \end{bmatrix} \sim \text{N}(0, \Omega_u) : \Omega_u = \begin{bmatrix} 68.770(9.469) \end{bmatrix}$$

$$\begin{bmatrix} e_{0ij} \end{bmatrix} \sim \text{N}(0, \Omega_e) : \Omega_e = \begin{bmatrix} 81.309(1.108) \end{bmatrix}$$

*-2*loglikelihood(IGLS Deviance)* = 83975.082(10870 of 10870 cases in use)

- Select **Weights** from the **Model** menu
- Under **Raw weights in**, specify **stand_L2wt** for level 2 and **stand_compwt** for level 1. (Note MLwiN does not compute the composite weight when raw weights are used, so, in contrast for the procedure when specifying standardised weights, the composite weight must be specified as the level 1 weight).
- Under **Standardised weight to** select **c1499** for level 2 and **c1500** for level 1
- Under **Weighting mode** select **Use raw weights** and click **Done**
- Press **Start** and store the model under the name **Manually standardised**

$$\text{y\_ss}_{ij} \sim \text{N}(XB, \Omega)$$
$$\text{y\_ss}_{ij} = \beta_{0ij}\text{cons\_ss} + 5.581(0.087)\text{x\_ss}_{ij}$$
$$\beta_{0ij} = 3.608(0.807) + u_{0j} + e_{0ij}$$

$$\begin{bmatrix} u_{0j} \end{bmatrix} \sim \text{N}(0, \Omega_u) : \Omega_u = \begin{bmatrix} 68.771(9.469) \end{bmatrix}$$

$$\begin{bmatrix} e_{0ij} \end{bmatrix} \sim \text{N}(0, \Omega_e) : \Omega_e = \begin{bmatrix} 81.308(1.108) \end{bmatrix}$$

*-2*loglikelihood(IGLS Deviance)* = 83975.082(10870 of 10870 cases in use)

- Add **stand_L2** and **stand_comp** to the model as explanatory variables
- Remove the random effect at both level 1 and level 2 from **cons_ss**
- Make **stand_L2** random at level 2 and **stand_comp** random at level 1
- Remove **stand_L2** and **stand_comp** from the fixed part of the model (untick the **Fixed parameter** box)
- Select **Weights** from the **Model** menu
- Under **Weighting mode** select **Off** and click **Done**
- Press **Start** and store the model under the name **Manually weighted**

$$\text{y\_ss}_{ij} \sim \text{N}(XB,\ \Omega)$$
$$\text{y\_ss}_{ij} = 3.608(0.807)\text{cons\_ss} + 5.581(0.087)\text{x\_ss}_{ij} + e_{3ij}\text{stand\_comp}_{ij} + u_{2j}\text{stand\_L2}_{j}$$

$$\left[u_{2j}\right] \sim \text{N}(0,\ \Omega_u) \ :\ \Omega_u = \left[68.771(9.469)\right]$$

$$\left[e_{3ij}\right] \sim \text{N}(0,\ \Omega_e) \ :\ \Omega_e = \left[81.308(1.108)\right]$$

$$-2{*}loglikelihood(IGLS\ Deviance) = 83975.083(10870\text{ of }10870\text{ cases in use})$$

Comparing the methods

Looking at the model comparison table (**Model** menu → **Compare stored models**), we can see that the 'Unweighted' model has very different estimates to the other three, and that the three weighted models have pretty much identical estimates, confirming that MLwiN is using the same standardisation process that we used above to standardise the weights ourselves.

This is confirmed if we compare **stand_L2wt** with **c14** and **stand_compwt** with **c15**.

Appendix B: Macros used in the simulation study

The macros presented here perform 60,000 replications of a simulation in which a population with 500 level 2 units each containing 50 level 1 units is generated, and a sample taken from the population in an informative design. Slight modifications were made to the macros in order to produce results for populations with other numbers and sizes of level 2 units; and in order to carry out other numbers of replications when investigating what the number of replications should be for the study (results not presented). The places and nature of these modifications are described in the comments within the macros themselves.

The macros explore a number of different ways of carrying out the weighted analysis, including various combinations of fixed and random sandwich estimators for the standard errors, and replicating the functionality of the MLwiN weights facility. The results from all these analyses are presented in Section 3.3; in Section 3.4 where only the weighted analysis using fixed and random sandwich estimators is presented, the same macros were used and just the results using both fixed and random sandwich estimators extracted from the output for presentation.

B.1 Macro 1

This macro is executed in MLwiN for the investigation into performance for the random intercepts model. It clears any existing data and models in the worksheet, gives the columns that will be used descriptive names, and sets up the aspects of the population that remain the same across replications (identifiers for level 2 and level 1; a **cons** column the length of the dataset; and a shortened version of the level 2 identifier). It then calls Macro 2, which performs 1 replication of the simulation, 60,000 times. Then it calculates the mean of the estimates for each parameter stored from the 60,000 replications, constructs confidence intervals for each parameter for each replication using the parameter estimate and the estimated standard error, and calculates for what proportion of replications the true (data-generating) value of the parameter lies inside the confidence interval.

```
Note ~~~~~~~~~~~~~~~
Note Preliminaries
Note ~~~~~~~~~~~~~~~
CLEAr
MWIP
```

```
BATCh 1
MAXI 100
ERASe c1-c75
ERASe c99-c410

Note ~~~~~~~~~~~~~~~~~~~~
Note Naming of variables
Note ~~~~~~~~~~~~~~~~~~~~
Name c1 'L2ID' c2 'L1ID' c3 'x1' c4 'x2' c5 'uj' c6 'eij' c7 'y' c8 'cons'
Name c10 'L2ID_short' c11 'x1bar' c12 'abso_uj' c13 'uj_short' c14 'L2prob_short'
Name c15 'L2pick_short' c16 'L2weight_short' c17 'L1avweight'
Name c20 'L2prob' c21 'L1prob' c22 'L2pick' c23 'L1pick' c24 'L2weight' c25 'L1weight'
Name c30 'SL2wt_man' c31 'SL1wt_man' c32 'Scompwt_man' c33 'inv_sqrt_wt_L2'
Name c34 'inv_sqrt_wt_L1' c35 'inv_sqrt_wt_comp'
Name c40 'L2ID_ss' c41 'L1ID_ss' c42 'y_ss' c43 'x1_ss' c44 'x2_ss' c45 'cons_ss'
Name c46 'L1pick_ss' c47 'L2prob_ss' c48 'L1prob_ss'
Name c55 'num_L2type_FS_ss_short' c56 'L2ID_ss_short' c57 'L2cons_ss' c58 'L2type_ss_short'
Name c59 'L2type_short' c60 'L2type' c61 'L1type' c62 'num_L2type_FS' c63 'num_L1type_FS'
Name c64 'num_L2type_FS_ss' c65 'num_L1type_FS_ss' c66 'L2type_ss' c67 'L1type_ss'
Name c68 'num_L2type_SS' c69 'num_L1type_SS' c70 'invL1type' c71 'num_L1type1_FS'
Name c72 'num_L1type0_FS' c73 'invL1type_ss' c74 'num_L1type1_ss' c75 'num_L1type0_ss'
Name c130 'beta_0|WNR' c131 'beta_1|WNR' c132 'beta_2|WNR' c133 'sigma^2_u|WNR'
Name c134 'sigma^2_e|WNR' c135 'se_beta0|WNR' c136 'se_beta1|WNR' c137 'se_beta2|WNR'
Name c138 'se_sigmasq_u|WNR' c139 'se_sigmasq_e|WNR' c141 'conv|WNR' c143 'beta0_upper_WNR'
Name c144 'beta0_lower_WNR' c145 'beta1_upper_WNR' c146 'beta1_lower_WNR'
Name c147 'beta2_upper_WNR' c148 'beta2_lower_WNR' c149 'sigmasq_u_upper_WNR'
Name c150 'sigmasq_u_lower_WNR' c151 'sigmasq_e_upper_WNR' c152 'sigmasq_e_lower_WNR'
Name c154 'beta_0_inside|WNR' c155 'beta_1_inside|WNR' c156 'beta_2_inside|WNR'
Name c157 'sigma^2_u_inside|WNR' c158 'sigma^2_e_inside|WNR'
Name c190 'beta_0|WNR_SE' c191 'beta_1|WNR_SE' c192 'beta_2|WNR_SE' c193 'sigma^2_u|WNR_SE'
Name c194 'sigma^2_e|WNR_SE' c195 'se_beta0|WNR_SE' c196 'se_beta1|WNR_SE'
Name c197 'se_beta2|WNR_SE' c198 'se_sigmasq_u|WNR_SE' c199 'se_sigmasq_e|WNR_SE'
Name c201 'conv|WNR_SE' c203 'beta0_upper_WNR_SE' c204 'beta0_lower_WNR_SE'
Name c205 'beta1_upper_WNR_SE' c206 'beta1_lower_WNR_SE' c207 'beta2_upper_WNR_SE'
```

```
Name c208 'beta2_lower_WNR_SE' c209 'sigmasq_u_upper_WNR_SE' c210 'sigmasq_u_lower_WNR_SE'
Name c211 'sigmasq_e_upper_WNR_SE' c212 'sigmasq_e_lower_WNR_SE' c214 'beta_0_inside|WNR_SE'
Name c215 'beta_1_inside|WNR_SE' c216 'beta_2_inside|WNR_SE' c217 'sigma^2_u_inside|WNR_SE'
Name c218 'sigma^2_e_inside|WNR_SE'
Name c220 'beta_0|UW' c221 'beta_1|UW' c222 'beta_2|UW' c223 'sigma^2_u|UW'
Name c224 'sigma^2_e|UW' c225 'se_beta0|UW' c226 'se_beta1|UW' c227 'se_beta2|UW'
Name c228 'se_sigmasq_u|UW' c229 'se_sigmasq_e|UW' c231 'conv|UW' c233 'beta0_upper_UW'
Name c234 'beta0_lower_UW' c235 'beta1_upper_UW' c236 'beta1_lower_UW' c237 'beta2_upper_UW'
Name c238 'beta2_lower_UW' c239 'sigmasq_u_upper_UW' c240 'sigmasq_u_lower_UW'
Name c241 'sigmasq_e_upper_UW' c242 'sigmasq_e_lower_UW' c244' 'beta_0_inside|UW'
Name c245 'beta_1_inside|UW' c246 'beta_2_inside|UW' c247 'sigma^2_u_inside|UW'
Name c248 'sigma^2_e_inside|UW'
Name c250 'beta_0|MW' c251 'beta_1|MW' c252 'beta_2|MW' c253 'sigma^2_u|MW'
Name c254 'sigma^2_e|MW' c255 'se_beta0|MW' c256 'se_beta1|MW' c257 'se_beta2|MW'
Name c258 'se_sigmasq_u|MW' c259 'se_sigmasq_e|MW' c261 'conv|MW' c263 'beta0_upper_MW'
Name c264 'beta0_lower_MW' c265 'beta1_upper_MW' c266 'beta1_lower_MW' c267 'beta2_upper_MW'
Name c268 'beta2_lower_MW' c269 'sigmasq_u_upper_MW' c270 'sigmasq_u_lower_MW'
Name c271 'sigmasq_e_upper_MW' c272 'sigmasq_e_lower_MW' c274 'beta_0_inside|MW'
Name c275 'beta_1_inside|MW' c276 'beta_2_inside|MW' c277 'sigma^2_u_inside|MW'
Name c278 'sigma^2_e_inside|MW'
Name c280 'beta_0|MW_SE' c281 'beta_1|MW_SE' c282 'beta_2|MW_SE' c283 'sigma^2_u|MW_SE'
Name c284 'sigma^2_e|MW_SE' c285 'se_beta0|MW_SE' c286 'se_beta1|MW_SE'
Name c287 'se_beta2|MW_SE' c288 'se_sigmasq_u|MW_SE' c289 'se_sigmasq_e|MW_SE'
Name c291 'conv|MW_SE' c293 'beta0_upper_MW_SE' c294 'beta0_lower_MW_SE'
Name c295 'beta1_upper_MW_SE' c296 'beta1_lower_MW_SE' c297 'beta2_upper_MW_SE'
Name c298 'beta2_lower_MW_SE' c299 'sigmasq_u_upper_MW_SE' c300 'sigmasq_u_lower_MW_SE'
Name c301 'sigmasq_e_upper_MW_SE' c302 'sigmasq_e_lower_MW_SE' c304 'beta_0_inside|MW_SE'
Name c305 'beta_1_inside|MW_SE' c306 'beta_2_inside|MW_SE' c307 'sigma^2_u_inside|MW_SE'
Name c308 'sigma^2_e_inside|MW_SE'
Name c320 'beta_0|WNR_FSE' c321 'beta_1|WNR_FSE' c322 'beta_2|WNR_FSE'
Name c323 'sigma^2_u|WNR_FSE' c324 'sigma^2_e|WNR_FSE' c325 'se_beta0|WNR_FSE'
Name c326 'se_beta1|WNR_FSE' c327 'se_beta2|WNR_FSE' c328 'se_sigmasq_u|WNR_FSE'
Name c329 'se_sigmasq_e|WNR_FSE' c331 'conv|WNR_FSE' c333 'beta0_upper_WNR_FSE'
Name c334 'beta0_lower_WNR_FSE' c335 'beta1_upper_WNR_FSE' c336 'beta1_lower_WNR_FSE'
```

```
Name c337 'beta2_upper_WNR_FSE' c338 'beta2_lower_WNR_FSE' c339 'sigmasq_u_upper_WNR_FSE'
Name c340 'sigmasq_u_lower_WNR_FSE' c341 'sigmasq_e_upper_WNR_FSE'
Name c342 'sigmasq_e_lower_WNR_FSE' c344 'beta_0_inside|WNR_FSE'
Name c345 'beta_1_inside|WNR_FSE' c346 'beta_2_inside|WNR_FSE'
Name c347 'sigma^2_u_inside|WNR_FSE' c348 'sigma^2_e_inside|WNR_FSE'
Name c350 'beta_0|MW_FSE' c351 'beta_1|MW_FSE' c352 'beta_2|MW_FSE'
Name c353 'sigma^2_u|MW_FSE' c354 'sigma^2_e|MW_FSE' c355 'se_beta0|MW_FSE'
Name c356 'se_beta1|MW_FSE' c357 'se_beta2|MW_FSE' c358 'se_sigmasq_u|MW_FSE'
Name c359 'se_sigmasq_e|MW_FSE' c361 'conv|MW_FSE' c363 'beta0_upper_MW_FSE'
Name c364 'beta0_lower_MW_FSE' c365 'beta1_upper_MW_FSE' c366 'beta1_lower_MW_FSE'
name c367 'beta2_upper_MW_FSE' c368 'beta2_lower_MW_FSE' c369 'sigmasq_u_upper_MW_FSE'
Name c370 'sigmasq_u_lower_MW_FSE' c371 'sigmasq_e_upper_MW_FSE'
Name c372 'sigmasq_e_lower_MW_FSE' c374 'beta_0_inside|MW_FSE'
Name c375 'beta_1_inside|MW_FSE' c376 'beta_2_inside|MW_FSE'
Name c377 'sigma^2_u_inside|MW_FSE' c378 'sigma^2_e_inside|MW_FSE'
Name c380 'beta_0|WNR_FRSE' c381 'beta_1|WNR_FRSE' c382 'beta_2|WNR_FRSE'
Name c383 'sigma^2_u|WNR_FRSE' c384 'sigma^2_e|WNR_FRSE' c385 'se_beta0|WNR_FRSE'
Name c386 'se_beta1|WNR_FRSE' c387 'se_beta2|WNR_FRSE' c388 'se_sigmasq_u|WNR_FRSE'
Name c389 'se_sigmasq_e|WNR_FRSE' c391 'conv|WNR_FRSE' c393 'beta0_upper_WNR_FRSE'
Name c394 'beta0_lower_WNR_FRSE' c395 'beta1_upper_WNR_FRSE' c396 'beta1_lower_WNR_FRSE'
Name c397 'beta2_upper_WNR_FRSE' c398 'beta2_lower_WNR_FRSE'
Name c399 'sigmasq_u_upper_WNR_FRSE' c400 'sigmasq_u_lower_WNR_FRSE'
Name c401 'sigmasq_e_upper_WNR_FRSE' c402 'sigmasq_e_lower_WNR_FRSE'
Name c404 'beta_0_inside|WNR_FRSE' c405 'beta_1_inside|WNR_FRSE'
Name c406 'beta_2_inside|WNR_FRSE' c407 'sigma^2_u_inside|WNR_FRSE'
Name c408 'sigma^2_e_inside|WNR_FRSE'




Note ~~~~~~~~~~~~~~~~
Note Data structure
Note ~~~~~~~~~~~~~~~~
Note Generate 500 level 2 units each with 50 level 1 units and cons of length 25000
```

```
Note To investigate populations with different numbers and sizes of level 2 units, change all the
Note 500s in this section to the desired number of level 2 units, all the 50s to the desired size of
Note level 2 units, and all the 25000s to the product of the desired number and the desired size
CODE 500 50 1 'L2ID'
GENE 1 25000 1 'L1ID'
PUT 25000 1 'cons'
GENE 1 500 1 'L2ID_short'


Note ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Note Generate populations and take samples
Note ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Note Change last value on this line to carry out different number of replications
LOOP b1 1 60000
OBEY 'Q:\ggzrjp\MLwiN weights\Thirdbash\NewPopEachReplication\Cluster size
         \50 (default)\500 clusters (default)\Correct\weights_inner_500c50_1b.txt'
ENDLoop


Note ~~~~~~~~~~~~~~~~~~~~~~~~~~
Note Calculate bias & coverage
Note ~~~~~~~~~~~~~~~~~~~~~~~~~~
Note
ECHO 1
Note ++++++++++++++++++
Note +      Bias      +
Note ++++++++++++++++++
AVER 5 'beta_0|WNR' 'beta_1|WNR' 'beta_2|WNR' 'sigma^2_u|WNR' 'sigma^2_e|WNR'
AVER 5 'beta_0|WNR_FSE' 'beta_1|WNR_FSE' 'beta_2|WNR_FSE' 'sigma^2_u|WNR_FSE' 'sigma^2_e|WNR_FSE'
AVER 5 'beta_0|WNR_SE' 'beta_1|WNR_SE' 'beta_2|WNR_SE' 'sigma^2_u|WNR_SE' 'sigma^2_e|WNR_SE'
AVER 5 'beta_0|WNR_FRSE' 'beta_1|WNR_FRSE' 'beta_2|WNR_FRSE' 'sigma^2_u|WNR_FRSE' 'sigma^2_e|WNR_FRSE'
AVER 5 'beta_0|UW' 'beta_1|UW' 'beta_2|UW' 'sigma^2_u|UW' 'sigma^2_e|UW'
AVER 5 'beta_0|MW' 'beta_1|MW' 'beta_2|MW' 'sigma^2_u|MW' 'sigma^2_e|MW'
AVER 5 'beta_0|MW_SE' 'beta_1|MW_SE' 'beta_2|MW_SE' 'sigma^2_u|MW_SE' 'sigma^2_e|MW_SE'
```

```
AVER 5 'beta_0|MW_FSE' 'beta_1|MW_FSE' 'beta_2|MW_FSE' 'sigma^2_u|MW_FSE' 'sigma^2_e|MW_FSE'
ECHO 0
Note
CALC 'beta0_upper_WNR' = 'beta_0|WNR' + 1.96*'se_beta0|WNR'
CALC 'beta0_lower_WNR' = 'beta_0|WNR' - 1.96*'se_beta0|WNR'
CALC 'beta1_upper_WNR' = 'beta_1|WNR' + 1.96*'se_beta1|WNR'
CALC 'beta1_lower_WNR' = 'beta_1|WNR' - 1.96*'se_beta1|WNR'
CALC 'beta2_upper_WNR' = 'beta_2|WNR' + 1.96*'se_beta2|WNR'
CALC 'beta2_lower_WNR' = 'beta_2|WNR' - 1.96*'se_beta2|WNR'
CALC 'sigmasq_u_upper_WNR' = 'sigma^2_u|WNR' + 1.96*'se_sigmasq_u|WNR'
CALC 'sigmasq_u_lower_WNR' = 'sigma^2_u|WNR' - 1.96*'se_sigmasq_u|WNR'
CALC 'sigmasq_e_upper_WNR' = 'sigma^2_e|WNR' + 1.96*'se_sigmasq_e|WNR'
CALC 'sigmasq_e_lower_WNR' = 'sigma^2_e|WNR' - 1.96*'se_sigmasq_e|WNR'
CALC 'beta0_upper_WNR_FSE' = 'beta_0|WNR_FSE' + 1.96*'se_beta0|WNR_FSE'
CALC 'beta0_lower_WNR_FSE' = 'beta_0|WNR_FSE' - 1.96*'se_beta0|WNR_FSE'
CALC 'beta1_upper_WNR_FSE' = 'beta_1|WNR_FSE' + 1.96*'se_beta1|WNR_FSE'
CALC 'beta1_lower_WNR_FSE' = 'beta_1|WNR_FSE' - 1.96*'se_beta1|WNR_FSE'
CALC 'beta2_upper_WNR_FSE' = 'beta_2|WNR_FSE' + 1.96*'se_beta2|WNR_FSE'
CALC 'beta2_lower_WNR_FSE' = 'beta_2|WNR_FSE' - 1.96*'se_beta2|WNR_FSE'
CALC 'sigmasq_u_upper_WNR_FSE' = 'sigma^2_u|WNR_FSE' + 1.96*'se_sigmasq_u|WNR_FSE'
CALC 'sigmasq_u_lower_WNR_FSE' = 'sigma^2_u|WNR_FSE' - 1.96*'se_sigmasq_u|WNR_FSE'
CALC 'sigmasq_e_upper_WNR_FSE' = 'sigma^2_e|WNR_FSE' + 1.96*'se_sigmasq_e|WNR_FSE'
CALC 'sigmasq_e_lower_WNR_FSE' = 'sigma^2_e|WNR_FSE' - 1.96*'se_sigmasq_e|WNR_FSE'
CALC 'beta0_upper_WNR_SE' = 'beta_0|WNR_SE' + 1.96*'se_beta0|WNR_SE'
CALC 'beta0_lower_WNR_SE' = 'beta_0|WNR_SE' - 1.96*'se_beta0|WNR_SE'
CALC 'beta1_upper_WNR_SE' = 'beta_1|WNR_SE' + 1.96*'se_beta1|WNR_SE'
CALC 'beta1_lower_WNR_SE' = 'beta_1|WNR_SE' - 1.96*'se_beta1|WNR_SE'
CALC 'beta2_upper_WNR_SE' = 'beta_2|WNR_SE' + 1.96*'se_beta2|WNR_SE'
CALC 'beta2_lower_WNR_SE' = 'beta_2|WNR_SE' - 1.96*'se_beta2|WNR_SE'
CALC 'sigmasq_u_upper_WNR_SE' = 'sigma^2_u|WNR_SE' + 1.96*'se_sigmasq_u|WNR_SE'
CALC 'sigmasq_u_lower_WNR_SE' = 'sigma^2_u|WNR_SE' - 1.96*'se_sigmasq_u|WNR_SE'
CALC 'sigmasq_e_upper_WNR_SE' = 'sigma^2_e|WNR_SE' + 1.96*'se_sigmasq_e|WNR_SE'
CALC 'sigmasq_e_lower_WNR_SE' = 'sigma^2_e|WNR_SE' - 1.96*'se_sigmasq_e|WNR_SE'
CALC 'beta0_upper_WNR_FRSE' = 'beta_0|WNR_FRSE' + 1.96*'se_beta0|WNR_FRSE'
```

```
CALC 'beta0_lower_WNR_FRSE' = 'beta_0|WNR_FRSE' - 1.96*'se_beta0|WNR_FRSE'
CALC 'beta1_upper_WNR_FRSE' = 'beta_1|WNR_FRSE' + 1.96*'se_beta1|WNR_FRSE'
CALC 'beta1_lower_WNR_FRSE' = 'beta_1|WNR_FRSE' - 1.96*'se_beta1|WNR_FRSE'
CALC 'beta2_upper_WNR_FRSE' = 'beta_2|WNR_FRSE' + 1.96*'se_beta2|WNR_FRSE'
CALC 'beta2_lower_WNR_FRSE' = 'beta_2|WNR_FRSE' - 1.96*'se_beta2|WNR_FRSE'
CALC 'sigmasq_u_upper_WNR_FRSE' = 'sigma^2_u|WNR_FRSE' + 1.96*'se_sigmasq_u|WNR_FRSE'
CALC 'sigmasq_u_lower_WNR_FRSE' = 'sigma^2_u|WNR_FRSE' - 1.96*'se_sigmasq_u|WNR_FRSE'
CALC 'sigmasq_e_upper_WNR_FRSE' = 'sigma^2_e|WNR_FRSE' + 1.96*'se_sigmasq_e|WNR_FRSE'
CALC 'sigmasq_e_lower_WNR_FRSE' = 'sigma^2_e|WNR_FRSE' - 1.96*'se_sigmasq_e|WNR_FRSE'
CALC 'beta0_upper_UW' = 'beta_0|UW' + 1.96*'se_beta0|UW'
CALC 'beta0_lower_UW' = 'beta_0|UW' - 1.96*'se_beta0|UW'
CALC 'beta1_upper_UW' = 'beta_1|UW' + 1.96*'se_beta1|UW'
CALC 'beta1_lower_UW' = 'beta_1|UW' - 1.96*'se_beta1|UW'
CALC 'beta2_upper_UW' = 'beta_2|UW' + 1.96*'se_beta2|UW'
CALC 'beta2_lower_UW' = 'beta_2|UW' - 1.96*'se_beta2|UW'
CALC 'sigmasq_u_upper_UW' = 'sigma^2_u|UW' + 1.96*'se_sigmasq_u|UW'
CALC 'sigmasq_u_lower_UW' = 'sigma^2_u|UW' - 1.96*'se_sigmasq_u|UW'
CALC 'sigmasq_e_upper_UW' = 'sigma^2_e|UW' + 1.96*'se_sigmasq_e|UW'
CALC 'sigmasq_e_lower_UW' = 'sigma^2_e|UW' - 1.96*'se_sigmasq_e|UW'
CALC 'beta0_upper_MW' = 'beta_0|MW' + 1.96*'se_beta0|MW'
CALC 'beta0_lower_MW' = 'beta_0|MW' - 1.96*'se_beta0|MW'
CALC 'beta1_upper_MW' = 'beta_1|MW' + 1.96*'se_beta1|MW'
CALC 'beta1_lower_MW' = 'beta_1|MW' - 1.96*'se_beta1|MW'
CALC 'beta2_upper_MW' = 'beta_2|MW' + 1.96*'se_beta2|MW'
CALC 'beta2_lower_MW' = 'beta_2|MW' - 1.96*'se_beta2|MW'
CALC 'sigmasq_u_upper_MW' = 'sigma^2_u|MW' + 1.96*'se_sigmasq_u|MW'
CALC 'sigmasq_u_lower_MW' = 'sigma^2_u|MW' - 1.96*'se_sigmasq_u|MW'
CALC 'sigmasq_e_upper_MW' = 'sigma^2_e|MW' + 1.96*'se_sigmasq_e|MW'
CALC 'sigmasq_e_lower_MW' = 'sigma^2_e|MW' - 1.96*'se_sigmasq_e|MW'
CALC 'beta0_upper_MW_SE' = 'beta_0|MW_SE' + 1.96*'se_beta0|MW_SE'
CALC 'beta0_lower_MW_SE' = 'beta_0|MW_SE' - 1.96*'se_beta0|MW_SE'
CALC 'beta1_upper_MW_SE' = 'beta_1|MW_SE' + 1.96*'se_beta1|MW_SE'
CALC 'beta1_lower_MW_SE' = 'beta_1|MW_SE' - 1.96*'se_beta1|MW_SE'
CALC 'beta2_upper_MW_SE' = 'beta_2|MW_SE' + 1.96*'se_beta2|MW_SE'
```

```
CALC 'beta2_lower_MW_SE' = 'beta_2|MW_SE' - 1.96*'se_beta2|MW_SE'
CALC 'sigmasq_u_upper_MW_SE' = 'sigma^2_u|MW_SE' + 1.96*'se_sigmasq_u|MW_SE'
CALC 'sigmasq_u_lower_MW_SE' = 'sigma^2_u|MW_SE' - 1.96*'se_sigmasq_u|MW_SE'
CALC 'sigmasq_e_upper_MW_SE' = 'sigma^2_e|MW_SE' + 1.96*'se_sigmasq_e|MW_SE'
CALC 'sigmasq_e_lower_MW_SE' = 'sigma^2_e|MW_SE' - 1.96*'se_sigmasq_e|MW_SE'
CALC 'beta0_upper_MW_FSE' = 'beta_0|MW_FSE' + 1.96*'se_beta0|MW_FSE'
CALC 'beta0_lower_MW_FSE' = 'beta_0|MW_FSE' - 1.96*'se_beta0|MW_FSE'
CALC 'beta1_upper_MW_FSE' = 'beta_1|MW_FSE' + 1.96*'se_beta1|MW_FSE'
CALC 'beta1_lower_MW_FSE' = 'beta_1|MW_FSE' - 1.96*'se_beta1|MW_FSE'
CALC 'beta2_upper_MW_FSE' = 'beta_2|MW_FSE' + 1.96*'se_beta2|MW_FSE'
CALC 'beta2_lower_MW_FSE' = 'beta_2|MW_FSE' - 1.96*'se_beta2|MW_FSE'
CALC 'sigmasq_u_upper_MW_FSE' = 'sigma^2_u|MW_FSE' + 1.96*'se_sigmasq_u|MW_FSE'
CALC 'sigmasq_u_lower_MW_FSE' = 'sigma^2_u|MW_FSE' - 1.96*'se_sigmasq_u|MW_FSE'
CALC 'sigmasq_e_upper_MW_FSE' = 'sigma^2_e|MW_FSE' + 1.96*'se_sigmasq_e|MW_FSE'
CALC 'sigmasq_e_lower_MW_FSE' = 'sigma^2_e|MW_FSE' - 1.96*'se_sigmasq_e|MW_FSE'
Note
CALC 'beta_0_inside|WNR' = ('beta0_lower_WNR' < 1) & ('beta0_upper_WNR' > 1)
CALC 'beta_1_inside|WNR' = ('beta1_lower_WNR' < 1) & ('beta1_upper_WNR' > 1)
CALC 'beta_2_inside|WNR' = ('beta2_lower_WNR' < 1) & ('beta2_upper_WNR' > 1)
CALC 'sigma^2_u_inside|WNR' = ('sigmasq_u_lower_WNR' < 0.25) & ('sigmasq_u_upper_WNR' > 0.25)
CALC 'sigma^2_e_inside|WNR' = ('sigmasq_e_lower_WNR' < 1) & ('sigmasq_e_upper_WNR' > 1)
CALC 'beta_0_inside|WNR_FSE' = ('beta0_lower_WNR_FSE' < 1) & ('beta0_upper_WNR_FSE' > 1)
CALC 'beta_1_inside|WNR_FSE' = ('beta1_lower_WNR_FSE' < 1) & ('beta1_upper_WNR_FSE' > 1)
CALC 'beta_2_inside|WNR_FSE' = ('beta2_lower_WNR_FSE' < 1) & ('beta2_upper_WNR_FSE' > 1)
CALC 'sigma^2_u_inside|WNR_FSE' = ('sigmasq_u_lower_WNR_FSE' < 0.25) &
                                  ('sigmasq_u_upper_WNR_FSE' > 0.25)
CALC 'sigma^2_e_inside|WNR_FSE' = ('sigmasq_e_lower_WNR_FSE' < 1) &
                                  ('sigmasq_e_upper_WNR_FSE' > 1)
CALC 'beta_0_inside|WNR_SE' = ('beta0_lower_WNR_SE' < 1) & ('beta0_upper_WNR_SE' > 1)
CALC 'beta_1_inside|WNR_SE' = ('beta1_lower_WNR_SE' < 1) & ('beta1_upper_WNR_SE' > 1)
CALC 'beta_2_inside|WNR_SE' = ('beta2_lower_WNR_SE' < 1) & ('beta2_upper_WNR_SE' > 1)
CALC 'sigma^2_u_inside|WNR_SE' = ('sigmasq_u_lower_WNR_SE' < 0.25) &
                                 ('sigmasq_u_upper_WNR_SE' > 0.25)
CALC 'sigma^2_e_inside|WNR_SE' = ('sigmasq_e_lower_WNR_SE' < 1) &
```

```
                                ('sigmasq_e_upper_WNR_SE' > 1)
CALC 'beta_0_inside|WNR_FRSE' = ('beta0_lower_WNR_FRSE' < 1) & ('beta0_upper_WNR_FRSE' > 1)
CALC 'beta_1_inside|WNR_FRSE' = ('beta1_lower_WNR_FRSE' < 1) & ('beta1_upper_WNR_FRSE' > 1)
CALC 'beta_2_inside|WNR_FRSE' = ('beta2_lower_WNR_FRSE' < 1) & ('beta2_upper_WNR_FRSE' > 1)
CALC 'sigma^2_u_inside|WNR_FRSE' = ('sigmasq_u_lower_WNR_FRSE' < 0.25) &
                                  ('sigmasq_u_upper_WNR_FRSE' > 0.25)
CALC 'sigma^2_e_inside|WNR_FRSE' = ('sigmasq_e_lower_WNR_FRSE' < 1) &
                                  ('sigmasq_e_upper_WNR_FRSE' > 1)
CALC 'beta_0_inside|UW' = ('beta0_lower_UW' < 1) & ('beta0_upper_UW' > 1)
CALC 'beta_1_inside|UW' = ('beta1_lower_UW' < 1) & ('beta1_upper_UW' > 1)
CALC 'beta_2_inside|UW' = ('beta2_lower_UW' < 1) & ('beta2_upper_UW' > 1)
CALC 'sigma^2_u_inside|UW' = ('sigmasq_u_lower_UW' < 0.25) &
                            ('sigmasq_u_upper_UW' > 0.25)
CALC 'sigma^2_e_inside|UW' = ('sigmasq_e_lower_UW' < 1) & ('sigmasq_e_upper_UW' > 1)
CALC 'beta_0_inside|MW' = ('beta0_lower_MW' < 1) & ('beta0_upper_MW' > 1)
CALC 'beta_1_inside|MW' = ('beta1_lower_MW' < 1) & ('beta1_upper_MW' > 1)
CALC 'beta_2_inside|MW' = ('beta2_lower_MW' < 1) & ('beta2_upper_MW' > 1)
CALC 'sigma^2_u_inside|MW' = ('sigmasq_u_lower_MW' < 0.25) & ('sigmasq_u_upper_MW' > 0.25)
CALC 'sigma^2_e_inside|MW' = ('sigmasq_e_lower_MW' < 1) & ('sigmasq_e_upper_MW' > 1)
CALC 'beta_0_inside|MW_SE' = ('beta0_lower_MW_SE' < 1) & ('beta0_upper_MW_SE' > 1)
CALC 'beta_1_inside|MW_SE' = ('beta1_lower_MW_SE' < 1) & ('beta1_upper_MW_SE' > 1)
CALC 'beta_2_inside|MW_SE' = ('beta2_lower_MW_SE' < 1) & ('beta2_upper_MW_SE' > 1)
CALC 'sigma^2_u_inside|MW_SE' = ('sigmasq_u_lower_MW_SE' < 0.25) &
                               ('sigmasq_u_upper_MW_SE' > 0.25)
CALC 'sigma^2_e_inside|MW_SE' = ('sigmasq_e_lower_MW_SE' < 1) & ('sigmasq_e_upper_MW_SE' > 1)
CALC 'beta_0_inside|MW_FSE' = ('beta0_lower_MW_FSE' < 1) & ('beta0_upper_MW_FSE' > 1)
CALC 'beta_1_inside|MW_FSE' = ('beta1_lower_MW_FSE' < 1) & ('beta1_upper_MW_FSE' > 1)
CALC 'beta_2_inside|MW_FSE' = ('beta2_lower_MW_FSE' < 1) & ('beta2_upper_MW_FSE' > 1)
CALC 'sigma^2_u_inside|MW_FSE' = ('sigmasq_u_lower_MW_FSE' < 0.25) &
                                ('sigmasq_u_upper_MW_FSE' > 0.25)
CALC 'sigma^2_e_inside|MW_FSE' = ('sigmasq_e_lower_MW_FSE' < 1) &
                                ('sigmasq_e_upper_MW_FSE' > 1)
Note
ECHO 1
```

```
Note ++++++++++++++++++
Note +    Coverage    +
Note ++++++++++++++++++
AVER 5 'beta_0_inside|WNR' 'beta_1_inside|WNR' 'beta_2_inside|WNR'
       'sigma^2_u_inside|WNR' 'sigma^2_e_inside|WNR'
AVER 5 'beta_0_inside|WNR_FSE' 'beta_1_inside|WNR_FSE' 'beta_2_inside|WNR_FSE'
       'sigma^2_u_inside|WNR_FSE' 'sigma^2_e_inside|WNR_FSE'
AVER 5 'beta_0_inside|WNR_SE' 'beta_1_inside|WNR_SE' 'beta_2_inside|WNR_SE'
       'sigma^2_u_inside|WNR_SE' 'sigma^2_e_inside|WNR_SE'
AVER 5 'beta_0_inside|WNR_FRSE' 'beta_1_inside|WNR_FRSE' 'beta_2_inside|WNR_FRSE'
       'sigma^2_u_inside|WNR_FRSE' 'sigma^2_e_inside|WNR_FRSE'
AVER 5 'beta_0_inside|UW' 'beta_1_inside|UW' 'beta_2_inside|UW'
       'sigma^2_u_inside|UW' 'sigma^2_e_inside|UW'
AVER 5 'beta_0_inside|MW' 'beta_1_inside|MW' 'beta_2_inside|MW'
       'sigma^2_u_inside|MW' 'sigma^2_e_inside|MW'
AVER 5 'beta_0_inside|MW_SE' 'beta_1_inside|MW_SE' 'beta_2_inside|MW_SE'
       'sigma^2_u_inside|MW_SE' 'sigma^2_e_inside|MW_SE'
AVER 5 'beta_0_inside|MW_FSE' 'beta_1_inside|MW_FSE' 'beta_2_inside|MW_FSE'
       'sigma^2_u_inside|MW_FSE' 'sigma^2_e_inside|MW_FSE'
ECHO 0
```

B.2 Macro 2

This macro performs 1 replication of the simulation study. It is not directly executed in MLwiN but is rather called by Macro 1. It first clears the model and makes sure that no weights or sandwich estimators are specified, and erases all the data which changes with every replication renaming the columns afterwards (renaming the columns since the names are lost when the data is erased). Then the explanatory variables x_1 and x_2 are generated, and the random effects at each level. The response is then calculated as a function of these. Selection probabilities are then generated as functions of the level 2 and level 1 random effects. Some of the work towards calculating the weights is done, and then the sample is selected using the calculated selection probabilities. The calculation of the weights is then completed. Finally, the model is fitted without weights; using the weights facility with all possible combinations of sandwich estimators or not for the fixed and random parts; and replicating the weights facility with macro commands with all possible combinations for the sandwich estimators; in each case the parameter estimates being added onto the end of the relevant storage column.

```
Note ~~~~~~~~~~~~~~~
Note Preliminaries
Note ~~~~~~~~~~~~~~~
WEIGhts 0
CLEAr
BATCh 1
MAXI 500
RSDE 0
FSDE 0
ERASe c3-c7
ERASe c22-c50
ERASe c55-c75
ERASe c11-c17
Name c3 'x1' c4 'x2' c5 'uj' c6 'eij' c7 'y'
Name c11 'x1bar' c12 'abso_uj' c13 'uj_short' c14 'L2prob_short' c15 'L2pick_short'
Name c16 'L2weight_short' c17 'L1avweight'
Name c22 'L2pick' c23 'L1pick' c24 'L2weight' c25 'L1weight'
Name c30 'SL2wt_man' c31 'SL1wt_man' c32 'Scompwt_man' c33 'inv_sqrt_wt_L2'
```

```
Name c34 'inv_sqrt_wt_L1' c35 'inv_sqrt_wt_comp'
Name c40 'L2ID_ss' c41 'L1ID_ss' c42 'y_ss' c43 'x1_ss' c44 'x2_ss' c45 'cons_ss'
Name c46 'L1pick_ss' c47 'L2prob_ss' c48 'L1prob_ss' c73 'invL1type_ss' c74 'num_L1type1_ss'
Name c75 'num_L1type0_ss'
Name c55 'num_L2type_FS_ss_short' c56 'L2ID_ss_short' c57 'L2cons_ss' c58 'L2type_ss_short'
Name c59 'L2type_short' c60 'L2type' c61 'L1type' c62 'num_L2type_FS' c63 'num_L1type_FS'
Name c64 'num_L2type_FS_ss' c65 'num_L1type_FS_ss' c66 'L2type_ss' c67 'L1type_ss'
Name c68 'num_L2type_SS' c69 'num_L1type_SS' c70 'invL1type' c71 'num_L1type1_FS'
Name c72 'num_L1type0_FS' c73 'invL1type_ss' c74 'num_L1type1_ss' c75 'num_L1type0_ss'


Note ~~~~~~~~~~
Note Generate variables
Note ~~~~~~~~~~
Note To investigate populations with different numbers and sizes of level 2 units, change all the
Note 500s in this section to the desired number of level 2 units, all the 50s to the desired size of
Note level 2 units, and all the 25000s to the product of the desired number and the desired size
NRAN 25000 'x1'
MLAV 'L2ID' 'x1' 'x1bar'
CALC 'x1' = 'x1' - 'x1bar'
CALC 'x1' = 'x1' + 1
BRAN 500 'x2' 0.5 1
NRAN 500 'uj_short'
CALC 'uj_short' = 'uj_short'*sqrt(0.25)
NRAN 25000 'eij'
MERGe 'L2ID_short' 'x2' 'uj_short' 'L2ID' 'x2' 'uj'
CALC 'y' = 1 + 'x1' + 'x2' + 'uj' + 'eij'

Note ~~~~~~~~~~~~~~~~~~~~~~~~
Note Selection probabilities
Note ~~~~~~~~~~~~~~~~~~~~~~~~
ABSO 'uj_short' 'abso_uj'
CALC 'L2prob_short' = 0.25*('abso_uj' > 0.5) + 0.75*('abso_uj' <= 0.5)
CALC 'L1prob' = 0.25*('eij' > 0) + 0.75*('eij' <= 0)
```

```
Note ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Note For use in calculating weights
Note ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
CALC 'L2type_short' = 'abso_uj' > 0.5
CALC 'L1type' = 'eij' > 0
SUM 'L2type_short' b30
CALC 'num_L2type_FS' = b30*'L2type_short' + (500 - b30)*(1 - 'L2type_short')
MERGe 'L2ID_short' 'L2type_short' 'num_L2type_FS' 'L2ID' 'L2type' 'num_L2type_FS'
MLSU 'L2ID' 'L1type' 'num_L1type1_FS'
CALC 'invL1type' = 1 - 'L1type'
MLSU 'L2ID' 'invL1type' 'num_L1type0_FS'
CALC 'num_L1type_FS' = ('L1type'*'num_L1type1_FS') + ('invL1type'*'num_L1type0_FS')

Note ~~~~~~~~~~~~
Note Take sample
Note ~~~~~~~~~~~~
Note To investigate populations with different numbers and sizes of level 2 units, change all the
Note 500s in this section to the desired number of level 2 units and all the 25000s to the product
Note of the desired number and the desired size
BRAN 500 'L2pick_short' 'L2prob_short' 1
MERGe 'L2ID_short' 'L2pick_short' 'L2prob_short' 'L2ID' 'L2pick' 'L2prob'
Note
CALC 'L1prob' = 'L1prob'*'L2pick'
BRAN 25000 'L1pick' 'L1prob' 1
Note
OMIT 0 'L1pick' 'L2ID' 'L1ID' 'y' 'x1' 'x2' 'cons' 'L2prob' 'L1prob' 'L2type' 'L1type'
       'invL1type' 'num_L2type_FS' 'num_L1type_FS' 'L1pick_ss' 'L2ID_ss' 'L1ID_ss'
       'y_ss' 'x1_ss' 'x2_ss' 'cons_ss' 'L2prob_ss' 'L1prob_ss' 'L2type_ss' 'L1type_ss'
       'invL1type_ss' 'num_L2type_FS_ss' 'num_L1type_FS_ss'
ERASe 'L1prob'
NAME c21 'L1prob'
CALC 'L1prob' = 0.25*('eij' > 0) + 0.75*('eij' <= 0)
```

```
Note ~~~~~~~~
Note Weights
Note ~~~~~~~~
TAKE 'L2ID_ss' 'L2type_ss' 'cons_ss' 'num_L2type_FS_ss' 'L2ID_ss_short' 'L2type_ss_short'
     'L2cons_ss' 'num_L2type_FS_ss_short'
SUM 'L2type_ss_short' b31
SUM 'L2cons_ss' b32
CALC 'num_L2type_SS' = b31*'L2type_ss_short' + (b32-b31)*(1-'L2type_ss_short')
CALC 'L2weight' = 'num_L2type_FS_ss_short'/'num_L2type_SS'
MERG 'L2ID_ss_short' 'L2weight' 'L2ID_ss' 'L2weight'
MLSU 'L2ID_ss' 'L1type_ss' 'num_L1type1_ss'
MLSU 'L2ID_ss' 'invL1type_ss' 'num_L1type0_ss'
CALC 'num_L1type_ss' = ('num_L1type1_ss'*'L1type_ss') + ('num_L1type0_ss'*'invL1type_ss')
CALC 'L1weight' = 'num_L1type_FS_ss'/'num_L1type_ss'
Note
TAKE 'L2ID_ss' 'L2weight' 'L2ID_ss_short'  'L2weight_short'
SUM 'L2pick_short' b4
SUM 'L2weight_short' b7
CALC 'SL2wt_man' = 'L2weight'*(b4/b7)
CALC 'inv_sqrt_wt_L2' = sqrt('SL2wt_man')
CALC 'inv_sqrt_wt_L2' = 1/'inv_sqrt_wt_L2'
Note
MLAV 'L2ID_ss' 'L1weight' 'L1avweight'
CALC 'SL1wt_man' = 'L1weight'/'L1avweight'
CALC 'inv_sqrt_wt_L1' = sqrt('SL1wt_man')
CALC 'inv_sqrt_wt_L1' = 1/'inv_sqrt_wt_L1'
Note
SUM 'SL2wt_man' b8
SUM 'cons_ss' b9
CALC 'inv_sqrt_wt_comp' = 'SL1wt_man'*'SL2wt_man'*(b9/b8)
CALC 'inv_sqrt_wt_comp' = sqrt('inv_sqrt_wt_comp')
CALC 'inv_sqrt_wt_comp' = 1/'inv_sqrt_wt_comp'

Note ~~~~~~~~~~~
```

```
Note Fit models
Note ~~~~~~~~~~~
Note **** Model structure ****
RESP 'y_ss'
IDEN 1 'L1ID_ss' 2 'L2ID_ss'
ADDT 'cons_ss'
SETV 1 'cons_ss'
SETV 2 'cons_ss'
ADDT 'x1_ss'
ADDT 'x2_ss'
Note **** Unweighted model ****
STARt
PICK 1 c1098 b24
JOIN 'beta_0|UW' b24 'beta_0|UW'
PICK 2 c1098 b25
JOIN 'beta_1|UW' b25 'beta_1|UW'
PICK 3 c1098 b28
JOIN 'beta_2|UW' b28 'beta_2|UW'
PICK 1 c1096 b26
JOIN 'sigma^2_u|UW' b26 'sigma^2_u|UW'
PICK 2 c1096 b27
JOIN 'sigma^2_e|UW' b27 'sigma^2_e|UW'
PICK 1 c1099 b24
SQRT b24 b24
JOIN 'se_beta0|UW' b24 'se_beta0|UW'
PICK 3 c1099 b25
SQRT b25 b25
JOIN 'se_beta1|UW' b25 'se_beta1|UW'
PICK 6 c1099 b28
SQRT b28 b28
JOIN 'se_beta2|UW' b28 'se_beta2|UW'
PICK 1 c1097 b26
SQRT b26 b26
JOIN 'se_sigmasq_u|UW' b26 'se_sigmasq_u|UW'
```

```
PICK 3 c1097 b27
SQRT b27 b27
JOIN 'se_sigmasq_e|UW' b27 'se_sigmasq_e|UW'
CONV b28
JOIN 'conv|UW' b28 'conv|UW'
Note **** Weighted model ****
WEIGhts 2 1 'L2weight'
WEIGhts 1 1 'L1weight'
WEIGhts 2 2 c50
WEIGhts 1 2 c51
WEIGhts
WEIGhts 2
STARt
PICK 1 c1098 b24
JOIN 'beta_0|WNR' b24 'beta_0|WNR'
PICK 2 c1098 b25
JOIN 'beta_1|WNR' b25 'beta_1|WNR'
PICK 3 c1098 b28
JOIN 'beta_2|WNR' b28 'beta_2|WNR'
PICK 1 c1096 b26
JOIN 'sigma^2_u|WNR' b26 'sigma^2_u|WNR'
PICK 2 c1096 b27
JOIN 'sigma^2_e|WNR' b27 'sigma^2_e|WNR'
PICK 1 c1099 b24
SQRT b24 b24
JOIN 'se_beta0|WNR' b24 'se_beta0|WNR'
PICK 3 c1099 b25
SQRT b25 b25
JOIN 'se_beta1|WNR' b25 'se_beta1|WNR'
PICK 6 c1099 b28
SQRT b28 b28
JOIN 'se_beta2|WNR' b28 'se_beta2|WNR'
PICK 1 c1097 b26
SQRT b26 b26
```

```
JOIN 'se_sigmasq_u|WNR' b26 'se_sigmasq_u|WNR'
PICK 3 c1097 b27
SQRT b27 b27
JOIN 'se_sigmasq_e|WNR' b27 'se_sigmasq_e|WNR'
CONV b28
JOIN 'conv|WNR' b28 'conv|WNR'
Note **** Weighted model, sandwich estimators (fixed part only) ****
FSDE 2
STARt
PICK 1 c1098 b24
JOIN 'beta_0|WNR_FSE' b24 'beta_0|WNR_FSE'
PICK 2 c1098 b25
JOIN 'beta_1|WNR_FSE' b25 'beta_1|WNR_FSE'
PICK 3 c1098 b28
JOIN 'beta_2|WNR_FSE' b28 'beta_2|WNR_FSE'
PICK 1 c1096 b26
JOIN 'sigma^2_u|WNR_SE' b26 'sigma^2_u|WNR_FSE'
PICK 2 c1096 b27
JOIN 'sigma^2_e|WNR_FSE' b27 'sigma^2_e|WNR_FSE'
PICK 1 c1099 b24
SQRT b24 b24
JOIN 'se_beta0|WNR_FSE' b24 'se_beta0|WNR_FSE'
PICK 3 c1099 b25
SQRT b25 b25
JOIN 'se_beta1|WNR_FSE' b25 'se_beta1|WNR_FSE'
PICK 6 c1099 b28
SQRT b28 b28
JOIN 'se_beta2|WNR_FSE' b28 'se_beta2|WNR_FSE'
PICK 1 c1097 b26
SQRT b26 b26
JOIN 'se_sigmasq_u|WNR_FSE' b26 'se_sigmasq_u|WNR_FSE'
PICK 3 c1097 b27
SQRT b27 b27
JOIN 'se_sigmasq_e|WNR_FSE' b27 'se_sigmasq_e|WNR_FSE'
```

```
CONV b28
JOIN 'conv|WNR_FSE' b28 'conv|WNR_FSE'
Note **** Weighted model, sandwich estimators (random part only) ****
FSDE 0
RSDE 2
STARt
PICK 1 c1098 b24
JOIN 'beta_0|WNR_SE' b24 'beta_0|WNR_SE'
PICK 2 c1098 b25
JOIN 'beta_1|WNR_SE' b25 'beta_1|WNR_SE'
PICK 3 c1098 b28
JOIN 'beta_2|WNR_SE' b28 'beta_2|WNR_SE'
PICK 1 c1096 b26
JOIN 'sigma^2_u|WNR_SE' b26 'sigma^2_u|WNR_SE'
PICK 2 c1096 b27
JOIN 'sigma^2_e|WNR_SE' b27 'sigma^2_e|WNR_SE'
PICK 1 c1099 b24
SQRT b24 b24
JOIN 'se_beta0|WNR_SE' b24 'se_beta0|WNR_SE'
PICK 3 c1099 b25
SQRT b25 b25
JOIN 'se_beta1|WNR_SE' b25 'se_beta1|WNR_SE'
PICK 6 c1099 b28
SQRT b28 b28
JOIN 'se_beta2|WNR_SE' b28 'se_beta2|WNR_SE'
PICK 1 c1097 b26
SQRT b26 b26
JOIN 'se_sigmasq_u|WNR_SE' b26 'se_sigmasq_u|WNR_SE'
PICK 3 c1097 b27
SQRT b27 b27
JOIN 'se_sigmasq_e|WNR_SE' b27 'se_sigmasq_e|WNR_SE'
CONV b28
JOIN 'conv|WNR_SE' b28 'conv|WNR_SE'
Note **** Weighted model, sandwich estimators (fixed and random) ****
```

```
FSDE 2
STARt
PICK 1 c1098 b24
JOIN 'beta_0|WNR_FRSE' b24 'beta_0|WNR_FRSE'
PICK 2 c1098 b25
JOIN 'beta_1|WNR_FRSE' b25 'beta_1|WNR_FRSE'
PICK 3 c1098 b28
JOIN 'beta_2|WNR_FRSE' b28 'beta_2|WNR_FRSE'
PICK 1 c1096 b26
JOIN 'sigma^2_u|WNR_FRSE' b26 'sigma^2_u|WNR_FRSE'
PICK 2 c1096 b27
JOIN 'sigma^2_e|WNR_FRSE' b27 'sigma^2_e|WNR_FRSE'
PICK 1 c1099 b24
SQRT b24 b24
JOIN 'se_beta0|WNR_FRSE' b24 'se_beta0|WNR_FRSE'
PICK 3 c1099 b25
SQRT b25 b25
JOIN 'se_beta1|WNR_FRSE' b25 'se_beta1|WNR_FRSE'
PICK 6 c1099 b28
SQRT b28 b28
JOIN 'se_beta2|WNR_FRSE' b28 'se_beta2|WNR_FRSE'
PICK 1 c1097 b26
SQRT b26 b26
JOIN 'se_sigmasq_u|WNR_FRSE' b26 'se_sigmasq_u|WNR_FRSE'
PICK 3 c1097 b27
SQRT b27 b27
JOIN 'se_sigmasq_e|WNR_FRSE' b27 'se_sigmasq_e|WNR_FRSE'
CONV b28
JOIN 'conv|WNR_FRSE' b28 'conv|WNR_FRSE'
Note **** Manually weighted model ****
RSDE 0
FSDE 0
WEIGhts 0
CLRV 2
```

```
CLRV 1
ADDT 'inv_sqrt_wt_L2'
ADDT 'inv_sqrt_wt_comp'
SETV 2 'inv_sqrt_wt_L2'
SETV 1 'inv_sqrt_wt_comp'
FPAR 0 'inv_sqrt_wt_L2'
FPAR 0 'inv_sqrt_wt_comp'
STARt
PICK 1 c1098 b24
JOIN 'beta_0|MW' b24 'beta_0|MW'
PICK 2 c1098 b25
JOIN 'beta_1|MW' b25 'beta_1|MW'
PICK 3 c1098 b28
JOIN 'beta_2|MW' b28 'beta_2|MW'
PICK 1 c1096 b26
JOIN 'sigma^2_u|MW' b26 'sigma^2_u|MW'
PICK 2 c1096 b27
JOIN 'sigma^2_e|MW' b27 'sigma^2_e|MW'
PICK 1 c1099 b24
SQRT b24 b24
JOIN 'se_beta0|MW' b24 'se_beta0|MW'
PICK 3 c1099 b25
SQRT b25 b25
JOIN 'se_beta1|MW' b25 'se_beta1|MW'
PICK 6 c1099 b28
SQRT b28 b28
JOIN 'se_beta2|MW' b28 'se_beta2|MW'
PICK 1 c1097 b26
SQRT b26 b26
JOIN 'se_sigmasq_u|MW' b26 'se_sigmasq_u|MW'
PICK 3 c1097 b27
SQRT b27 b27
JOIN 'se_sigmasq_e|MW' b27 'se_sigmasq_e|MW'
CONV b28
```

```
JOIN 'conv|MW' b28 'conv|MW'
Note **** Manually weighted model, sandwich estimators random part only****
RSDE 2
STARt
PICK 1 c1098 b24
JOIN 'beta_0|MW_SE' b24 'beta_0|MW_SE'
PICK 2 c1098 b25
JOIN 'beta_1|MW_SE' b25 'beta_1|MW_SE'
PICK 3 c1098 b28
JOIN 'beta_2|MW_SE' b28 'beta_2|MW_SE'
PICK 1 c1096 b26
JOIN 'sigma^2_u|MW_SE' b26 'sigma^2_u|MW_SE'
PICK 2 c1096 b27
JOIN 'sigma^2_e|MW_SE' b27 'sigma^2_e|MW_SE'
PICK 1 c1099 b24
SQRT b24 b24
JOIN 'se_beta0|MW_SE' b24 'se_beta0|MW_SE'
PICK 3 c1099 b25
SQRT b25 b25
JOIN 'se_beta1|MW_SE' b25 'se_beta1|MW_SE'
PICK 6 c1099 b28
SQRT b28 b28
JOIN 'se_beta2|MW_SE' b28 'se_beta2|MW_SE'
PICK 1 c1097 b26
SQRT b26 b26
JOIN 'se_sigmasq_u|MW_SE' b26 'se_sigmasq_u|MW_SE'
PICK 3 c1097 b27
SQRT b27 b27
JOIN 'se_sigmasq_e|MW_SE' b27 'se_sigmasq_e|MW_SE'
CONV b28
JOIN 'conv|MW_SE' b28 'conv|MW_SE'
Note **** Manually weighted model, sandwich estimators fixed and random ****
FSDE 2
STARt
```

```
PICK 1 c1098 b24
JOIN 'beta_0|MW_FSE' b24 'beta_0|MW_FSE'
PICK 2 c1098 b25
JOIN 'beta_1|MW_FSE' b25 'beta_1|MW_FSE'
PICK 3 c1098 b28
JOIN 'beta_2|MW_FSE' b28 'beta_2|MW_FSE'
PICK 1 c1096 b26
JOIN 'sigma^2_u|MW_FSE' b26 'sigma^2_u|MW_FSE'
PICK 2 c1096 b27
JOIN 'sigma^2_e|MW_FSE' b27 'sigma^2_e|MW_FSE'
PICK 1 c1099 b24
SQRT b24 b24
JOIN 'se_beta0|MW_FSE' b24 'se_beta0|MW_FSE'
PICK 3 c1099 b25
SQRT b25 b25
JOIN 'se_beta1|MW_FSE' b25 'se_beta1|MW_FSE'
PICK 6 c1099 b28
SQRT b28 b28
JOIN 'se_beta2|MW_FSE' b28 'se_beta2|MW_FSE'
PICK 1 c1097 b26
SQRT b26 b26
JOIN 'se_sigmasq_u|MW_FSE' b26 'se_sigmasq_u|MW_FSE'
PICK 3 c1097 b27
SQRT b27 b27
JOIN 'se_sigmasq_e|MW_FSE' b27 'se_sigmasq_e|MW_FSE'
CONV b28
JOIN 'conv|MW_FSE' b28 'conv|MW_FSE'
```

References

References

Carle, A. (2009). Fitting multilevel models in complex survey data with design weights: Recommendations. *BMC Medical Research Methodology*, **9**(1):49. ISSN 1471-2288. doi:10.1186/1471-2288-9-49. URL http://www.biomedcentral.com/1471-2288/9/49

Pfeffermann, D., Skinner, C.J., Holmes, D.J., 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. ISSN 1467-9868. doi:10.1111/1467-9868.00106. URL http://dx.doi.org/10.1111/1467-9868.00106

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. ISSN 1467-985X. doi:10.1111/j.1467-985X.2006.00426.x. URL http://dx.doi.org/10.1111/j.1467-985X.2006.00426.x

Rodríguez, G. & Goldman, N. (2001). Improved estimation procedures for multilevel models with binary response: a case-study. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, **164**(2):339–355. ISSN 1467-985X. doi:10.1111/1467-985X.00206. URL http://dx.doi.org/10.1111/1467-985X.00206