Paper 438-2013

A SAS® Macro for Applying Multiple Imputation to Multilevel Data

Stephen A. Mistler, Arizona State University

ABSTRACT

Single-level multiple imputation procedures (e.g., PROC MI) are not appropriate for multilevel data sets where observations are nested within clusters. Analyzing multilevel data imputed with a single-level procedure yields variance estimates that are biased toward zero and may yield other biased parameters. Given the prevalence of clustered data (e.g., children within schools; employees within companies; observations within people), a general approach is needed for handling missing data in multilevel data sets. This paper describes a SAS macro, MMI_IMPUTE, that performs multiple imputation for clustered data sets with two levels. The macro uses a Bayesian implementation of the mixed linear model to generate imputations for lower-level incomplete variables, and uses single-level procedures similar to those used in PROC MI to generate imputations for cluster-level variables.

INTRODUCTION

Missing data are common in many fields of research and can lead to reduced power and increased bias if not properly handled. The two state-of-the-art missing data methods recommended by methodologists are multiple imputation and maximum likelihood estimation (Schafer & Graham, 2002). Although these techniques have advanced greatly in recent years, methods for handling missing data in multilevel data have received less attention. For example, maximum likelihood estimation remains an imperfect solution because (in almost all software implementations) it allows only dependent variables to be missing in multilevel data and eliminates all observations with missing predictor variables prior to estimation. Although multiple imputation does not suffer from this limitation, most software packages (including SAS®) include only single-level multiple imputation procedures. This paper aims to make the analysis of incomplete multilevel data easier by presenting a new SAS macro for performing multilevel imputation, MMI_IMPUTE. This paper demonstrates the use of this new tool in the context of a two-level multilevel model.

IGNORING MISSING DATA

The most common approach to handling missing data is simply to ignore it. The MIXED procedure in SAS is capable of handling missing values on the dependent variable through maximum likelihood estimation. So, it is reasonable to ignore the missing data if exclusively running PROC MIXED with complete predictor variables. However, PROC MIXED performs listwise deletion on the predictor variables (any case that is missing on one of the predictor variables is excluded from the analysis). SAS does not generate a warning for these cases, but reports the number of excluded cases in the output under "Number of Observations Not Used." Listwise deletion is problematic because it often results in biased parameter estimates if the data are missing at random (MAR) but not missing completely at random (MCAR) (Little, 1992). Additionally, listwise deletion results in decreased power to detect effects. "It is not uncommon in real life applications that more than half of the original sample is lost" (van Buuren, 2012, p 8). Because it is not reasonable simply to ignore the missing data and because maximum likelihood estimation (as implemented in SAS) can only handle missing values on the dependent variables, I recommend handling missing data in multilevel data sets through multilevel multiple imputation.

OVERVIEW OF MULTIPLE IMPUTATION

A multiple imputation (MI) analysis consists of two distinct phases: the imputation phase and the analysis/pooling phase. In the imputation phase, the algorithm draws parameters for the imputation model from a distribution and then uses these parameter estimates to fill in the missing data. This process is repeated for a very large number of iterations. Whenever a pre-determined number of iterations has passed, the algorithm outputs an imputed data set. The algorithm continues in this vein until *m* distinct data sets have been generated, where *m* is a number chosen by the analyst (usually between 20 and 50). This process amounts to sampling missing values from a posterior predictive distribution. In the analysis/pooling phase, the data analyst runs an identical analysis on each of the *m* data sets. Because each of the *m* data sets contains no missing data, the analyst can employ standard complete-data analysis methods. The point estimates and standard errors from these analyses are then pooled using methods described later in this document. The analysis/pooling phase can be repeated for multiple analyses without repeating the imputation phase.

An important point about multiple imputation is that the imputation model is distinct from the analysis models. That is, the model used to fill in the missing data may bear little relation to the analysis models. This is not problematic, provided that all of the necessary variables/effects are included in the imputation model. For example, an analyst with a single-level data set might use the MI procedure to impute the data. The analyst might specify that variables a, b, c, d, e, and f be included in the imputation process. The MI procedure would include all of the linear relationships between the specified variables in the imputation model. The analyst could then run any analyses whose variables/effects were in the imputation model. So, the analyst could run a regression of a on b, c, d, and e. The analyst could also run a simpler model, such as the regression of e on a. However, the analyst should not perform a regression of variable a on variable g, because g was not included in the imputation model. The analyst should also avoid analysis models with effects that were not included in the imputation model, such as a regression of a on b^2. Note that the

EXAMPLE DATA

Throughout the paper, I use an artificial data set that is loosely based on a math problem solving intervention described by Montague, Enders, and Dietz (2011) to demonstrate multilevel imputation. The data set mimics a scenario where researchers randomly assign 40 schools, each with 25 students, to an intervention and a control condition (i.e., a cluster-randomized design). In addition to a level-2 intervention code (TX; 0 = comparison, 1 = intervention), the data set contains a number of student-level covariates, including standardized math scores (MATH), math self-efficacy scores (SELFE), problem-solving pre-test scores (PREPSOLVE), a binary lunch assistance indicator (FRLUNCH), and a binary ethnicity indicator (MINORITY). The outcome variable is end-of-year math problem-solving test score (PSTPSOLVE). The end-of-year math problem-solving test scores and the self-efficacy ratings are each missing completely at random for 20% of students in the dataset. I use variables from this data set to illustrate the use of the MMI_IMPUTE macro throughout the paper. For readers that are interested in duplicating the illustrative analyses, the data set is available upon request.

Because all variables/effects to be analyzed must be included in the imputation model to preserve the effects in the imputed data, it is necessary to plan one's analyses prior to performing multiple imputation. In our example, the analysis of interest consists of end-of-year math problem-solving test scores (PSTPSOLVE) predicted by treatment (TX) and math self-efficacy (SELFE). The standardized math score (MATH) is not directly of interest, but is included in the model as a covariate. The multilevel regression equation for this model is

$$\mathrm{PSTPSOLVE}_{ij} = \beta_0 + \beta_1(\mathrm{MATH}_{ij}) + \beta_2(\mathrm{SELFE}_{ij}) + \beta_3(\mathrm{TX}_j) + b_{0j} + \varepsilon_{ij} \tag{1}$$

where $\mathrm{PSTPSOLVE}_{ij}$ is the outcome for case *i* in cluster *j*. β_0 is the intercept. β_1, β_2, and β_3 are slope coefficients. b_{0j} is a level-2 residual that allows the intercepts to vary across clusters, and ε_{ij} is the within-cluster residual for case *i*.

MULTILEVEL MULTIPLE IMPUTATION VS. SINGLE-LEVEL IMPUTATION

Single-level multiple imputation generates plausible values for missing data using a series of multivariate regression equations. Though this method is appropriate for single-level data sets where all observations share a common mean vector and covariance matrix, it fails to account for the clustering present in multilevel datasets. To illustrate, consider our math problem solving data set. Suppose that it is of interest to estimate a two-level random slope model where standardized math scores (MATH) predict end-of-year math problem-solving (PSTPSOLVE), as follows.

$$\mathrm{PSTPSOLVE}_{ij} = \beta_0 + \beta_1(\mathrm{MATH}_{ij}) + b_{0j} + b_{1j}(\mathrm{MATH}_{ij}) + \varepsilon_{ij} \tag{2}$$

In addition to the overall intercept and slope terms (β_0 and β_1, respectively), the analysis model also contains a level-2 residual intercept term (b_{0j}) and a level-2 residual slope term (b_{1j}). These terms effectively allow each cluster to have a unique regression line.

A single-level imputation algorithm that fills in the end-of-year math problem-solving test variable (PSTPSOLVE) using the standardized math test score (MATH) would employ a single regression equation that is common to every student, regardless of the school to which they belonged. This ignores the fact that the schools potentially differ in their intercepts (i.e., dependent variable means) and slopes. Specifically, the single-level regression imputation equation would contain a common intercept term (β_0), a common slope term (β_1) and a single common residual distribution, as follows.

$$\mathrm{PSTPSOLVE}_{i} = \beta_0 + \beta_1(\mathrm{MATH}_{i}) + \varepsilon_{i} \tag{3}$$

Equation 3 is a problematic imputation equation because it effectively omits an important main effect (i.e., cluster-level mean differences—the b_{0j} term in Equation 2) and an important interaction (i.e., the moderating effect of cluster membership on the regression equation—the b_{1j} term in Equation 2). It is well-established in the multiple imputation literature that omitting variables or effects from the imputation model can bias the subsequent estimates of those effects (because the single-level imputation algorithm imputes under a model where the variances of b_{0j} and b_{1j} equal zero). Multilevel imputation, on the other hand, would essentially use a separate regression line for each school, thereby eliminating bias that occurs due to model misspecification.

MMI_IMPUTE, A TWO-LEVEL IMPUTATION MACRO

The MI procedure is designed for single-level imputation. To address the lack of SAS software capable of handling missing data in multilevel data, I developed a macro for use in Version 9.3 of SAS. The SAS macro, called MMI_IMPUTE, implements multiple imputation for two-level data. The two-level imputation algorithm is a combination of three existing multiple imputation algorithms. The top level of the data (level 2) is imputed using an adaptation of the multiple imputation algorithm developed by Tanner and Wong (1987) and popularized by Schafer (1997). This imputation algorithm is similar to the algorithm used in the MI procedure for single-level data. The bottom level of the data (level 1) is imputed using an adaptation of Joseph Schafer's PAN algorithm (Schafer, 2001; Schafer & Yucel, 2002). The two algorithms are combined to create a single multilevel algorithm (called a Gibbs Sampler) using techniques described in Yucel (2008).

The MMI_IMPUTE macro is capable of handling both incomplete level-1 and level-2 variables. It is also capable of modeling random effects of level-1 variables on other level-1 variables. All variables to be imputed by the macro should be in their raw (i.e., uncentered) metrics. Finally, the imputation algorithm assumes that the incomplete variables are normally distributed. As such, the macro may introduce bias when applied to variables that are non-normal or discrete (categorical variables, ordinal variables, percentages, etc.). A small body of research suggests that normality violations may not introduce substantial bias, provided that fractional imputations are left unrounded (e.g., a binary variable with an imputed value of .74; Allison, 2005). However, this issue has not been examined in the context of multilevel imputation.

USING THE MACRO

The macro is available at the following website: http://www.mistlerconsulting.com/software.html. Because the macro is very long, I recommend setting up the macro using the %INCLUDE function. The syntax below shows the macro invocation for running MMI_IMPUTE with all parameters omitted.

Example 1.

```
%mmi_impute(
    data          = ,
    id_l2         = ,
    var           = ,
    random        = ,
    nimpute       = ,
    iterations    = ,
    out           = ,
    seed          = ,
    graphs        = );
```

The above code shows the macro call with all of the parameters left blank. DATA specifies the name of the input SAS data set. All missing values in the data set must be represented with a SAS missing value code. The missing data code can be a single decimal point ("."), a decimal point followed by a letter (e.g., ".B"), or a decimal point followed by an underscore ("._"). The macro does not differentiate between these three types of missing values. The input dataset must also be in stacked format. For example, in a cross-sectional study each participant would occupy a separate row. So, row 1 would contain person 1 from cluster 1. Row 2 would contain person 2 from cluster 1, etc. In a longitudinal study, each observation would occupy a separate row. So, row 1 would contain observation 1 from person 1. Row 2 would contain observation 2 from person 1, etc.

The macro variable ID_L2 specifies the name of the level-2 ID variable, or cluster ID variable. The level-2 ID variable should contain a unique value (numeric or character) for each cluster in the dataset. The numbers do not need to be in a particular order, and they do not need to be consecutive. For example, the numbers -1, 1.5, 3, 50, 200, and 201 would all be acceptable. Note that it is not necessary to specify an ID variable for level 1.

The parameter VAR specifies all of the variables (except for the ID variable) to be used in the imputation procedure. Note that all variables should be included here, regardless of whether they will be predictors or dependent variables in later analyses. This includes all complete and incomplete variables at both level 1 and level 2. The macro automatically identifies the level of each variable, and whether the variable is complete or incomplete. The macro outputs a list of the variables to the log, with information on the level of each variable ad whether the variable is complete.

The parameter RANDOM specifies the subset of complete level-1 variables to be used as random effects for imputing the incomplete level-1 variables. Adding a complete variable to RANDOM allows the variable's relationship with all of the incomplete level-1 variables to differ between clusters. It is necessary to add a variable to RANDOM under each of the following conditions:

1. The random effect of a complete level-1 variable will be used to predict one or more incomplete level-1 variables in a subsequent analysis.
2. The random effect of an incomplete level-1 variable will be used to predict a complete level-1 variable in a subsequent analysis.

In both of the above situations, the complete variable should be added to the RANDOM line.

Returning to our example data set, let's say we plan to estimate the fixed and random effects for the regression of end-of-year problem-solving test scores (PSTPSOLVE) on problem-solving pre-test scores (PREPSOLVE) because we expected that this association might vary across clusters. PREPSOLVE is a complete level-1 variable, whereas PSTPSOLVE is an incomplete level-1 variable. We would need to add PREPSOLVE to the RANDOM line (it should also be included on the VAR line). PSTPSOLVE should still be listed only on the VAR line.

As a second example, let's say we plan to estimate the fixed and random effects for the regression of PSTPSOLVE on standardized math scores (MATH). MATH is a complete level-1 variable, whereas PSTPSOLVE is an incomplete level-1 variable. Even though MATH will be a dependent variable in the subsequent analysis, it should still be included on the RANDOM line for the imputation (and also on the VAR line). PSTPSOLVE should be listed only on the VAR line.

As a final example, let's say that we plan to perform two follow up analyses: the regression of PSTPSOLVE on MATH, and the regression of self-efficacy ratings (SELFE) on MATH. In both cases, we wish to estimate the fixed and random effects. PSTPSOLVE and SELFE are incomplete level-1 variables, so should be listed only on the VAR line. Including MATH on the RANDOM line will allow its relationship with both incomplete variables to vary between clusters.

Though it is necessary to add variables to the RANDOM line if the analyst wishes to preserve random effects in the imputation process, this should be done with caution. Random effects are substantially more difficult for the macro to calculate. Recall that adding a variable to the RANDOM line allows the relationship between that variable and all of the incomplete level-1 variables to vary across clusters. This may cause convergence problems if:

1. The number of observations per cluster is small.
2. The number of incomplete variables is large.
3. The relationship between the variable(s) on the RANDOM line and the incomplete variable(s) does not vary across clusters (i.e., there is no random effect in the data).

Convergence problems arising from this issue can range from minor (e.g., the algorithm takes a few more iterations to converge) to extreme (e.g., the algorithm completely fails to converge). If the algorithm fails to converge, then it is advisable to remove a parameter from the RANDOM line.

Note that for VAR and RANDOM, variables can be included individually (e.g., x1 x2 x3) or as a list (e.g., x1-x3). If a list is used, the macro follows SAS conventions for determining what variables belong in the list. Specifically, a single dash denotes variables that with sequential names (e.g., x1-x4 includes variables x1 x2 x3 x4), whereas a double dash extracts variables in order of variable definition (e.g., x--a includes all variables from x to a, in order of definition). If no random variables are part of the imputation process, then the space to the right of the equal sign on associated line should be left blank (e.g., RANDOM = ,).

The macro variable NIMPUTE specifies the number of datasets to be imputed. The recommended as a minimum number of imputations (NIMPUTE) for single-level multiple imputation is 20 (Graham, Olchowski, & Gilreath, 2007). Because the number of imputations for multilevel imputation has not yet been systematically examined and because increasing the number of imputations generally leads to an increase in power for subsequent analyses, a larger number of imputations may be advisable (e.g., NIMPUTE = 50,). The primary downside to a large number of imputations is an increase in computational time.

The macro variable ITERATIONS specifies the number of iterations of the Gibbs Sampler to be performed between imputations and prior to the first imputation. Although we use 500 between-imputation iterations in the subsequent example, this number is unique to each data set and so should be determined by assessing convergence of the imputation procedure (see below). The macro variable OUT specifies the name of the output dataset. The imputations are stacked into a single file with a variable _IMPUTATION_ that identifies each data set. The macro variable SEED is used to designate a seed for random number generation in the macro. Setting the value of SEED is to zero results in different imputations for each run of the macro. Specifying a value other than zero will result in identical output across multiple runs of the macro (if the input is the same for each). The macro variable GRAPHS is used to request convergence plots of the parameters in the Gibbs Sampler. A value of "yes" requests that plots be printed, whereas "no" requests that plots be suppressed. Note that convergence plots should only be requested when assessing convergence, as they can increase the macro's memory usage and computational time.

EXAMPLE

Let's walk through an example of multiple imputation for our math problem solving data set. Recall that the analysis consists of end-of-year math problem-solving test scores (PSTPSOLVE) predicted by treatment (TX) and math self-efficacy (SELFE). The standardized math score (MATH) is not directly of interest, but is included in the model as a covariate (see Equation 1). The following syntax invokes the macro for the example data.

Example 2.

```
%mmi_impute(
    data          = example.exampledata,
    id_l2         = cluster,
    var           = pstpsolve selfe minority frlunch prepsolve math tx,
    random        = ,
    nimpute       = 1,
    iterations    = 5000,
    out           = example.imputed,
    seed          = 65406540,
    graphs        = yes);
```

We begin by specifying the name of the cluster variable (ID_L2 = CLUSTER). Recall from Equation 1 that we wish to perform an analysis predicting PSTPSOLVE from MATH, SELFE, and TX. To ensure that the effects of interest are preserved in the imputation process, all of the aforementioned variables must be included in the imputation model, although it is generally a good idea to include additional variables that are potential correlates of missingness or correlates of the analysis variables (we include two additional variables not in the analysis model). The variables in our analysis model (PSTPSOLVE , SELFE, MATH, and TX) as well as auxiliary variables (FRLUNCH, MINORITY, and PREPSOLVE) are included in VAR. FRLUNCH, MINORITY, and PREPSOLVE are not directly of interest (they won't be included in the subsequent analysis), but they help us to generate more accurate estimates in the imputation process.

Before generating imputed datasets to be analyzed, it is necessary to assess the convergence of the Gibbs sampler. This is done through generating one or more exploratory data augmentation chains and examining the resulting time-series plots. An exploratory data augmentation chain can be created by running the macro for a single imputation for a large number of iterations. A data augmentation chain of a few thousand iterations can be used to check for pathological trends that may last for 1000s of cycles. In this example, we generated a single imputation and 5000 iterations. If no pathological trends are present in the resulting convergence plots, the macro should be re-run for a smaller number of iterations (e.g., 500) to more accurately determine the number of iterations needed for convergence. When GRAPHS is set to "yes", the macro produces graphs of the following: the level-2 means (for incomplete level-2 variables and cluster-averaged incomplete level-1 variables), the level-2 variances (for incomplete level-2 variables and cluster-averaged incomplete level-1 variables), the diagonal elements of the random effects covariance matrix (labeled psi on the plot), and the residual variances (labeled sigma on the plot). Figure 1 contains the example time-series plot for 5000 iterations generated by the above code. Notice that the parameter appears to bounce around randomly without any long-term trends. The absence of a trend suggests rapid convergence to a stable distribution. Running the same syntax again with 500 between-imputation iterations (code not shown) allows the user to check for short-term trends. Though short-term trends appeared in the plot (not shown), they seemed to disappear in less than 500 iterations. We chose to use 500 iterations between each imputation, as it should be more than sufficient to remove any dependencies across iterations. See Schafer (1997) or Schafer and Olson (1998) for a more thorough description of convergence diagnostics and convergence problems.

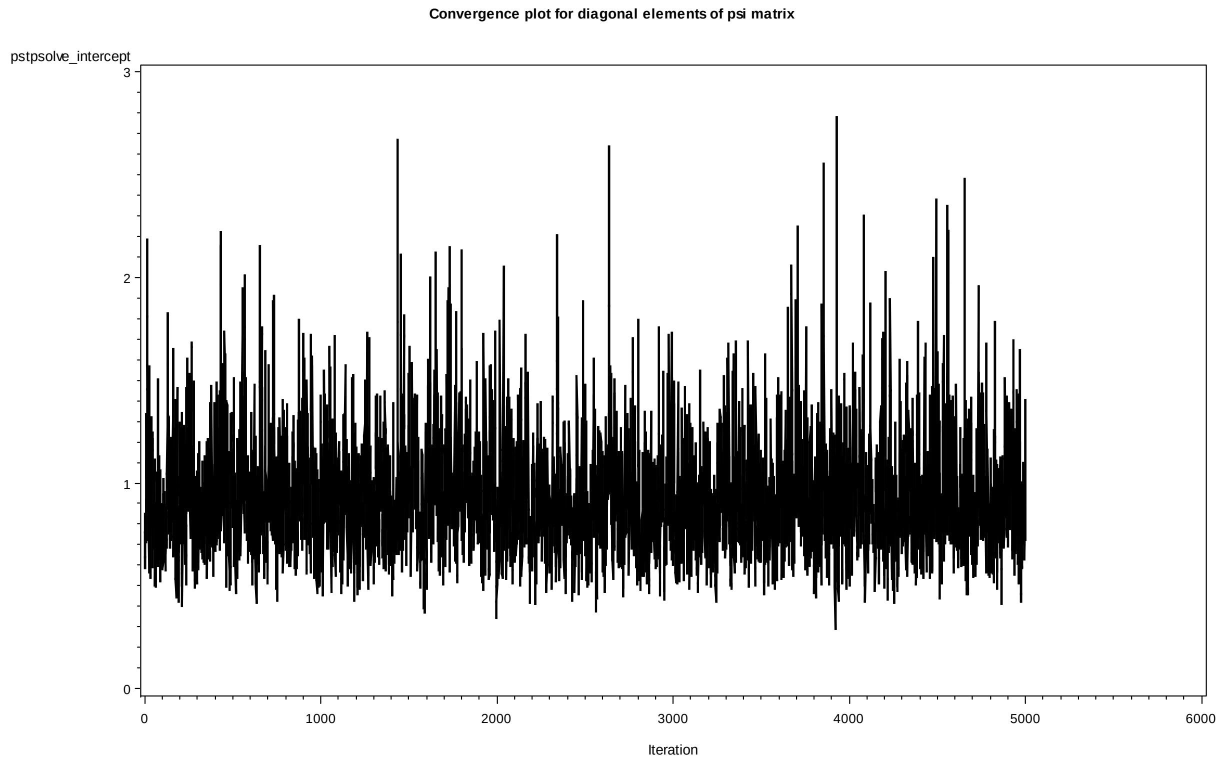

Figure 1. Example time-series plot for good convergence across 5000 iterations

After we have assessed the convergence of the algorithm, we run the same code but with convergence plots suppressed for a larger number of imputations and a smaller number of between-imputation iterations to generate the imputed datasets. In Example 3, we request 50 imputed datasets (NIMPUTE = 50) and 500 between-imputation iterations (ITER_BETWEEN = 500).

Example 3.

```
%mmi_impute(
    data          = example.exampledata,
    id_l2         = cluster,
    var           = pstpsolve selfe minority frlunch prepsolve math tx,
    random        = ,
    nimpute       = 50,
    iterations    = 500,
    out           = example.imputed,
    seed          = 2452455,
    graphs        = no);
```

The imputed data sets generated by the syntax in Example 3 are ready to be analyzed.

ANALYSIS & POOLING

After the missing values have been imputed, it is necessary to analyze the data and to pool the results. The imputed data sets can be analyzed with PROC MIXED and the subsequent results can be pooled using the MIANALYZE procedure. However, MIANALYZE does not automatically pool the random effects, which are often of interest in multilevel analyses. Although MIANALYZE can be tricked into pooling the random effects, this takes some extra work on the part of the user. MIANALYZE also does not include functionality for pooling likelihood ratio tests. This is problematic in the context of multilevel modeling, as likelihood ratio tests are the preferred method for testing random effects (Singer & Willett, 2003). To address these problems, I created a second macro that analyzes multilevel imputed data in PROC MIXED, pools the results, and implements a pooled likelihood ratio test. This macro is described in a second SAS Global Forum paper (Mistler, 2013). I recommend using the multilevel imputation and pooling macros together.

CONCLUSION

I developed a two-level multiple imputation macro for SAS. This paper demonstrates the use of the macro in the context of a two-level multilevel model. The macro presented in this paper is available at http://www.mistlerconsulting.com/software.html. I plan to update the macro to increase user-friendliness and add greater functionality. I welcome any comments and suggestions that you may have for updates to the macro.

REFERENCES

- Aiken, L.S., & West, S.G. (1991). *Multiple regression: Testing and interpreting interactio*ns. Newbury Park, CA: Sage.
- Allison, P.D. (2005). *Imputation of categorical variables with PROC MI.* Paper presented at the SAS Users Group International, 30th Meeting (SUGI 30).
- Enders, C.K. (2010). *Applied missing data analysis*. New York: The Guilford Press.
- Graham, J.W., Olchowski, A.E., & Gilreath, T.D. (2007). How many imputations are really needed? Some practical clarifications of multiple imputation theory. *Prevention Science*, 8, 206-213.
- Little, R.J.A. (1992). Regression with missing X's: a review. *Journal of the American Statistical Association*, 87(420), 1227-1237.
- Mistler, S.A. (2013). *A SAS macro for computing pooled likelihood ratio tests with multiply imputed data*. in Proceedings of the SAS Global Forum 2013, San Francisco, California: Contributed Paper (Statistics and Data Analysis) 440-2013.
- Montague, M., Enders, C., & Dietz, S. (2011). Effects of cognitive strategy instruction on math problem solving of middle school students with learning disabilities. *Learning Disability Quarterly*, 34(4), 262-272.
- Raudenbush, S.W., & Bryk, A.S. (2002). *Hierarchical linear models: Applications and data analysis methods* (2nd Ed.). Thousand Oaks, CA: Sage Publications.
- Rubin, D.B. (1987). *Multiple imputation for nonresponse in surveys*. Hoboken, NJ: Wiley.
- Schafer, J.L. (1997). *Analysis of incomplete multivariate data*. Boca Raton, FL: Chapman & Hall.
- Schafer, J.L. (2001). Multiple imputation with PAN. In A.G. Sayer & L.M. Collins (Eds.), *New methods for the analysis of change* (pp. 355-377). Washington, DC: American Psychological Association.
- Schafer, J.L. & Graham, J.W. (2002). Missing data: Our view of the state of the art. *Psychological Methods*, 7, 147-177.
- Schafer, J.L., & Olsen, M.K. (1998). Multiple imputation for multivariate missing-data problems: A data analyst's perspective. *Multivariate Behavioral Research*, 33, 545-571.
- Schafer, J.L., & Yucel, R.M. (2002). Computational strategies for multivariate linear mixed-effects models with missing data. *Journal of Computational and Graphical Statistics*, *11*, 437-457.
- Singer, J.D., & Willett, J.B. (2003). *Applied longitudinal data analysis: Modeling change and event occurrence*. Oxford, NY: Oxford University Press.
- Tanner, M.A., & Wong, W.H. (1987). The calculation of posterior distributions by data augmentation. *Journal of the American Statistical Association*, 82, 528-540.
- van Buuren, S. (2011). Multiple imputation of multilevel data. In J. K. Roberts & J. J. Hox (Eds.), *The Handbook of Advanced Multilevel Analysis* (pp. 173-196). New York: Routledge.
- van Buuren, S. (2012). *Flexible imputation of missing data*. New York: Chapman & Hall.
- Yucel, R.M. (2008). Multiple imputation inference for multivariate multilevel continuous data with ignorable non-response. *Philosophical Transactions of the Royal Society, Series A*, Volume 366, No 1874, 2389-2403.

ACKNOWLEDGMENTS

I would like to thank my SAS mentor, David Pasta, for his careful review and valuable comments. I would also like to thank my advisor, Dr. Craig Enders, for his insights into missing data theory.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Stephen Mistler
E-mail: stephen.mistler@asu.edu
Web: www.mistlerconsulting.com