$\mathfrak{A}melia$: A Program for Missing Data

James Honaker, Gary King, and Matthew Blackwell

June 22, 2006

Contents

1 Introduction

This manual presents a quick introduction to the $\mathfrak{A}melia$ multiple imputation package. Our software remedies the discrepancy between the way social scientists analyze data with missing values and the recommendations of the statistics community. With a few notable exceptions, statisticians and methodologists have agreed on a widely applicable approach to many missing data problems based on the concept of "multiple imputation," yet most social scientists still use listwise deletion (deleting all cases with at least one missing cell) to make inferences in the presence of missing data. This practice is always inefficient and often biased. Unfortunately, many of the existing multiple imputation methods are unfit for handling the common problems of social science data and can be difficult to use even for experts. $\mathfrak{A}melia$ combines a fast, efficient algorithm for multiple imputation with a set of features designed for social science data.

2 What $\mathfrak{A}melia$ Does

Multiple imputation involves imputing m values for each missing cell in your data matrix and creating m "completed" data sets. (Across these completed data sets, the observed values are the same, but the missing values are filled in with different imputations that reflect the uncertainty about the missing data.) After imputation with our algorithm, you can apply whatever statistical method you would have used if there had been no missing values to each of the m data sets, and use a simple procedure, described in the next paragraph, to combine the results. (If you use the Stata package for statistical analysis, you may be interested in our MI procedures, or the CLARIFY package, both of which can combine the results automatically.) Under normal circumstances, you only need to impute once and can then analyze the m imputed data sets as many times and for as many purposes as you wish. The advantage of $\mathfrak{A}melia$ is that it combines the comparative speed and ease-of-use of our algorithm with the power of multiple imputation, to let you focus on your substantive research questions rather than spending time developing complex application-specific models for nonresponse in each new data set. Unless the rate of missingness is exceptionally high, $m = 5$ (the program default) is probably adequate.

In order to combine the results across m data sets, first decide on the

quantity of interest to compute, such as univariate mean, regression coefficient, predicted probability, or first difference. The multiple imputation estimate of this parameter, $\bar{q}$, is the average of the m separate estimates, q_j $(j = 1, ..., m)$:

$$\bar{q} = \frac{1}{m} \sum_{j=1}^{m} q_j. \tag{1}$$

The variance of the point estimate is the average of the estimated variances from *within* each completed data set, plus the sample variance in the point estimates *across* the data sets (multiplied by a factor that corrects for the bias because $m < \infty$). Let $SE(q_j)^2$ denote the estimated variance (squared standard error) of q_j from the data set j, and $S_q^2 = \Sigma_{j=1}^{m}(q_j - \bar{q})^2/(m-1)$ be the sample variance across the m point estimates. The standard error of the multiple imputation point estimate is the square root of

$$SE(q)^2 = \frac{1}{m} \sum_{j=1}^{m} SE(q_j)^2 + S_j^2(1 + 1/m). \tag{2}$$

If, instead of point estimates and standard errors, simulations of q are desired (as would be used to compute quantities of interest directly; see King, Tomz, and Wittenberg 2000), use each completed data set to create $1/m$ of the needed number of simulations and then combine them all into one set of simulations.

Users should see especially Pp. 57-58 of our article for a variety of practical suggestions in making imputations, such as what variables to include in the imputation stage, how to keep imputations within logically possible ranges, etc.

3 Versions of $\mathfrak{A}melia$

Two versions of $\mathfrak{A}melia$ are available, each with its own advantages and drawbacks. First, $\mathfrak{A}melia$ exists as a package, or collection of functions, for the R statistical software package. Users can utilize their knowledge of the R language to run $\mathfrak{A}melia$ at the command line or to create scripts that will run $\mathfrak{A}melia$ and preserve the commands for future use. Alternatively, AmeliaView, an interactive Graphical User Interface (GUI), allows users to set options and run $\mathfrak{A}melia$ without any knowledge of the R programming language. AmeliaView enables users to set all of the $\mathfrak{A}melia$ options and,

thus, empowers those with limited or no coding experience to create expert level imputations.

Both versions of *Amelia* are available on the Windows and Linux platforms and *Amelia* for R runs in any environment that R can. All versions of *Amelia* require the R software, which is freely available at `http://www.r-project.org/`.

4 Installation and Updates

Before installing *Amelia*, you must have installed R version 2.1.0 or higher, which is freely available at `http://www.r-project.org/`. *Amelia* cannot run without R properly installed.

4.1 Windows - AmeliaView

To install AmeliaView in the Windows environment, simply download the installer `setup.exe` from `http://gking.harvard.edu/stats.shtml` and run it. The installer will ask you to choose a location to install *Amelia*. If you have installed *R* with the default options, *Amelia* will automatically find the location of R. If the installer cannot find R, it will ask you to locate the directory of the most current version of R. Make sure you choose the directory name that includes the version number of R (e.g. C:/Program Files/R/R-2.2.0) and contains a subdirectory named `bin`. The installer will also put shortcuts on your Desktop and Start Menu.

4.2 Windows - *Amelia* for R

Users familiar with the *R* language, and who intend to use *Amelia* primarily as a function within other *R* code can install all the Amelia functions as one would install any other Amelia library. There is a .zip package install, which can be added to the library path. Additionally the functions are provided in a set of five source files which can be sourced into any *R* code directly.

Even users familiar with the *R* language may find it useful to utilize AmeliaView to set options on variables, change arguments, or run diagnostics. From the command line, AmeliaView can be brought up with the call:

```
> ameliagui()
```

4.3 Linux

To install $\mathfrak{A}melia$ in a Linux OS, you must install the $\mathfrak{A}melia$ library into your version of R. This is true even if users only wish to use AmeliaView. Download the package from `http://gking.harvard.edu`. Then, in the same directory as the file, at the Linux/Unix command line type

```
> R CMD INSTALL --library=.R/library amelia_1.0.tar.gz .
```

If you do not have access to the root, you can install the package locally. Create a directory (i.e. myrlibrary) to be the local storage space for R packages. Once this directory is created you can install the package to that local library:

```
> R CMD INSTALL --library=~/myrlibrary amelia_1.0.tar.gz .
```

Once this is complete you need to edit or create your R profile. Locate or create `/.Rprofile` in your home directory and add this line:

```
.libPath(``~/myrlibrary'')
```

This will add your local library to the list of library paths that R searches in when you load libraries.

Linux users can use AmeliaView in the same way as Windows users of $\mathfrak{A}melia$ for R. From the command line, AmeliaView can be brought up with the call:

```
> ameliagui()
```

5 Program Overview

5.1 AmeliaView

Built around the core of the R library, AmeliaView has a graphic interface for users to input data, set options, create the imputed datasets, and run diagnostics. The structure of the program moves logically through these steps. In Windows, this can be accessed from the desktop shortcut created

by the installer. In Linux, accessing the GUI requires calling a function from within R[1]. Once AmeliaView is open, no further use of R is required.

The main window of AmeliaView is divided into three steps. In Step 1, you indicate the location of your data and load it into the program. In Step 2, you can set options about your data by using the Options dialogs. In Step 3, you can set options about the output of $\mathfrak{A}melia$ and execute the $\mathfrak{A}melia$ code. You have the option to save your output data files for further use.

In most simple applications, all you will need to do is load an input data file, set any options you desire, and hit the "Run Amelia" button. For example:

1. Specify the type of data file you wish to input by using the "Input Data Type" drop-down menu. Next, use the "Browse..." button to find the location of your data file. Once this is specified, you can hit the "Load Data" button to load the data. If your data loads correctly, the status bar at the bottom of the program will show the filename, number of rows and number of columns. At this point you can use the "Summarize Data" button to view summary statistics about the variables along with histogram plots of them.

2. In the Options step, you can specify the time series and cross sectional variables in your data set (if any) by using the appropriate drop-down menus. Each of the "Variables", "TSCS", and "Priors" buttons open a separate dialog box in which you can set options in each of these categories.

3. In the Output step, you can specify what file type (if any) in which you would like to save your output data. You can also set the name of the output data and the number of data sets you would like. You can now run $\mathfrak{A}melia$ by pressing the "Run Amelia" button. A dialog will open that tracks the progress of $\mathfrak{A}melia$ and will let you know when it has finished. Once this is complete, you can either use the diagnostics or close $\mathfrak{A}melia$.

[1] AmeliaView can be called from within R in Windows, as well, using the `ameliagui()` function.

5.2 $\mathfrak{A}melia$ for R

Using $\mathfrak{A}melia$ in *R* you will first have to load the library into R using the `library(amelia)` command. Once the package is loaded, you can set options using the arguments of the `amelia()` function. The output data will be either a list of m data frames or matrices depending on the your input data. Please refer to the end of this manual for detailed documentation on adjusting the optional arguments from their default values.

6 Data Input and Output

6.1 AmeliaView

AmeliaView uses the "foreign" package in R to import and export various types of files. In the Input box, you can choose the file type you wish to use from the group of supported file types: Comma Separated Values (.csv), Tab-Delimited (.txt), Stata (.dta), SPSS (.dat), and SAS Transport (.xport). Once this is set, you can proceed to locate your file in one of two ways. You can type the location of the file in the "Input Data File" entry and press the "Load Data" button. Or you can locate your file by using the "Browse..." button to select the file of your choice. Once you have entered the file name, press the "Load Data" button to load the file into $\mathfrak{A}melia$.

When you have loaded the data, you can use the "Summarize Data" button to see the data. This includes seeing the minimum and maximum values along with the mean and standard deviation. Another feature is the ability to view a plot of the histogram of each individual variable. This can help you get a graphical sense of the observed values in your dataset.

Fewer options exist for the output data files due to the limitations of the "foreign" package. You can only save the imputed datasets as either a CSV, Tab-Delimited, or Stata file. However, most statistical packages will allow you to read in datasets as either CSV or Tab-delimited. The names of the files will be the name you specify, plus a number appended at the end to distinguish between successive imputed datasets. For example, if you set the name to be "mydata" and the output file type to be CSV, your files would be:

```
mydata1.csv
mydata2.csv
```

...

There will be as many datasets as you indicate in the output options box.

Another output file that is produced when running *Amelia* is the Replication Archive. This file can be used to reproduce the same set of options you ran *Amelia* with so that you or anyone else can replicate the same findings. This file automatically saves to "amarchive.r". This file can also serve as a session save that can be opened later to load the same options that you ran *Amelia* under. This allows you to pick up where you left off in the next run of *Amelia*, adjust options and rerun the imputations, or return at a later point and use the diagnostic routines.

6.2 *Amelia* for R

Data input in the command-line version of *Amelia* for R is identical to any data input in R. You must have your data in either a text format such as Comma Separated Values (.csv) or Tab-Delimited (.txt) and then read them into R (generally this would involve the functions `read.csv` or `read.table`, respectively). You can read about the specifics of how to load data into R in the R documentation. Of course your data may be generated by code and amelia called as a function later in that code.

Besides various commercial packages to transfer your data between formats, a viable option is using the `foreign` package. This package can greatly expand the number of different formats that R can input, including Stata, SPSS, and SAS transport.

Whichever way you get your data into R, when passing it to the `amelia` function, it can either be in the form of a data frame or matrix. After *Amelia* has run, you will get datasets returned in the same type as the format of the dataset given to the amelia function. Once you have these datasets, you can manipulate them in R or save them to files using the R function `write`.

For example, a simple session or code fragment might be:

```
> library(amelia)
> x<-read.table("mydata.csv")
> empri<-10
> amords<-c(3,4)
> output<-amelia(data=x,empri=empri,amords=amords)
> save(output,file="output.rData")
```

where `amords` and `empri` are options detailed later in this manual.

7 Options

There are a variety of options in $\mathfrak{A}melia$ that customize the imputation model to handle problems that are common in social science data. In AmeliaView, these options are set using the dialog boxes "Variables," "TSCS," and "Priors." At these dialogs, the user can visually inspect and set each option. Please refer to the Menu Guide below for details on the content of the dialogs.

In $\mathfrak{A}melia$ for R, these options must be set on the command line. From within R, users can set the transformations by including a vector of column numbers or names (if column names exist), that $\mathfrak{A}melia$ should transform. For example,

```
> amelia(mydata, logs=c(2,3,7))
```

or

```
> amelia(mydata, noms=c(‘‘gdp’’,‘‘population’’)).
```

7.1 Transformations of Variables

Social science data commonly includes variables that fail to fit into a multivariate normal distribution. Indeed, numerous model have been introduced specifically to deal with the problems they present. As it turns out, much evidence in the literature (discussed in our paper) indicates that the multivariate normal model used in $\mathfrak{A}melia$ usually works well for the imputation stage even when discrete or non-normal variables are included and when the analysis stage involves these limited dependent variable models. Nevertheless, $\mathfrak{A}melia$ includes some limited capacity to deal directly with ordinal and nominal variables and to variables that require other transformations. In general nominal and log transform variables must be declared to $\mathfrak{A}melia$, whereas ordinal (including dichotomous) variables often need not be, as described below. (For harder cases, see Schafer, 1997, for specialized MCMC-based imputation models for discrete variables.)

Although these transformations are taken internally on these variables to better fit the data to the multivariate normal assumptions of the imputation

model, all the imputations that are created will be returned in the original untransformed form of the data. The fully imputed datasets that are returned will always be in the form of the original data that is passed to the amelia routine.

7.1.1 Ordinal

In much statistical research, researchers treat independent ordinal (including dichotomous) variables as if they were really continuous. If the analysis model to be employed is of this type, then nothing extra is required of the of the imputation model. Users are advised to allow $\mathfrak{A}melia$ to impute non-integer values for any missing data, and to use these non-integer values in their analysis. Sometimes this makes sense, and sometimes this defies intuition. One particular imputation of 2.35 for a missing value on a seven point scale carries the intuition that the respondent is between a 2 and a 3 and most probably would have responded 2 had the data been observed. This is easier to accept than an imputation of 0.79 for a dichotomous variable where a zero represents a male and a one represents a female respondent. However, in both cases the non-integer imputations carry more information about the underlying distribution than would be carried if we were to force the imputations to be integers. Thus whenever the analysis model permits, missing ordinal observations should be allowed to take on continuously valued imputations.

Often, however, analysis models require some variables to be strictly ordinal, as for example the dependent variable must be in a logistical regression. Imputations for variables set as ordinal are created by taking the continuously valued imputation and using an appropriately scaled version of this as the probability of success in a binomial distribution. The draw from this binomial distribution is then translated back into one of the ordinal categories.

7.1.2 Nominal

Nominal variables (other than dichotomous) must be treated quite differently than ordinal variables. Any multinomial variables in the data set (such as religion coded 1 for Catholic, 2 for Jewish, and 3 for Protestant) must be specified to $\mathfrak{A}melia$.

For a p-category multinomial variable, $\mathfrak{A}melia$ will find p (as long as your data contain at least one value in each category), and substitute $p-1$ binary

variables to specify each possible category. These new $p-1$ variables will be treated as the other variables in the multivariate normal imputation method chosen, and receive continuous imputations. These continuously valued imputations will then be appropriately scaled into probabilities for each of the p possible categories, and one of these categories will be drawn, where upon the original p-category multinomial variable will be reconstructed and returned to the user. Thus all imputations will be appropriately multinomial.

Since $\mathfrak{A}melia$ properly treats a p-category multinomial variable as $p-1$ variables, one should understand the number of parameters that are quickly accumulating if many multinomial variables are being used. If the square of the number of real and constructed variables is large relative to the number of observations, the user is recommended to implement a ridge prior distribution on the parameter space.

7.1.3 Natural Log

If one of your variables is heavily skewed or has outliers that may alter the imputation in an unwanted way, you can use a natural logarithm transformation of that variable in order to normalize its distribution. This transformed distribution help $\mathfrak{A}melia$ to avoid imputing values that depend too heavily on outlying data points. Log transformations are common in expenditure and economic variables where we have strong beliefs that the marginal relationship between two variables decreases as we move across the range.

7.1.4 Square Root

Event count data is often heavily skewed and has nonlinear relationships with other variables. One common transformation to tailor the linear model to count data is to take the square roots of the counts. This is a transformation that can be set as an option in amelia.

7.1.5 Logistic

Proportional data is sharply bounded between 0 and 1. A logistic transformation is one possible option in amelia to make the distribution symmetric and relatively unbounded.

7.2 Identification Variables

Datasets often contain identification variables, such as country names, respondent numbers, or other id numbers, codes or abbreviations. Sometimes these are text and sometimes these are numeric. Often it is not appropriate to include these variables in the imputation model, but it is useful to have them remain in the imputed datasets (However, there are models that would include the ID variables in the imputation model, such as fixed effects model for data with repeated observations of the same countries). Identification variables which are not to be included in the imputation model can be identified with the argument `idvars`. These variables will not be used in the imputation model, but will be kept in the imputed datasets.

In order to conserve memory, it is wise to remove unnecessary variables from a data set before loading it into $\mathfrak{A}melia$. The only variables you should include in your data when running $\mathfrak{A}melia$ are variables you will use in the analysis stage and those variables that will help in the imputation model. While it may be tempting to simply mark unneeded variables as IDs, it only serves to waste memory and slow down the imputation procedure.

7.3 Ridge Priors for High Missingness, Small n's, or Large Correlations

When the data to be analyzed contain a high degree of missingness or very strong correlations among the variables, or when the number of observations is only slightly greater than $p(p+3)/2$ (where p is the number of variables), results from your analysis model will be more dependent on the choice of imputation model. This suggests more testing in these cases of alternative specifications under $\mathfrak{A}melia$.

In addition, in these circumstances, we recommend adding a ridge prior which will help with numerical stability by shrinking the covariances among the variables toward zero without changing the means or variances. The ridge prior can be implemented by setting the option `empri` in R and the Ridge Prior entry under the "Priors" button in the standalone. Including this prior as a positive number is roughly equivalent to adding `empri` artificial observations to the data set with the same means and variances as the existing data but with zero covariances. Thus, increasing the `empri` setting results in more shrinkage of the covariances thus putting more a priori structure on the estimation problem. In general, keep the value on this prior relatively

small and increase it only when necessary.

7.4 Priors of Relationships Among Cases

8 Diagnostics

Amelia currently provides one three diagnostic tools to inspect the imputations that are created. These routines `compare`, `overimpute` and `overdisperse` graphically investigate the distribution of the imputations, the fit of the imputation model, and modality of the Likelihood space optimized by the EM chain, respectively. We are currently developing other diagnostic routines and tests.

8.1 Compare

In the diagnostic window of AmeliaView, the compare function will, for a given variable, generate a plot of the relative frequencies of the observed data with an overlay of the relative frequency of the imputed values. The imputed curve plots the density of the *mean* imputation over the m datasets. That is, for each cell that is missing in the variable the diagnostic will find the mean of the that cell in each of the m datasets and use that value for the density plot. These graphs will allow you to inspect how the density of imputations compares to the density of observed data. Some discussion of these graphs can be found in Gelman et al (2005). Minimally, these graphs can be used to check that the mean imputation falls within known bounds, when such bounds exist in certain variables or settings.

Figure 1: Two examples of the compare diagnostic graph. In the left graph all the imputations, the density of which is shown in black, are contained within the range of the observed data for this variable (in red). In the right graph some of the mean imputed values are larger than the largest observed value. This may be reasonable (if for example, high income respondents to a survey do not report their income) or may be a warning to change the imputation model, if for example the data has strong bounds and these imputations fall well outside these bounds.

This graph can also be called from *R* as: