Paper SP10

A SAS® Macro for Single Imputation

Shuping Zhang, Jane Liao and Xingshu Zhu
Merck & Co Inc., Upper Gwynedd, PA. 19454-2505

ABSTRACT
Single imputation is often used to replace the missing value of a variable in a dataset because this approach is both simple and efficient. Single imputation is particularly useful when working with an extensive dataset containing millions of records and a large number of variables. In this case, it would be highly impractical, or maybe even impossible, to use the multiple imputation method, performed by the procedure PROC MI *[ref1]*, because of the overwhelmingly large number of datasets that would have to be created. In this paper, we introduce a simple SAS macro that allows the user to create a "complete" SAS dataset through single imputation by selecting different statistical methods and applying them to data on a given patient or to information from the entire dataset.

KEY WORDS: Missing data, Mean, Multiple Imputation, data imputation methods, RANNOR

INTRODUCTION
One particularly challenging task for researchers working in pharmaceutical companies is to choose the most appropriate method for handling missing data. This challenge is especially familiar to health economists and epidemiologists working with social science research data from surveys and other questionnaires, where missing values occur all the time. Proper handling of these missing values is crucial to a successful analysis.

Often times, missing values are just ignored for reasons of convenience, which might be acceptable when working with a large dataset and a relatively small amount of missing data. However, this simple treatment can yield biased findings if the percentage of missing data is relatively large, resulting in lost information on the incomplete cases. Moreover, when dealing with relatively small datasets, it becomes impractical to just ignore missing values or to delete incomplete observations from the dataset. In these situations, more reliable imputation methods must be pursued in order to perform meaningful analyses.

Though there are different methods for handling missing data, one common approach is single imputation, which substitutes a missing value with a definite value, following an established procedure. For example, each missing value can be imputed with the variable mean of the complete cases or imputed by carrying forward or backward the non-missing values in an incomplete case. After such substitution, standard statistical methods for complete data analysis can then be used on the full data set.

Instead of providing a single value for each missing value, the multiple imputation method *[ref2]* makes multiple predictions. The predictions are made for each of the missing values within a variable by using the existing values from other variables to produce multiple imputed datasets. These datasets are then able to provide plausible representations of the missing data. One attractive feature of multiple imputation is its ability to introduce appropriate random error into the imputation process. This feature makes it possible to get practically unbiased estimates of all parameters and standard errors.

Unlike the multiple imputation method, however, single imputation generally does not reflect the uncertainty of the predictions about the unknown missing values. Nonetheless, the single imputation method is still used widely because of its simplicity and efficiency. Since multiple imputation creates multiple datasets requiring more advanced statistical analysis, it is difficult for users with insufficient statistical knowledge to implement and utilize this method. In fact, in some cases, the output from simple and multiple imputation is not substantially different, making it unnecessary to use the additional resources required by multiple imputation. Moreover, the SAS procedure used in multiple imputation, PROC MI, only works well for samples of small to medium-sized datasets *[ref3]*. As researchers deal with

large numbers of variables or millions of records, as in a typical epidemiology study, PROC MI can be slow or even incapable of imputing the data at all. In response to the needs for the single imputation method, and as an alternative to PROC MI, we have developed a SAS macro for single imputation, called %SingleImpute, that provides an easy and quick approach for dealing with missing data. In this approach, the missing values are substituted using one of ten statistical methods, resulting in the creation of a complete SAS dataset for further analysis.

THE STRUCTURE OF THE SAS MACRO

The macro %SingleImpute contains the following five parameters:

- the name of the input dataset;
- the name of the output dataset;
- a pair of measurable variables specifying the name of the variable that needs to be imputed and the imputation method;
- the name of the variable denoting the imputation range.

If the parameter for the imputation range, ByVar, is left blank, the imputation method is applied to the information from the entire sample population. Otherwise, the imputation is performed on the patient or visit level specified by ByVar. In addition, there are ten imputation methods included in this macro that can be denoted in the parameter "mmPairs." Users need to specify all of these parameters (with the possible exception of ByVar), which are listed in the macro call below, in order for a complete dataset to be created.

```
%macro SingleImpute(
 inds    =    /* name of input SAS data set                          */
,mmPairs = /* pair of Measure-Variable/Impute-Method                */
,ByVar   =   /* variable name denotes the imputation range          */
,Visit   =    /* variable name denotes visit within parameter &ByVar */
,outds   =    /* name of output SAS data set                         */
);
```

IMPUTATION METHODS

Consider listing the 10 different imputation methods included within %SingleImpute.
In this paper, we use a plasma concentration dataset, called "sample," to illustrate how the SAS system's rich tools offer different approaches for imputing missing data. For demonstration purposes, the variable that we use, called "value," has missing values from two patients with ten visits each, as displayed in the following table.

Plasma Concentration Dataset "sample"

Time (hours)	Original Data Value	
	Patient (1)	Patient (2)
0	98	.
1	50	83
2	50	61
3	.	.
4	26	61
5	.	45
6	.	21
7	26	11
8	12	5
9	6	.

Before applying an imputation method to this dataset, we copy the input dataset to a temporary dataset ("_temp_") and rename the imputing variable (in our case, "value") by adding an underscore before and after the variable name ("_value_"). Once the missing values are imputed with a selected method, the variable name will be changed back to it original name ("value").

```
data _temp_;
   set sample (rename=(value=_value_));
run;
```

The remainder of this section discusses in detail the ten imputation methods included in the macro %SingleImpute. These methods are categorized according to their imputation functionalities into four different groups.

1.) Replacing missing values with mean, minimum or maximum values.

Assuming there are multiple visits per patient, and we would like to replace the missing values with the mean of the non-missing values from the data on a given patient, the following SQL code can be used:

```
PROC SQL noprint;
 create table sample as
 select *
      ,CASE _value_
          when . then MEAN(_value_) else _value_
       end as value
  from _temp_
GROUP BY patient;
Quit;
```

The original dataset and the "complete" dataset are shown side by side for comparison in the table below, with the imputed values in bold.

Time (hours)	Original Data		'Complete' Data					
	Value		**MEAN**		**MIN**		**MAX**	
	Patient (1)	Patient (2)	Patient (1)	Patient (2)	Patient (1)	Patient (2)	Patient (1)	Patient (2)
0	98	.	98	**41.0**	98	**5**	98	**83**
1	50	83	50	83	50	83	50	83
2	50	61	50	61	50	61	50	61
3	.	.	**38.3**	**41.0**	**6**	**5**	**98**	**83**
4	26	61	26	61	26	61	26	61
5	.	45	**38.3**	45	**6**	45	**98**	45
6	.	21	**38.3**	21	**6**	21	**98**	21
7	26	11	26	11	26	11	26	11
8	12	5	12	5	12	5	12	5
9	6	.	6	**41.0**	6	**5**	6	**83**

The function MEAN in the code above could also be replaced by the functions MIN or MAX. Using the MIN function in our example, the imputed values 38.3 and 41.0 would be replaced by 6 and 5, which are the minimum values for patients (1) and (2), respectively. With the MAX function, the imputed values would be 98 and 83, representing the maximum values for these two patients.

2.) Replacing missing values with the mean, minimum or maximum of the most frequently appearing values.

Depending on the nature of the study, we are sometimes interested in the most or least frequently appearing values. With the macro %SingleImpute, we can replace missing values with the mean, minimum or maximum of the most commonly appearing values in the data for a given patient in our sample dataset. If there is only one most frequently appearing value for a patient, then this value itself is actually the mean, minimum and maximum of the most frequently appearing value for that patient. If there are equal numbers of the most commonly appearing values, then the average of

these values is the mean of the most frequently appearing value. The following three steps show the partial SAS codes used in the macro to determine the mean of the most frequently appearing values:

a. Calculate the frequency of each value appearing in the data for a given patient.

```
proc sql noprint;
  create table FreqVals as
  select patient, freq(_value_) as frq, _value_
    from _temp_
   where _value_ is not null
group by patient, _value_;
```

b. Determine the mean of the most frequently appearing values for that patient.

```
  create table Target as
  select patient, MEAN(_value_) as _value_
    from (select distinct patient, _value_
            from FreqVals
        group by patient
          having frq eq max(frq) )
GROUP BY patient
order by patient;
```

c. Replace the missing values for that patient by selecting the imputed values calculated in Step b.

```
  create table sample as
  select e.*
       ,case e._value_
            when . then f._value_ else e._value_
        end as value
    from _temp_ as e left join Target as f
      on f.patient eq e.patient
order by patient, time;
quit;
```

The table below illustrates the most frequently appearing values for each of two patients, as calculated in step a.

Patient	_value_	Frequency of appearance
1	**26**	**2**
	50	**2**
	6	1
	12	1
	98	1
2	**61**	**2**
	5	1
	11	1
	21	1
	45	1
	83	1

The values that appear most often for patient (1) are 26 and 50 because they both appear twice. Consequently, the average of these two values is used for the mean replacement. As for patient (2), the only value that appears most often is 61. Therefore, the mean of the most frequently appearing value is 61, which is then used to replace the missing values for patient(2). In the MEAN column of the "complete" dataset shown below, the missing values are filled in with the mean of the most frequently appearing values for each patient.

Time (hours)	Original Data		'*Complete*' Data					
	Value		**MEAN**		**MIN**		**MAX**	
	Patient (1)	Patient (2)	Patient (1)	Patient (2)	Patient (1)	Patient (2)	Patient (1)	Patient (2)

0	98	.	98	**61**	98	**61**	98	**61**
1	50	83	50	83	50	83	50	83
2	50	61	50	61	50	61	50	61
3	.	.	**38**	**61**	**26**	**61**	**50**	**61**
4	26	61	26	61	26	61	26	61
5	.	45	**38**	45	**26**	45	**50**	45
6	.	21	**38**	21	**26**	21	**50**	21
7	26	11	26	11	26	11	26	11
8	12	5	12	5	12	5	12	5
9	6	.	6	**61**	6	**61**	6	**61**

Similarly, we can also use the functions of MIN or MAX in step (b) of the SAS code above to impute the missing values with the minimum or maximum of the most frequently appearing values. For the minimum, the imputed values of 38 and 61 would be replaced with 26 and 61, respectively, as indicated in the MIN column in the table above. For the maximum of the most frequently appearing values, the imputed values would be 50 and 61, respectively, as in the MAX column above.

3.) Replacing missing values by carrying forward or backward, or by averaging the values adjacent to missing data.

Instead of replacing missing values with summary statistics of a dataset, using the MEAN, MIN, or MAX functions, we can also replace missing data with adjacent non-missing values. For example, a missing value can be imputed by applying the carry-forward or carry-backward methods to a patient's information. Another approach is to replace a patient's missing data with the average of the two values that are adjacent to the missing data. If there are consecutive missing values, then the carry-forward method uses the next preceding non-missing value in the dataset; the carry-backward method uses the next succeeding non-missing value in the dataset; and the averaging method uses a combination of both.

When replacing missing values in a patient's data with the carry-forward method, the macro %SingleImpute uses the following SAS code:

```
proc sort data=_temp_;
    by patient DESCENDING time;
data _temp_;
    set _temp_;
          by patient DESCENDING time;
          retain BackWard;
          if first.patient then BackWard=.;
          if _value_ ne . then BackWard=_value_;
run;

proc sort data=_temp_;
    by patient time;
data _temp_;
    set _temp_;
           by patient time;
           retain ForWard;
           if first.patient then ForWard=.;
           if _value_ ne . then ForWard=_value_;
run;

data sample (drop=ForWard BackWard);
    set _temp_;
          if _value_ ne . then value=_value_;
          else value=(ForWard);
run;
```

In the last data step of the above code, a user can replace the formula "value=ForWard" with "value=BackWard" for the carry-backward method. It is also possible to change the formula in this last step of the code to "value=(ForWard+BackWard)/2." This formula replaces the missing values with the average of the adjacent non-missing values.

Again, using the plasma dataset as an example, the table below demonstrates how the carry-forward, carry-backward, and averaging methods can be used to construct a more complete dataset.

Time (hours)	Original Data _Value_		'*Complete*' Data					
			ForWard		**BackWard**		**Averaging**	
	Patient (1)	Patient (2)	Patient (1)	Patient (2)	Patient (1)	Patient (2)	Patient (1)	Patient (2)
0	98	.	98	**.**	98	**83**	98	**.**
1	50	83	50	83	50	83	50	83
2	50	61	50	61	50	61	50	61
3	.	.	**50**	**61**	**26**	**61**	**38**	**61**
4	26	61	26	61	26	61	26	61
5	.	45	**26**	45	**26**	45	**26**	45
6	.	21	**26**	21	**26**	21	**26**	21
7	26	11	26	11	26	11	26	11
8	12	5	12	5	12	5	12	5
9	6	.	6	**5**	6	**.**	6	**.**

However, as indicated in this table, the first missing value for patient (2) was not replaced in the carry-forward or averaging methods. The last observation for patient (2) also has a missing value that similarly could not be replaced by the carry-backward or averaging methods. In such situations, carry-backward is automatically applied for a missing value in the first observation, and carry-forward is applied for a missing value in the last observation. Therefore, we need to insert the following code in the macro so that any missing values in the first or last observations can be imputed:

```
if value eq . then value = max(ForWard, BackWard);
```

The final output, using this additional formula, is shown below. The missing value in the ForWard column is replaced with 83, in the BackWard column with 5, and in the averaging column with 83 and 5, respectively.

Time (hours)	Original Data _Value_		'*Complete*' Data					
			ForWard		**BackWard**		**Averaging**	
	Patient (1)	Patient (2)	Patient (1)	Patient (2)	Patient (1)	Patient (2)	Patient (1)	Patient (2)
0	98	.	98	**83**	98	**83**	98	**83**
1	50	83	50	83	50	83	50	83
2	50	61	50	61	50	61	50	61
3	.	.	**50**	**61**	**26**	**61**	**38**	**61**
4	26	61	26	61	26	61	26	61
5	.	45	**26**	45	**26**	45	**26**	45
6	.	21	**26**	21	**26**	21	**26**	21
7	26	11	26	11	26	11	26	11
8	12	5	12	5	12	5	12	5
9	6	.	6	**5**	6	**5**	6	**5**

4.) Random generation of the missing value based on a sample mean.

This last imputation method in the macro %SingleImpute is similar to multiple imputation in its approach. Therefore, it is useful to begin this section with a brief discussion of how multiple imputation works. The multiple imputation method has a completely different approach from the single imputation method. Multiple imputation does not attempt to fill in each missing value with a simulated value, as in single imputation. Instead, it attempts to replace each missing value with a set of plausible values that represent the uncertainty about the correct value, which is then imputed under the assumptions of missing at random (MAR) or missing completely at random (MCAR).

In the multiple imputation approach, multiple datasets are created with the missing values filled in from the distribution of the data, with different values for each of the missing items. This step is typically performed by a procedure called PROC MI, which generates multiple datasets. A standard statistical analysis must be conducted and repeated on all of these multiple datasets. Finally, the procedure PROC MIANALYZE will be applied to combine the results from the analyses on these multiple datasets. By reflecting the uncertainty about the predictions of the unknown missing values, the multiple imputation method yields valid statistical inferences regarding the missing values. However, the process is lengthy, and, depending on the type of analysis performed, the outcome may not be significantly different from that of the single imputation approach. In addition, there is a limitation with the procedure PROC MI: it only works well with medium-sized datasets or a limited number of variables. It becomes very slow when it encounters a large dataset with tens of thousands of records or more.

"Random generation based on a sample mean," the tenth and final imputation method available in the macro %SingleImpute, offers an alternative to PROC MI for creating a "complete" dataset. Unlike the other nine methods discussed in this paper, where missing values were replaced with one fixed value (such as the mean, minimum, or maximum), this last method imputes the missing values with a set of random observations with the SAS function RANNOR. The RANNOR function returns a variate that is generated from a normal distribution with mean 0 and variance 1. To create a random observation from a normal distribution, each missing value in the dataset is replaced by multiplying the standard deviation by the RANNOR function, and then adding the product to the non-missing mean, as in the equation, X=mean +(standard deviation)*RANNOR(seed). The following SAS codes would be used to apply this equation to the plasma dataset that we have been using as an example throughout this paper:

```
proc sort data=_temp_;
   by patient ;
run;

proc means data=_temp_ noprint;
   var _value_;
   by patient;
   output out=Target mean=m std=s;
run;

proc sql noprint;
   create table sample as
   select e.*
   ,case e._value_ when .
   then  f.m+f.s*rannor(0)
   else e._value_
   end as value
   from _temp_ as e left join Target as f
   on f.patient eq e.patient
   order by patient, time;
quit;
```

In the code above, the value zero is used for the seed in the RANNOR function. The resulting completed dataset is constructed as shown in the table below.

Time (hours)	Original Data _Value_		'*Complete*' Data **Random Mean**	
	Patient (1)	Patient (2)	Patient (1)	Patient (2)
0	98	.	98	**26.64**
1	50	83	50	83

2	50	61	50	61
3	.	.	**72.88**	**52.60**
4	26	61	26	61
5	.	45	**34.71**	45
6	.	21	**47.42**	21
7	26	11	26	11
8	12	5	12	5
9	6	.	6	**4.45**

Compared with the nine other single imputation methods that we have discussed, this last method is capable of reflecting a certain uncertainty in imputing missing data with a set of random sample observations. This method also takes into account more information about the dataset in predicting missing values.

CONCLUSION

In this paper, we introduced the single imputation macro %SingleImpute, which offers the efficiency and simplicity of the widely used single imputation approach to imputing missing data. This macro includes ten of the most commonly applied statistical methods used for imputation, allowing a user to generate a complete SAS dataset from an incomplete one by filling in the missing data values. One of the statistical methods used in this macro is called "random generation based on a sample mean." This method uses a set of random numbers to impute missing data values, reflecting some of the uncertainty of these missing values, without using the SAS procedure PROC MI. Multiple imputation, a very popular approach for imputing missing data, also reflects uncertainty in its imputations. However, unlike the methods in macro %SingleImpute, multiple imputation uses PROC MI, which has difficulty handling datasets with extremely large numbers of records or variables. While users should select the approach that best fits their particular needs for creating a complete SAS dataset from an incomplete one, the macro %SingleImpute is a versatile single imputation macro that offers users a wide range of options for imputation.

REFERENCES

1. SAS/STAT Documentation
2. Little, R.J.A. and Rubin, D.B. (1987), Statistical Analysis with Missing Data, New York; John Wiley &Sons, Inc.
3. D. Lanning, and B. Doug "An Alternative to PROC MI for Large Samples", SUGI 28 Conference Proceedings.

ACKNOWLEDGMENTS

The authors would like to thank Jason Liao and Jodi Benjamin for their valuable suggestions and reviewing of this paper.

CONTACT INFORMATION

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

Xingshu Zhu
Merck &Co., Inc.
UG1CD-14,
Upper Gwynedd, PA. 19454-2505
Phone: 267 305 2689
Fax: 267 305 6538
Email: xingshu_zhu@merck.com